Data Access 3

Materialized view commands

Date of Publish: 2019-12-17

https://docs.hortonworks.com

Contents

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 for the materialized view in dot notation.

Description

Hive performs view maintenance incrementally if possible, refreshing the view to reflect any data inserted into ACID tables. The rebuild operation preserves the low-latency analytical processing (LLAP) cache for existing data in the materialized view. Hive does a full rebuild if an incremental one is impossible.

Hive does not rewrite a query based on a stale materialized view 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 Hive cannot determine the freshness of such a table. 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

ALTER MATERIALIZED VIEW REWRITE

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

Syntax

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

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 use the command 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
 (serde_property_name=serde_property_value, ...)]
   ]
   [LOCATION hdfs_path]
   [TBLPROPERTIES (tbl_property_name=tbl_property_value, ...)]
   AS
   <query>;
```

db_name.materialized_view_name	The database name followed by a name, unique among materialized view names, for the materialized view in dot notation. The name must conform to Apache Hive specifications for 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. When not specified, Hive uses the default hive.materializedview.fileformat.
serde_property_name	A property supported by SERDEPROPERTIES that you specify as part of the STORED BY clause and passed to the serde provided by the storage handler. When not specifed, Hive uses the default hive.materializedview.serde.
serde_property_value	A value of the SERDEPROPERTIES property.

hdfs_path	The location on the HDFS file system for storing the materialized view.
tbl_property_name	A key that conforms to the Apache Hive specification to TBLPROPERTIES keys in a table.
tbl_property_value	The value of a TBLPROPERTIES key.
query	The query to execute for results that populate the contents of the materialized view

Description

The materialized view creation statement is atomic (not visible unitl all results are populated). 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.

Example

CREATE MATERIALIZED VIEW druid_tSTORED AS
'org.apache.hadoop.hive.druid.DruidStorageHandler'ASSELECT a, b, cFROM src;

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
# 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 22 19:46:08 UTC 2018	NULL
LastAccessTime:	UNKNOWN	NULL
Retention:	0	NULL
Location:	hdfs://myserver:8020/warehouse/ tablespace/ 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.OrcInputFormat	NULL
OutputFormat:	org.apache.hadoop.hive.ql.io.orc.OrcOutputFormat	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 WHERE deptno > 100 AND deptno < 200	NULL
Expanded Query:	SELECT `hire_date`, `deptno` FROM (SELECT `emps`.`hire_date`, `emps`.`deptno` FROM `default`.`emps` WHERE `emps`.`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