Hive 语法介绍

## 一、建表

CREATE [EXTERNAL] TABLE [IF NOT EXISTS] table\_name   
  [(col\_name data\_type [COMMENT col\_comment], ...)]   
  [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]   
  [ROW FORMAT row\_format]   
  [STORED AS file\_format]   
  [LOCATION hdfs\_path]

•CREATE TABLE 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

•EXTERNAL 关键字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）

•LIKE 允许用户复制现有的表结构，但是不复制数据

•COMMENT可以为表与字段增加描述

•ROW FORMAT

    DELIMITED [FIELDS TERMINATED BY char] [COLLECTION ITEMS TERMINATED BY char]

        [MAP KEYS TERMINATED BY char] [LINES TERMINATED BY char]

   | SERDE serde\_name [WITH SERDEPROPERTIES (property\_name=property\_value, property\_name=property\_value, ...)]

  用户在建表的时候可以自定义 SerDe 或者使用自带的 SerDe。如果没有指定 ROW FORMAT 或者 ROW FORMAT DELIMITED，将会使用自带的 SerDe。在建表的时候，用户还需要为表指定列，用户在指定表的列的同时也会指定自定义的 SerDe，Hive 通过 SerDe 确定表的具体的列的数据。

•STORED AS

            SEQUENCEFILE

            | TEXTFILE

            | RCFILE

            | INPUTFORMAT input\_format\_classname OUTPUTFORMAT             output\_format\_classname

       如果文件数据是纯文本，可以使用 STORED AS TEXTFILE。如果数据需要压缩，使用 STORED AS SEQUENCE 。

**建立外部表**

CREATE EXTERNAL TABLE page\_view(viewTime INT, userid BIGINT,

     page\_url STRING, referrer\_url STRING,

     ip STRING COMMENT 'IP Address of the User',

     country STRING COMMENT 'country of origination')

 COMMENT 'This is the staging page view table'

 ROW FORMAT DELIMITED FIELDS TERMINATED BY '\054'

 STORED AS TEXTFILE

 LOCATION '<hdfs\_location>';

**建分区表**

CREATE TABLE par\_table(viewTime INT, userid BIGINT,

     page\_url STRING, referrer\_url STRING,

     ip STRING COMMENT 'IP Address of the User')

 COMMENT 'This is the page view table'

 PARTITIONED BY(date STRING, pos STRING)

ROW FORMAT DELIMITED ‘\t’

   FIELDS TERMINATED BY '\n'

STORED AS SEQUENCEFILE;

**建Bucket表**

CREATE TABLE par\_table(viewTime INT, userid BIGINT,

     page\_url STRING, referrer\_url STRING,

     ip STRING COMMENT 'IP Address of the User')

 COMMENT 'This is the page view table'

 PARTITIONED BY(date STRING, pos STRING)

 CLUSTERED BY(userid) SORTED BY(viewTime) INTO 32 BUCKETS

 ROW FORMAT DELIMITED ‘\t’

   FIELDS TERMINATED BY '\n'

STORED AS SEQUENCEFILE;

**复制一个空表**

CREATE TABLE empty\_key\_value\_store

LIKE key\_value\_store;

**二、删除表**

DROP TABLE table\_name

**修改表结构**

•增加分区、删除分区

•重命名表

•修改列的名字、类型、位置、注释

•增加/更新列

•增加表的元数据信息

**增加、删除分区**

•增加

ALTER TABLE table\_name ADD [IF NOT EXISTS] partition\_spec [ LOCATION 'location1' ] partition\_spec [ LOCATION 'location2' ] ...

      partition\_spec:

  : PARTITION (partition\_col = partition\_col\_value, partition\_col = partiton\_col\_value, ...)

•删除

ALTER TABLE table\_name DROP partition\_spec, partition\_spec,...

**重命名表**

•ALTER TABLE table\_name RENAME TO new\_table\_name

**修改列的名字、类型、位置、注释**

•ALTER TABLE table\_name CHANGE [COLUMN] col\_old\_name col\_new\_name column\_type [COMMENT col\_comment] [FIRST|AFTER column\_name]

•这个命令可以允许改变列名、数据类型、注释、列位置或者它们的任意组合

**增加/更新列**

•ALTER TABLE table\_name ADD|REPLACE COLUMNS (col\_name data\_type [COMMENT col\_comment], ...)

• ADD是代表新增一字段，字段位置在所有列后面(partition列前)

     REPLACE则是表示替换表中所有字段。

**增加表的元数据信息**

•ALTER TABLE table\_name SET TBLPROPERTIES table\_properties table\_properties:

         :[property\_name = property\_value…..]

•用户可以用这个命令向表中增加metadata

**改变表文件格式与组织**

•ALTER TABLE table\_name SET FILEFORMAT file\_format

•ALTER TABLE table\_name CLUSTERED BY(userid) SORTED BY(viewTime) INTO num\_buckets BUCKETS

•这个命令修改了表的物理存储属性

**创建／删除视图**

•CREATE VIEW [IF NOT EXISTS] view\_name [ (column\_name [COMMENT column\_comment], ...) ][COMMENT view\_comment][TBLPROPERTIES (property\_name = property\_value, ...)] AS SELECT

•增加视图

•如果没有提供表名，视图列的名字将由定义的SELECT表达式自动生成

•如果修改基本表的属性，视图中不会体现，无效查询将会失败

•视图是只读的，不能用LOAD/INSERT/ALTER

•DROP VIEW view\_name

•删除视图

**创建数据库**

•CREATE DATABASE name

**显示命令**

•show tables;

•show databases;

•show partitions ;

•show functions

•describe extended table\_name dot col\_name

**第二部分：ＤＭＬ**

**DML**

•向数据表内加载文件

•将查询结果插入到Hive表中

•0.8新特性 insert into

**向数据表内加载文件**

•LOAD DATA [LOCAL] INPATH 'filepath' [OVERWRITE] INTO TABLE tablename [PARTITION (partcol1=val1, partcol2=val2 ...)]

