Cloudera Runtime 1.5.5

Configuring Apache Hive

Date published: 2020-11-30
Date modified: 2025-06-06

https://docs.cloudera.com/

Legal Notice

Contents

Configuring Hive in Cloudera Data Warehouse

Configuring Hive performance in the Cloud is rarely, if ever, required relative to the extensive performance tuning typical in a bare metal environment. You might occasionally want to configure Hive properties unrelated to performance, such as HiveServer (HS2) administration or logging level properties.

About this task

Changing one of the vast array of configuration parameters from Cloudera Data Warehouse is recommended only when following Cloudera instructions. For example, you follow instructions in a compaction alert or the product documentation that describes the property and value to change, and how to do it.

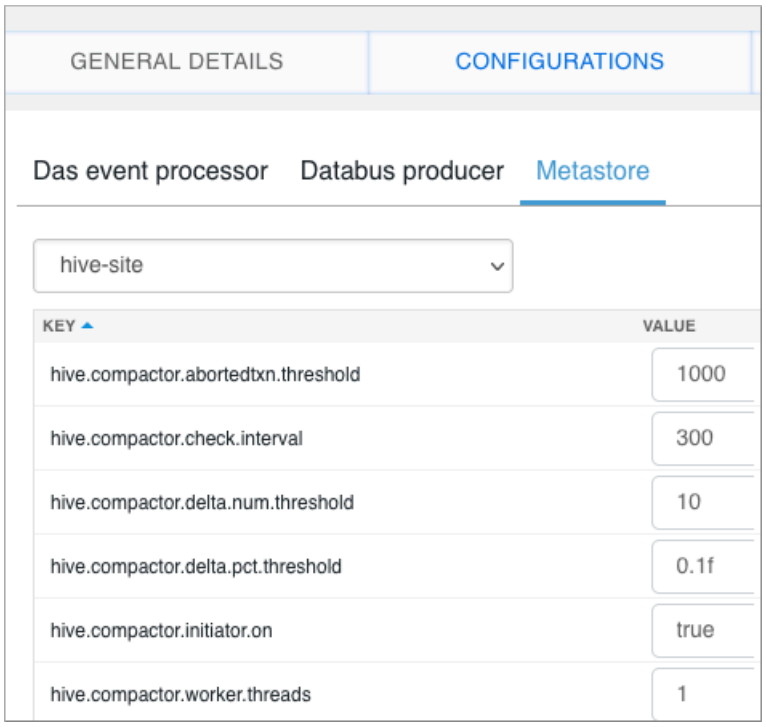
In Cloudera Data Warehouse, if you have required permissions, you can configure Hive, Hive metastore, or HiveServer properties from a Database Catalog or Virtual Warehouse. In this task, you see how to change a property and add a property from the Database Catalog, and then from a Virtual Warehouse.

Before you begin

- You have the appropriate Admin role to make the configuration change.

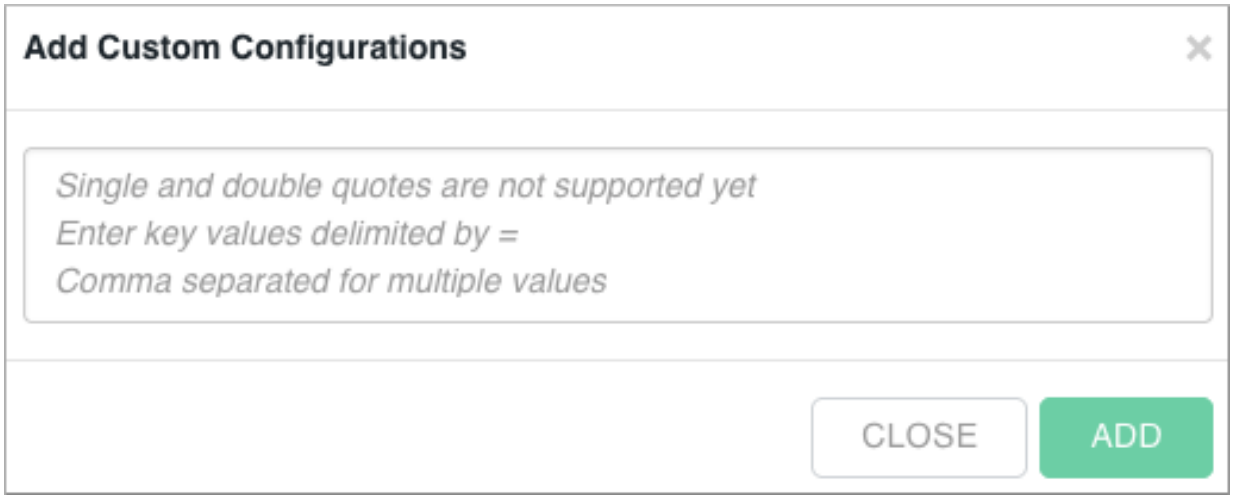
Procedure

1. Click Cloudera Data Warehouse Overview Database Catalogs Options ⋮ Edit Configurations Metastore .
2. From the drop-down, select a configuration category, for example hive-site.

