Cloudera Runtime 7.0.0

Working with Apache Hive metastore

Date published: 2019-08-21
Date modified:

https://docs.cloudera.com/

Legal Notice

Contents

HMS table storage

You need to understand how HMS stores Hive tables when you run a CREATE TABLE statement or migrate a table to Cloudera Data Platform. The success or failure of the statement, the resulting table type, and the table location depends on a number of factors.

HMS table transformations

The HMS includes the following Hive metadata about tables that you create:

- A table definition
- Column names
- Data types
- Comments in a central schema repository

When you use EXTERNAL keyword in the CREATE TABLE statement, HMS stores the table as an external table. When you omit the EXTERNAL keyword and create a managed table, or ingest a managed table, HMS might translate the table into an external table or the table creation can fail, depending on the table properties. An important table property that affects table transformations is the ACID or Non-ACID table type:

Non-ACID

This property is true if table properties do not contain any ACID related properties. For example, the table does not contain such properties transactional=true or insert_only=true.

ACID

This property is true if table properties do contain one or more ACID properties.

Full ACID

This property is true if table properties contain transactional=true but not insert_only=true.

Insert-only ACID

Table properties contain insert_only=true.

The following matrix shows the table type and whether or not the location property is supported.

ACID	Managed	Location Property	Comments	Action
Non-ACID	Yes	Yes	Migrated to CDP, for example from an HDP or CDH cluster	Table stored as external
Non-ACID	Yes	No	Table location is null	Table stored in subdirectory metastore.warehouse.external.dir

HMS detects type of client for interacting with HMS, for example Hive or Spark, and compares the capabilities of the client with the table requirement. HMS performs the following actions, depending on the results of the comparison:

Table requirement	Client meets requirements	Managed Table	ACID table type	Action
Client can write to any type of ACID table	No	Yes	Yes	CREATE TABLE fails
Client can write to full ACID table	No	Yes	insert_only=true	CREATE TABLE fails
Client can write to insert-only ACID table	No	Yes	insert_only=true	CREATE TABLE fails

If, for example, a Spark client does not have the capabilities required, the following type of error message appears:

```
Spark has no access to table `mytable`. Clients can access this table only i
f
they have the following capabilities: CONNECTORREAD,HIVEFULLACIDREAD, HIVE
FULLACIDWRITE,
HIVEMANAGESTATS, HIVECACHEINVALIDATE, ...
```

Configuring HMS for high availability

To provide failover to a secondary Hive metastore if your primary instance goes down, you need to know how to add a Metastore role in Cloudera Manager and configure a property.

About this task

Multiple HMS instances run in active/active mode. Load balancing is done at the Hive client side (like HiveServer or Spark) as the HMS client picks an HMS instance randomly. By default, the hive.metastore.uri.selection property is set to RANDOM. If that HMS instance is down, then the client randomly picks another instance from the list of HMS instances specified through the hive.metastore.uris property.

Before you begin

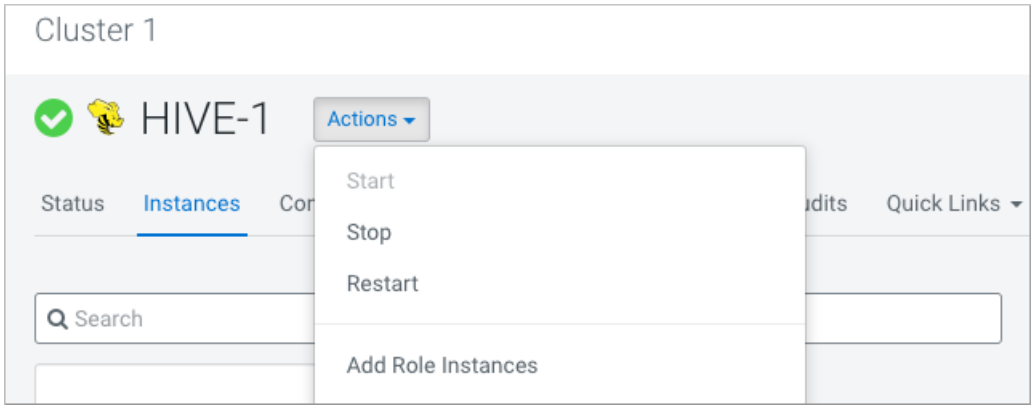
Minimum Required Role: Configurator (also provided by Cluster Administrator, Full Administrator)

