

## Music Data Analysis

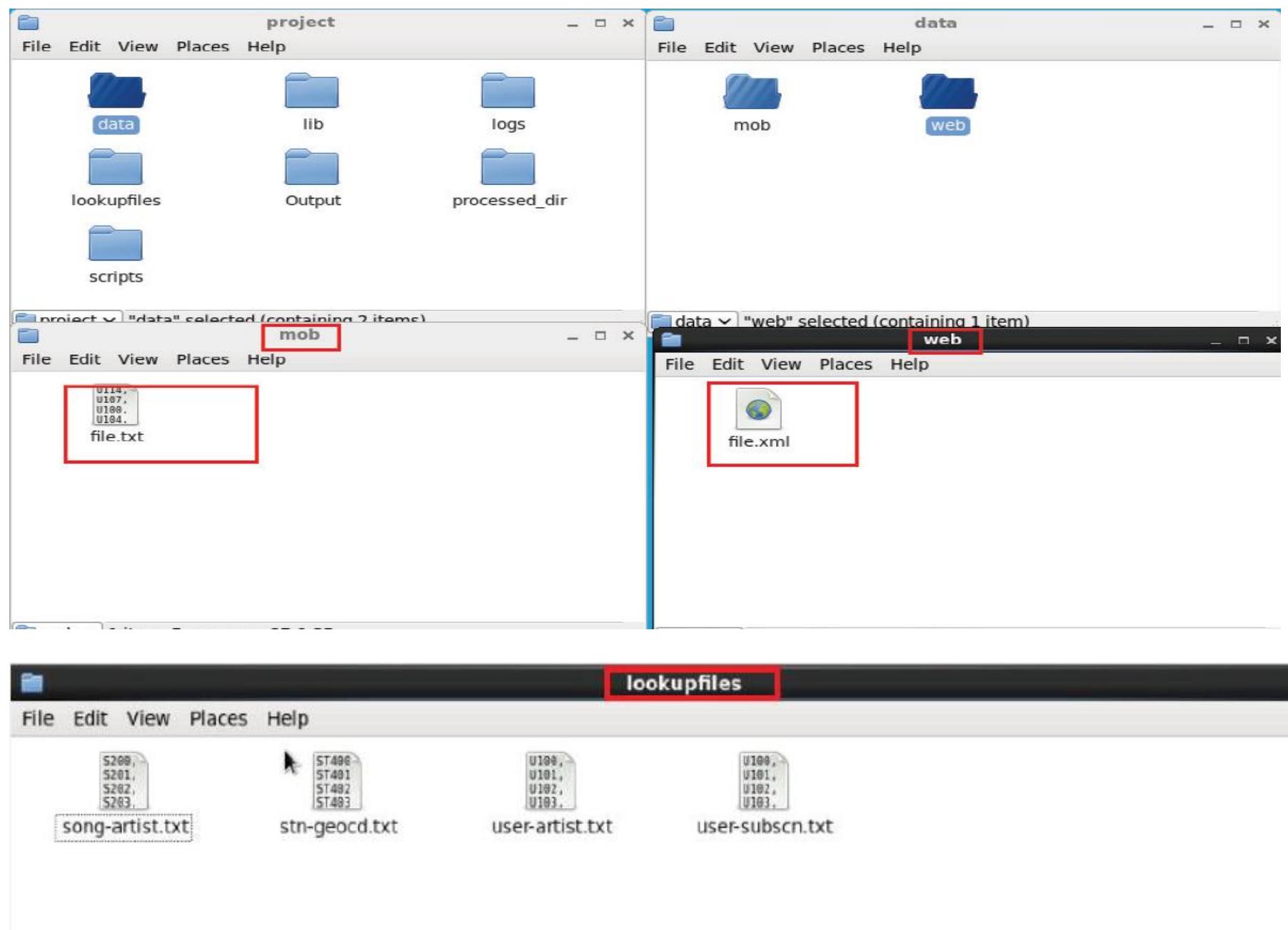
### **1. Introduction**

A leading music-catering company is planning to analyse large amount of data received from varieties of sources, namely mobile app and website to track the behaviour of users, classify users, calculate royalties associated with the song and make appropriate business strategies. The file server receives data files periodically after every 3 hours.

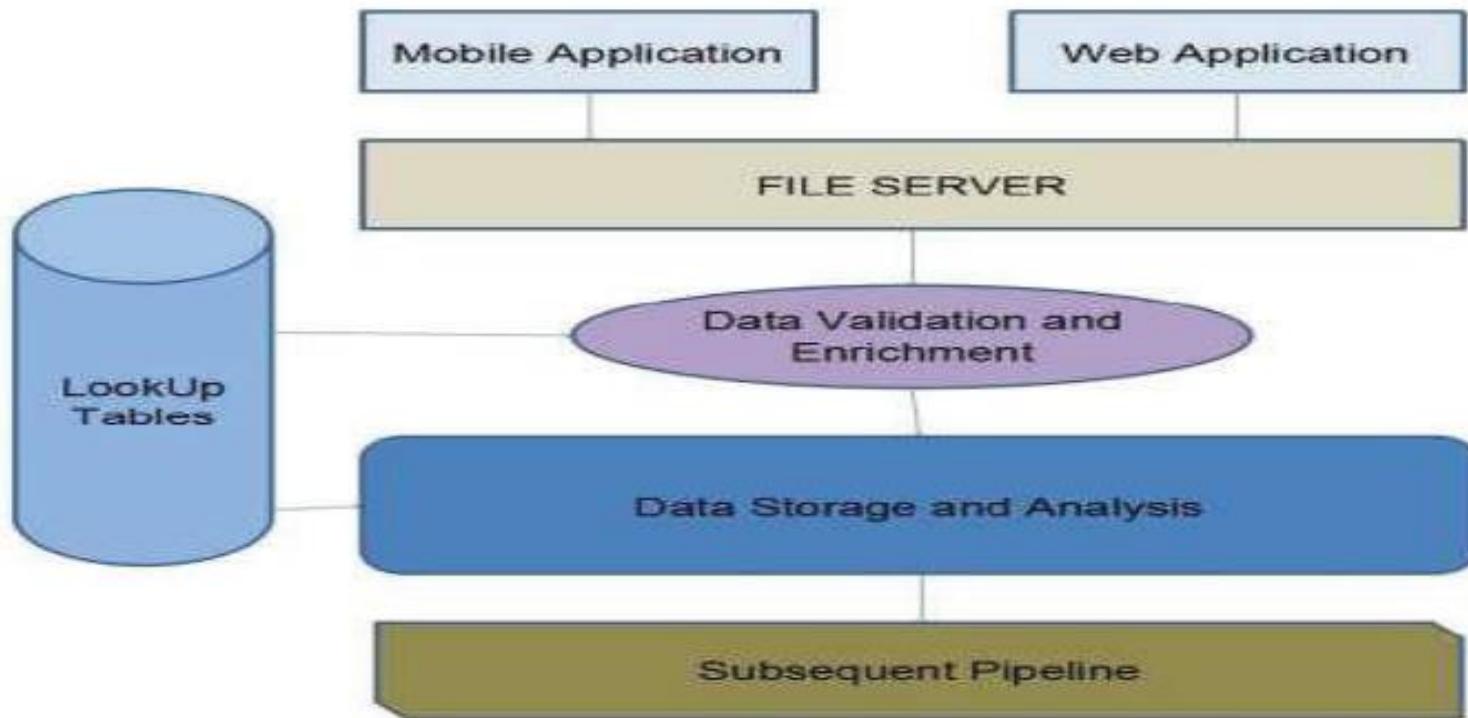
So as per periodic data for every 3 hours, we need to calculate the queries and return the results to business/customer.

### **2. DATASET:**

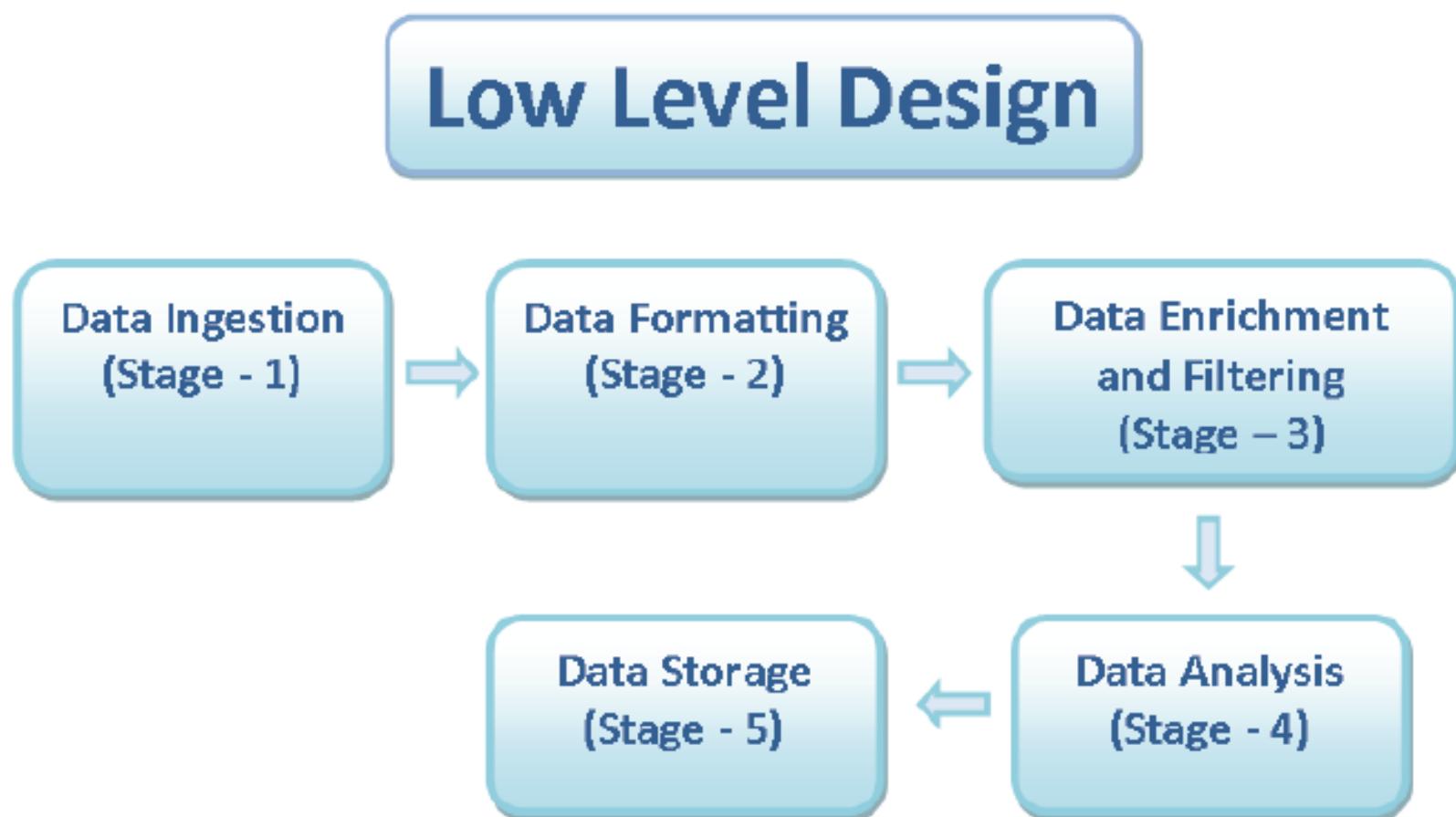
1. Data coming from web applications reside in /data/web and has xml format.
2. Data coming from mobile applications reside in /data/mob and has csv format.
3. Data present in lookup directory should be used in HBase.



