基于Hadoop的数据仓库Hive

Northeastern University

HORIZON 昊宸科技

目录

HiveQL-Database

Create Database

```
CREATE (DATABASE|SCHEMA) [IF NOT EXISTS] database_name
  [COMMENT database_comment]
  [LOCATION hdfs_path]
  [WITH DBPROPERTIES (property_name=property_value, ...)];
```

Drop Database

```
DROP (DATABASE|SCHEMA) [IF EXISTS] database_name [RESTRICT|CASCADE];
```

Alter Database

```
ALTER (DATABASE|SCHEMA) database_name SET DBPROPERTIES (property_name=property_value, ...);   -- (Note: SCHEMA added in Hive 0

ALTER (DATABASE|SCHEMA) database_name SET OWNER [USER|ROLE] user_or_role;   -- (Note: Hive 0.13.0 and later; SCHEMA added in H
```

Use Database

```
USE database_name;
USE DEFAULT;
```

HiveQL-创建表

Create Table

```
CREATE [TEMPORARY] [EXTERNAL] TABLE [IF NOT EXISTS] [db_name.]table_name    -- (Note: TEMPORARY available in Hive 0.14.0 and later)
  [(col_name data_type [COMMENT col_comment], ... [constraint_specification])]
  [COMMENT table_comment]
  [PARTITIONED BY (col_name data_type [COMMENT col_comment], ...)]
  [CLUSTERED BY (col_name, col_name, ...) [SORTED BY (col_name [ASC|DESC], ...)] INTO num_buckets BUCKETS]
  [SKEWED BY (col_name, col_name, ...)                  -- (Note: Available in Hive 0.10.0 and later)]
     ON ((col_value, col_value, ...), (col_value, col_value, ...), ...)
     [STORED AS DIRECTORIES]
  [
   [ROW FORMAT row_format]
   [STORED AS file_format]
     | STORED BY 'storage.handler.class.name' [WITH SERDEPROPERTIES (...)]  -- (Note: Available in Hive 0.6.0 and later)
  ]
  [LOCATION hdfs_path]
  [TBLPROPERTIES (property_name=property_value, ...)]   -- (Note: Available in Hive 0.6.0 and later)
  [AS select_statement];   -- (Note: Available in Hive 0.5.0 and later; not supported for external tables)
```

```
CREATE [TEMPORARY] [EXTERNAL] TABLE [IF NOT EXISTS] [db_name.]table_name
  LIKE existing_table_or_view_name
  [LOCATION hdfs_path];
```

HiveQL-创建表

[EXTERNAL]

说明：可以让用户创建一个外部表，在建表的同时指定一个指向实际数据的路径（LOCATION），Hive 创建内部表时，会将数据移动到数据仓库指向的路径；若创建外部表，仅记录数据所在的路径，不对数据的位置做任何改变。在删除表的时候，内部表的元数据和数据会被一起删除，而外部表只删除元数据，不删除数据。配合LOCATION HDFS_PATH使用。

[IF NOT EXISTS]

说明：创建一个指定名字的表。如果相同名字的表已经存在，则抛出异常；用户可以用 IF NOT EXIST 选项来忽略这个异常。

[PARTITIONED BY]

用法：[PARTITIONED BY (col_name data_type [COMMENT col_comment], ...)]

说明：一个表可以拥有一个或者多个分区，每一个分区单独存在一个目录下。而且，表和分区都可以对某个列进行 CLUSTERED BY 操作，将若干个列放入一个桶（bucket） 中。

Example:PARTITIONED BY (`windcode` string comment '风场ID', `fancode` string comment '风机编号')

[CLUSTERED BY]

用法：[CLUSTERED BY (列1, 列2, ...) [SORTED BY (列名 [ASC|DESC], ...)] INTO 桶数量 BUCKETS]

说明：桶，把表或者分区组成成桶（bucket）有两个理由，第一个理由是获得更高的查询效率。例如map端连接。第二个理由是使“取样”更高效。物理上，每个桶就是表或分区中的一个文件。桶的每个文件对应于MapReduce的输出分区。

HiveQL-创建表

[ROW FORMAT]

用法：ROW FORMAT DELIMITED

[FIELDS TERMINATED BY char [ESCAPED BY char]]

[COLLECTION ITEMS TERMINATED BY char]

[MAP KEYS TERMINATED BY char]

[LINES TERMINATED BY char]

[NULL DEFINED AS char]

|SERDE serde_name [WITH SERDEPROPERTIES (property_name=property_value, property_name=property_value, ...)]