Procedure

1. In Cloudera Manager, click Clusters Hive Configuration .
2. Take one of the following actions:
   - If you have a cluster secured by Kerberos, search for Hive Delegation Token Store, which specifies storage for the Kerberos token as described below.
   - If you have an unsecured cluster, skip the next step.
3. Select org.apache.hadoop.hive.thrift.DBTokenStore, and save the change.

Storage for the Kerberos delegation token is defined by the hive.cluster.delegation.token.store.class property. The available choices are Zookeeper, the Metastore, and memory. Cloudera recommends using the database by setting the org.apache.hadoop.hive.thrift.DBTokenStore property.

4. Click Instances Actions Add Role Instances



5. In Assign Roles, in Metastore Server, click Select Hosts.
6. In Hosts Selected, scroll and select the host that you want to serve as the backup Metastore, and click OK.
7. Click Continue until you exit the wizard.
8. Start the Metastore role on the host from the Actions menu.

   The hive.metastore.uris property is updated automatically. To verify, go to /etc/hive/config directory in your cluster node and look for the updated property in the hive-site.xml file.
9. To check or to change the hive.metastore.uri.selection property, go to Clusters Hive Configurations in Cloudera Manager, and search for 'Hive Service Advanced Configuration Snippet (Safety Valve) for hive-site.xml'.
10. Add the property and value (SEQUENTIAL or RANDOM).

Related Information

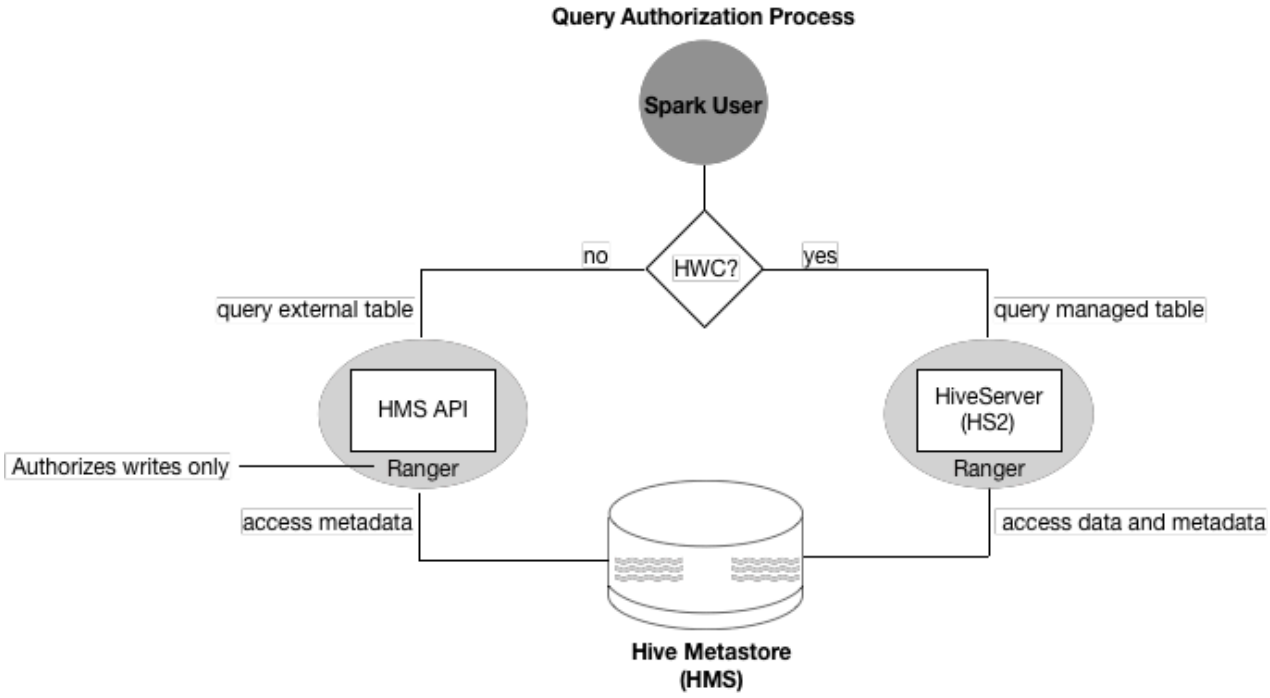
Example of using the Cloudera Manager Safety Valve

