

Cloudera Runtime 7.2.11

Configuring Apache Hive

Date published: 2019-08-21

Date modified: 2022-08-10

CLOUDERA

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

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

Configuring Hive in Cloudera Data Warehouse.....	4
Understanding CREATE TABLE behavior.....	6
Generating Hive statistics in Cloudera Data Warehouse.....	7
Generating and viewing Apache Hive statistics in Cloudera Data Warehouse.....	8
Statistics generation and viewing commands in Cloudera Data Warehouse.....	8

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.

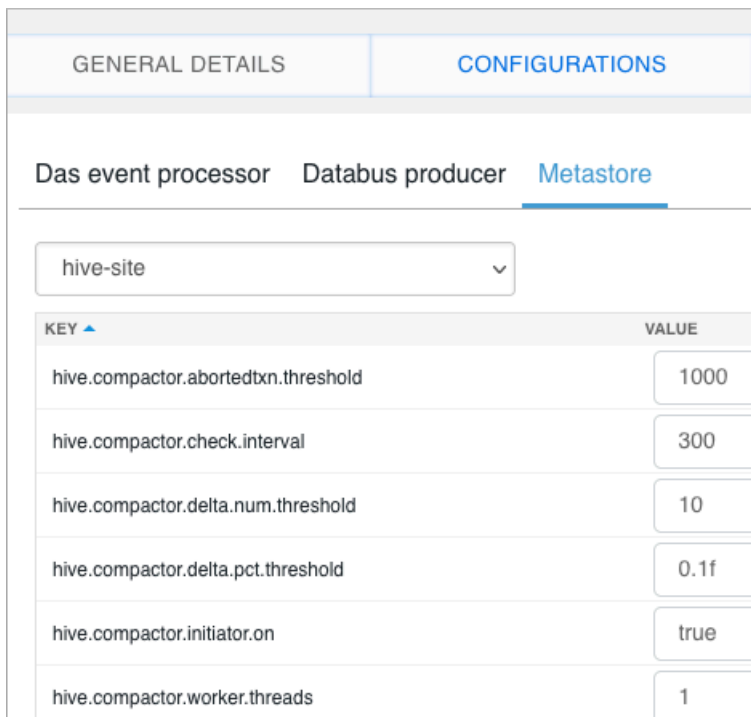
In Cloudera Data Warehouse, if you have required permissions, you can configure Hive, Hive metastore, or HiveServer properties from a Database Catalog or Virtual Warehouse. In this task, you see how to change a property and add a property from the Database Catalog, and then from a Virtual Warehouse.

Before you begin

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

Procedure

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



KEY	VALUE
hive.compactor.abortedtxn.threshold	1000
hive.compactor.check.interval	300
hive.compactor.delta.num.threshold	10
hive.compactor.delta.pct.threshold	0.1f
hive.compactor.initiator.on	true
hive.compactor.worker.threads	1

- Change, or set, the property value.

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

Add Custom Configurations ×

Single and double quotes are not supported yet
Enter key values delimited by =
Comma separated for multiple values

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 .

SIZING AND SCALING **CONFIGURATIONS**

Das webapp **Hiveserver2** Hue Query coordinator

hadoop-mapred-site ▼

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

The screenshot shows a configuration management interface. A dropdown menu is open under the 'CONFIGURATIONS' tab, listing various properties. The property 'hive-kerberos-config' is selected and highlighted in blue. The background shows a table with columns for 'KEY' and 'VALUE'. The table contains the following entries:

KEY	VALUE
hive-site	
hive-site-standalone-compute	
hadoop-mapred-site	
hive-log4j2	
hive-beeline-log4j2	
✓ hive-kerberos-config	
hive-atlas-config	
tez-site	
hive-llap-log4j2	
hive-llap-log4j2-standalone-compute	
hive-ranger-security	
hive-ranger-audit	
hive-ranger-policymgr	
hadoop-core-site	
hadoop-core-site-default-warehouse	
hive.auto.convert.sortmerge.join	true

Understanding CREATE TABLE behavior

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

Hive has changed table creation in the following ways:

- Creates ACID-compliant table, which is the default in
- 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 , 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'='EXTERNAL');
```

In this example, tables created under the `test_db` database using the `CREATE TABLE` statement creates external tables with the purge functionality 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, `EXTERNAL_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 `hive-site.xml` 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 `as` 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](#).

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 \(about Cloudera Manager Safety Valve\)](#)