

Cloudera Runtime 7.1.1

Configuring Apache Hive

Date published: 2019-08-21

Date modified:

CLOUdera

<https://docs.cloudera.com/>

Legal Notice

© Cloudera Inc. 2024. 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

Limit concurrent connections.....	4
Configuring HiveServer high availability using a load balancer.....	5
Configuring the Hive Delegation Token Store.....	5
Adding a HiveServer role.....	6
Configuring the HiveServer load balancer.....	7
Configuring HiveServer high availability using ZooKeeper.....	8
Generating statistics.....	8
Set up the cost-based optimizer and statistics.....	9
Generate and view Apache Hive statistics.....	10
Statistics generation and viewing commands.....	10
Removing scratch directories.....	11

Limit concurrent connections

To prevent a rogue application from repeatedly connecting to and monopolizing HiveServer, you can limit concurrent connections to HiveServer.

About this task

As administrator, you can limit concurrent connections using the Cloudera Manager Safety Valve to add one or more of the following properties to the hive-site.xml configuration file:

hive.server2.limit.connections.per.user

Maximum number of HiveServer concurrent connections per user

hive.server2.limit.connections.per.ipaddress

Maximum number of HiveServer concurrent connections per IP address

hive.server2.limit.connections.per.user.ipaddress

Maximum number of HiveServer concurrent connections per user and IP address combination

The default of each parameter is 0. You can change the value of each parameter to any number. You must configure concurrent connections on the server side; therefore, a hive --hiveconf command does not work.

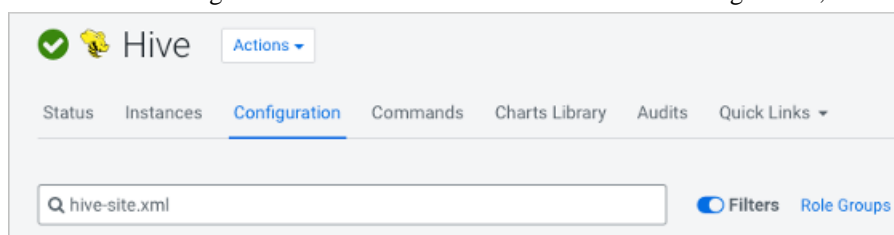
In this task, limit the number of connections per user to 25.

Before you begin

- The following components are running:
 - HiveServer
 - Hive Metastore
 - Hive client
- Minimum Required Role: Configurator (also provided by Cluster Administrator, Full Administrator)

Procedure

- In Cloudera Manager Clusters select the Hive service. Click Configuration, and search for hive-site.xml.



- In HiveServer2 Advanced Configuration Snippet (Safety Valve) for hive-site.xml, click + and add the hive.server2.limit.connections.per.user property.
- Enter a value representing the maximum number of concurrent connections: for example 25.

- Click Save.

5. Click **Actions** **Deploy Client Configuration** .
6. Restart HIVE.

Configuring HiveServer high availability using a load balancer

To enable high availability for multiple HiveServer (HS2) hosts, you need to know how to configure a load balancer to manage them. First, you configure the Hive Delegation Token Store, next you add HS2 roles, and finally, you configure the load balancer.

About this task

HiveServer HA does not automatically fail and retry long-running Hive queries. If any of the HS2 instances fail, all queries running on that instance fail and are not retried. The client application must re-submit the queries.

After you enable HS2 high availability, ensure that all your clients reflect the load balancer's principal in the connection string. On Kerberos-enabled clusters, you must use the load balancer's principal to connect to HS2 directly; otherwise, after you enable HS2 high availability, direct connections to HS2 instances fail.

Before you begin

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

Configuring the Hive Delegation Token Store

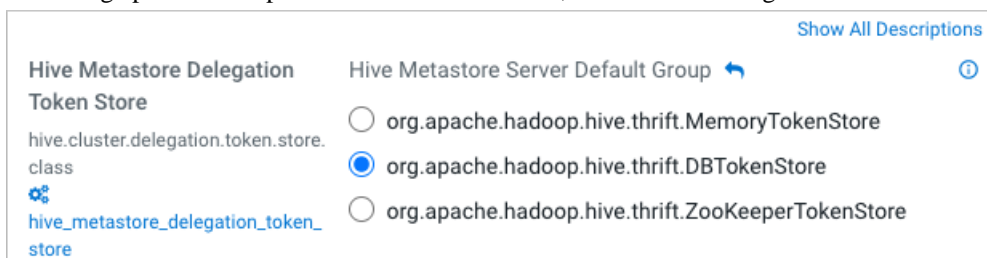
You need to enable Hive Delegation Token Store implementation as the first step in configuring HiveServer high availability using a load balancer. You also need to understand the interaction between Oozie and HS2 with regard to the delegation token.