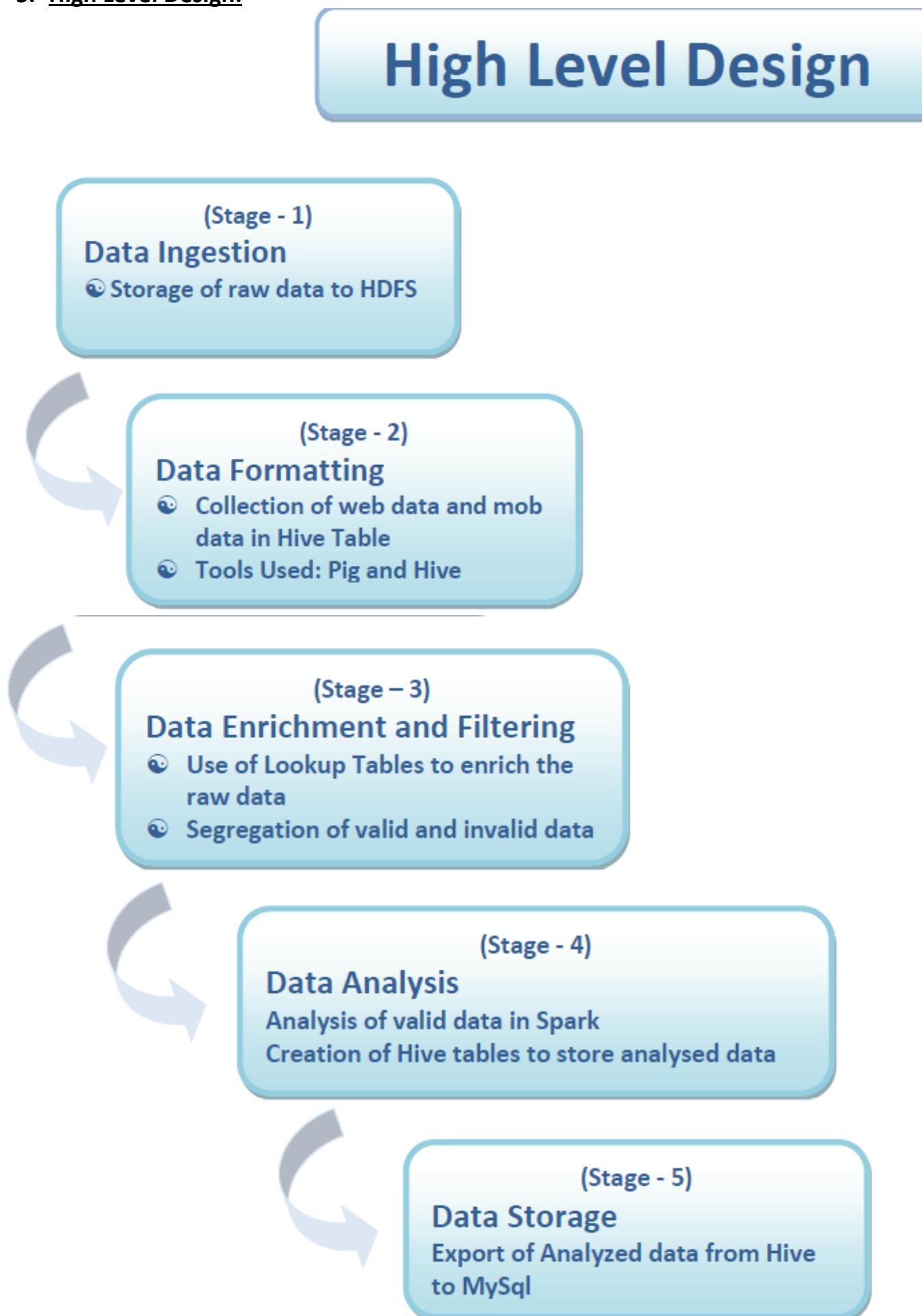
### 3. Operational Flow



### 4. Low Level Design:



5. High Level Design:



## **6. Implementation-**

### **a. Starting all daemons in hadoop-**

```
#!/bin/bash

if [ -f "/home/acadgild/project/logs/current-batch.txt" ]
then
echo "Batch File Found!"
else
echo -n "1" > "/home/acadgild/project/logs/current-batch.txt"
fi

chmod 775 /home/acadgild/project/logs/current-batch.txt
batchid=`cat /home/acadgild/project/logs/current-batch.txt`
LOGFILE=/home/acadgild/project/logs/log_batch_$batchid
echo "Starting daemons...." >> $LOGFILE

# To start the hadoop daemons:
/usr/local/hadoop-2.6.0/sbin/start-all.sh

# To start the Hmaster service:
/usr/local/hbase/bin/start-hbase.sh

# To start the Jobhistory server service:
mr-jobhistory-daemon.sh start historyserver

# To start mysql service:
sudo service mysqld start

# To start hive metastore:
hive --service metastore
```

```
[acadgild@localhost scripts]$ sh start-daemons.sh
Batch file found!
This script is Deprecated. Instead use start-dfs.sh and start-yarn.sh
17/09/11 12:42:30 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Starting namenodes on [localhost]
acadgild@localhost's password:
localhost: starting namenode, logging to /usr/local/hadoop-2.6.0/logs/hadoop-acadgild-namenode-localhost.localdomain.out
acadgild@localhost's password:
localhost: starting datanode, logging to /usr/local/hadoop-2.6.0/logs/hadoop-acadgild-datanode-localhost.localdomain.out
Starting secondary namenodes [0.0.0.0]
```

Same can be seen with jps command-

```
File Edit View Search Terminal Help
[acadgild@localhost ~]$
[acadgild@localhost ~]$ jps
3441 ResourceManager
5234 JobHistoryServer
5298 RunJar
2995 NameNode
3124 DataNode
3877 HMaster
3545 NodeManager
3292 SecondaryNameNode
5453 Jps
[acadgild@localhost ~]$ sudo service mysqld status
[sudo] password for acadgild:
mysqld (pid 5186) is running...
```

The above script will check whether the **current-batch.txt** file is available in the logs folder or not. If not it will create the file and dump value '1' in that file and create LOGFILE with the current batchid.

### **b. Lookup Tables creation in HBASE:**

By using the populate-lookup.sh script we will create lookup tables in Hbase. These tables have to be used in data formatting, data enrichment and analysis stage.

<b><u>Table Name</u></b>	<b><u>Description</u></b>
<b><u>Station Geo Map</u></b>	Contains mapping of a geo_cd with station_id
<b><u>Subscribed Users</u></b>	Contains user_id, subscription_start_date and subscription_end_date. Contains details only for subscribed users
<b><u>Song Artist Map</u></b>	Contains mapping of song_id with artist_id alongwith royalty associated with each play of the song.
<b><u>User Artist Map</u></b>	Contains an array of artist_id(s) followed by a user_id.

These 4 lookup files are used to create Hbase tables using populate-lookup.sh script and then with the help of data\_enrichment\_filtering\_schema.sh file we will create hive tables on the top of Hbase tables using create\_hive\_hbase\_lookup.hql

```
#!/bin/bash

batchid=`cat /home/cloudera/acadgild/project/logs/current-batch.txt`

LOGFILE=/home/cloudera/acadgild/project/logs/log_batch_$batchid

echo "Creating Lookup Tables ..." >> $LOGFILE

echo "create 'station-geo-map', 'geo'" | hbase shell
echo "create 'subscribed-users', 'subscn'" | hbase shell
echo "create 'song-artist-map', 'artist'" | hbase shell
echo "create 'user-artist-map', 'artists'" | hbase shell

echo "Populating Lookup Tables..." >> $LOGFILE

file="/home/cloudera/acadgild/project/lookupfiles/stn-geocd.txt"
while IFS= read -r line
do
  stnid=`echo $line | cut -d',' -f1`
  geocd=`echo $line | cut -d',' -f2`
  echo "put 'station-geo-map', '$stnid', 'geo:geo_cd', '$geocd'" | hbase shell
done <"$file"

file="/home/cloudera/acadgild/project/lookupfiles/song-artist.txt"
while IFS= read -r line
do
  songid=`echo $line | cut -d',' -f1`
  artistid=`echo $line | cut -d',' -f2`
  echo "put 'song-artist-map', '$songid', 'artist:artistid', '$artistid'" | hbase shell
done <"$file"

file="/home/cloudera/acadgild/project/lookupfiles/user-subscn.txt"
while IFS= read -r line
do
  userid=`echo $line | cut -d',' -f1`
  startdt=`echo $line | cut -d',' -f2`
  enddt=`echo $line | cut -d',' -f3`
  echo "put 'subscribed-users', '$userid', 'subscn:startdt', '$startdt'" | hbase shell
  echo "put 'subscribed-users', '$userid', 'subscn:enddt', '$enddt'" | hbase shell
done <"$file"

file="/home/cloudera/acadgild/project/lookupfiles/user-artist.txt"
```

```

touch /home/cloudera/acadgild/project/lookupfiles/user-artist1.txt
chmod 775 /home/cloudera/acadgild/project/lookupfiles/user-artist1.txt
file1="/home/cloudera/acadgild/project/lookupfiles/user-artist1.txt"
awk '$1=$1' FS="&" OFS=" " $file > $file1
num=1
while IFS= read -r line
do
    userid=`echo $line | cut -d',' -f1`
    artists=`echo $line | cut -d',' -f2`
    for row in $artists
    do
        #artist= `echo $row | cut -d',' -f$num
        echo "put 'user-artist-map', 'userid', 'artists:artist$num', '$row'" | hbase shell
        let "num=num+1"
    done
    num=1
done <"$file1"

