

## Working with Apache Hive metastore

Date published: 2019-08-21

Date modified: 2022-02-24



# Legal Notice

© Cloudera Inc. 2025. All rights reserved.

The documentation is and contains Cloudera proprietary information protected by copyright and other intellectual property rights. No license under copyright or any other intellectual property right is granted herein.

Unless otherwise noted, scripts and sample code are licensed under the Apache License, Version 2.0.

Copyright information for Cloudera software may be found within the documentation accompanying each component in a particular release.

Cloudera software includes software from various open source or other third party projects, and may be released under the Apache Software License 2.0 (“ASLv2”), the Affero General Public License version 3 (AGPLv3), or other license terms. Other software included may be released under the terms of alternative open source licenses. Please review the license and notice files accompanying the software for additional licensing information.

Please visit the Cloudera software product page for more information on Cloudera software. For more information on Cloudera support services, please visit either the Support or Sales page. Feel free to contact us directly to discuss your specific needs.

Cloudera reserves the right to change any products at any time, and without notice. Cloudera assumes no responsibility nor liability arising from the use of products, except as expressly agreed to in writing by Cloudera.

Cloudera, Cloudera Altus, HUE, Impala, Cloudera Impala, and other Cloudera marks are registered or unregistered trademarks in the United States and other countries. All other trademarks are the property of their respective owners.

Disclaimer: EXCEPT AS EXPRESSLY PROVIDED IN A WRITTEN AGREEMENT WITH CLOUDERA, CLOUDERA DOES NOT MAKE NOR GIVE ANY REPRESENTATION, WARRANTY, NOR COVENANT OF ANY KIND, WHETHER EXPRESS OR IMPLIED, IN CONNECTION WITH CLOUDERA TECHNOLOGY OR RELATED SUPPORT PROVIDED IN CONNECTION THEREWITH. CLOUDERA DOES NOT WARRANT THAT CLOUDERA PRODUCTS NOR SOFTWARE WILL OPERATE UNINTERRUPTED NOR THAT IT WILL BE FREE FROM DEFECTS NOR ERRORS, THAT IT WILL PROTECT YOUR DATA FROM LOSS, CORRUPTION NOR UNAVAILABILITY, NOR THAT IT WILL MEET ALL OF CUSTOMER’S BUSINESS REQUIREMENTS. WITHOUT LIMITING THE FOREGOING, AND TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, CLOUDERA EXPRESSLY DISCLAIMS ANY AND ALL IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, QUALITY, NON-INFRINGEMENT, TITLE, AND FITNESS FOR A PARTICULAR PURPOSE AND ANY REPRESENTATION, WARRANTY, OR COVENANT BASED ON COURSE OF DEALING OR USAGE IN TRADE.

# Contents

|                                                   |          |
|---------------------------------------------------|----------|
| <b>HMS table storage.....</b>                     | <b>4</b> |
| <b>Configuring HMS for high availability.....</b> | <b>5</b> |

## HMS table storage

You need to understand how the Hive metastore (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 the `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

Table properties do not contain any ACID related properties set to true. For example, the table does not contain such properties `transactional=true` or `insert_only=true`.

#### ACID

Table properties do contain one or more ACID properties set to true.

#### Full ACID

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 of external warehouse* |

\* `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           | <code>insert_only=true</code> | CREATE TABLE fails |
| Client can write to insert-only ACID table | No                        | Yes           | <code>insert_only=true</code> | 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 if
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

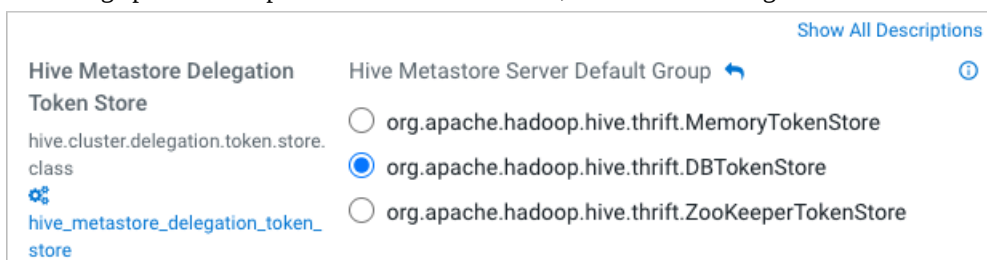
Multiple HMS instances run in active/active mode. Load balancing is done at the Hive client side (like HiveServer or Spark) as the HMS client picks an HMS instance randomly. By default, the `hive.metastore.uri.selection` property is set to `RANDOM`. If that HMS instance is down, then the client randomly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



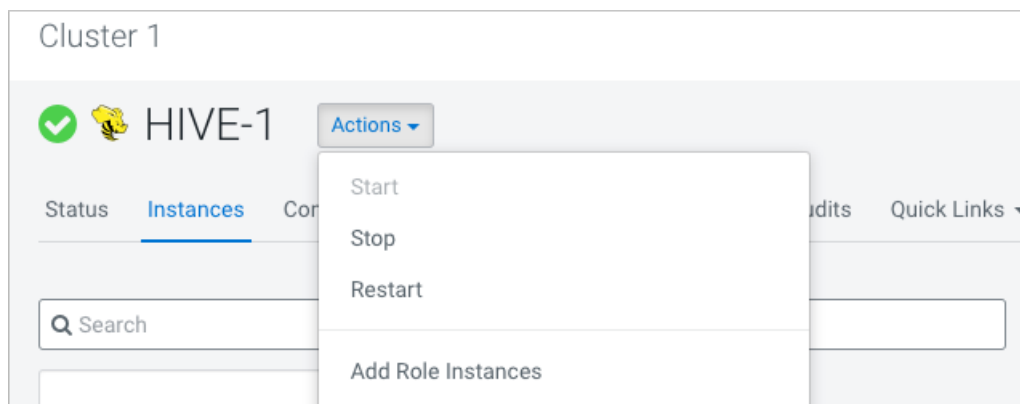
The screenshot shows the configuration page for the **Hive Metastore Delegation Token Store**. On the left, the property `hive.cluster.delegation.token.store.class` is listed with a gear icon and a link to `hive_metastore_delegation_token_store`. On the right, under the heading **Hive Metastore Server Default Group**, there are three radio button options:
 

- ☐ `org.apache.hadoop.hive.thrift.MemoryTokenStore`
- ☒ `org.apache.hadoop.hive.thrift.DBTokenStore`
- ☐ `org.apache.hadoop.hive.thrift.ZooKeeperTokenStore`

 A [Show All Descriptions](#) link is visible in the top right corner of the configuration area.

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

[Custom Configuration \(about Cloudera Manager Safety Valve\)](#)