Cloudera Runtime 7.3.3

Apache Hive Materialized View Commands

Date published: 2019-08-21
Date modified: 2026-11-13



https://docs.cloudera.com/

Legal Notice

© Cloudera Inc. 2026. 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

ALTER MATERIALIZED VIEW REBUILD....4
ALTER MATERIALIZED VIEW REWRITE....4
CREATE MATERIALIZED VIEW....5
DESCRIBE EXTENDED and DESCRIBE FORMATTED....6
DROP MATERIALIZED VIEW....8
SHOW MATERIALIZED VIEWS....8

ALTER MATERIALIZED VIEW REBUILD

You must rebuild the materialized view to keep it up-to-date when changes to the data occur.

Syntax

```
ALTER MATERIALIZED VIEW [db_name.]materialized_view_name REBUILD;
```

DB_NAME.MATERIALIZED_VIEW_NAME

The database name followed by the name of the materialized view in dot notation.

Description

A rewrite of a query based on a stale materialized view does not occur automatically. If you want a rewrite of a stale or possibly stale materialized view, you can force a rewrite. For example, you might want to use the contents of a materialized view of a non-transactional table because the freshness of such a table is unknown. To enable rewriting of a query based on a stale materialized view, you can run the rebuild operation periodically and set the following property: hive.materializedview.rewriting.time.window. For example, SET hive.materializedview.rewriting.time.window=10min;

Example

ALTER MATERIALIZED VIEW mydb.mv1 REBUILD;

Related Information

Using materialized views

Periodically rebuild a materialized view

ALTER MATERIALIZED VIEW REWRITE

You can enable or disable the rewriting of queries based on a particular materialized view.

Syntax

```
ALTER MATERIALIZED VIEW [db_name.]materialized_view_name ENABLE|DISABLE REWR
ITE;
```

DB_NAME.MATERIALIZED_VIEW_NAME

The database name followed by the name for the materialized view in dot notation.

Description

To optimize performance, by default, Hive rewrites a query based on materialized views. You can change this behavior to manage query planning and execution manually. By setting the hive.materializedview.rewriting global property, you can manage query rewriting based on materialized views for all queries.

Example

ALTER MATERIALIZED VIEW mydb.mv1 DISABLE REWRITE;

Related Information

Using materialized views

CREATE MATERIALIZED VIEW

If you are familiar with the CREATE TABLE AS SELECT (CTAS) statement, you can quickly master how to create a materialized view.

Syntax

```
CREATE MATERIALIZED VIEW [IF NOT EXISTS] [db_name.]materialized_view_name
   [DISABLE REWRITE]
   [COMMENT materialized_view_comment]
   [PARTITIONED ON (column_name, ...)]
   [
     [ROW FORMAT row_format]
     [STORED AS file_format]
        | STORED BY 'storage.handler.class.name' [WITH SERDEPROPERTIES (ser
de_property_name=serde_property_value, ...)]
   ]
   [LOCATION file_path]
   [TBLPROPERTIES (tbl_property_name=tbl_property_value, ...)]
   AS
   <query>;
```

Required Parameters

QUERY

The query to run for results that populate the contents of the materialized view.

Optional Parameters

DB_NAME.MATERIALIZED_VIEW_NAME

The database name followed by a name, unique among materialized view names, for the materialized view. The name must be a valid a table name, including case-insensitive alphanumeric and underscore characters.

MATERIALIZED_VIEW_COMMENT

A string literal enclosed in single quotation marks.

COLUMN_NAME

A key that determines how to do the partitioning, which divides the view of the table into parts.

'STORAGE.HANDLER.CLASS.NAME'

The name of a storage handler, such as org.apache.hadoop.hive.druid.DruidStorageHandler, that conforms to the Apache Hive specifications for storage handlers in a table definition that uses the STORED BY clause. The default is hive.materializedview.fileformat.

SERDE_PROPERTY_NAME

A property supported by SERDEPROPERTIES that you specify as part of the STORED BY clause. The property is passed to the serde provided by the storage handler. When not specified, Hive uses the default hive.materializedview.serde.

SERDE_PROPERTY_VALUE

A value of the SERDEPROPERTIES property.

FILE_PATH

The location on the file system for storing the materialized view.

TBL_PROPERTY_NAME

A key that conforms to the Apache Hive specification for TBLPROPERTIES keys in a table.

TBL_PROPERTY_VALUE

The value of a TBLPROPERTIES key.

Usage

The materialized view creation statement meets the criteria of being atomic: it does not return incomplete results. By default, the optimizer uses materialized views to rewrite the query. You can store a materialized view in an external storage system using the STORED AS clause followed by a valid storage handler class name. You can set the DISABLE REWRITE option to alter automatic rewriting of the query at materialized view creation time. The table on which you base the materialized view, src in the example below, must be an ACID, managed table.

Examples

```
CREATE MATERIALIZED VIEW druid_t
  STORED BY 'org.apache.hadoop.hive.druid.DruidStorageHandler'
  AS SELECT a, b, c
  FROM src;
```

```
CREATE MATERIALIZED VIEW mv4
  LOCATION '/user/csso_max'
  AS SELECT empid, deptname, hire_date
  FROM emps JOIN depts
  ON (emps.deptno = depts.deptno)
  WHERE hire_date >= '2017-01-01';
```



Note: Renaming materialized view in Hive is currently not supported.