说明：是用来设置创建的表在加载数据的时候，支持的列分隔符。

[COLLECTION ITEMS TERMINATED BY char]:参照Hive复杂数据类型使用文档。

[MAP KEYS TERMINATED BY char]:参照Hive复杂数据类型使用文档。

[NULL DEFINED AS char]:用于指定hive数据NULL数据表示方式，模式是\N。一般指定使用空字符串。**参照Hive空值处理。**

SERDE serde_name [WITH SERDEPROPERTIES (propertiy_name=property_value)]:SERDE是序列化和反序列化的缩写，SERDE表示行格式是如何存储的。HIVE从来两个维度对表的存储进行管理，“行格式”和“文件格式”。ROW FORMAT这里指的便是行格式。

当作为反序列化工具使用时，也就是查询表时，SERDE将把文件中的自己而形式数据行反序列化为Hive内部操作数据行时所使用的对象形式。当作为序列化工具使用 (INSERT或者CTAS)时，SERDE会把Hive的数据行内部表示形式序列化成字节形式并写入到输出文件中。**Hive行格式详解。**

文件格式我们会在后面的[STORED AS format]语句中介绍。

HiveQL-创建表

[STORED AS]

用法： [STORED AS file_format]

说明：用于指定hive存储的文件格式。一般是结合[ROW FORMAT]使用。

Hive支持的文件格式有：

SEQUENCEFILE

TEXTFILE

RCFILE

ORC

PARQUET

AVRO

INPUTFORMAT

参照： https://cwiki.apache.org/confluence/display/Create/Drop/Alter/UseDatabase

Example:

CREATE TABLE my_table(a string, b bigint, ...)

ROW FORMAT SERDE 'org.apache.hive.hcatalog.data.

STORED AS TEXTFILE;

```
CREATE TABLE apachelog (
  host STRING,
  identity STRING,
  user STRING,
  time STRING,
  request STRING,
  status STRING,
  size STRING,
  referer STRING,
  agent STRING)
ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.RegexSerDe'
WITH SERDEPROPERTIES (
  "input.regex" = "([^]*) ([^]*) ([^]*) (-|\\[^\\]*\\]) ([^ \"]*|\"[^\"]*\") (-|[0-9]*) (-|[0-
9]*)(?: ([^ \"]*|\".*\") ([^ \"]*|\".*\"))?"
)
```

HiveQL-创建表

[CTAS]

用法： [CREATE TABLE ...AS SELECT ...]

说明：借助以已存在的表，生成一张新表。

Example:

create table person2 as select * from person;

```
hive> create table person2 as select * from person;
Query ID = hive_20170318151621_5b27a0f3-fae6-4e8f-bdd5-2664b1e42874
Total jobs = 1
Launching Job 1 out of 1
Tez session was closed. Reopening...
Session re-established.


Status: Running (Executing on YARN cluster with App id application_1489626712521_1653)

--------------------------------------------------------------------------------
        VERTICES      STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 ..........   SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 01/01  [==========================>>] 100%  ELAPSED TIME: 6.94 s
--------------------------------------------------------------------------------
Moving data to directory hdfs://idh104:8020/apps/hive/warehouse/hivetest.db/person2
Table hivetest.person2 stats: [numFiles=1, numRows=3, totalSize=101, rawDataSize=98]
OK
Time taken: 17.017 seconds
```

HiveQL-创建表

[CREATE TABLE LIKE]