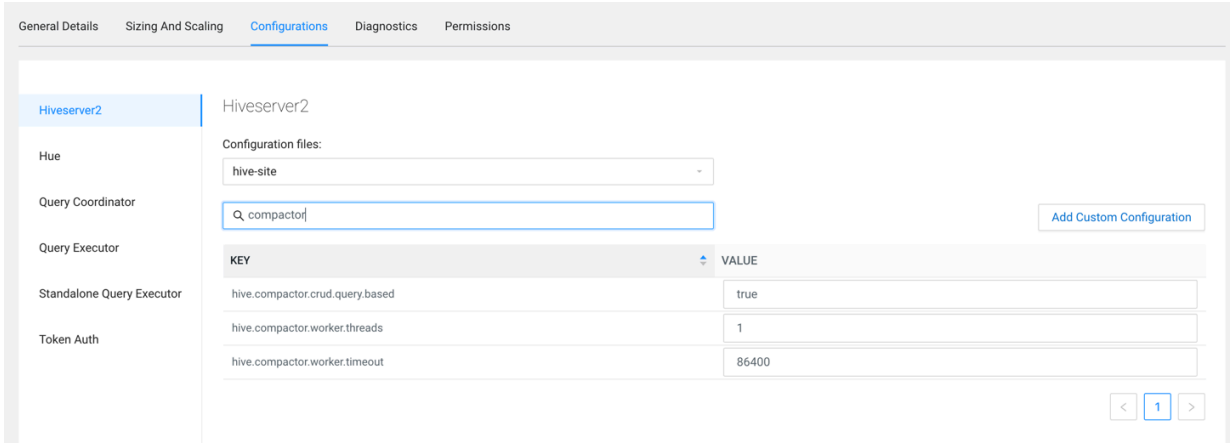


3. Change, or set, the property value.

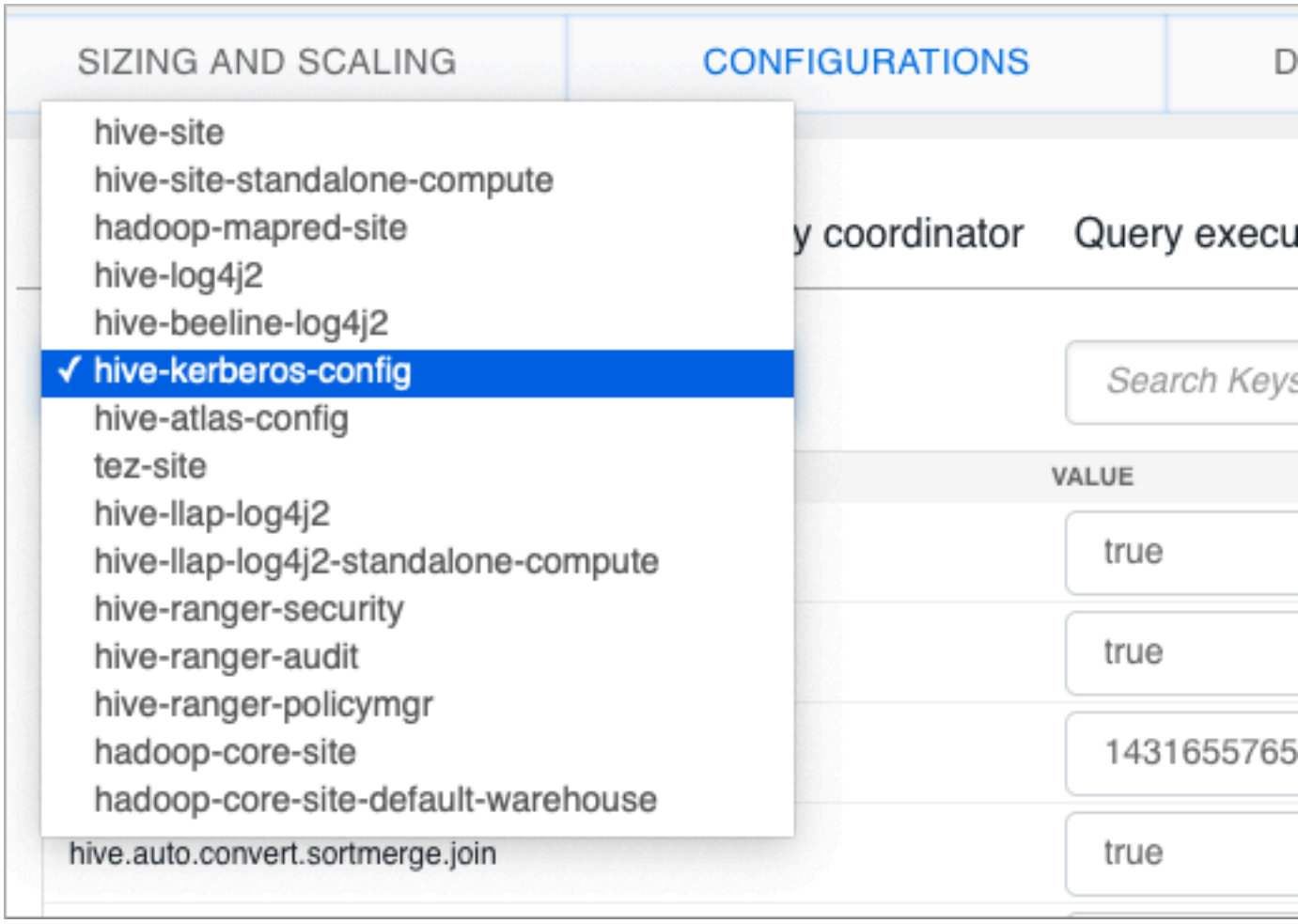
4. Click + to add a property to hive-site, for example.



