Lab 6: Manual Hive Partitioning

These labs assume you are using the class repository from the Nuvelab VM. You can also clone the repo to your HDP VM but that variation is not covered in the lab instructions.

Data Setup

1. Log into the Ambari file explorer
2. Create a new “staging_area” directory in /user/maria_dev that is fully writeable, just like you did in previous labs. You should have a directory named “/user/maria_dev/parts”.
3. Also make sure you have a directory “/user/maria_dev/data” that is writeable by everyone.
4. Make sure the git repo is up to date but executing the command “git pull” in the Nuvelab VM.
5. Upload the two file from the data directory in this lab in the repository named "transaction-2015-01-01.csv" and "transaction-2015-01-02.csv" (There are more data files available if you want to experiment with more partitions).
6. Ensure the files are writable by everyone and that they are in the staging area.
7. Use LibreOffice in the Nuvelab VM to open one of the files so can see what the data looks like.

Creating the Table

The table represents a series of credit card transactions. The schema is

```
transid     integer
account-id  integer
vendor-id   integer
tdsmpt      string    // time/date stamp
city        string
state       string
total       double
```

1. For this lab, we will just use the default database.
2. In this lab, we are imposing an external partition on the table – we are not using the table data itself to decided on partitions. In this case, we are deciding to load the different CSV files into different partitions corresponding to the year of the transactions. This is

useful when we have a lot a data and often want to both analyze the data within specific years and across specific years.

3. Create an external table trans_p

```
Create external table trans_p(
      id          int,
      account_id  int,
      vendor_id   int,
      tstamp      string,
      city        string,
      state       string,
      amount      double)
partitioned by (dt string)
row format delimited terminated by ',' stored as textfile
location '/user/maria_dev/data/in-part'  ;
```

4. Notice that partitioning parameter is NOT a column value in the table.

Load Data

Load the data from the two files into two different partitions.

```
load data inpath '/user/maria_dev/staging_area/transaction-2015-01-1.csv'
      into table trans_p partition (dt='2015-01-01');

load data inpath '/user/maria_dev/staging_area/transaction-2015-01-2.csv'
      into table trans_p partition (dt='2015-01-02');
```

Visually confirm the data has been moved from the staging area to the partitions.

View the data

1. You can view the partitions by doing:

```
show partitions trans_p;
```

2. You can also count the number of rows in each partition with:

```
select count(*) from trans_p where dt='2015-01-01';

select count(*) from trans_p where dt='2015-01-01';
```

3. However, because this is still all a single table, you can count the total number of rows as if the partitions did not exist.

```
select count(*) from trans_p;
```

4. The partitions allow you to use the equivalent to a where clause without any computational overhead.