Related Information

Apache Hive Wiki Hive Data Definition Language > Create Table and CTAS

Apache Hive Wiki StorageHandlers > DDL

Using materialized views

DESCRIBE EXTENDED and DESCRIBE FORMATTED

You can get extensive formatted and unformatted information about a materialized view.

Syntax

```
DESCRIBE [EXTENDED | FORMATTED] [db_name.]materialized_view_name;
```

DB_NAME

The database name.

MATERIALIZED_VIEW_NAME

The name of the materialized view.

Examples

Get summary, details, and formatted information about the materialized view in the default database and its partitions.

```
DESCRIBE FORMATTED default.partition_mv_1;
```

Example output is:

col_name	data_type	comment
# col_name	data_type	comment
name	varchar(256)	
	NULL	NULL
# Partition Information	NULL	NULL
# col_name	data_type	comment
deptno	int	
	NULL	NULL
# Detailed Table Information	NULL	NULL
Database:	default	NULL
OwnerType:	USER	NULL
Owner:	hive	NULL
CreateTime:	Wed Aug 24 19:46:08 UTC 2022	NULL
LastAccessTime:	UNKNOWN	NULL
Retention:	0	NULL
Location:	hdfs://myserver:8020/warehouse/ tables pace/managed/hive/partition_mv_1	NULL
Table Type:	MATERIALIZED_VIEW	NULL
Table Parameters:	NULL	NULL
	COLUMN_STATS_ACCURATE	{\"BASIC_STATS\":\"true\"}
	bucketing_version	2
	numFiles	2
	numPartitions	2
	numRows	4
	rawDataSize	380
	totalSize	585
	transient_lastDdlTime	1534967168
	NULL	NULL
# Storage Information	NULL	NULL
SerDe Library:	org.apache.hadoop.hive.ql.io.orc.OrcSerde	NULL
InputFormat:	org.apache.hadoop.hive.ql.io.orc.OrcInputFor mat	NULL
OutputFormat:	org.apache.hadoop.hive.ql.io.orc.OrcOutputFo rmat	NULL
Compressed:	No	NULL
Num Buckets:	-1	NULL
Bucket Columns:	[]	NULL
Sort Columns:	[]	NULL
	NULL	NULL
# Materialized View Information	NULL	NULL
Original Query:	SELECT hire_date, deptno FROM emps W HERE deptno > 100 AND deptno < 200	NULL

col_name	data_type	comment
Expanded Query:	SELECT `hire_date`, `deptno` FROM (S ELECT `emps`.`hire_date`, `emps`.`dept no` FROM `default`.`emps` WHERE `e mps`.`deptno` > 100 AND `emps`.`deptno ` < 200) `default.partition_mv_1`	NULL
Rewrite Enabled:	Yes	NULL
Outdated for Rewriting:	No	NULL

Related Information

Using materialized views

DROP MATERIALIZED VIEW

You can avoid making a table name unusable by dropping a dependent materialized view before dropping a table.

Syntax

```
DROP MATERIALIZED VIEW [db_name.]materialized_view_name;
```

DB_NAME.MATERIALIZED_VIEW_NAME

The database name followed by a name for the materialized view in dot notation.

Description

Dropping a table that is used by a materialized view is not allowed and prevents you from creating another table of the same name. You must drop the materialized view before dropping the tables.

Example

DROP MATERIALIZED VIEW mydb.mv1;

Related Information

Using materialized views

SHOW MATERIALIZED VIEWS

You can list all materialized views in the current database or in another database. You can filter a list of materialized views in a specified database using regular expression wildcards.

Syntax

```
SHOW MATERIALIZED VIEWS [IN db_name];
```

DB_NAME

The database name.

'IDENTIFIER_WITH_WILDCARDS'

The name of the materialized view or a regular expression consisting of part of the name plus wildcards. The asterisk and pipe (* and |) wildcards are supported. Use single quotation marks to enclose the identifier.

Examples

SHOW MATERIALIZED VIEWS;

```
+-----------------+--------------------+-----------------+
|     mv_name     |  rewrite_enabled   |      mode       |
+-----------------+--------------------+-----------------+
| # MV Name       | Rewriting Enabled  | Mode            |
| partition_mv_1  | Yes                | Manual refresh  |
| partition_mv_2  | Yes                | Manual refresh  |
| partition_mv_3  | Yes                | Manual refresh  |
+-----------------+--------------------+-----------------+
```

SHOW MATERIALIZED VIEWS '*1';

```
+-----------------+--------------------+-----------------+
|     mv_name     |  rewrite_enabled   |      mode       |
+-----------------+--------------------+-----------------+
| # MV Name       | Rewriting Enabled  | Mode            |
| partition_mv_1  | Yes                | Manual refresh  |
|                 | NULL               | NULL            |
+-----------------+--------------------+-----------------+
```

Related Information

Using materialized views