Custom Configuration (about Cloudera Manager Safety Valve)

Accessing Hive tables from Spark in CDP Data Center

You need to understand authorization access to Hive tables from Spark. You might need to, or want to, use the Hive Warehouse Connector (HWC), depending on your use case.

Ranger authorizes access to Hive tables from Spark through HiveServer (HS2) or the Hive metastore API (HMS API). The following diagram shows the typical query authorization process in CDP Data Center:



Managed tables

To read or write managed tables from Spark, you need to use the HWC. You explicitly use HWC by calling the HiveWarehouseConnector API for writes. You can implicitly use HWC for reads by simply running a Spark SQL query on a managed table. A Spark job impersonates the end user when attempting to access a Hive managed table. As an end user, you do not have permission to these secure files in the Hive warehouse.

Managed table queries go through HiveServer, which is integrated with Ranger. As Ranger Administrator, you configure permissions to access the managed tables. You can fine-tune Ranger to protect specific data. For example, you can mask data in certain columns, or set up tag-based access control.

External tables

To read or write external tables from Spark, can use the HWC, or not, based on your use case. The Spark job accesses the metadata first, and then indirectly (no HWC), or directly (through HWC) accesses the data. Of course, file system permissions must be set to allow Spark direct access to the data.

External table queries go through the HMS API, which is integrated with Ranger. This release supports Ranger authorization of external table writes. This release of CDP Data Center does not support Ranger authorization of external table reads. Reads are supported in the CDP Public Cloud release. You need to check for, and add, a few properties to enable authorization of external table writes, as described in the next section.

Creating an external table stores the metadata in HMS. Using HWC to create the external table, HMS keeps track of the location of table names and columns. Dropping an external table deletes the metadata from HMS and from the file system.

Not using HWC, dropping an external table deletes only the metadata from HMS. If you do not have permission to access the file system, and you want to drop table data in addition to metadata, you need to use HWC.

Related Information

Hive Warehouse Connector for accessing Apache Spark data

Configure access to managed tables from Spark

You need to configure properties in Cloudera Manager for write authorization to Hive managed tables when accessing the tables from Spark.

About this task

You add the following properties and values to hive-site.xml for HiveServer-Ranger integration:
hive.security.authorization.manager

Value: org.apache.ranger.authorization.hive.authorizer.RangerHiveAuthorizerFactory

hive.security.authorization.enabled

Value: true

Procedure

1. In Cloudera Manager, to configure Hive Metastore properties click Clusters Hive-on-Tez Configuration .
2. Search for each property.
3. Set the value given above.
4. Save changes.

Related Information

Hive Warehouse Connector for accessing Apache Spark data

Authorize read access to tables from Spark

You need to configure properties in Cloudera Manager for read authorization to Hive external tables when accessing the tables from Spark.

About this task

You add the following properties and values to hive-site.xml for HMS API-Ranger integration:
hive.security.authenticator.manager

Value: org.apache.hadoop.hive.ql.security.SessionStateUserAuthenticator

hive.metastore.pre.event.listeners

Value: org.apache.hadoop.hive.ql.security.authorization.plugin.metastore.HiveMetaStoreAuthorizer

Add properties as described in the next section.