```

By using the shell scripting we are creating 4 lookup tables in Hbase NoSQL Database.

```

File Edit View Search Terminal Help
2017-08-31 20:28:01,616 INFO [main] Configuration.deprecation: hadoop.native.lib is deprecated. Instead, use io.native.lib.available
HBase Shell; enter 'help<RETURN>' for list of supported commands.
Type "exit<RETURN>" to leave the HBase Shell
Version 0.98.14-hadoop2, r4e4aabb93b52f1b0fef6b66edd06ec8923014dec, Tue Aug 25 22:35:44 PDT 2015

'ut 'song-artist-map', 'S209', 'artist:artistid', 'A305'
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/local/hbase/lib/slf4j-log4j12-1.6.4.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/local/hadoop-2.6.0/share/hadoop/common/lib/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
2017-08-31 20:28:03,442 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
0 row(s) in 0.2850 seconds

2017-08-31 20:28:07,451 INFO [main] Configuration.deprecation: hadoop.native.lib is deprecated. Instead, use io.native.lib.available
HBase Shell; enter 'help<RETURN>' for list of supported commands.
Type "exit<RETURN>" to leave the HBase Shell
Version 0.98.14-hadoop2, r4e4aabb93b52f1b0fef6b66edd06ec8923014dec, Tue Aug 25 22:35:44 PDT 2015

put 'subscribed-users', 'U100', 'subscn:startdt', '1465230523'
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/local/hbase/lib/slf4j-log4j12-1.6.4.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/local/hadoop-2.6.0/share/hadoop/common/lib/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
2017-08-31 20:28:09,229 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
0 row(s) in 0.2700 seconds

2017-09-06 04:21:59,787 INFO [main] Configuration.deprecation: hadoop.native.lib is deprecated. Instead, use io.native.lib.available
HBase Shell; enter 'help<RETURN>' for list of supported commands.
Type "exit<RETURN>" to leave the HBase Shell
Version 1.2.0-cdh5.10.0, rUnknown, Fri Jan 20 12:13:18 PST 2017

'ut 'user-artist-map', 'U112', 'artists:artist2', 'A301
0 row(s) in 0.2180 seconds

2017-09-06 04:22:06,838 INFO [main] Configuration.deprecation: hadoop.native.lib is deprecated. Instead, use io.native.lib.available
HBase Shell; enter 'help<RETURN>' for list of supported commands.
Type "exit<RETURN>" to leave the HBase Shell
Version 1.2.0-cdh5.10.0, rUnknown, Fri Jan 20 12:13:18 PST 2017

put 'user-artist-map', 'U113', 'artists:artist1', 'A305
0 row(s) in 0.2250 seconds

2017-09-06 04:22:14,019 INFO [main] Configuration.deprecation: hadoop.native.lib is deprecated. Instead, use io.native.lib.available
HBase Shell; enter 'help<RETURN>' for list of supported commands.
Type "exit<RETURN>" to leave the HBase Shell
Version 1.2.0-cdh5.10.0, rUnknown, Fri Jan 20 12:13:18 PST 2017

'ut 'user-artist-map', 'U113', 'artists:artist2', 'A302
0 row(s) in 0.2180 seconds

2017-09-06 04:22:20,818 INFO [main] Configuration.deprecation: hadoop.native.lib is deprecated. Instead, use io.native.lib.available
HBase Shell; enter 'help<RETURN>' for list of supported commands.
Type "exit<RETURN>" to leave the HBase Shell
Version 1.2.0-cdh5.10.0, rUnknown, Fri Jan 20 12:13:18 PST 2017

put 'user-artist-map', 'U114', 'artists:artist1', 'A300
0 row(s) in 0.2260 seconds

2017-09-06 04:22:27,997 INFO [main] Configuration.deprecation: hadoop.native.lib is deprecated. Instead, use io.native.lib.available
HBase Shell; enter 'help<RETURN>' for list of supported commands.
Type "exit<RETURN>" to leave the HBase Shell
Version 1.2.0-cdh5.10.0, rUnknown, Fri Jan 20 12:13:18 PST 2017

put 'user-artist-map', 'U114', 'artists:artist2', 'A301
0 row(s) in 0.2220 seconds

2017-09-06 04:22:35,159 INFO [main] Configuration.deprecation: hadoop.native.lib is deprecated. Instead, use io.native.lib.available
HBase Shell; enter 'help<RETURN>' for list of supported commands.
Type "exit<RETURN>" to leave the HBase Shell
Version 1.2.0-cdh5.10.0, rUnknown, Fri Jan 20 12:13:18 PST 2017

```

Now we can see that the lookup tables have been created in hbase-

```
hbase(main):027:0> list
TABLE
song-artist-map
station-geo-map
subscribed-users
user-artist-map
4 row(s) in 0.0220 seconds

=> ["song-artist-map", "station-geo-map", "subscribed-users", "user-artist-map"]
hbase(main):028:0>
```

Same can be seen in below screenshots-

```
File Edit View Search Terminal Help
hbase(main):023:0> scan 'user-artist-map'
ROW
U100
U100
U100
U101
U101
U102
U103
U103
U103
U104
U104
U105
U105
U105
U105
U106
U106
U106
U107
U108
U108
U108
U109
U109
U110
U110
U111
U111
U111
U112
U112
U113
U113
U113
U114
U114
U114
15 row(s) in 0.1300 seconds

hbase(main):024:0> scan 'song-artist-map'
ROW
S200
S201
S202
S203
S204
S205
S206
S207
S208
S209
10 row(s) in 0.0530 seconds

File Edit View Search Terminal Help
hbase(main):025:0> scan 'station-geo-map'
ROW
ST400
ST401
ST402
ST403
ST404
ST405
ST406
ST407
ST408
ST409
ST410
ST411
ST412
ST413
ST414
15 row(s) in 0.0480 seconds
```

```

hbase(main):026:0> scan 'subscribed-users'
ROW
U100
U100
U101
U101
U102
U102
U103
U103
U104
U104
U105
U105
U106
U106
U107
U107
U108
U108
U109
U109
U110
U110
U111
U111
U112
U112
U113
U113
U114
U114
15 row(s) in 0.8590 seconds

```

COLUMN+CELL

column=subscn:enddt, timestamp=1504715225823, value=1465130523  
 column=subscn:startdt, timestamp=1504715218773, value=1465230523  
 column=subscn:enddt, timestamp=1504715239964, value=1475130523  
 column=subscn:startdt, timestamp=1504715232971, value=1465230523  
 column=subscn:enddt, timestamp=1504715254171, value=1475130523  
 column=subscn:startdt, timestamp=1504715246875, value=1465230523  
 column=subscn:enddt, timestamp=1504715268395, value=1475130523  
 column=subscn:startdt, timestamp=1504715261847, value=1465230523  
 column=subscn:enddt, timestamp=1504715282788, value=1475130523  
 column=subscn:startdt, timestamp=1504715275591, value=1465230523  
 column=subscn:enddt, timestamp=1504715297219, value=1475130523  
 column=subscn:startdt, timestamp=1504715298108, value=1465230523  
 column=subscn:enddt, timestamp=1504715311183, value=1485130523  
 column=subscn:startdt, timestamp=1504715304199, value=1465230523  
 column=subscn:enddt, timestamp=1504715325247, value=1455130523  
 column=subscn:startdt, timestamp=1504715318471, value=1465230523  
 column=subscn:enddt, timestamp=1504715339075, value=1465230623  
 column=subscn:startdt, timestamp=1504715332878, value=1465230523  
 column=subscn:enddt, timestamp=1504715352911, value=1475130523  
 column=subscn:startdt, timestamp=1504715345989, value=1465230523  
 column=subscn:enddt, timestamp=1504715367212, value=1475130523  
 column=subscn:startdt, timestamp=1504715368147, value=1465230523  
 column=subscn:enddt, timestamp=1504715381081, value=1475130523  
 column=subscn:startdt, timestamp=1504715374138, value=1465230523  
 column=subscn:enddt, timestamp=1504715394817, value=1475130523  
 column=subscn:startdt, timestamp=1504715387953, value=1465230523  
 column=subscn:enddt, timestamp=1504715408681, value=1485130523  
 column=subscn:startdt, timestamp=1504715401790, value=1465230523  
 column=subscn:enddt, timestamp=1504715422632, value=1468130523  
 column=subscn:startdt, timestamp=1504715415809, value=1465230523

### c. Creating Hive tables on top of Hbase

Now we shall link these lookup tables from hbase to the hive external tables by matching the columns of Hbase tables to HIVE tables using below shown scripts-

```

#!/bin/bash

batchid=`cat /home/acadgild/project/logs/current-batch.txt`
LOGFILE=/home/acadgild/project/logs/log-batch_$batchid

echo "Creating hive tables on top of hbase tables for data enrichment and filtering..." >> $LOGFILE

hive -f /home/acadgild/project/scripts/create_hive_hbase_lookup.hql

```

```

CREATE DATABASE IF NOT EXISTS project;
USE project;

create external table if not exists station_geo_map
(
station_id STRING,
geo_cd STRING
)
STORED By 'org.apache.hadoop.hive.hbase.HBaseStorageHandler'
with serdeproperties
("hbase.columns.mapping"":key,geo:geo cd")
tblproperties("hbase.table.name""station-geo-map");

create external table if not exists subscribed_users
(
user_id STRING,
subscn_start_dt STRING,
subscn_end_dt STRING
)
STORED BY 'org.apache.hadoop.hive.hbase.HBaseStorageHandler'
with serdeproperties
("hbase.columns.mapping"":key,subscn:startdt,subscn:enddt")
tblproperties("hbase.table.name""subscribed-users");

create external table if not exists song_artist_map
(
song_id STRING,
artist_id STRING
)
STORED BY 'org.apache.hadoop.hive.hbase.HBaseStorageHandler'
with serdeproperties
("hbase.columns.mapping"":key,artist:artistid")
tblproperties("hbase.table.name""song-artist-map");

create external table if not exists user_artist_map
(
user_id STRING,
artists array MAP<STRING,STRING>
)
STORED BY 'org.apache.hadoop.hive.hbase.HBaseStorageHandler'
with serdeproperties
("hbase.columns.mapping"":key,artists:")
tblproperties("hbase.table.name""user-artist-map");

```

Below screenshot shows tables getting created in hive by running the [data\\_enrichement\\_filtering\\_schema.sh](#) file.

```

File Edit View Search Terminal Help
[cloudera@quickstart scripts]$ ./data_enrichment_filtering_schema.sh

Logging initialized using configuration in file:/etc/hive/conf.dist/hive-log4j.properties
OK
Time taken: 0.395 seconds
OK
Time taken: 0.102 seconds
OK
Time taken: 0.021 seconds
OK
Time taken: 0.018 seconds
OK
Time taken: 1.085 seconds
WARN: The method class org.apache.commons.logging.impl.SLF4JLogFactory#release() was invoked.
WARN: Please see http://www.slf4j.org/codes.html#release for an explanation.
[cloudera@quickstart scripts]$ 