5. Enter the name of the property you want to add, the equals symbol, and one or more comma-separated values. Do not use single or double quotation marks to enclose the property or value.
6. Click Add.
7. Save, and restart the Database Catalog.
8. Click Virtual Warehouses, and select a Hive Virtual Warehouse.
9. Click Actions Edit Configurations HiveServer2 .



10. Search for a specific property, or view categories of properties by selecting a category in the dropdown.



Understanding CREATE TABLE behavior

Hive table creation has changed significantly since Hive 3 to improve useability and functionality.

Hive has changed table creation in the following ways:

- Creates ACID-compliant table, which is the default in Cloudera
- Supports simple writes and inserts
- Writes to multiple partitions
- Inserts multiple data updates in a single SELECT statement
- Eliminates the need for bucketing.

If you have an ETL pipeline that creates tables in Hive, the tables will be created as ACID. Hive now tightly controls access and performs compaction periodically on the tables. Using ACID-compliant, transactional tables causes no performance or operational overload. The way you access managed Hive tables from Spark and other clients changes. In Cloudera, access to external tables requires you to set up security access permissions.

Modify the default CREATE TABLE behavior

Override default behavior when creating the table

Irrespective of the database, session, or site-level settings, you can override the default table behavior by using the MANAGED or EXTERNAL keyword in the CREATE TABLE statement.

```
CREATE [MANAGED][EXTERNAL] TABLE foo (id INT);
```

Set the default table type at a database level

You can use the database property, defaultTableType=EXTERNAL or ACID to specify the default table type to be created using the CREATE TABLE statement. You can specify this property when creating the database or at a later point using the ALTER DATABASE statement. For example:

```
CREATE DATABASE test_db WITH DBPROPERTIES ('defaultTableType'='E
XTERNAL');
```

In this example, tables created under the test_db database using the CREATE TABLE statement creates external tables with the purge fucntionality enabled (external.table.purge = 'true').

You can also choose to configure a database to allow only external tables to be created and prevent creation of ACID tables. While creating a database, you can set the database property, EXTER NAL_TABLES_ONLY=true to ensure that only external tables are created in the database. For example:

```
CREATE DATABASE test_db WITH DBPROPERTIES ('EXTERNAL_TABLES_ONLY
'='true');
```

Set the default table type at a session level

You can configure the CREATE TABLE behavior within an existing beeline session by setting hive.create.as.external.legacy to true or false. Setting the value to true results in configuring the CREATE TABLE statement to create external tables by default.

When the session ends, the default CREATE TABLE behavior also ends.

Set the default table type at a site level

You can configure the CREATE TABLE behavior at the site level by configuring the hive.create.as.insert.only and hive.create.as.acid properties in Cloudera Manager under Hive configuration. When configured at the site level, the behavior persists from session to session.