Related Information

Hive Warehouse Connector for accessing Apache Spark data

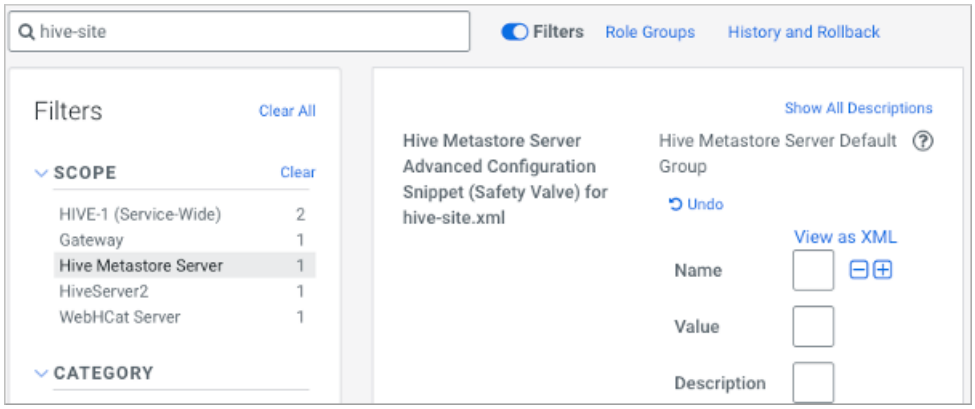
Add HMS properties to hive-site.xml

To add HMS properties and values to hive-site.xml, you use Cloudera Manager.

Procedure

1. In Cloudera Manager, to configure Hive Metastore properties click Clusters Hive-1 Configuration .
2. Search for hive-site.

3. In Hive Metastore Server Advanced Configuration Snippet (Safety Valve) for hive-site.xml, click +.



4. Add a property name and value.
5. Repeat steps to add other properties.
6. Save changes.

Related Information

Hive Warehouse Connector for accessing Apache Spark data

Filter HMS results

HMS can perform server-side filtering of data returned by a read operation. Enabling the filter can show results of statements, such as SHOW TABLES or SHOW DATABASES, based on who the user is. You enable filtering by setting a boolean flag and hook. The hook identifies the class name that implements the filtering.

About this task

You add the following properties and values in hive-site.xml for HMS API-Ranger integration:

metastore.server.filter.enabled

Value: true (to do filtering) or false (no filtering)

metastore.filter.hook

Value: org.apache.hadoop.hive.ql.security.authorization.plugin.metastore.HiveMetaStoreAuthorizer

Add properties as described in the previous section.

Setting up the metastore database

In CDP Private Cloud Base, you need to install a supported database for Hive metastore (HMS) to store the metadata. You configure Hive metastore by modifying hive-site.xml using the Cloudera Manager Safety Valve feature, described later, instead of using hive set key=value on the command line.

Related Information

Apache Wiki: Hive Metastore Administration

Example of using the Cloudera Manager Safety Valve

Custom Configuration (about Cloudera Manager Safety Valve)

Setting up the backend Hive metastore database

In CDP Private Cloud Base you need to install, start, and configure a backend database for Hive metastore.

About this task

In this procedure, you install a database on a different node/cluster from the HiveServer for sharing the Hive metastore (HMS) with Hive, Impala, Spark, and other components. Do not put HiveServer and the database on the same node. You can have one or more HMS instances in your cluster that can take over in the event of a problem.

Procedure

Install the backend database.

Set up a PostgreSQL database

You install a Postgres database to serve as the backend database for the Hive metastore. You also install the Postgres driver on your cluster, and then configure the database.