```

```

File Edit View Search Terminal Help
[cloudera@quickstart ~]$ hive

Logging initialized using configuration in file:/etc/hive/conf.dist/hive-log4j.properties
WARNING: Hive CLI is deprecated and migration to Beeline is recommended.
hive> show databases;
OK
default
project
Time taken: 0.499 seconds, Fetched: 2 row(s)
hive> use project;
OK
Time taken: 0.047 seconds
hive> show tables;
OK
song_artist_map
station_geo_map
subscribed_users
user_artist_map
Time taken: 0.049 seconds, Fetched: 4 row(s)
hive> 

```

#### d. Data Formatting

Now since we know that the data is coming from two sources namely mobile and web, we need to format them. So here we are merging data coming from both web applications and mobile applications and create a common table for analyzing purpose and create partitioned data based on batchid, since we are running this script for every 3 hours.

```
#!/bin/bash
batchid=`cat /home/acadgild/project/logs/current-batch.txt`
LOGFILE=/home/acadgild/project/logs/log_batch_${batchid}

echo "Placing data files from local to HDFS..." >> $LOGFILE
hadoop fs -rm -r /user/acadgild/project/batch${batchid}/web/
hadoop fs -rm -r /user/acadgild/project/batch${batchid}/formattedweb/
hadoop fs -rm -r /user/acadgild/project/batch${batchid}/mob/
hadoop fs -mkdir -p /user/acadgild/project/batch${batchid}/web/
hadoop fs -mkdir -p /user/acadgild/project/batch${batchid}/mob/
hadoop fs -put /home/acadgild/project/data/web/* /user/acadgild/project/batch${batchid}/web/
hadoop fs -put /home/acadgild/project/data/mob/* /user/acadgild/project/batch${batchid}/mob/
echo "Running pig script for data formatting..." >> $LOGFILE
pig -param batchid=$batchid /home/acadgild/project/scripts/dataformatting.pig
echo "Running hive script for formatted data load..." >> $LOGFILE
hive -hiveconf batchid=$batchid -f /home/acadgild/project/scripts/formatted_hive_load.hql
```

Here we are running two scripts to format the data namely- 1. Dataformatting.pig  
2. formatted\_hive\_load.hql

We are running the pig script to parse the data coming from web.xml to csv format and partition both web and mob data based on batch Ids.

Below are the scripts-

```
REGISTER /home/acadgild/project/lib/piggybank.jar;

DEFINE XPath org.apache.pig.piggybank.evaluation.xml.XPath();

A = LOAD '/user/acadgild/project/batch${batchid}/web/' using org.apache.pig.piggybank.storage.XMLLoader('record') as (x:chararray);

B = Foreach A GENERATE TRIM(XPath(x, 'record/user_id')) AS user_id,
      TRIM(XPath(x, 'record/song_id')) AS song_id,
      TRIM(XPath(x, 'record/artist_id')) AS artist_id,
      ToUnixTime(ToDate(TRIM(XPath(x, 'record/timestamp')),'yyyy-mm-dd HH:mm:ss')) AS timestamp,
      ToUnixTime(ToDate(TRIM(XPath(x, 'record/start_ts')),'yyyy-mm-dd HH:mm:ss')) AS start_ts,
      ToUnixTime(ToDate(TRIM(XPath(x, 'record/end_ts')),'yyyy-mm-dd HH:mm:ss')) AS end_ts,
      TRIM(XPath(x, 'record/geo_cd')) AS geo_cd,
      TRIM(XPath(x, 'record/station_id')) AS station_id,
      TRIM(XPath(x, 'record/song_end_type')) AS song_end_type,
      TRIM(XPath(x, 'record/like')) AS like,
      TRIM(XPath(x, 'record/dislike')) AS dislike;

STORE B INTO '/user/acadgild/project/batch${batchid}/formattedweb/' USING PigStorage(',');
```

Here we are running the HIVE script to load the parsed data from pig to hive and do analysis based on this data.

USE project;

```
CREATE TABLE IF NOT EXISTS formatted_input
(
  user_id STRING,
  song_id STRING,
  artist_id STRING,
  timestamp STRING,
  start_ts STRING,
  end_ts STRING,
  geo_cd STRING,
  station_id STRING,
  song_end_type INT,
  like INT,
  dislike INT
)
PARTITIONED BY
(batchid INT)
ROW FORMAT DELIMITED FIELDS TERMINATED BY ',';

LOAD DATA INPATH '/user/acadgild/project/batch${hiveconf:batchid}/formattedweb/'
INTO TABLE formatted_input PARTITION (batchid=${hiveconf:batchid});

LOAD DATA| INPATH '/user/acadgild/project/batch${hiveconf:batchid}/mob/'
INTO TABLE formatted_input PARTITION (batchid=${hiveconf:batchid});
```

In below screenshot we can see that first pig script is parsing the data and then hive script is loading the data into hive terminal successfully-

```
[WARN: Please see http://www.slf4j.org/codes.html#release for an explanation.
[cloudera@quickstart scripts]$ ./dataformatting.sh
Deleted /user/acadgild/project/batch1/web
`Lrm: '/user/acadgild/project/batch1/formattedweb/': No such file or directory
Deleted /user/acadgild/project/batch1/mob
log4j:WARN No appenders could be found for logger (org.apache.hadoop.util.Shell).
log4j:WARN Please initialize the log4j system properly.
log4j:WARN See http://logging.apache.org/log4j/1.2/faq.html#noconfig for more info.
2017-09-06 04:41:07,677 [main] INFO org.apache.pig.Main - Apache Pig version 0.12.0-cdh5.10.0 (rexported) compiled Jan 20 2017, 12:05:43
2017-09-06 04:41:07,678 [main] INFO org.apache.pig.Main - Logging error messages to: /home/cloudera/acadgild/project/scripts/pig_1504690067660.log
2017-09-06 04:41:08,930 [main] INFO org.apache.pig.impl.util.Utils - Default bootstrap file /home/cloudera/.pigbootup not found
2017-09-06 04:41:09,049 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - mapred.job.tracker is deprecated. Instead, use mapreduce.jobtracker.address
2017-09-06 04:41:09,050 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS
2017-09-06 04:41:09,056 [main] INFO org.apache.pig.backend.hadoop.executionengine.HExecutionEngine - Connecting to hadoop file system at: hdfs://quickstart.cloudera:8020
2017-09-06 04:41:09,501 [main] INFO org.apache.pig.backend.hadoop.executionengine.HExecutionEngine - Connecting to map-reduce job tracker at: localhost:8021
2017-09-06 04:41:09,564 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS
2017-09-06 04:41:09,565 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - mapred.job.tracker is deprecated. Instead, use mapreduce.jobtracker.address
2017-09-06 04:41:09,599 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS
2017-09-06 04:41:09,601 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - mapred.job.tracker is deprecated. Instead, use mapreduce.jobtracker.address
2017-09-06 04:41:09,652 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS
2017-09-06 04:41:09,652 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - mapred.job.tracker is deprecated. Instead, use mapreduce.jobtracker.address
2017-09-06 04:41:09,688 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS
2017-09-06 04:41:09,690 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - mapred.job.tracker is deprecated. Instead, use mapreduce.jobtracker.address
2017-09-06 04:41:09,737 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS
2017-09-06 04:41:09,738 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - mapred.job.tracker is deprecated. Instead, use mapreduce.jobtracker.address
2017-09-06 04:41:09,790 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS
2017-09-06 04:41:09,796 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - mapred.job.tracker is deprecated. Instead, use mapreduce.jobtracker.address
2017-09-06 04:41:09,818 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS
2017-09-06 04:41:09,818 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - mapred.job.tracker is deprecated. Instead, use mapreduce.jobtracker.address
2017-09-06 04:41:09,856 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS
2017-09-06 04:41:09,856 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - mapred.job.tracker is deprecated. Instead, use mapreduce.jobtracker.address
2017-09-06 04:41:09,895 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS
2017-09-06 04:41:09,896 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - mapred.job.tracker is deprecated. Instead, use mapreduce.jobtracker.address
2017-09-06 04:41:10,498 [main] INFO org.apache.pig.tools.pigstats.ScriptState - Pig features used in the script: UNKNOWN
2017-09-06 04:41:10,547 [main] INFO org.apache.pig.newplan.logical.optimizer.LogicalPlanOptimizer - {RULES_ENABLED=[AddForEach, ColumnMapKeyPrune, DuplicateForEachColumnRewrite, GroupByOptimizer, LoadTypeCastInserter, MergeFilter, MergeForEach, NewPartitionFilterOptimizer, PushDownForEachFlatten, PushUpFilter, SplitFilter, StreamTypeCastInserter], RULES_DISABLED=[Filter]
`l

2017-09-05 23:29:02,615 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - 0% complete
2017-09-05 23:29:16,031 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - 50% complete
2017-09-05 23:29:17,800 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - mapred.reduce.tasks is deprecated. Instead, use mapreduce.job.reduces
2017-09-05 23:29:17,850 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - 100% complete
2017-09-05 23:29:17,851 [main] INFO org.apache.pig.tools.pigstats.SimplePigStats - Script Statistics:
HadoopVersion PigVersion UserId StartedAt FinishedAt Features
2.6.0-cdh5.10.0 0.12.0-cdh5.10.0 cloudera 2017-09-05 23:28:55 2017-09-05 23:29:17 UNKNOWN

Success!
Job Stats (time in seconds):
JobId Maps Reduces MaxMapTime MinMapTime AvgMapTime MedianMapTime MaxReduceTime MinReduceTime AvgReduceTime MedianReducetime Alias Feature 0
job_1504678625866_0001 1 0 4 4 4 n/a n/a n/a n/a A,B MAP_ONLY /user/acadgild/project/batch1/formattedweb,
Input(s):
Successfully read 20 records (7353 bytes) from: "/user/acadgild/project/batch1/web"
Output(s):
Successfully stored 20 records (1236 bytes) in: "/user/acadgild/project/batch1/formattedweb"
Counters:
Total records written : 20
Total bytes written : 1236
Spillable Memory Manager spill count : 0
Total bags proactively spilled: 0
Total records proactively spilled: 0
Job DAG:
job_1504678625866_0001

2017-09-05 23:29:17,930 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - Success!
Logging initialized using configuration in file:/etc/hive/conf.dist/hive-log4j.properties
OK
Time taken: 0.393 seconds
OK
Time taken: 0.446 seconds
Loading data to table project.formatted input partition {batchid=1}
Partition project.formatted_input{batchid=1} stats: [numFiles=1, numRows=0, totalSize=1236, rawDataSize=0]
OK
Time taken: 0.908 seconds
Loading data to table project.formatted_input partition {batchid=1}
Partition project.formatted_input{batchid=1} stats: [numFiles=2, numRows=0, totalSize=2493, rawDataSize=0]
OK
Time taken: 0.742 seconds
WARN: The method class org.apache.commons.logging.impl.SLF4JLogFactory#release() was invoked.
WARN: Please see http://www.slf4j.org/codes.html#release for an explanation.
```

In the above screenshot we can see the dataformatting.pig along with the formatted\_hive\_load.hql executed successfully.