If you are a Spark user, switching to legacy behavior is unnecessary. Calling ‘create table’ from SparkSQL, for example, creates an external table after upgrading to Cloudera as it did before the upgrade. You can connect to Hive using the Hive Warehouse Connector (HWC) to read Hive ACID tables from Spark. To write ACID tables to Hive from Spark, you use the HWC and HWC API. Spark creates an external table with the purge property when you do not use the HWC API. For more information, see Hive Warehouse Connector for accessing Spark data.

Configuring legacy CREATE TABLE behavior

After you upgrade to Cloudera Base on premises and migrate old tables, the legacy CREATE TABLE behavior of Hive is no longer available by default and you might want to switch to the legacy behavior. Legacy behavior might solve compatibility problems with your scripts during data migration, for example, when running ETL.

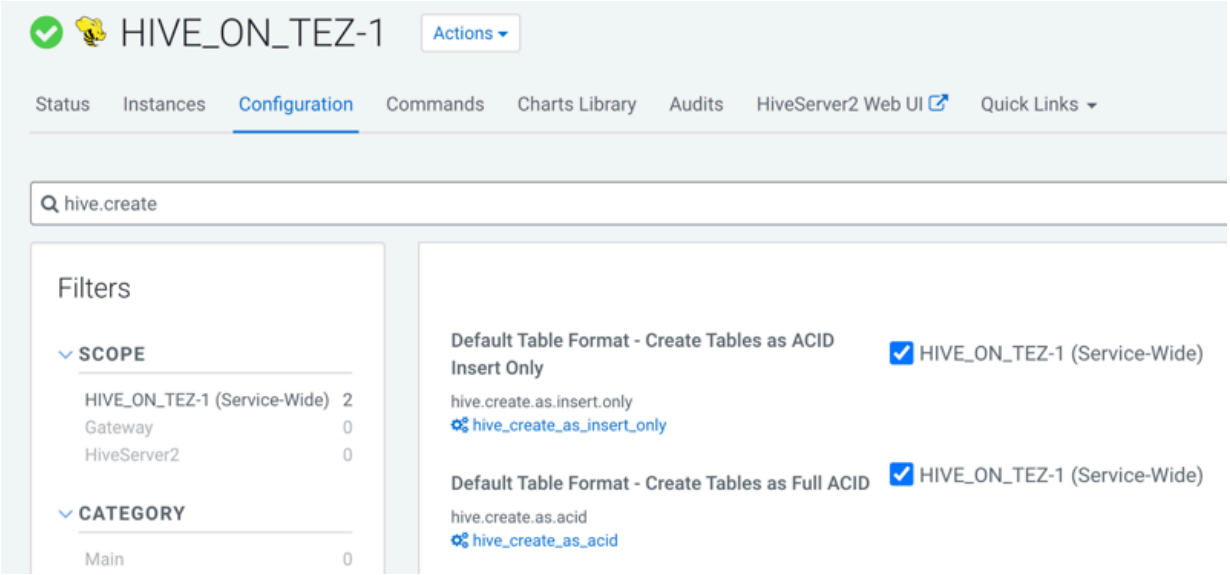
About this task

In Cloudera, running a CREATE TABLE statement by default creates a full ACID table for ORC file format and insert-only ACID table for other file formats. You can change the default behavior to use the legacy CREATE TABLE behavior. When you configure legacy behavior, CREATE TABLE creates external tables with the purge functionality enabled (external.table.purge = 'true'). Therefore, when the table is dropped, data is also deleted from the file system.

You can configure legacy CREATE TABLE behavior at the site level by configuring properties in Cloudera Manager. When configured at the site level, the behavior persists from session to session.

Procedure

1. In Cloudera Manager, click Clusters and select the Hive on Tez service.
2. From the Hive on Tez service, go to the Configuration tab and search for hive.create.



3. If the following properties are selected, clear the selection to enable legacy CREATE TABLE behavior.
   - Default Table Format - Create Tables as ACID Insert Only (hive.create.as.insert.only)
   - Default Table Format - Create Tables as Full ACID (hive.create.as.acid)

Results

Legacy behavior is enabled and the CREATE TABLE statement now creates external tables with the external.table.purge table property set to true.

Limiting 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.

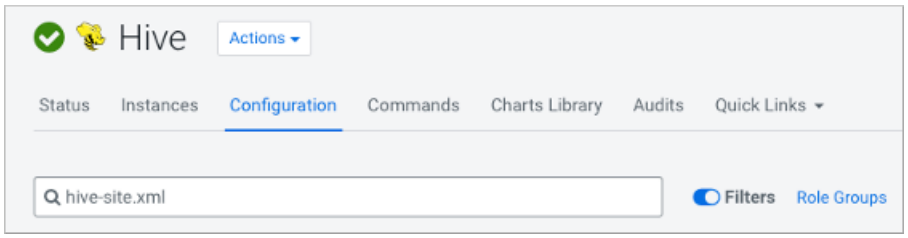
In this task, limit the number of connections per user to 25.