Procedure

1. Install the database from the command line of a node in your cluster.
   - RHEL:
     ```
     sudo yum install postgresql-server
     ```
   - SLES:
     ```
     sudo zypper install postgresql-server
     ```
   - Ubuntu:
     ```
     sudo apt-get install postgresql
     ```
2. If your system is RHEL, initialize database files; otherwise, skip this step.
   ```
   sudo service postgresql initdb
   ```
3. Set the listen_addresses property to * and set the standard_conforming_strings property off.
   - RHEL and SLES:
     ```
     sudo cat /var/lib/pgsql/data/postgresql.conf  | grep -e listen -e standa
     rd_conforming_strings
     listen_addresses = '*'
     standard_conforming_strings = off
     ```
   - Ubuntu:
     ```
     cat /etc/postgresql/9.1/main/postgresql.conf | grep -e listen -e standar
     d_conforming_strings
     listen_addresses = '*'
     standard_conforming_strings = off
     ```

4. Add a new line into pg_hba.con to ensure that the Postgres user can access the server remotely.
   For example, to allow all users to connect from all hosts to all your databases:

   ```
   host  <database>  <user>  <network address>  <mask>   md5
   ```

   This configuration is applicable only for a network listener. Using this configuration does not open all your databases to the entire world; you must provide a password to authenticate yourself, and privilege restrictions configured in PostgreSQL are effective.
5. Start the Postgres server.

   ```
   sudo service postgresql start
   ```

6. Set Postgres to start when the cluster comes up, and check the configuration.

   ```
   chkconfig postgresql on
   chkconfig --list postgresql
   ```

7. On the Hive metastore server node in your cluster, install the latest Postgres JDBC driver (the postgresql-jdbc package), and create symbolic link to the /usr/lib/hive/lib/ directory.
   - RHEL

     ```
     sudo yum install postgresql-jdbc
     -- For Package install method:
     ln -s /usr/share/java/postgresql-jdbc.jar /usr/lib/hive/lib/postgresql-j
     dbc.jar
     -- For Parcel install method:
     ln -s /usr/share/java/postgresql-jdbc.jar /opt/cloudera/parcels/CDH/lib/
     hive/lib/postgresql-jdbc.jar
     ```

   - SLES

     ```
     sudo zypper install postgresql-jdbc
     -- For Package install method:
     ln -s /usr/share/java/postgresql-jdbc.jar /usr/lib/hive/lib/postgresql-
     jdbc.jar
     -- For Parcel install method:
     ln -s /usr/share/java/postgresql-jdbc.jar /opt/cloudera/parcels/CDH/lib/
     hive/lib/postgresql-jdbc.jar
     ```

   - Ubuntu

     ```
     sudo apt-get install libpostgresql-jdbc-java
     -- For Package install method:
     ln -s /usr/share/java/postgresql-jdbc4.jar /usr/lib/hive/lib/postgresql-
     jdbc4.jar
     -- For Parcel install method:
     ln -s /usr/share/java/postgresql-jdbc4.jar /opt/cloudera/parcels/CDH/lib
     /hive/lib/postgresql-jdbc4.jar
     ```

8. Create a metastore database and user account to access the metastore, using the script for your Hive version represented by n.n.n.

   ```
   sudo -u postgres psql
   postgres=# CREATE USER hiveuser WITH PASSWORD 'mypassword';
   postgres=# CREATE DATABASE metastore;
   postgres=# \c metastore;
   You are now connected to database 'metastore'.
   -- For Package install method:
   postgres=# \i /usr/lib/hive/scripts/metastore/upgrade/postgres/hive-schem
   a-n.n.n.postgres.sql
   -- For Parcel install method:
   ```

```
postgres=# \i /opt/cloudera/parcels/CDH/lib/hive/scripts/metastore/upgrad
e/postgres/hive-schema-n.n.n.postgres.sql
SET
SET
...
```

Configure metastore database properties

In CDP Private Cloud Base, you configure Hive and Hive metastore by modifying hive-site.xml indirectly using the Cloudera Manager Safety Valve feature. Using hive set key=value on the command line is not supported.

About this task

This task assumes the database is running on myhost, the user account is hiveuser, and the password is mypassword. Substitute the following connection URLs and driver names.