About this task

Oozie needs this implementation for secure HiveServer high availability (HA). Otherwise, the Oozie server can get a delegation token from one HS2 server, but the actual query might run against another HS2 server, which does not recognize the HS2 delegation token.

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. Do not use the `MemoryTokenStore`. This can cause failures because one HS2 does not recognize the delegation token issued by another.

4. Add HiveServer (HS2) roles as described in the next topic.

Adding a HiveServer role

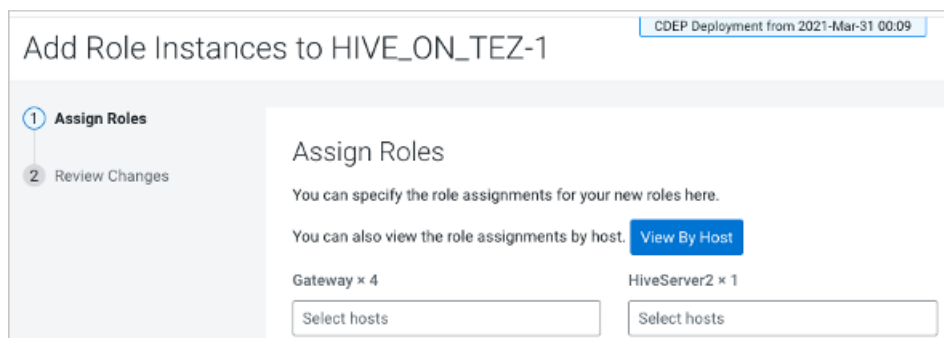
You can add a HiveServer (HS2) role to the Hive-on-Tez service, not to the Hive service.

Before you begin

You configured the Hive Delegation Token Store.

Procedure

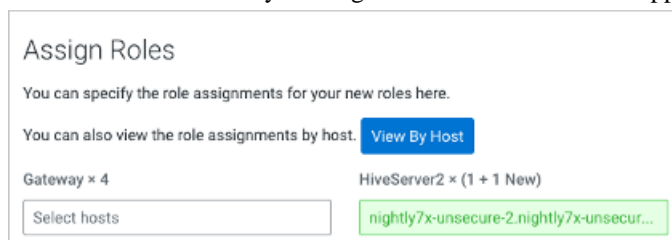
1. In Cloudera Manager, click **Clusters** **Hive on Tez** .
Do not click **Clusters** **Hive** by mistake. Only the Hive on Tez service supports the HiveServer2 role.
2. Click **Actions** **Add Role Instances** .



3. Click in the HiveServer2 box to select hosts.

<input type="checkbox"/>	Hostname	IP Address	Rack	Cores	Physical Memory	Existing Roles
<input checked="" type="checkbox"/>	nightly7x-unsecure-1.nightly7x-unsecure.root.hwx.site	172.27.75.0	/default	64	503.6 GiB	CCS G HS2 LB HS AP ES HM RM SCM QS SRS SS G G JHS RM
<input type="checkbox"/>	nightly7x-unsecure-2.nightly7x-	172.27.75.2	/default	64	503.6 GiB	RS DN G G NM SS G G

4. In the Host name column, select a host for the HiveServer2 role, and click OK.
The selected host name you assigned the HiveServer2 role appears under HiveServer2.



5. Click **Continue**.
The new HiveServer2 role state is stopped.

6. Select the new HiveServer2 role.

Actions for Selected (1) ▾				
<input type="checkbox"/>	Status	Role Type	State	Hostname
<input checked="" type="checkbox"/>		HiveServer2	Stopped	nightly7x-unsecure-2
<input type="checkbox"/>		HiveServer2	Started	nightly7x-unsecure-1

7. In Actions for Selected, select Start, and then click Start to confirm. You see that the service successfully started.

Start

Status **Finished** Context [HiveServer2 \(nightly7x-unsecure-2\)](#) Apr 1, 3:08:53 AM

23.15s

Successfully started service.

▾ **Completed 1 of 1 step(s)**

Show All Steps Show Only Failed Steps Show Only Running Steps

> Starting 1 roles on service

Configuring the HiveServer load balancer

Cloudera Manager exposes the HiveServer load balancer property. You see how to access the property and set it.

Before you begin

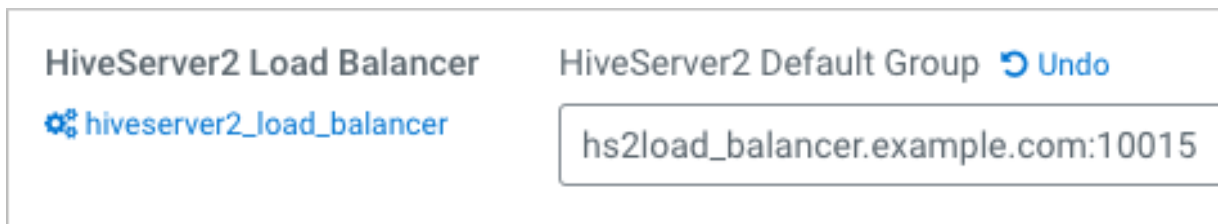
- You configured the Hive Delegation Token Store.
- You added one or more HiveServer roles.