The output of **dataformatting.pig** script gets stored in HDFS folders-

```
[cloudera@quickstart scripts]$ hadoop fs -ls /user/acadgild/project/batch1/
Found 3 items
drwxr-xr-x  - cloudera supergroup          0 2017-09-05 12:14 /user/acadgild/project/batch1/formattedweb
drwxr-xr-x  - cloudera supergroup          0 2017-09-05 12:13 /user/acadgild/project/batch1/mob
drwxr-xr-x  - cloudera supergroup          0 2017-09-05 12:13 /user/acadgild/project/batch1/web
[cloudera@quickstart scripts]$ hadoop fs -ls /user/acadgild/project/batch1/formattedweb
Found 2 items
-rw-r--r--  1 cloudera supergroup      0 2017-09-05 12:14 /user/acadgild/project/batch1/formattedweb/_SUCCESS
-rw-r--r--  1 cloudera supergroup  2597 2017-09-05 12:14 /user/acadgild/project/batch1/formattedweb/part-m-00000
[cloudera@quickstart scripts]$ hadoop fs -ls /user/acadgild/project/batch1/web
Found 3 items
-rw-r--r--  1 cloudera supergroup  6716 2017-09-05 12:13 /user/acadgild/project/batch1/web/file.xml
-rw-r--r--  1 cloudera supergroup  1067 2017-09-05 12:13 /user/acadgild/project/batch1/web/web_data.xml.txt~
-rw-r--r--  1 cloudera supergroup  6962 2017-09-05 12:13 /user/acadgild/project/batch1/web/web_data.xml-
[cloudera@quickstart scripts]$ hadoop fs -ls /user/acadgild/project/batch1/mob
Found 2 items
-rw-r--r--  1 cloudera supergroup  1267 2017-09-05 12:13 /user/acadgild/project/batch1/mob/file.txt
-rw-r--r--  1 cloudera supergroup  1267 2017-09-05 12:13 /user/acadgild/project/batch1/mob/file.txt~
```

U106,S205,A388,1452457462,1452457462,1483978162,AP,ST407,2,1,1  
U114,S209,A383,1452406356,1452457462,1483978162,U,ST411,2,1,0  
U113,S203,A384,1452406356,1452406356,1452457462,U,ST405,0,0,1  
U108,S200,A382,1452418689,1452457462,1452418689,U,ST414,0,0,1  
U102,S203,A385,1452406356,1452406356,1483978162,U,ST404,2,0,0  
,S208,A368,1452406356,1483978162,1452406356,U,ST411,1,0,1  
U115,S200,A380,1452406356,1483978162,1452406356,AU,ST404,3,0,0  
U111,S204,A380,1452406356,1452406356,1452418689,U,ST418,3,1,1  
U120,S201,A380,1483978162,1452406356,1452418689,,ST410,3,0,1  
U113,S203,,1452406356,1452406356,1452406356,A,ST402,1,1,0  
U109,S203,A384,1452457462,1483978162,1452418689,E,ST405,1,1,1  
U110,S202,A383,1483978162,1483978162,1452418689,AU,ST402,2,1,0  
U100,S200,A381,1483978162,1483978162,1483978162,AP,ST410,3,1,1  
U101,S208,A380,1452457462,1452418689,1452457462,E,ST408,0,1,1  
U106,S206,A388,1483978162,1452406356,1452457462,A,ST405,3,1,0  
U107,S202,A384,1483978162,1452418689,1452457462,U,ST409,0,0,0  
U103,S204,A380,1452418689,1483978162,1452406356,AU,ST411,2,1,0  
U103,S202,A380,1452406356,1452406356,1452406356,A,ST415,2,1,1  
U113,S203,A383,1452457462,1452418689,1483978162,U,ST408,2,0,0  
U113,S204,A381,1483978162,1483978162,1452406356,E,ST415,3,0,1  
U106,S205,A388,1452457462,1452457462,1483978162,AP,ST407,2,1,1  
U114,S209,A383,1452406356,1452457462,1483978162,U,ST411,2,1,0  
U113,S203,A384,1452406356,1452406356,1452457462,U,ST405,0,0,1  
U108,S200,A382,1452418689,1452457462,1452418689,U,ST414,0,0,1  
U102,S203,A385,1452406356,1452406356,1483978162,U,ST404,2,0,0  
,S208,A368,1452406356,1483978162,1452406356,U,ST411,1,0,1  
U115,S200,A380,1452406356,1483978162,1452406356,AU,ST404,3,0,0  
U111,S204,A380,1452406356,1452406356,1452418689,U,ST418,3,1,1  
U120,S201,A380,1483978162,1452406356,1452418689,,ST410,3,0,1  
U113,S203,,1452406356,1452406356,1452406356,A,ST402,1,1,0

The script dataformatting.sh produces output in Hive-

```
Time taken: 0.020 seconds, Fetched: 2 row(s)
```

```
hive> select * from formatted input;
```

```
OK
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| U114 | S207 | A303 | 1465130523 | 1465230523 | 1475130523 | A | ST415 | 3 | 1 | 0 | 1 |
| U107 | S202 | A303 | 1495130523 | 1465230523 | 1465230523 | U | ST415 | 0 | 1 | 1 | 1 |
| U100 | S204 | A302 | 1495130523 | 1475130523 | 1465130523 | AU | ST408 | 2 | 1 | 1 | 1 |
| U104 | S202 | A303 | 1465230523 | 1475130523 | 1465130523 | A | ST409 | 2 | 0 | 1 | 1 |
| U102 | S207 | A301 | 1465230523 | 1485130523 | 1465230523 | AU | ST403 | 3 | 1 | 1 | 1 |
| S203 | A302 | 1495130523 | 1475130523 | 1465230523 | E | ST400 | 0 | 0 | 1 | 1 |
| U106 | S202 | A302 | 1465230523 | 1465130523 | 1465130523 | AU | ST408 | 0 | 1 | 1 | 1 |
| U105 | S207 | A300 | 1465230523 | 1485130523 | 1465130523 | U | ST400 | 2 | 0 | 1 | 1 |
| U108 | S205 | A304 | 1465130523 | 1475130523 | 1465130523 | ST410 | 2 | 1 | 0 | 1 |
| U105 | S203 | 1475130523 | 1465230523 | 1465130523 | AU | ST408 | 2 | 0 | 1 | 1 |
| U110 | S203 | A300 | 1465230523 | 1465130523 | 1485130523 | A | ST415 | 0 | 1 | 1 | 1 |
| U113 | S200 | A303 | 1465230523 | 1475130523 | 1465130523 | E | ST413 | 3 | 1 | 1 | 1 |
| U119 | S208 | A302 | 1495130523 | 1465230523 | 1465230523 | U | ST415 | 3 | 0 | 0 | 1 |
| U118 | S208 | A303 | 1475130523 | 1465130523 | 1465230523 | E | ST415 | 3 | 0 | 0 | 1 |
| U107 | S210 | A302 | 1475130523 | 1485130523 | 1485130523 | AP | ST404 | 2 | 1 | 0 | 1 |
| U118 | S202 | A300 | 1495130523 | 1465230523 | 1465230523 | AP | ST410 | 1 | 0 | 0 | 1 |
| U111 | S206 | A305 | 1465130523 | 1485130523 | 1485130523 | AU | ST415 | 0 | 1 | 1 | 1 |
| U116 | S208 | A305 | 1465230523 | 1485130523 | 1475130523 | A | ST413 | 1 | 0 | 1 | 1 |
| U101 | S202 | A300 | 1465230523 | 1465130523 | 1475130523 | U | ST401 | 0 | 0 | 1 | 1 |
| U120 | S206 | A303 | 1495130523 | 1485130523 | 1465130523 | AU | ST414 | 0 | 0 | 0 | 1 |
| U106 | S205 | A300 | 1452457462 | 1452457462 | 1483978162 | AP | ST407 | 2 | 1 | 1 | 1 |
| U114 | S209 | A303 | 1452406356 | 1452457462 | 1483978162 | U | ST411 | 2 | 1 | 0 | 1 |
| U113 | S203 | A304 | 1452406356 | 1452406356 | 1452457462 | U | ST405 | 0 | 0 | 1 | 1 |
| U108 | S200 | A302 | 1452418689 | 1452457462 | 1452418689 | U | ST414 | 0 | 0 | 1 | 1 |
| U102 | S203 | A305 | 1452406356 | 1452406356 | 1483978162 | U | ST404 | 2 | 0 | 0 | 1 |
| S208 | A300 | 1452406356 | 1483978162 | 1452406356 | U | ST411 | 1 | 0 | 1 | 1 |
| U115 | S200 | A300 | 1452406356 | 1483978162 | 1452406356 | AU | ST404 | 3 | 0 | 0 | 1 |
| U111 | S204 | A300 | 1452406356 | 1452406356 | 1452418689 | U | ST410 | 3 | 1 | 1 | 1 |
| U120 | S201 | A300 | 1483978162 | 1452406356 | 1452418689 | ST410 | 3 | 0 | 1 | 1 |
| U113 | S203 | 1452406356 | 1452406356 | 1452406356 | A | ST402 | 1 | 1 | 0 | 1 |
| U109 | S203 | A304 | 1452457462 | 1483978162 | 1452418689 | E | ST405 | 1 | 1 | 1 | 1 |
| U110 | S202 | A303 | 1483978162 | 1483978162 | 1452418689 | AU | ST402 | 2 | 1 | 0 | 1 |
| U100 | S200 | A301 | 1483978162 | 1483978162 | 1483978162 | AP | ST410 | 3 | 1 | 1 | 1 |
| U101 | S208 | A300 | 1452457462 | 1452418689 | 1452457462 | E | ST408 | 0 | 1 | 1 | 1 |
| U106 | S206 | A300 | 1483978162 | 1452406356 | 1452457462 | A | ST405 | 3 | 1 | 0 | 1 |
| U107 | S202 | A304 | 1483978162 | 1452418689 | 1452457462 | U | ST409 | 0 | 0 | 0 | 1 |
| U103 | S204 | A300 | 1452418689 | 1483978162 | 1452406356 | AU | ST411 | 2 | 1 | 0 | 1 |
| U103 | S202 | A300 | 1452406356 | 1452406356 | 1452406356 | A | ST415 | 2 | 1 | 1 | 1 |
| U113 | S203 | A303 | 1452457462 | 1452418689 | 1483978162 | U | ST408 | 2 | 0 | 0 | 1 |
| U113 | S204 | A301 | 1483978162 | 1483978162 | 1452406356 | E | ST415 | 3 | 0 | 1 | 1 |

```

- In the above screenshot we can see the formatted input data with some null values in user\_id, artist\_id and geo\_cd columns which we will fill the enrichment script based on rules of enrichment for artist\_id and geo\_cd only. We will get neglect user\_id because they didn't mentioned anything about user\_id for enrichment purpose.
- Data Formatting phase is executed successfully by loading both mobile and web data and partitioned based on batchid.