-
- Postgres connection URL: jdbc:postgresql://myhost/metastore

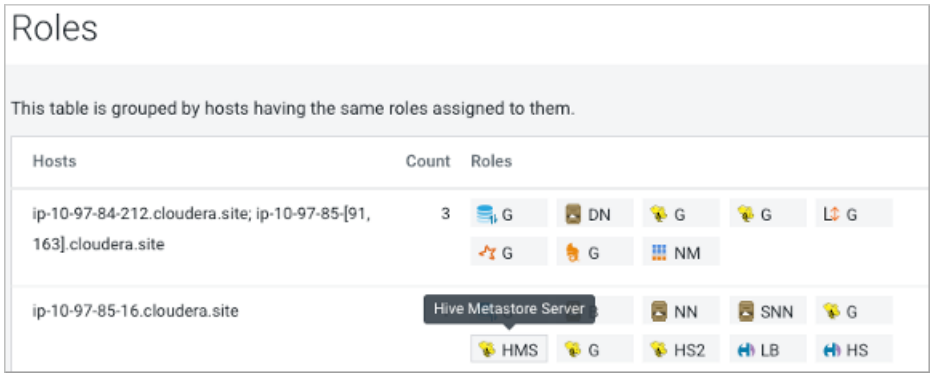
  Postgres driver name: jdbc:postgresql://myhost/metastore
-

Before you begin

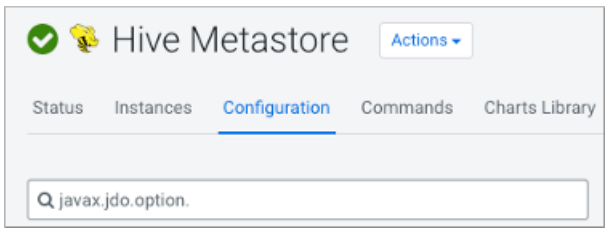
- The following components are running:
  - HiveServer
  - Hive Metastore
  - A database for the metastore, such as the default MySQL Server
  - Hive clients
- Minimum Required Role: Configurator (also provided by Cluster Administrator, Full Administrator)

Procedure

1. Find the fully qualified domain name or IP address of Hive metastore by navigating to Cloudera Manager Hosts Role(s) and looking through the list of roles to find Hive Metastore Server.



2. Navigate to the metastore host configuration in Clusters Hive Metastore Configuration , and search for javax.jdo.option.ConnectionURL.



3. In Value, specify the database connection string using the following syntax: <connection protocol>://<metastore host>/<metastore database>?createDatabaseIfNotExist=true
4. Optionally, repeat the previous steps on all hosts in the cluster.
5. In the same manner, specify other required connection properties on the metastore host (required), or on all hosts (optional) as shown in the following example.

```
<property>
<name>javax.jdo.option.ConnectionDriverName</name>
<value>(your driver name)</value>
</property>
<property>
<name>javax.jdo.option.ConnectionUserName</name>
<value>hive</value>
</property>

<property>
<name>javax.jdo.option.ConnectionPassword</name>
<value>mypassword</value>
</property>

<property>
<name>datanucleus.autoCreateSchema</name>
<value>false</value>
</property>

<property>
<name>datanucleus.fixedDatastore</name>
<value>true</value>
</property>

<property>
<name>datanucleus.autoStartMechanism</name>
<value>SchemaTable</value>
</property>

<property>
<name>hive.metastore.schema.verification</name>
<value>true</value>
</property>
```

Related Information

Example of using the Cloudera Manager Safety Valve

Custom Configuration (about Cloudera Manager Safety Valve)

Configure metastore location and HTTP mode

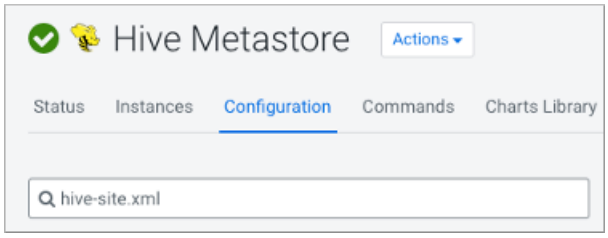
In addition to the database properties you need to set in CDP Private Cloud Base, you must configure the metastore URI property. This property defines one or more metastore locations. In CDP Public Cloud, you need to check the HTTP transport mode property. This property is set to http only for HiveServer and not for the metastore.

