Hive语句

建表

```
# hive中的表对应的都是文件夹，数据存储在文件夹下
# 建表一：全部使用默认建表方式
create table students
(
    id bigint,
    name string,
    age int,
    gender string,
    clazz string
)
row format delimited fields terminated by ",";

# 建表二：指定location
create table students2
(
    id bigint,
    name string,
    age int,
    gender string,
    clazz string
)
row format delimited fields terminated by ","
location '/input1'; // 指定Hive表的数据的存储位置，一般数据已经上传到HDFS上了，所以想要使用的话，就会指定Location。

# 建表三：指定存储格式
create table students3
(
    id bigint,
    name string,
    age int,
    gender string,
    clazz string
)
row format delimited fields terminated by ","
stored as rcfile; // 指定存储格式为rcfile，如果你不指定存储格式，默认为textFile

# 建表四：create table xxxx as select * from xxx(sql语句)
# 这种建表方式，会将select所查到的内容也带过去
create table students4 as select * from students2;

# 建表五：create table xxxx like table_name 只建立一样的表结构，不会有数据
create table students5 like students;
```

上传数据

```
1.使用hdfs dfs -put 上传数据至hive表对应的目录下

2.load data inpath 是从HDFS上导入数据
// 将HDFS上/input1/ 下的数据移动到students表对应的HDFS目录下。
进入hive shell
load data inpath '/input1/students.txt' into table students;

// 清空表
truncate table students;
// load data local inpath 可以从Linux本地将文件导入，上传到hive表对应的HDFS目录下，原文件不会被删除。
load data local inpath '/usr/local/soft/data/students.txt' into table students;
重复导入会进行数据追加
// overwrite 覆盖加载
load data local inpath '/usr/local/soft/data/students.txt' overwrite into table students;

3.insert into table select xxx (SQL语句)
insert into table students2 select * from students;

insert overwrite table students2 select * from students;
```

Hive内部表(Managed tables)和外部表(External tables)

```
// 内部表
create table students_internal
(
    id bigint,
    name string,
    age int,
    gender string,
    clazz string
)
row format delimited fields terminated by ","
location '/input2';

// 外部表
create external table students_external
(
    id bigint,
    name string,
    age int,
    gender string,
    clazz string
)
row format delimited fields terminated by ","
location '/input3';

// 加载数据
hadoop dfs -put /usr/local/soft/data/students.txt /input2/
hadoop dfs -put /usr/local/soft/data/students.txt /input3/

// 删除表
drop table students_internal;
```

```
drop table students_external;
```

可以看出，删除内部表的时候，表中的数据(HDFS上的文件)会被同表的元数据一起删除，删除外部表的时候，只会删除表的元数据，不会删除表中的数据(HDFS上的文件)
一般在公司中，使用外部表会多一些，因为数据可以需要被多个程序使用，避免误删，外部表通常会结合location一起使用。

Hive分区表

1. 分区表实际上是在表的目录下再以分区命名，建立子目录
2. 作用: 进行分区裁剪，避免全表扫描，减少MapReduce处理的数据量，可以提高效率
3. 在公司中，基本上所有的表 都是分区表，通常按照日期分区，地域分区
4. 分区也不是越多越好，一般来说不超过3级，根据实际业务衡量。

建立分区表

```
create external table students_pt1
(
    id bigint,
    name string,
    age int,
    gender string,
    clazz string
)
partitioned by(pt string)
row format delimited fields terminated by ","
location '/student/input1';
```

对分区操作

```
// 增加一个分区
alter table students_pt1 add partition(pt='20220218');

// 删除一个分区 如果创建的表是外部表的话，删除分区的时候，也不会删除HDFS上分区的数据文件
alter table students_pt1 drop partition(pt='20220216');

// 查看某个表的所有分区
show partitions students_pt1; // (推荐采用这种方式，直接从元数据中获取)

select distinct pt from students_pt1; // 不推荐，需要跑MapReduce

// 往分区中插入数据
insert into table students_pt1 partition(pt='20220218') select * from students;
load data local inpath '/usr/local/soft/data/students.txt' into table
students_pt1 partition(pt='20220217')

// 查询某个分区的数据
select count(*) from students_pt1; // 全表扫描，分区没有产生作用，不推荐

// 使用where条件进行分区裁剪，避免全表扫描，效率会变高
select count(*) from students_pt1 where pt='20220218';

// 也可以在where条件中使用非等值判断
select count(*) from students_pt1 where pt<='20220120' and pt>='20220219';
```

Hive动态分区

- 有的时候大家原始表中的数据包含了时间日期字段 dt，我们需要根据dt中不同的日期，分为不同的分区，将原始表改造成分区表
- hive默认是不开启动态分区的
- 动态分区：根据数据中某几列的不同的取值划分，不同的分区

```
# 开启动态分区支持
set hive.exec.dynamic.partition=true;
# 表示动态分区模式: strict(需要配合静态分区一起使用)  nostrict
set hive.exec.dynamic.partition.mode=nostrict;
# 表示最大支持的分区数量
set hive.exec.max.dynamic.partitions.pernode=1000;

# 建立原始表并且加载数据
create table students_dt
(
    id bigint,
    name string,
    age int,
    gender string,
    clazz string,
    dt string
)
row format delimited fields terminated by ",";
load data local inpath '/usr/local/soft/data/studentsdt.txt' into table
students_dt;


# 建立分区表并加载数据
create table students_dt_p
(
    id bigint,
    name string,
    age int,
    gender string,
    clazz string
)
partitioned by (dt string)
row format delimited fields terminated by ",";

# 使用动态分区插入数据
# 注意这里分区字段在匹配的时候，不是按照名字匹配，是按照位置匹配，默认匹配select的最后一位
insert into table students_dt_p partition(dt) select id,name,age,gender,clazz,dt
from students_dt;
insert into table students_dt_p partition(dt) select id,name,dt,gender,clazz,age
from students_dt;
```

多级分区

```
create table students_year_month
(
    id bigint,
    name string,
    age int,
    gender string,
```

```
    clazz string,
    year string,
    month string
)
ROW FORMAT DELIMITED FIELDS TERMINATED BY ',';

create table students_year_month_pt
(
    id bigint,
    name string,
    age int,
    gender string,
    clazz string
)
PARTITIONED BY(year string,month string)
ROW FORMAT DELIMITED FIELDS TERMINATED BY ',';

load data local inpath '/usr/local/soft/data/studentsym.txt' into table
students_year_month;

insert into table students_year_month_pt partition(year,month) select
id,name,age,gender,clazz,year,month from students_year_month;
```

Hive分桶

分桶实际上是对数据的进一步划分

hive默认是关闭分桶

作用：在往分桶表中插入数据的时候，会根据clustered by指定的字段进行hash分区对指定的buckets个数进行取余，进而可以将数据分割成buckets个数个文件，以达到数据均匀分布，可以解决一些数据倾斜问题，可以提高效率

```
# 开启分桶开关
set hive.enforce.bucketing=true;

# 建立分桶表
create table students_buks
(
    id bigint,
    name string,
    age int,
    gender string,
    clazz string
)
clustered by (clazz) into 12 buckets
row format delimited fields terminated by ",";

# 往分桶表中插入数据直接使用load data并不能将数据打散
# 需要使用insert into
insert into students_buks select * from students;
```