Procedure

- In Cloudera Manager, click Clusters Hive On Tez Configuration , and search for HiveServer2 Load Balancer.

The screenshot shows the Cloudera Manager configuration interface for a cluster named HIVE_ON_TEZ-1. The 'Configuration' tab is active, and a search for 'Load Bal' has been performed. The search results display the 'HiveServer2 Load Balancer' property, which is currently set to 'hiveserver2_load_balancer'. The 'Filters' section on the left shows the 'SCOPE' filter expanded, with 'HiveServer2' selected under the 'HIVE_ON_TEZ-1 (Service-Wide)' category. The page also shows the 'HiveServer2 Default Group' property and a 'Show All Descriptions' link.

2. Set the value, using the following format: <hostname>:<port number>.
For example, hs2load_balancer.example.com:10015.



3. Save the change.

Configuring HiveServer high availability using ZooKeeper

You need to know how to configure your Hive-on-Tez to use ZooKeeper for HiveServer high availability.

When a client requests an HS2 instance, ZooKeeper returns one randomly-selected registered HS2. If more than one HS2 instance is registered with ZooKeeper, and all instances fail except one, ZooKeeper passes the link to the instance that is running and the client can connect successfully. (Failed instances must be restarted manually.)

Automatic failover does not occur. If an HS2 instance failed while a client is connected, the session is lost. Since this situation need to be handed at the client, there is no automatic failover; the client needs to reconnect using ZooKeeper.

Using binary transport mode in HiveServer (HS2), Knox, and Dynamic Discovery, possibly supported on your platform before upgrading to CDP, are not supported on CDP. Use alternate solutions, such as HAProxy.

Related Information

[Adding a Role Instance](#)

Generating statistics

A cost-based optimizer (CBO) generates efficient query plans. Hive does not use the CBO until you generate column statistics for tables. By default, Hive gathers only table statistics. You need to configure Hive to enable gathering of column statistics.

The CBO, powered by Apache Calcite, is a core component in the Hive query processing engine. The CBO optimizes plans for executing a query, calculates the cost, and selects the least expensive plan to use. In addition to increasing the efficiency of execution plans, the CBO conserves resources.

How the CBO works

After parsing a query, a process converts the query to a logical tree (Abstract Syntax Tree) that represents the operations to perform, such as reading a table or performing a JOIN. Calcite applies optimizations, such as query rewrite, JOIN re-ordering, JOIN elimination, and deriving implied predicates to the query to produce logically equivalent plans. Bushy plans provide maximum parallelism. Each logical plan is assigned a cost that is based on distinct, value-based heuristics.

The Calcite plan pruner selects the lowest-cost logical plan. Hive converts the chosen logical plan to a physical operator tree, optimizes the tree, and converts the tree to a Tez job for execution on the Hadoop cluster.

Explain plans

You can generate explain plans by running the EXPLAIN query command. An explain plan shows you the execution plan of a query by revealing the operations that occur when you run the query. Having a better understanding of the plan, you might rewrite the query or change Tez configuration parameters.

Set up the cost-based optimizer and statistics

You can use the cost-based optimizer (CBO) and statistics to develop efficient query execution plans that can improve performance. You must generate column statistics to make CBO functional.

About this task

In this task, you enable and configure the cost-based optimizer (CBO) and configure Hive to gather column statistics as well as table statistics for evaluating query performance. Column and table statistics are critical for estimating predicate selectivity and the cost of the plan. Certain advanced rewrites require column statistics.

In this task, you check, and set the following properties:

- `hive.stats.autogather`
Controls collection of table-level statistics.
- `hive.stats.fetch.column.stats`
Controls collection of column-level statistics.
- `hive.compute.query.using.stats`
Instructs Hive to use statistics when generating query plans.

You can manually generate the table-level statistics for newly created tables and table partitions using the `ANALYZE TABLE` statement.

Before you begin

- The following components are running:
 - HiveServer
 - Hive Metastore
 - Hive clients
- Minimum Required Role: Configurator (also provided by Cluster Administrator, Full Administrator)

Procedure

1. In Cloudera Manager, click `Clusters Hive On Tez Configuration`.
2. In the Configuration page, search for the `hive.cbo.enable` property.