Before you begin

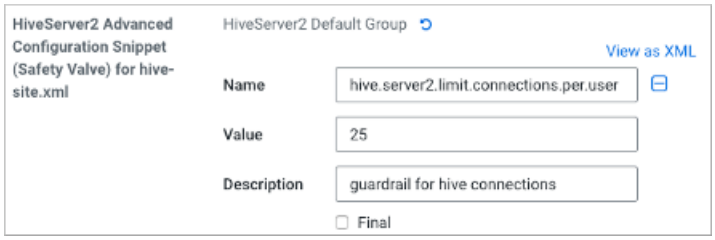
- The following components are running:
  - HiveServer
  - Hive Metastore
  - Hive client
- Minimum Required Role: Configurator (also provided by Cluster Administrator, Full Administrator)

Procedure

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



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



4. Click Save.
5. Click Actions Deploy Client Configuration .
6. Restart HIVE.

Hive on Tez configurations

Understanding key Hive on Tez properties might help you tune performance or troubleshoot problems, such as running multiple TEZ Application Master (AM) when your default sessions configuration allows running only one. After upgrading, the number of default sessions allowed might be only one. Making Hive on Tez configuration changes is recommended for users who know what they are doing.

Property and Default Value	Description	How to Check and Configure
hive.server2.tez.default.queues (default: "default")	A list of comma separated values corresponding to YARN queues for which to maintain a Tez session pool	Use the Cloudera Manager Safety Valve. When specifying additional queues, they must already exist in YARN.

Property and Default Value	Description	How to Check and Configure
hive.server2.tez.sessions.per.default.queue (default:1)	The number of Tez sessions (DAGAppMaster) to maintain in the pool per YARN queue The total number of concurrent Tez session running can be calculated with: (Tez Sessions)total = HiveServer2instances x (default.queues) x (sessions.per.default.queue) The pooled Tez Sessions are always running, even on an idle cluster.	Use the Cloudera Manager Safety Valve. A value of 1 means only one query can run at a time
hive.server2.tez.initialize.default.sessions (default: false)	If enabled, HiveServer (HS2), at startup, will launch all necessary Tez sessions within the specified default.queues to meet the sessions .per.default.queue requirements. Disable this property if you want to improve query performance and do not want to limit all the queries to the same Tez AM pool. You can enable this property and increase the sessions.per.default.queue value based on your resource availability and query concurrency/ load on the server.	Use the Cloudera Manager Safety Valve.

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 you add one or more additional HiveServer (HS2) role instances to the Hive-on-Tez service, the multiple role instances can register themselves with ZooKeeper. The JDBC client (client driver) can find a HiveServer through ZooKeeper. Using Beeline, you connect to Hive, and the ZooKeeper discovery mechnism locates and connects to one of the running HiveServer instances.

If more than one HiveServer 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 needs 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 Cloudera, are not supported on Cloudera. Use alternate solutions, such as HAProxy.

Related Information

Adding a Role Instance

Generating Hive statistics in Cloudera Data Warehouse

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.

Generating and viewing Apache Hive statistics in Cloudera Data Warehouse

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 or editor.
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 in Cloudera Data Warehouse

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

Custom Configuration (about Cloudera Manager Safety Valve)

Configuring query audit logs to include caller context

Learn how you can configure Hive to audit partitions that are scanned as part of a Hive query. The audit information comprises the Query ID and User ID, which helps in meeting compliance requirements, such as controlling user access to data from specific regions or access only to particular time periods.

About this task

The audit information that is logged is stored in the HDFS audit log entry. You can access the audit logs from either Cloudera Manager or through the command line:

- From Cloudera Manager

  Access the log file from Clusters > HDFS-1 > NameNode Web UI > Utilities > logs > hdfs-audit.log
- From the command line:

  Access the log file from /var/log/hadoop-hdfs/hdfs-auidt.log

Procedure

1. Log in to Cloudera Manager as an administrator and go to Clusters HDFS-1 .
2. In the HDFS-1 service page, click the Configuration tab and search for the "hadoop.caller.context.enabled" property.
3. Enable the property to allow the caller context to be included in the audit logs.

   This allows additional fields to be written into HDFS NameNode audit log records for auditing coarse granularity operations.

4. Click Save Changes and restart the HDFS-1 service.
5. After the service restarts, run your Hive queries.

What to do next
You can view the audit information of the queries by accessing the hdfs-audit.log file.

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, 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, 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. InName enter the property name and in value enter the value.