Big Data And Hadoop

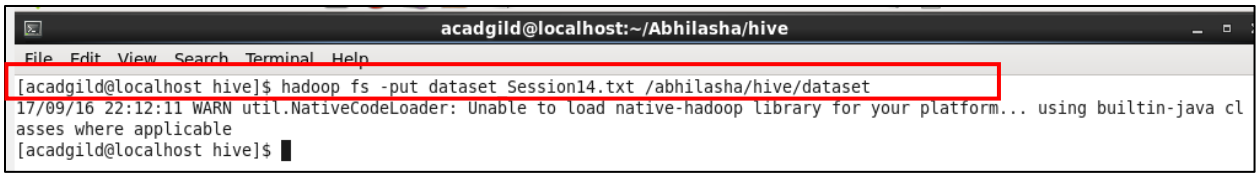
Session 14 - Assignment 2

Problem Statement:

- Fetch date and temperature from temperature_data where zip code is greater than 300000 and less than 399999.
- Calculate maximum temperature corresponding to every year from temperature_data table.
- Calculate maximum temperature from temperature_data table corresponding to those years which have at least 2 entries in the table.
- Create a view on the top of last query, name it temperature_data_vw.
- Export contents from temperature_data_vw to a file in local file system, such that each file is '|' delimited.

Solution:

Input File: The input file is downloaded and placed on the local system at **/home/acadgild/Abhilasha/hive**. We put this file on HDFS using the **put** command at location **/abhilasha/hive** and renamed the file to **dataset** as follows:



Start hive: We start the hive command line by executing the command **hive** as shown below:

The above snapshot also shows that hive prompt has started. A pre-requisite to use hive is to start mysql server. This was done using the command **sudo service mysqld start.**

1. **Fetch date and temperature from temperature_data where zip code is greater than 300000 and less than 399999.**

Solution:

i. The database we are using is named **custom.** It can be listed using the command **SHOW DATABASES;**

```
acadgild@localhost:~
File  Edit  View  Search  Terminal  Help
hive> show databases;
OK
b1
custom
default
Time taken: 0.064 seconds, Fetched: 3 row(s)
hive>
acadgild@localhost:~/...   conf - File Browser   acadgild@localhost:~
```

ii. Next is to mention which database we want to work on. This is done using the command **USE custom;**

```
acadgild@localhost:~
File  Edit  View  Search  Terminal  Help
hive> USE custom;
OK
Time taken: 0.065 seconds
hive>
```

iii. **SHOW TABLES;** command lists all the tables in the current database and the table we will be using is **temperature_data** and is appearing in the list as follows:

```
acadgild@localhost:~
File  Edit  View  Search  Terminal  Help
hive> SHOW TABLES;
OK
temperature_data
Time taken: 0.068 seconds, Fetched: 1 row(s)
hive>
```

iv. Using **DESCRIBE** command gives the schema of the table as shown below:

```
acadgild@localhost:~
File  Edit  View  Search  Terminal  Help
hive> DESCRIBE temperature_data;
OK
date                    string
zipcode                 int
temperature             int
Time taken: 0.224 seconds, Fetched: 3 row(s)
hive>
```

v. We can also use **DESCRIBE FORMATTED** command to get detailed description of the as follows:

```
acadgild@localhost:~
File  Edit  View  Search  Terminal  Help
hive> DESCRIBE FORMATTED temperature_data;
OK
# col_name              data_type               comment

date                    string
zipcode                 int
temperature             int

# Detailed Table Information
Database:               custom
Owner:                  acadgild
CreateTime:             Sun Sep 17 15:16:09 IST 2017
LastAccessTime:         UNKNOWN
Protect Mode:           None
Retention:              0
Location:               hdfs://localhost:9000/user/hive/warehouse/custom.db/temperature_data
Table Type:             MANAGED_TABLE
Table Parameters:
        transient_lastDdlTime   1505641569

# Storage Information
SerDe Library:          org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe
InputFormat:            org.apache.hadoop.mapred.TextInputFormat
OutputFormat:           org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat
Compressed:             No
Num Buckets:            -1
Bucket Columns:         []
Sort Columns:           []
Storage Desc Params:
        field.delim             ,
        serialization.format    ,
Time taken: 0.199 seconds, Fetched: 29 row(s)
hive> LOAD DATA INPATH '/abhilasha/hive/dataset'
OVERWRITE INTO TABLE temperature_data;
Loading data to table custom.temperature_data
Table custom.temperature_data stats: [numFiles=1, numRows=0, totalSize=437, rawDataSize=0]
OK
Time taken: 0.706 seconds
hive>
```