用法： [CREATE TABLE … LIKE …]

说明：借助以已存在的表，创建一张空表

Example:

create table person3 like person;

[TEMPORARY]

用法：CREATE TEMPORARY TABLE table_name…

说明：创建临时表，临时表只对当前session生效，session退出后表自动删除。

注意点：

如果创建的临时表表名已存在，那么当前session引用到该表名时实际用的是临时表，只有drop或rename临时表名才能使用原始表。

不支持分区字段和创建索引

HiveQL-删除表

[DROP TABLE]

用法： DROP TABLE [IF EXISTS] table_name [PURGE];

说明：删除表

Example:

drop table person3;

[TRUNCATE TABLE]

用法： TRUNCATE TABLE table_name [PARTITION partition_spec];

说明：截断表

Example:

truncate table person2;

HiveQL-修改表

➢ 重命名表

ALTER TABLE table_name RENAME TO new_table_name;

重命名表，除了更新元数据外，会将表目录移动到新名称所对应的目录下。但对于外部表，只更新元数据。

➢ 修改表属性

ALTER TABLE table_name SET TBLPROPERTIES table_properties;

✓ 修改表注释：

ALTER TABLE table_name SET TBLPROPERTIES ('comment' = new_comment);

✓ 修改表serde的列分隔符

ALTER TABLE table_name SET SERDEPROPERTIES ('field.delim' = ',');

✓ 修改表sede

ALTER TABLE table_name [PARTITION partition_spec] SET SERDE serde_class_name [WITH SERDEPROPERTIES serde_properties];

ALTER TABLE table_name [PARTITION partition_spec] SET SERDEPROPERTIES serde_properties;

HiveQL-修改表

- 修改表分区
  - ✓ 新增分区：

```
ALTER TABLE table_name ADD [IF NOT EXISTS] PARTITION partition_spec
  [LOCATION 'location1'] partition_spec [LOCATION 'location2'] ...;

partition_spec:
  : (partition_column = partition_col_value, partition_column = partition_col_value, ...)
```

  - ✓ 重命名分区：

```
ALTER TABLE table_name PARTITION partition_spec RENAME TO PARTITION partition_spec;
```

  - ✓ 自动恢复分区：

```
MSCK REPAIR TABLE table_name;
```

  - ✓ 删除分区

```
ALTER TABLE table_name DROP [IF EXISTS] PARTITION partition_spec[, PARTITION partition_spec, ...]
  [IGNORE PROTECTION] [PURGE];           -- (Note: PURGE available in Hive 1.2.0 and later, IGNORE PROTECTION not available 2
```

HiveQL-修改表

修改列名称/类型/位置/注释

```
ALTER TABLE table_name [PARTITION partition_spec] CHANGE [COLUMN] col_old_name col_new_name column_type
  [COMMENT col_comment] [FIRST|AFTER column_name] [CASCADE|RESTRICT];
```

添加或替换列

```
ALTER TABLE table_name
  [PARTITION partition_spec]                 -- (Note: Hive 0.14.0 and later)
  ADD|REPLACE COLUMNS (col_name data_type [COMMENT col_comment], ...)
  [CASCADE|RESTRICT]                         -- (Note: Hive 1.1.0 and later)
```

HiveQL-视图

和其他数据库一样，hive从0.6版本之后也支持了视图功能。但是hive中的视图和关系数据库中的视图是有区别的。

- ✓ **只有逻辑视图，没有物化视图。**
- ✓ 视图只能查询，不能Load/Insert/Update/Delete数据；
- ✓ 视图在创建时候，只是保存了一份元数据，当查询视图的时候，才开始执行视图对应的那些子查询；

```
CREATE VIEW [IF NOT EXISTS] [db_name.]view_name [(column_name [COMMENT column_comment], ...) ]
  [COMMENT view_comment]
  [TBLPROPERTIES (property_name = property_value, ...)]
  AS SELECT ...;
```

```
DROP VIEW [IF EXISTS] [db_name.]view_name;
```

```
ALTER VIEW [db_name.]view_name SET TBLPROPERTIES table_properties;

table_properties:
  : (property_name = property_value, property_name = property_value, ...)
```