### e. Data Enrichment

In this phase we will enrich the data coming from web and mobile applications using the lookup table stored in Hbase and divide the records based on the enrichment rules into 'pass' and 'fail' records.

Rules for data enrichment

- If any of like or dislike is NULL or absent, consider it as 0.
- If fields like Geo\_cd and Artist\_id are NULL or absent, consult the lookup tables for fields Station\_id and Song\_id respectively to get the values of Geo\_cd and Artist\_id.
- If corresponding lookup entry is not found, consider that record to be invalid.

So based on the enrichment rules we will fill the null geo\_cd and artist\_id values with the help of corresponding lookup values in song-artist-map and station-geo-map tables in Hive-Hbase tables.

```
#!/bin/bash
batchid=`cat /home/cloudera/acadgild/project/logs/current-batch.txt`
LOGFILE=/home/cloudera/acadgild/project/logs/log_batch_$batchid
VALIDDIR=/home/cloudera/acadgild/project/processed_dir/valid/batch_$batchid
INVALIDDIR=/home/cloudera/acadgild/project/processed_dir/invalid/batch_$batchid

echo "Running hive script for data enrichment and filtering..." >> $LOGFILE

hive -hiveconf batchid=$batchid -f /home/cloudera/acadgild/project/scripts/data_enrichment.hql

if [ ! -d "$VALIDDIR" ]
then
mkdir -p "$VALIDDIR"
fi

if [ ! -d "$INVALIDDIR" ]
then
mkdir -p "$INVALIDDIR"
fi

echo "Copying valid and invalid records in local file system..." >> $LOGFILE

hadoop fs -get /user/hive/warehouse/project.db/enriched_data/batchid=$batchid/status=pass/* $VALIDDIR
hadoop fs -get /user/hive/warehouse/project.db/enriched_data/batchid=$batchid/status=fail/* $INVALIDDIR

echo "Deleting older valid and invalid records from local file system..." >> $LOGFILE

find /home/cloudera/acadgild/project/processed_dir/ -mtime +7 -exec rm {} \;
```

```

SET hive.auto.convert.join=false;
SET hive.exec.dynamic.partition.mode=nonstrict;

USE project;

CREATE TABLE IF NOT EXISTS enriched_data
(
User_id STRING,
Song_id STRING,
Artist_id STRING,
Timestamp STRING,
Start_ts STRING,
End_ts STRING,
Geo_cd STRING,
Station_id STRING,
Song_end_type INT,
Like INT,
Dislike INT
)
PARTITIONED BY (batchid INT,status STRING)
STORED AS ORC;

INSERT OVERWRITE TABLE enriched_data
PARTITION (batchid,status)
SELECT
i.user_id,
i.song_id,
IF(i.artist_id is NULL OR i.artist_id='',sa.artist_id,i.artist_id) AS artist_id,
i.timestamp,
i.start_ts,
i.end_ts,
IF(i.geo_cd is NULL OR i.geo_cd='',sg.geo_cd,i.geo_cd) AS geo_cd,
i.station_id,
IF (i.song_end_type IS NULL,3,i.song_end_type) AS song_end_type,
IF (i.like IS NULL,0,i.like) AS like,
IF (i.dislike IS NULL,0,i.dislike) AS dislike,
i.batchid,
IF((i.like=1 AND i.dislike=1)
OR i.user_id IS NULL
OR i.song_id IS NULL
OR i.timestamp IS NULL
OR i.start_ts IS NULL
OR i.end_ts IS NULL
OR i.user_id=''
OR i.song_id=''

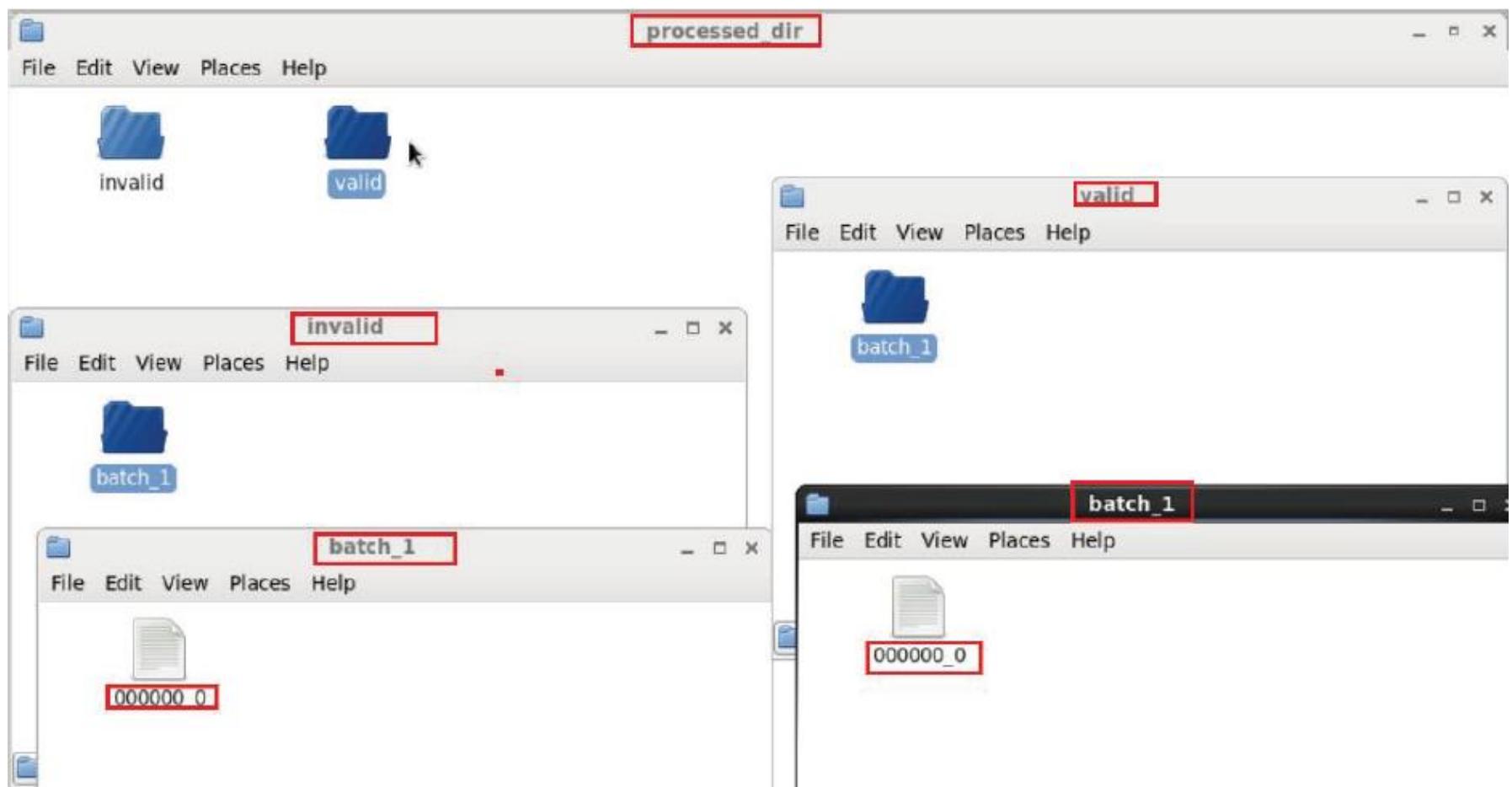
```

```
[cloudera@quickstart scripts]$ ./data_enrichment.sh

Logging initialized using configuration in file:/etc/hive/conf.dist/hive-log4j.properties
OK
Time taken: 0.391 seconds
OK
Time taken: 0.178 seconds
Query ID = cloudera_20170905233838_f151d27a-601b-405e-9313-6d9df9bd18f2
Total jobs = 2
Launching Job 1 out of 2
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 hive.exec.reducers.bytes.per.reducer=<number>

Loading data to `table project.enriched_data partition (batchid=null, status=null)
  Time taken for load dynamic partitions : 222
  Loading partition {batchid=1, status=fail}
  Loading partition {batchid=1, status=pass}
  Time taken for adding to write entity : 2
Partition project.enriched_data{batchid=1, status=fail} stats: [numFiles=1, numRows=18, totalSize=1481, rawDataSize=12435]
Partition project.enriched_data{batchid=1, status=pass} stats: [numFiles=1, numRows=22, totalSize=1487, rawDataSize=16126]
MapReduce Jobs Launched:
Stage-Stage-1: Map: 3  Reduce: 1  Cumulative CPU: 5.5 sec  HDFS Read: 45891 HDFS Write: 3097 SUCCESS
Stage-Stage-2: Map: 2  Reduce: 1  Cumulative CPU: 4.58 sec  HDFS Read: 21135 HDFS Write: 3135 SUCCESS
Total MapReduce CPU Time Spent: 10 seconds 80 msec
OK
Time taken: 63.499 seconds
WARN: The method class org.apache.commons.logging.impl.SLF4JLogFactory#release() was invoked.
WARN: Please see http://www.slf4j.org/codes.html#release for an explanation.
```

So here in above script we are dividing the records based on status pass and fail and dump the result into processed\_dir folder with valid and invalid folders



Below screenshot shows that data has been loaded in hive-

```
hive> show tables;
OK
enriched_data
formatted_input
song_artist_map
station_geo_map
subscribed_users
users_artists
Time taken: 0.034 seconds, Fetched: 6 row(s)
```