vi. Now, we execute the query **SELECT date, temperature from temperature_data WHERE zipCode>300000 AND zipCode<3999999;**
We need to get only those records that have zip code is greater than 300000 and less than 399999. So we use **WHERE** clause to apply this filter and get the records accordingly.

```
acadgild@localhost:~
File  Edit  View  Search  Terminal  Help
hive> SELECT date, temperature from temperature_data WHERE zipCode>300000 AND zipCode<3999999;
OK
10-03-1990      15
10-01-1991      22
12-02-1990      9
10-03-1991      16
10-01-1990      23
12-02-1991      10
10-03-1993      16
10-01-1994      23
12-02-1991      10
10-03-1991      16
10-01-1990      23
12-02-1991      10
Time taken: 0.239 seconds, Fetched: 12 row(s)
hive>
```

2. **Calculate maximum temperature corresponding to every year from temperature_data table.**

Solution:

We need to group the records based on the year in the date and get maximum temperature for every such group.

The date field is in STRING and does not follow the default date format. Hence we convert it to unix time using **unix_timestamp(date,'mm-dd-yyyy').** Here, we mention the date format in which the data is present. This function gives time in unix epoch.

However, we need the year part of the date and hence convert this epoch to default date format of hive using the function **from_unixtime** on the output of the function **unix_timestamp.**

Now that we have the date in hive's default format, we use the function **YEAR().** This gives the year part of the date in the data.

We have used **GROUP BY** clause to group data by year.

We have used **MAX** function to get max of temperature in every group.

Hence, the complete query is :

SELECT YEAR(from_unixtime(unix_timestamp(date,'mm-dd-yyyy'))), MAX(temperature) FROM temperature_data GROUP BY YEAR(from_unixtime(unix_timestamp(date,'mm-dd-yyyy')))

```
acadgild@localhost:~
File  Edit  View  Search  Terminal  Help

hive> SELECT YEAR(from_unixtime(unix_timestamp(date,'mm-dd-yyyy'))), MAX(temperature) FROM temperature_data GROUP BY YEAR(fro
m_unixtime(unix_timestamp(date,'mm-dd-yyyy')));
Query ID = acadgild_20170917155757_d7a08d11-a055-4474-ac8a-bea7bc6d1792
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks not specified. Estimated from input data size: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1505629817517_0006, Tracking URL = http://localhost:8088/proxy/application_1505629817517_0006/
Kill Command = /home/acadgild/hadoop-2.6.0/bin/hadoop job  -kill job_1505629817517_0006
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2017-09-17 15:57:11,525 Stage-1 map = 0%,  reduce = 0%
2017-09-17 15:57:19,291 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 3.08 sec
2017-09-17 15:57:27,914 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 5.43 sec
MapReduce Total cumulative CPU time: 5 seconds 430 msec
Ended Job = job_1505629817517_0006
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 5.43 sec   HDFS Read: 667 HDFS Write: 32 SUCCESS
Total MapReduce CPU Time Spent: 5 seconds 430 msec
OK
1990    23
1991    22
1993    16
1994    23
Time taken: 26.03 seconds, Fetched: 4 row(s)
hive>
```

The above screenshot also shows the output of the query executed.

We can also get the detailed flow of execution, the plan of execution using the **EXPLAIN** command as follows:

```
acadgild@localhost:~
File  Edit  View  Search  Terminal  Help
hive> EXPLAIN
SELECT YEAR(from_unixtime(unix_timestamp(date,'mm-dd-yyyy'))), MAX(temperature) FROM temperature_data GROUP BY YEAR(from_unix
time(unix_timestamp(date,'mm-dd-yyyy')));
OK
STAGE DEPENDENCIES:
  Stage-1 is a root stage
  Stage-0 depends on stages: Stage-1

STAGE PLANS:
  Stage: Stage-1
    Map Reduce
      Map Operator Tree:
          TableScan
            alias: temperature_data
            Statistics: Num rows: 4 Data size: 437 Basic stats: COMPLETE Column stats: NONE
            Select Operator
              expressions: date (type: string), temperature (type: int)
              outputColumnNames: date, temperature
              Statistics: Num rows: 4 Data size: 437 Basic stats: COMPLETE Column stats: NONE
              Group By Operator
                aggregations: max(temperature)
                keys: year(from_unixtime(unix_timestamp(date,'mm-dd-yyyy'))) (type: int)
                mode: hash
                outputColumnNames: _col0, _col1
                Statistics: Num rows: 4 Data size: 437 Basic stats: COMPLETE Column stats: NONE
                Reduce Output Operator
                  key expressions: _col0 (type: int)
                  sort order: +
                  Map-reduce partition columns: _col0 (type: int)
                  Statistics: Num rows: 4 Data size: 437 Basic stats: COMPLETE Column stats: NONE
                  value expressions: _col1 (type: int)
      Reduce Operator Tree:
        Group By Operator
          aggregations: max(VALUE._col0)
          keys: KEY._col0 (type: int)
          mode: mergepartial
          outputColumnNames: _col0, _col1
          Statistics: Num rows: 2 Data size: 218 Basic stats: COMPLETE Column stats: NONE
          Select Operator
```

