Hive

- What?Why?How?

Hive SQL

Hive的数据类型

- : primitive_type
- | array_type
- | map_type
- | struct_type
- : primitive_type
- |TINYINT
- | SMALLINT
- | INT
- | BIGINT
- | BOOLEAN
- | FLOAT
- | DOUBLE
- | STRING

Hive SQL

- Hive完整的DDL建表语法规则

```
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)
```

Hive SQL

- Hive 内部表
  - CREATE TABLE [IF NOT EXISTS] table_name
    - 删除表时，元数据与数据都会被删除
- Hive 外部表
  - CREATE EXTERNAL TABLE [IF NOT EXISTS] table_name LOCATION hdfs_path
    - 删除外部表只删除metastore的元数据，不删除hdfs中的表数据

Hive SQL

- Hive 建表
  - CREATE TABLE person(
  - id INT,
  - name STRING,
  - age INT,
  - likes ARRAY<STRING>,
  - address MAP<STRING,STRING>
  -)
  - ROW FORMAT DELIMITED
  - FIELDS TERMINATED BY ','
  - COLLECTION ITEMS TERMINATED BY '-'
  - MAP KEYS TERMINATED BY ':'
  - LINES TERMINATED BY '\n';
- Hive字段的默认值

Hive SQL

- Hive 查看表描述
  - DESCRIBE [EXTENDED|FORMATTED] table_name

Hive SQL

- Hive 建表
  - Create Table Like:
    - CREATE TABLE empty_key_value_store LIKE key_value_store;
  - Create Table As Select (CTAS)
    - CREATE TABLE new_key_value_store
      AS
      SELECT columA, columB FROM key_value_store;

Hive 分区

- Hive 分区partition
  - 必须在表定义时指定对应的partition字段
  - a、单分区建表语句：
    - create table day_table (id int, content string) partitioned by (dt string);
    - 单分区表，按天分区，在表结构中存在id，content，dt三列。
    - 以dt为文件夹区分
  - b、 双分区建表语句：
    - create table day_hour_table (id int, content string) partitioned by (dt string, hour string);
    - 双分区表，按天和小时分区，在表结构中新增加了dt和hour两列。
    - 先以dt为文件夹，再以hour子文件夹区分

Hive 分区

- Hive添加分区表语法
  - （表已创建，在此基础上添加分区）：
  - 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, ...)

  - 例：
  - ALTER TABLE day_table ADD PARTITION (dt='2008-08-08', hour='08')

Hive 分区

- Hive删除分区语法：
  - ALTER TABLE table_name DROP partition_spec, partition_spec,...
  - partition_spec:
    - : (partition_column = partition_col_value, partition_column = partition_col_value, ...)
  - 用户可以用 ALTER TABLE DROP PARTITION 来删除分区。
  - 内部表中、对应分区的元数据和数据将被一并删除。
  - 例：
    - ALTER TABLE day_hour_table DROP PARTITION (dt='2008-08-08', hour='09');

Hive 分区

- Hive向指定分区添加数据语法：
  - LOAD DATA [LOCAL] INPATH 'filepath' [OVERWRITE] INTO TABLE tablename [PARTITION (partcol1=val1, partcol2=val2 ...)]
  - 例：
    - LOAD DATA INPATH '/user/pv.txt' INTO TABLE day_hour_table PARTITION(dt='2008-08- 08', hour='08');
    - LOAD DATA local INPATH '/user/hua/*' INTO TABLE day_hour partition(dt='2010-07- 07');
  - 当数据被加载至表中时，不会对数据进行任何转换。Load操作只是将数据复制至Hive表对应的位置。数据加载时在表下自动创建一个目录

Hive 分区

- Hive查询执行分区语法
  - SELECT day_table.* FROM day_table WHERE day_table.dt>= '2008-08-08';
  - 分区表的意义在于优化查询。查询时尽量利用分区字段。如果不使用分区字段，就会全部扫描。
- Hive查询表的分区信息语法：
  - SHOW PARTITIONS day_hour_table;
- 预先导入分区数据，但是无法识别怎么办
  - Msck repair table tablename
  - 直接添加分区

Hive DML

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

Hive DML

- Hive DML

  FROM from_statement

  INSERT OVERWRITE TABLE tablename1 [PARTITION (partcol1=val1, partcol2=val2 ...) [IF NOT EXISTS]] select_statement1

  [INSERT OVERWRITE TABLE tablename2 [PARTITION ... [IF NOT EXISTS]] elect_statement2]

  [INSERT INTO TABLE tablename2 [PARTITION ...] select_statement2] ...;

Hive DML

- Hive DML
  - Delete
  - Update
  - Deletes can only be performed on tables that support ACID. See Hive Transactions for details.

Hive Serde

- Hive SerDe - Serializer and Deserializer
  - SerDe 用于做序列化和反序列化。
  - 构建在数据存储和执行引擎之间，对两者实现解耦。
  - Hive通过ROW FORMAT DELIMITED以及SERDE进行内容的读写。

```
row_format
: DELIMITED
        [FIELDS TERMINATED BY char [ESCAPED 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, ...)]
```

Hive Serde

- Hive正则匹配
  - CREATE TABLE logtbl (
  - host STRING,
  - identity STRING,
  - t_user STRING,
  - time STRING,
  - request STRING,
  - referer STRING,
  - agent STRING)
  - ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.RegexSerDe'
  - WITH SERDEPROPERTIES (
  - "input.regex" = "([^]*) ([^]*) ([^]*) \\[(.*)\\] \"(.*)\" (-|[0-9]*) (-|[0-9]*)"
  -)
  - STORED AS TEXTFILE;