The screenshot shows the Cloudera Manager configuration interface for a cluster named 'HIVE_ON_TEZ-1'. The search bar at the top contains the text 'hive.cbo.enable'. Below the search bar, there are tabs for 'Filters', 'Role Groups', and 'History & Rollback'. The main content area displays a configuration entry: 'Enable Cost-Based Optimizer for Hive'. This entry has a checked checkbox next to 'HiveServer2 Default Group'. Below this, the property name 'hive.cbo.enable' and its configuration file 'hiveserver2_enable.cbo' are listed. On the left side, there is a 'Filters' section with a 'SCOPE' dropdown menu. Under 'SCOPE', there are three entries: 'HIVE_ON_TEZ-1 (Service-Wide)' with a value of 0, 'Gateway' with a value of 0, and 'HiveServer2' with a value of 1. A 'Show All Descriptions' link is visible in the top right of the configuration list.

If the property is not visible in your version of Cloudera Manager, add the property to Hive site using the Cloudera Manager Safety Valve (see links below). Set the property to enabled.

3. Accept the default (enabled), or check to enable the `hive.cbo.enable` property for the HiveServer Default Group.

4. Search for and enable, if necessary, `hive.stats.fetch.column.stats`.
5. Search for and enable, if necessary, `hive.compute.query.using.stats`.
6. Click **Actions Restart** to restart the Hive on Tez service.

Related Information

[Example of using the Cloudera Manager Safety Valve](#)

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

Generate and view Apache Hive statistics

You can use statistics to optimize queries for improved performance. The cost-based optimizer (CBO) also uses statistics to compare query plans and choose the best one. By viewing statistics instead of running a query, you can often get answers to your data questions faster.

About this task

This task shows how to generate different types of statistics about a table.

Procedure

1. Launch a Hive shell and log in.
2. Gather table statistics for the non-partitioned table `mytable`:

```
ANALYZE TABLE mytable COMPUTE STATISTICS;
```

3. View table statistics you generated:

```
DESCRIBE EXTENDED mytable;
```

4. Gather column-level statistics for the table:

```
ANALYZE TABLE mytable COMPUTE STATISTICS FOR COLUMNS;
```

5. View column statistics for the `col_name` column in `my_table` in the `my_db` database:

```
DESCRIBE FORMATTED my_db.my_table col_name;
```

Related Information

[Apache Hive Wiki language reference](#)

[Apache Hive Wiki - Statistics in Hive](#)

Statistics generation and viewing commands

You can manually generate table and column statistics, and then view statistics using Hive queries. By default, Hive generates table statistics, but not column statistics, which you must generate manually to make cost-based optimization (CBO) functional.

Commands for generating statistics

The following `ANALYZE TABLE` command generates statistics for tables and columns:

```
ANALYZE TABLE [table_name] COMPUTE STATISTICS;
```

Gathers table statistics for non-partitioned tables.

```
ANALYZE TABLE [table_name] PARTITION(partition_column) COMPUTE STATISTICS;
```

Gathers table statistics for partitioned tables.

```
ANALYZE TABLE [table_name] COMPUTE STATISTICS for COLUMNS [comma_separated_column_list];
```

Gathers column statistics for the entire table.

```
ANALYZE TABLE partition2 (col1="x") COMPUTE STATISTICS for COLUMNS;
```

Gathers statistics for the partition2 column on a table partitioned on col1 with key x.

Commands for viewing statistics

You can use the following commands to view table and column statistics:

```
DESCRIBE [EXTENDED] table_name;
```

View table statistics. The EXTENDED keyword can be used only if the hive.stats.autogather property is enabled in the hive-site.xml configuration file. Use the Cloudera Manager Safety Valve feature (see link below).

```
DESCRIBE FORMATTED [db_name.]table_name [column_name] [PARTITION (partition_spec)];
```

View column statistics.

Related Information

[Example of using the Cloudera Manager Safety Valve](#)

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

Removing scratch directories

You need to know how to periodically clear scratch directories used by Apache Hive to prevent problems, such as failing jobs.

About this task

Scratch directories where Hive stores intermediate, or temporary, files accumulate too much data over time and overflow. You can configure Hive to remove scratch directories periodically and without user intervention. Using Cloudera Manager, you add the following properties as shown in the procedure:

hive.start.cleanup.scratchdir

Value: true

Cleans up the Hive scratch directory while starting the HiveServer.

hive.server2.clear.dangling.scratchdir

Value: true

Starts a thread in HiveServer to clear out the dangling directories from the file system, such as HDFS.

hive.server2.clear.dangling.scratchdir.interval

Example Value: 1800s

Procedure

1. In Cloudera Manager, click **Clusters Hive on Tez Configuration** . **Clusters > Hive on Tez > Configuration**.
2. Search for the **Hive Service Advanced Configuration Snippet (Safety Valve) for hive-site.xml** setting.
3. In the **Hive Service Advanced Configuration Snippet (Safety Valve) for hive-site.xml** setting, click **+**.
4. In **Name** enter the property name and in **Value** enter the value.