3. **Calculate maximum temperature from temperature_data table corresponding to those years which have at least 2 entries in the table.**

Solution: The solution to this is an extension of the previous query. After the grouping is done, we need to apply filter to get only those records that have year with at least 2 entries in the table. Hence, we use **COUNT** function and apply the predicate on it as follows:

SELECT YEAR(from_unixtime(unix_timestamp(date,'mm-dd-yyyy'))), MAX(temperature) FROM temperature_data GROUP BY YEAR(from_unixtime(unix_timestamp(date,'mm-dd-yyyy'))) HAVING COUNT(*)>1;

The results are as follows:

```
acadgild@localhost:~
File  Edit  View  Search  Terminal  Help
hive> SELECT YEAR(from_unixtime(unix_timestamp(date,'mm-dd-yyyy'))), MAX(temperature) FROM temperature_data GROUP BY YEAR(fro
m_unixtime(unix_timestamp(date,'mm-dd-yyyy'))) HAVING COUNT(*)>1;
Query ID = acadgild_20170917160202_32744a7f-7c4a-46d2-97c6-8f4656aae6a4
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks not specified. Estimated from input data size: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1505629817517_0008, Tracking URL = http://localhost:8088/proxy/application_1505629817517_0008/
Kill Command = /home/acadgild/hadoop-2.6.0/bin/hadoop job  -kill job_1505629817517_0008
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2017-09-17 16:02:24,253 Stage-1 map = 0%,  reduce = 0%
2017-09-17 16:02:32,202 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 3.23 sec
2017-09-17 16:02:41,128 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 7.12 sec
MapReduce Total cumulative CPU time: 7 seconds 120 msec
Ended Job = job_1505629817517_0008
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 7.12 sec   HDFS Read: 667 HDFS Write: 32 SUCCESS
Total MapReduce CPU Time Spent: 7 seconds 120 msec
OK
1990    23
1991    22
1993    16
1994    23
Time taken: 31.975 seconds, Fetched: 4 row(s)
hive>
```

4. **Create a view on the top of last query, name it temperature_data_vw.**

Solution: A common use case for views is restricting the result rows based on the value of one or more columns. When a query becomes long or complicated, a view may be used to hide the complexity by dividing the query into smaller, more manageable pieces; similar to writing a function in a programming language or the concept of layered design in software.
The query used to create the view is as follows:

CREATE VIEW temperature_data_vw as
SELECT YEAR(from_unixtime(unix_timestamp(date,'mm-dd-yyyy'))), MAX(temperature) FROM temperature_data GROUP BY YEAR(from_unixtime(unix_timestamp(date,'mm-dd-yyyy'))) HAVING COUNT(*)>1;

```
acadgild@localhost:~
File  Edit  View  Search  Terminal  Help
hive> CREATE VIEW temperature_data_vw as
SELECT YEAR(from_unixtime(unix_timestamp(date,'mm-dd-yyyy'))), MAX(temperature) FROM temperature_data GROUP BY YEAR(from_unix
time(unix_timestamp(date,'mm-dd-yyyy'))) HAVING COUNT(*)>1;
OK
Time taken: 0.194 seconds
hive>
```

The view that got created can be listed using the **SHOW** command as follows:

```
acadgild@localhost:~
File  Edit  View  Search  Terminal  Help
hive> SHOW tables;
OK
temperature_data
temperature_data_vw
Time taken: 0.1 seconds, Fetched: 2 row(s)
hive>
```

The schema of this view, using **DESCRIBE** command is as follows:

```
acadgild@localhost:~
File  Edit  View  Search  Terminal  Help
hive> DESCRIBE temperature_data_vw;
OK
_c0                     int
_c1                     int
Time taken: 0.193 seconds, Fetched: 2 row(s)
hive>
```