- In the below screenshot we have data for data enrichment table where we filled the null values of artist\_id and geo\_cd of formatted input with the help of lookup tables

```
Time taken: 0.214 seconds, Fetched: 7 row(s)
hive> select * from enriched_data;
OK
U113 S200 A303 1465230523 1475130523 1465130523 E ST413 3 1 1 1 1 fail
U100 S200 A301 1494342562 1494342562 1494342562 AP ST410 3 1 1 1 1 fail
U107 S202 A303 1495130523 1465230523 1465230523 U ST415 0 1 1 1 1 fail
U103 S202 A300 1465535556 1465535556 1465535556 A ST415 2 1 1 1 1 fail
U106 S202 A302 1465230523 1465130523 1465130523 AU ST408 0 1 1 1 1 fail
U109 S203 A304 1462908262 1494342562 1468139889 E ST405 1 1 1 1 1 fail
S203 A302 1495130523 1475130523 1465230523 E ST400 0 0 1 1 1 fail
U110 S203 A300 1465230523 1465130523 1485130523 A ST415 0 1 1 1 1 fail
U111 S204 A300 1465535556 1465535556 1468139889 U ST410 3 1 1 1 1 fail
U113 S204 A301 1494342562 1494342562 1465535556 E ST415 3 0 1 1 1 fail
U100 S204 A302 1495130523 1475130523 1465130523 AU ST408 2 1 1 1 1 fail
U106 S205 A300 1462908262 1462908262 1494342562 AP ST407 2 1 1 1 1 fail
U111 S206 A305 1465130523 1465130523 1485130523 AU ST415 0 1 1 1 1 fail
U114 S207 A303 1465130523 1465230523 1475130523 A ST415 3 1 0 1 1 fail
U102 S207 A301 1465230523 1485130523 1465230523 AU ST403 3 1 1 1 1 fail
S208 A300 1465535556 1494342562 1465535556 U ST411 1 0 1 1 1 fail
U118 S208 A303 1475130523 1465130523 1465230523 E ST415 3 0 0 1 1 fail
U119 S208 A302 1495130523 1465230523 1465230523 U ST415 3 0 0 1 1 fail
U101 S208 A300 1462908262 1468139889 1462908262 E ST408 0 1 1 1 1 fail
U107 S210 A302 1475130523 1485130523 1485130523 AP ST404 2 1 0 1 1 fail
U115 S200 A300 1465535556 1494342562 1465535556 AU ST404 3 0 0 1 1 pass
U108 S200 A302 1468139889 1462908262 1468139889 U ST414 0 0 1 1 1 pass
U120 S201 A300 1494342562 1465535556 1468139889 A ST410 3 0 1 1 1 pass
U107 S202 A304 1494342562 1468139889 1462908262 U ST409 0 0 0 1 1 pass
U101 S202 A300 1465230523 1465130523 1475130523 U ST401 0 0 1 1 1 pass
U110 S202 A303 1494342562 1494342562 1468139889 AU ST402 2 1 0 1 1 pass
U118 S202 A300 1495130523 1465230523 1465230523 AP ST410 1 0 0 1 1 pass
U104 S202 A303 1465230523 1475130523 1465130523 A ST409 2 0 1 1 1 pass
U102 S203 A305 1465535556 1465535556 1494342562 U ST404 2 0 0 1 1 pass
U113 S203 A304 1465535556 1465535556 1462908262 U ST405 0 0 1 1 1 pass
U113 S203 A303 1462908262 1468139889 1494342562 U ST408 2 0 0 1 1 pass
U105 S203 A303 1475130523 1465230523 1465130523 AU ST408 2 0 1 1 1 pass
U113 S203 A303 1465535556 1465535556 1465535556 A ST402 1 1 0 1 1 pass
U103 S204 A300 1468139889 1494342562 1465535556 AU ST411 2 1 0 1 1 pass
U108 S205 A304 1465130523 1465130523 1475130523 A ST410 2 1 0 1 1 pass
U106 S206 A300 1494342562 1465535556 1462908262 A ST405 3 1 0 1 1 pass
U120 S206 A303 1495130523 1485130523 1465130523 AU ST414 0 0 0 1 1 pass
U105 S207 A300 1465230523 1485130523 1465130523 U ST400 2 0 1 1 1 pass
U116 S208 A303 1465230523 1485130523 1475130523 A ST413 1 0 1 1 1 pass
U114 S209 A303 1465535556 1462908262 1494342562 U ST411 2 1 0 1 1 pass
Time taken: 0.419 seconds. Fetched: 48 row(s)
```

#### f. Data Analysis using Spark-

In this stage we will do analysis on enriched data using Spark SQL and run the program using **Spark-Submit** command.

Before running the spark-submit command we have to **zip -d command** to remove the bad manifests in created spark project jar file to avoid the invalid Signature exception.

We have used two spark-submits for analysis-

- 1) Spark\_analysis for creating tables for each query/problem statement.
- 2) Spark\_analysis\_2 for displaying results for each query in terminal.

```
#!/bin/bash
|
batchid=`cat /home/acadgild/project/logs/current-batch.txt`

LOGFILE=/home/acadgild/project/logs/log_batch_$batchid

echo "Running script for data analysis using spark..." >> $LOGFILE
chmod 775 /home/acadgild/project/Spark_Practice-1/out/artifacts/Spark_practice_1_jar/Spark_practice-1.jar

zip -d /home/acadgild/project/Spark_Practice-1/out/artifacts/Spark_practice_1_jar/Spark_practice-1.jar META-INF/*.DSA META-INF/*.RSA META-INF/*.SF

spark-submit \
--class Spark_analysis \
--master local[2] \
--driver-class-path /usr/local/hive/lib/hive-hbase-handler-0.14.0.jar:/usr/local/hbase/lib/* \
/home/acadgild/project/Spark_Practice-1/out/artifacts/Spark_practice_1_jar/Spark_practice-1.jar $batchid

spark-submit \
--class Spark_analysis_2 \
--master local[2] \
--driver-class-path /usr/local/hive/lib/hive-hbase-handler-0.14.0.jar:/usr/local/hbase/lib/* \
/home/acadgild/project/Spark_Practice-1/out/artifacts/Spark_practice_1_jar/Spark_practice-1.jar $batchid

echo "Exporting data to MYSQL using sqoop export..." >> $LOGFILE
sh /home/acadgild/project/scripts/data_export.sh

echo "Incrementing batchid..." >> $LOGFILE
batchid= `expr $batchid + 1`
echo -n $batchid > /home/acadgild/project/logs/current-batch.txt
```

Spark code-

We have one scala file for creating tables as per query-

```
1  import org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe
2  import org.apache.spark.sql.SparkSession
3
4  object Spark_analysis {
5
6    def main(args: Array[String]): Unit = {
7      val sparkSession = SparkSession.builder()
8        .master("local[2]")
9        .appName("Data Analysis Main_1")
10       .config("spark.sql.warehouse.dir","/user/hive/warehouse")
11       .config("hive.metastore.uris","thrift://127.0.0.1:9083")
12       .enableHiveSupport()
13       .getOrCreate()
14
15      val batchid = args(0)
16
17      //<<<<<----- PROBLEM 1 - Creation of table and Insertion of data ----->>>>>>
18      //Determine top 10 station_id(s) where maximum number of songs were played, which were liked by unique users.
19
20      val set_properties = sparkSession.sqlContext.sql("set hive.auto.convert.join=false")
21
22      val use_project_database = sparkSession.sqlContext.sql("USE project")
23
24      val create_hive_table_top_10_stations = sparkSession.sqlContext.sql("CREATE TABLE IF NOT EXISTS project.top_10_stations"+
25        "("+
26        " station_id STRING,"+
27        " total_distinct_songs_played INT,"+
28        " distinct_user_count INT"+
29        ")" +
30        " PARTITIONED BY (batchid INT)" +
31        " ROW FORMAT DELIMITED" +
32        " FIELDS TERMINATED BY ','" +
33        " STORED AS TEXTFILE")
34
35
36      val insert_into_top_10_stations = sparkSession.sqlContext.sql("INSERT OVERWRITE TABLE project.top_10_stations"+
37        " PARTITION (batchid=$batchId)" +
38        " SELECT" +
39        " station_id," +
40        " COUNT(DISTINCT song_id) AS total_distinct_songs_played," +
41        " COUNT(DISTINCT user_id) AS distinct_user_count" +
```

```

16
17 //<<<<<----- PROBLEM 1 - Creation of table and Insertion of data ----->>>>>>>
18 //Determine top 10 station_id(s) where maximum number of songs were played, which were liked by unique users.
19
20 val set_properties = sparkSession.sqlContext.sql("set hive.auto.convert.join=false")
21
22 val use_project_database = sparkSession.sqlContext.sql("USE project")
23
24 val create_hive_table_top_10_stations = sparkSession.sqlContext.sql("CREATE TABLE IF NOT EXISTS project.top_10_stations"+
25 " ("+
26 " station_id STRING,"+
27 " total_distinct_songs_played INT,"+
28 " distinct_user_count INT"+
29 ")"+
30 " PARTITIONED BY (batchid INT)" +
31 " ROW FORMAT DELIMITED" +
32 " FIELDS TERMINATED BY ','" +
33 " STORED AS TEXTFILE")
34
35
36 val insert_into_top_10_stations = sparkSession.sqlContext.sql("INSERT OVERWRITE TABLE project.top_10_stations"+
37 "s" PARTITION (batchid=$batchId)" +
38 " SELECT" +
39 " station_id," +
40 " COUNT(DISTINCT song_id) AS total_distinct_songs_played," +
41 " COUNT(DISTINCT user_id) AS distinct_user_count" +
42 " FROM project.enriched_data" +
43 " WHERE status='pass'" +
44 "s" AND (batchid=$batchId)" +
45 " AND like=1" +
46 " GROUP BY station_id" +
47 " ORDER BY total_distinct_songs_played DESC" +
48 " LIMIT 10")
49
50 //<<<<<----- PROBLEM 2 - Creation of table and Insertion of data ----->>>>>>>
51 //Determine total duration of songs played by each type of user, where type of user can be 'subscribed' or 'unsubscribed'.
52 An unsubscribed user is the one whose record is either not present in Subscribed_users lookup table or has subscription_end_date
53 earlier than the timestamp of the song played by him.*/
54
55 val create_hive_table_song_duration = sparkSession.sqlContext.sql("CREATE TABLE IF NOT EXISTS project.song_duration"+
56 " ("+
57 " user_id STRING,"+
58 " user_type STRING,"+
59 " song_id STRING,"+
60 " artist_id STRING,"+
61 " total_duration_in_minutes DOUBLE" +
62 ")" +
63 " PARTITIONED BY (batchid INT)" +
64 " ROW FORMAT DELIMITED" +
65 " FIELDS TERMINATED BY ','" +
66 " STORED AS TEXTFILE")
67
68
69 val insert_into_song_duration = sparkSession.sqlContext.sql("INSERT OVERWRITE TABLE project.song_duration"+
70 "s" PARTITION (batchid=$batchId)" +
71 " SELECT" +
72 " e.user_id" +
73 " IF(e.user_id!=s.user_id" +
74 " OR (CAST(s.subscription_end_dt as BIGINT) < CAST(e.start_ts as BIGINT)), 'unsubscribed', 'subscribed') AS user_type" +
75 " e.song_id" +
76 " e.artist_id" +
77 " (cast(e.end_ts as BIGINT)-cast(e.start_ts as BIGINT))/60 AS total_duration_in_minutes" +
78 " FROM project.enriched_data e" +
79 " LEFT OUTER JOIN project.subscribed_users s" +
80 " ON e.user_id=s.user_id" +
81 " WHERE e.status='pass'" +
82 "s" AND (batchid=$batchId)")
83
84 //<<<<<----- PROBLEM 3 - Creation of table and Insertion of data ----->>>>>>>
85 //Determine top 10 connected artists.
86 //Connected artists are those whose songs are most listened by the unique users who follow them.
87
88
89 val create_hive_table_top_10_connected_artists = sparkSession.sqlContext.sql("CREATE TABLE IF NOT EXISTS project.connected_artists"+
90 " ("+
91 " artist_id STRING,"+
92 " total_distinct_songs INT,"+
93 " unique_followers INT"+
94 ")" +
95 " PARTITIONED BY (batchid INT)" +
96 " ROW FORMAT DELIMITED" +
97 " FIELDS TERMINATED BY ','" +
98 " STORED AS TEXTFILE")
99
100
101 val insert_into_top_10_connected_artists = sparkSession.sqlContext.sql("INSERT OVERWRITE TABLE project.connected_artists"+
102 "s" PARTITION (batchid=$batchId)" +
103 " SELECT" +
104 " artist_id," +
105 " COUNT(DISTINCT song_id) AS total_distinct_songs," +
106 " COUNT(DISTINCT user_id) AS unique_followers" +
107 " FROM project.enriched_data" +
108 " WHERE status='pass'" +
109 "s" AND (batchid=$batchId)" +
110 " GROUP BY artist_id" +
111 " ORDER BY unique_followers desc,total_distinct_songs desc" +
112 " LIMIT 10")
113