•Load 操作只是单纯的复制/移动操作，将数据文件移动到 Hive 表对应的位置。

•filepath

•相对路径，例如：project/data1

•绝对路径，例如： /user/hive/project/data1

•包含模式的完整 URI，例如：hdfs://namenode:9000/user/hive/project/data1

**向数据表内加载文件**

•加载的目标可以是一个表或者分区。如果表包含分区，必须指定每一个分区的分区名

•filepath 可以引用一个文件（这种情况下，Hive 会将文件移动到表所对应的目录中）或者是一个目录（在这种情况下，Hive 会将目录中的所有文件移动至表所对应的目录中）

**LOCAL关键字**

•指定了LOCAL

•load 命令会去查找本地文件系统中的 filepath。如果发现是相对路径，则路径会被解释为相对于当前用户的当前路径。用户也可以为本地文件指定一个完整的 URI，比如：file:///user/hive/project/data1.

•load 命令会将 filepath 中的文件复制到目标文件系统中。目标文件系统由表的位置属性决定。被复制的数据文件移动到表的数据对应的位置

• 没有指定LOCAL

         如果 filepath 指向的是一个完整的 URI，hive 会直接使用这个 URI。 否则

•如果没有指定 schema 或者 authority，Hive 会使用在 hadoop 配置文件中定义的 schema 和 authority，fs.default.name 指定了 Namenode 的 URI

•如果路径不是绝对的，Hive 相对于 /user/ 进行解释。 Hive 会将 filepath 中指定的文件内容移动到 table （或者 partition）所指定的路径中

**OVERWRITE**

•指定了OVERWRITE

•目标表（或者分区）中的内容（如果有）会被删除，然后再将 filepath 指向的文件/目录中的内容添加到表/分区中。

•如果目标表（分区）已经有一个文件，并且文件名和 filepath 中的文件名冲突，那么现有的文件会被新文件所替代。

**将查询结果插入Hive表**

•将查询结果插入Hive表

•将查询结果写入HDFS文件系统

•基本模式

     INSERT OVERWRITE TABLE tablename1 [PARTITION (partcol1=val1, partcol2=val2 ...)] select\_statement1 FROM from\_statement

•多插入模式

 FROM from\_statement

INSERT OVERWRITE TABLE tablename1 [PARTITION (partcol1=val1, partcol2=val2 ...)] select\_statement1

[INSERT OVERWRITE TABLE tablename2 [PARTITION ...] select\_statement2] ...

•自动分区模式

 INSERT OVERWRITE TABLE tablename PARTITION (partcol1[=val1], partcol2[=val2] ...) select\_statement FROM from\_statement

**将查询结果写入HDFS文件系统**

•INSERT OVERWRITE [LOCAL] DIRECTORY directory1 SELECT ... FROM ...

        FROM from\_statement

        INSERT OVERWRITE [LOCAL] DIRECTORY directory1 select\_statement1

     [INSERT OVERWRITE [LOCAL] DIRECTORY directory2 select\_statement2]

•

•数据写入文件系统时进行文本序列化，且每列用^A 来区分，\n换行

**INSERT INTO**

•INSERT INTO  TABLE tablename1 [PARTITION (partcol1=val1, partcol2=val2 ...)] select\_statement1 FROM from\_statement

**第三部分：HiveQL 查询操作**

**SQL操作**

•基本的Select 操作

•基于Partition的查询

•Join

**基本的Select 操作**

SELECT [ALL | DISTINCT] select\_expr, select\_expr, ...

FROM table\_reference

[WHERE where\_condition]

[GROUP BY col\_list [HAVING condition]]

[   CLUSTER BY col\_list

  | [DISTRIBUTE BY col\_list] [SORT BY| ORDER BY col\_list]

]

[LIMIT number]

•使用ALL和DISTINCT选项区分对重复记录的处理。默认是ALL，表示查询所有记录。DISTINCT表示去掉重复的记录

•

•Where 条件

•类似我们传统SQL的where 条件

•目前支持 AND,OR ,0.9版本支持between

•IN, NOT IN

•不支持EXIST ,NOT EXIST

**ORDER BY与SORT BY的不同**

•ORDER BY 全局排序，只有一个Reduce任务

•SORT BY 只在本机做排序

**Limit**

•Limit 可以限制查询的记录数

SELECT \* FROM t1 LIMIT 5

•实现Top k 查询

•下面的查询语句查询销售记录最大的 5 个销售代表。

SET mapred.reduce.tasks = 1   
  SELECT \* FROM test SORT BY amount DESC LIMIT 5

•REGEX Column Specification

SELECT 语句可以使用正则表达式做列选择，下面的语句查询除了 ds 和 hr 之外的所有列：

SELECT `(ds|hr)?+.+` FROM test

**基于Partition的查询**

•一般 SELECT 查询会扫描整个表，使用 PARTITIONED BY 子句建表，查询就可以利用分区剪枝（input pruning）的特性

•Hive 当前的实现是，只有分区断言出现在离 FROM 子句最近的那个WHERE 子句中，才会启用分区剪枝