We can display the data of view by executing the select * query as follows:

```
acadgild@localhost:~
File  Edit  View  Search  Terminal  Help
hive> SHOW tables;
OK
temperature_data
temperature_data_vw
Time taken: 0.1 seconds, Fetched: 2 row(s)
hive> DESCRIBE temperature_data_vw;
OK
_c0                     int
_c1                     int
Time taken: 0.193 seconds, Fetched: 2 row(s)
hive> SELECT * FROM temperature_data_vw;
Query ID = acadgild_20170917215050_bb5aee97-b2a4-4a9c-b772-4e833aeb3306
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks not specified. Estimated from input data size: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1505664911475_0001, Tracking URL = http://localhost:8088/proxy/application_1505664911475_0001/
Kill Command = /home/acadgild/hadoop-2.6.0/bin/hadoop job  -kill job_1505664911475_0001
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2017-09-17 21:51:06,412 Stage-1 map = 0%,  reduce = 0%
2017-09-17 21:51:15,462 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 3.24 sec
2017-09-17 21:51:25,368 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 7.11 sec
MapReduce Total cumulative CPU time: 7 seconds 110 msec
Ended Job = job_1505664911475_0001
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 7.11 sec   HDFS Read: 667 HDFS Write: 32 SUCCESS
Total MapReduce CPU Time Spent: 7 seconds 110 msec
OK
1990    23
1991    22
1993    16
1994    23
Time taken: 38.413 seconds, Fetched: 4 row(s)
hive>
hive - File Browser
```

5. **Export contents from temperature_data_vw to a file in local file system, such that each file is '|' delimited.**

Solution: The content of the view is displayed in the previous snap shot. To export this into local file system, we use the following query:

```
insert overwrite local directory '/home/acadgild/Abhilasha/hive/output'
row format delimited
fields terminated by '|'
select * from temperature_data_vw;
```

Here, we use the **overwrite** flag to overwrite the content of the destination folder if its already existing.

The destination directory is **/home/acadgild/Abhilasha/hive/output.** The delimiter user to separate the fields is '|'.

```
acadgild@localhost:~
File  Edit  View  Search  Terminal  Help

hive> insert overwrite local directory '/home/acadgild/Abhilasha/hive/output'
row format delimited
fields terminated by '|'
select * from temperature_data_vw;
Query ID = acadgild_20170917160707_d6cc363d-4d4a-4a1e-9bd8-43e2b2d180d7
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks not specified. Estimated from input data size: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1505629817517_0009, Tracking URL = http://localhost:8088/proxy/application_1505629817517_0009/
Kill Command = /home/acadgild/hadoop-2.6.0/bin/hadoop job  -kill job_1505629817517_0009
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2017-09-17 16:08:05,220 Stage-1 map = 0%,  reduce = 0%
2017-09-17 16:08:13,016 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 3.03 sec
2017-09-17 16:08:21,647 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 6.42 sec
MapReduce Total cumulative CPU time: 6 seconds 420 msec
Ended Job = job_1505629817517_0009
Copying data to local directory /home/acadgild/Abhilasha/hive/output
Copying data to local directory /home/acadgild/Abhilasha/hive/output
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 6.42 sec   HDFS Read: 667 HDFS Write: 32 SUCCESS
Total MapReduce CPU Time Spent: 6 seconds 420 msec
OK
Time taken: 28.152 seconds
hive>
acadgild@localhost:~
```

On execution of the above query, the resultant folder can be listed on local file system as follows:

```
acadgild@localhost:~/Abhilasha/hive
File  Edit  View  Search  Terminal  Help
[acadgild@localhost hive]$ pwd
/home/acadgild/Abhilasha/hive
[acadgild@localhost hive]$ ls -l
total 32
-rw-rw-r--. 1 acadgild acadgild 1895 Sep 17 15:09 commands
-rw-rw-r--. 1 acadgild acadgild 1774 Sep 17 15:04 commands~
-rw-rw-r--. 1 acadgild acadgild  170 Sep 17 14:17 complexData
-rw-rw-r--. 1 acadgild acadgild  437 Sep 16 19:29 dataset_Session14.txt
-rw-rw-r--. 1 acadgild acadgild   84 Sep 17 13:43 empDetails
-rw-rw-r--. 1 acadgild acadgild    0 Sep 17 13:42 empDetails~
drwxrwxr-x. 2 acadgild acadgild 4096 Sep 17 16:08 output
-rw-rw-r--. 1 acadgild acadgild  170 Sep 17 14:17 Unsaved Document 1
-rw-rw-r--. 1 acadgild acadgild  170 Sep 17 14:17 Unsaved Document 1~
[acadgild@localhost hive]$
```

Listing the files of output directory as follows:

```
acadgild@localhost:~/Abhilasha/hive
File  Edit  View  Search  Terminal  Help
[acadgild@localhost hive]$ ls -l output;
total 4
-rw-r--r--. 1 acadgild acadgild 32 Sep 17 16:08 000000_0
[acadgild@localhost hive]$
```

Its content is as follows:

```
acadgild@localhost:~/Abhilasha/hive
File  Edit  View  Search  Terminal  Help
[acadgild@localhost hive]$ cat output/000000_0
1990|23
1991|22
1993|16
1994|23
[acadgild@localhost hive]$
```