HiveQL-索引

创建索引

```
CREATE INDEX index_name
  ON TABLE base_table_name (col_name, ...)
  AS index_type
  [WITH DEFERRED REBUILD]
  [IDXPROPERTIES (property_name=property_value, ...)]
  [IN TABLE index_table_name]
  [
     [ ROW FORMAT ...] STORED AS ...
     | STORED BY ...
  ]
  [LOCATION hdfs_path]
  [TBLPROPERTIES (...)]
  [COMMENT "index comment"];
```

Hint: double-click to select code

删除索引

```
DROP INDEX [IF EXISTS] index_name ON table_name;
```

修改索引

```
ALTER INDEX index_name ON table_name [PARTITION partition_spec] REBUILD;
```

HiveQL-SHOW

列出所有的库

```
SHOW (DATABASES|SCHEMAS) [LIKE 'identifier_with_wildcards'];
```

列出库下的所有表

```
SHOW TABLES [IN database_name] ['identifier_with_wildcards'];
```

列出所有视图

```
SHOW VIEWS [IN/FROM database_name] [LIKE 'pattern_with_wildcards'];
```

列出表分区

```
SHOW PARTITIONS table_name;
```

```
SHOW PARTITIONS [db_name.]table_name [PARTITION(partition_spec)];   -- (Note: Hive 0.13.0 and later)
```

显示表/视图结构

```
SHOW CREATE TABLE ([db_name.]table_name|view_name);
```

HiveQL-SHOW

显示索引信息

```
SHOW [FORMATTED] (INDEX|INDEXES) ON table_with_index [(FROM|IN) db_name];
```

显示表的所有列信息

```
SHOW COLUMNS (FROM|IN) table_name [(FROM|IN) db_name];
```

显示Function信息

```
SHOW FUNCTIONS "a.*";
```

显示锁表信息

```
SHOW LOCKS <table_name>;
SHOW LOCKS <table_name> EXTENDED;
SHOW LOCKS <table_name> PARTITION (<partition_spec>);
SHOW LOCKS <table_name> PARTITION (<partition_spec>) EXTENDED;
SHOW LOCKS (DATABASE|SCHEMA) database_name;     -- (Note: Hive 0.13.0 and later;
SCHEMA added in Hive 0.14.0)
```

Hive用户自定义函数

Hive中有三种UDF。

- UDF(user-defined function)

操作作用于单个数据行，且产生了一个数据行作为输出。大多数函数(数学函数和字符串函数)都属于这一类。

- UDAF(user-defined aggregate function)

用户定义聚集函数，接受多个输入数据行，并产生一个输出数据行。像COUNT和MAX这样的函数都是聚集函数。

- UDTF(user-defined table-generating function)

用户定义表生成函数，操作作用于单个数据行，且产生多个数据行。

参照：

https://cwiki.apache.org/confluence/display/Hive/HivePlugins
https://cwiki.apache.org/confluence/display/Hive/LanguageManual+DDL#LanguageManualDDL-CreateFunction
https://cwiki.apache.org/confluence/display/Hive/GenericUDAFCaseStudy

作业

~~1.调研ORCFile、RCFile、PARQUETFILE、SEQUENCEFILE、PARQUETFILE这几中存储格式，总结它们的特点，优缺点。~~
2.比较HQL中的order by;sort by;cluster by;distribute by的作用，并分别尝试使用它们。
3.一下二选一。
实现一个UDF去查找hive表中array类型列的值中是否包含某一项。
SELECT FIND_IN_ARRAY(列名, 搜索字符) FROM users;
实现一个UDF去链接Array字段的值。
SELECT ARRAY_CONTACT(列名, 搜索字符) FROM users;
4.学习UDAF，并尝试使用UDAF去实现TOPK算法。

THANKS