Before you begin

- The following components are running:
  - HiveServer
  - Hive Metastore
  - A database for the metastore, such as the default MySQL Server
  - Hive clients
- Minimum Required Role: Configurator (also provided by Cluster Administrator, Full Administrator)

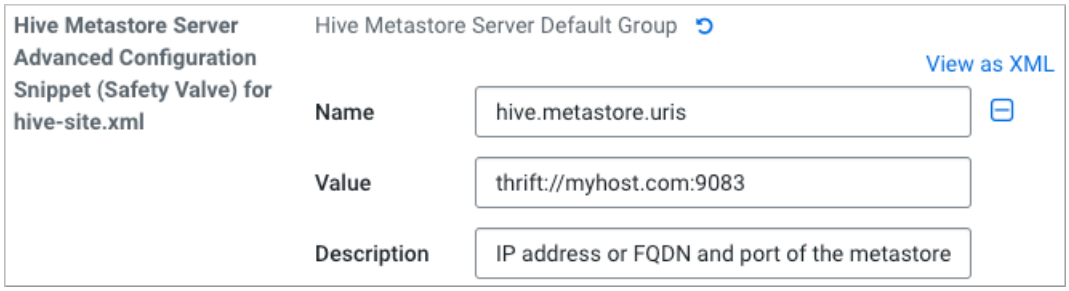
Procedure

1. In Clusters Hive Metastore Configuration , search for hive-site.xml.



2. In the Hive Metastore Server Advanced Configuration Safety Valve, which you use to change properties in hive-site.xml, click + and add the hive.metastore.uris property using the following syntax: thrift://<n.n.n.n>:9083

   Substitute for <n.n.n.n> an IP address or fully qualified domain name (FQDN) of the metastore host.



   Only the Hive Metastore Server Default Group in hive-site.xml should define this property.

Set up a JDBC URL connection override

You can configure fine-grained tuning of the HMS database connection. You specify a JDBC URL override for establishing a connection to the Hive metastore database.

About this task

This task is intended for advanced database users only. When using this override, the following properties are overwritten

- Hive Metastore Database Name
- Hive Metastore Database Host
- Hive Metastore Database Port
- Enable TLS/SSL to the Hive Metastore Database

Before you begin

- The required default user role is Configurator.
- You know the values for setting the following properties:
  - Hive Metastore Database Type
  - Hive Metastore Database User
  - Hive Metastore Database Password

Procedure

1. Set the value of the Hive Metastore Database JDBC URL Override property to jdbc:postgresql://<host>:<port>/<metastore_db>?key=value.
2. Click Save.
3. Click Actions Deploy Client Configuration .

4. Restart Hive Metastore.

Tuning the metastore

Generally, you need to limit concurrent connections to Hive metastore. As the number of open connections increases, so does latency. Issues with the backend database, improper Hive use, such as extremely complex queries, a connection leak, and other factors can affect performance.

General Metastore Tuning in CDP Private Cloud Base

Try making the following changes to tune HMS performance:

- Ensure that a single query accesses no more than 10,000 table partitions. If the query joins tables, calculate the combined partition count accessed across all tables.
- Tune the backend (the RDBMS). HiveServer connects to HMS, and only HMS connects to the RDBMS. The longer the backend takes, the more memory the HMS needs to respond to the same requests. Limit the number of connections in the backend database.
- Use default thrift properties (8K):

```
hive.server2.async.exec.threads 8192
hive.server2.async.exec.wait.queue.size 8192
hive.server2.thrift.max.worker.threads 8192
```

- Set datanucleus.connectionPool.maxPoolSize for your applications. For example, if poolSize = 100, with 3 HMS instances (one dedicated to compaction), and with 4 pools per server, you can accommodate 1200 connections.