Lab 7: Hive Auto Partitioning

The usual preambles from the previous labs about having your repo up to date apply. The lab also uses the '/user/mara_dev/data' directory as a data staging area as in previous labs. The have commands are listed in text format in the commands.txt file in this lab directorty.

In this lab, Hive will automatically partition at table for you based on the values in a column you specify.

Data Setup

1. This lab will use the the same data as for lab #1 Hive Basics
2. Upload the CSV file sampledatafoodsales.csv into the folder which he have used before: /user/maria_dev/data
3. You can either create a database or use the default database. The rest of the instructions assume the default database.
4. The schema for the data that will be loaded into the table is:

```
orderdate   string;
region      string;
city        string;
category    string;
product     string;
quantity    int;
unitprice   double;
total       double;
```

Creating the Base Table

1. The base table is the data without partitioning. The following is the Hive command to create the table.

```
create table foodbase(
      orderdate   string,
      region      string,
      city        string,
      category    string,
      product     string,
      quantity    int,
      unitprice   double,
      total       double)
row format delimited fields terminated by ','
lines terminated by '\n' stored as textfile;
```

2. Modify the table to skip the first header row when reading.

```
alter table foodbase set tblproperties("skip.header.line.count"="1");
```

3. Load the data

```
load data inpath'/user/maria_dev/data/sampledatafoodsales.csv'
    into table foodbase;
```

4. Check to see that the data has been loaded correctly.

Create a Partitioned Table

1. Create a second table that tells Hive to partition on the region column. Notice that the column definition for region is not in its usual place but is specified as the partition value.

```
create table food1(
      orderdate   string,
      city        string,
      category    string,
      product     string,
      quantity    int,
      unitprice   double,
      total       double)
partitioned by (region string)
row format delimited fields terminated by ','
lines terminated by '\n' stored as textfile;
```

2. Insert the base table into the partitioned table. Notice region is listed last in the select statement.

```
insert into table food1 partition(region) select orderdate,
      city, category, product, quantity, unitprice, total, region
      from foodbase;
```

3. Examine the resulting partitions as well as checking the directories in the managed file directories in the warehouse directory.

```
show partitions food1;

select count(*) from food1;

select count(*) from food1 where region='East';

select count(*) from food1 where region='West';

select count* from food1; // Note the ordering of the rows and where
                             the region column is
```

Other Partitions

Repeat this exercise using a different column to partition the foodbase table.

1. Create a new table with the partitioning defined with the column you choose.
2. Insert the data from foodbase
3. Examine the results like you did before

Why would choosing “unitprice” be a bad choice for a column to partition on.