```

```

115 //<<<<<----- PROBLEM 4 - Creation of table and Insertion of data ----->>>>>
116 //Determine top 10 songs who have generated the maximum revenue.
117 //NOTE: Royalty applies to a song only if it was liked or was completed successfully or both.
118
119 val create_hive_table_top_10_songs_maxrevenue = sparkSession.sqlContext.sql("CREATE TABLE IF NOT EXISTS project.top_10_songs_maxrevenue"+
120   " ("+
121     " song_id STRING,"+
122     " artist_id STRING,"+
123     " total_duration_in_minutes DOUBLE"'+
124     " )"+
125     " PARTITIONED BY (batchid INT)"'+
126     " ROW FORMAT DELIMITED"'+
127     " FIELDS TERMINATED BY ','"'+
128     " STORED AS TEXTFILE")
129
130
131 val insert_into_top_10_songs_maxrevenue = sparkSession.sqlContext.sql("INSERT OVERWRITE TABLE project.top_10_songs_maxrevenue"+
132   " s" PARTITION (batchid=$batchId)" +
133   " SELECT" +
134     " song_id,"+
135     " artist_id,"+
136     " (cast(end_ts as BIGINT)-cast(start_ts as BIGINT))/60 AS total_duration_in_minutes" +
137     " FROM project.enriched_data" +
138     " WHERE status='pass'" +
139     " s" AND (batchid=$batchId)" +
140     " AND (like=1 OR song_end_type=0 OR (like=1 and song_end_type=0))" +
141     " ORDER BY total_duration_in_minutes desc" +
142     " LIMIT 10")
143
144
145 //<<<<<----- PROBLEM 5 - Creation of table and Insertion of data ----->>>>>
146 //Determine top 10 unsubscribed users who listened to the songs for the longest duration.
147
148 val create_hive_table_top_10_unsubscribed_users = sparkSession.sqlContext.sql("CREATE TABLE IF NOT EXISTS project.top_10_unsubscribed_users"+
149   " ("+
150     " user_id STRING,"+
151     " song_id STRING,"+
152     " artist_id STRING,"+
153     " total_duration_in_minutes DOUBLE"'+
154     " )"+
155     " PARTITIONED BY (batchid INT)"'+
156     " ROW FORMAT DELIMITED"'+
157     " FIELDS TERMINATED BY ','"'+
158     " STORED AS TEXTFILE")
159
160
161 val insert_into_unsubscribed_users = sparkSession.sqlContext.sql("INSERT OVERWRITE TABLE project.top_10_unsubscribed_users"+
162   " s" PARTITION (batchid=$batchId)" +
163   " SELECT" +
164     " user_id,"+
165     " song_id,"+
166     " artist_id,"+
167     " total_duration_in_minutes" +
168     " FROM project.song_duration" +
169     " WHERE user_type='unsubscribed'" +
170     " AND total_duration_in_minutes>=0" +
171     " s" AND (batchid=$batchId)" +
172     " ORDER BY total_duration_in_minutes desc" +
173     " LIMIT 10")
174

```

This second Scala file is for displaying results of each query in the terminal.

```

1 import ...
2 object Spark_analysis_2 {
3   def main(args: Array[String]): Unit = {
4     val sparkSession = SparkSession.builder.master("local").appName("Spark Session example")
5       .config("spark.sql.warehouse.dir", "/user/hive/warehouse")
6       .config("hive.metastore.uris", "thrift://localhost:9083")
7       .enableHiveSupport().getOrCreate()
8     val batchId = args(0)
9
10
11     sparkSession.sqlContext.sql("USE project")
12     sparkSession.sqlContext.sql("SELECT station_id from top_10_stations").show()
13     sparkSession.sqlContext.sql("SELECT user_type,total_duration_in_minutes from song_duration").show()
14     sparkSession.sqlContext.sql("SELECT artist_id from connected_artists").show()
15     sparkSession.sqlContext.sql("SELECT song_id from top_10_songs_maxrevenue").show()
16     sparkSession.sqlContext.sql("SELECT user_id from top_10_unsubscribed_users").show()
17
18   }
19 }
20

```

Now in order to run we will follow below steps-

- We will create jar for the above source code using File -> project structure -> Artifacts -> create jar using Build Artifacts command.
- Then we can observe **out** folder getting created with artifacts inside it.
- Now we will export this jar to VM where we will run using spark submit command and get the analysis result.
- When we run the Data Analysis First it will submit **Spark\_analysis** file and then it will run the **Spark-analysis\_2** file which displays the result for each query.

## Output-

```
[acadgild@localhost scripts]$ ./DataAnalysis.sh
      zip warning: name not matched: META-INF/*.RSA
deleting: META-INF/DUMMY.SF
deleting: META-INF/DUMMY.DSA
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/local/hbase/lib/slf4j-log4j12-1.6.4.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/acadgild/Downloads/spark-2.0.0-bin-hadoop2.6/jars/slf4j-log4j12-1.7.16.jar!/org/slf4
/impl/StaticLoggerBinder.class]
```

We can see below tables being created in HIVE-

```
USE project
2017-09-10 16:57:07,332 INFO  [main] execution.SparkSqlParser: Parsing command:
CREATE TABLE IF NOT EXISTS project.top_10_stations( station_id STRING, total_distinct_songs_played INT, distinct_user_count INT) PARTITIONED BY (batchid INT) ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' STORED AS TEXTFILE
2017-09-10 16:57:08,926 INFO  [main] execution.SparkSqlParser: Parsing command:
INSERT OVERWRITE TABLE project.top_10_stations PARTITION (batchid= 1) SELECT station_id, COUNT(DISTINCT song_id) AS total_distinct_songs_played, COUNT(DISTINCT user_id) AS distinct_user_count FROM project.enriched_data WHERE status='pass' AND (batchid= 1) AND like=1 GROUP BY station_id ORDER BY total_distinct_songs_played DESC LIMIT 10

CREATE TABLE IF NOT EXISTS project.top_10_unsubscribed_users( user_id STRING, song_id STRING, artist_id STRING, total_duration_in_minutes DOUBLE) PARTITIONED BY (batchid INT) ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' STORED AS TEXTFILE
2017-09-10 16:59:00,057 INFO  [main] execution.SparkSqlParser: Parsing command:
INSERT OVERWRITE TABLE project.top_10_unsubscribed_users PARTITION (batchid= 1) SELECT user_id, song_id, artist_id, total_duration_in_minutes FROM project.song_duration WHERE user_type='unsubscribed' AND total_duration_in_minutes>=0 AND (batchid= 1) ORDER BY total_duration_in_minutes desc LIMIT 10
```

## Spark analysis 2 output:

Query-1: Determine top 10 station\_id(s) where maximum number of songs were played, which were liked by unique users.

```
+-----+
|station_id|
+-----+
| ST402 |
| ST411 |
| ST405 |
| ST410 |
+-----+
```

Query-2: Determine total duration of songs played by each type of user, where type of user can be 'subscribed' or 'unsubscribed'. An unsubscribed user is the one whose record is either not present in Subscribed\_users lookup table or has subscription\_end\_date earlier than the timestamp of the song played by him

```
+-----+-----+
| user_type|total_song_duration|
+-----+-----+
| subscribed| 1904665.650000001|
+-----+-----+
```

**Query-3:** Determine top 10 connected artists. Connected artists are those whose songs are most listened by the unique users who follow them

artist_id
A300
A303
A304
A305
A302

**Query-4:** Determine top 10 songs who have generated the maximum revenue. Royalty applies to a song only if it was liked or was completed successfully or both.

song_id
S209
S202
S205
S200
S203
S203
S206
S202
S206
S202

**Query-5:** Determine top 10 unsubscribed users who listened to the songs for the longest duration.

: Removed TaskSet 7.0, whose tasks have all completed, from pool
user_id

For this query we got no result because there is no unsubscribed user whose has got positive total duration\_minutes\_in\_minutes. All the unsubscribed users got negative total\_duration means their end timestamp is less than the start time stamp of the song which is not correct.

We can see output of above queries being stored in HIVE tables-

```
hive> show tables;
OK
enriched_data
formatted_input
song_artist_map
station_geo_map
subscribed_users
users_artists

Time taken: 0.04 seconds, Fetched: 6 row(s)
hive> show tables;
OK
connected_artists
enriched_data
formatted_input
song_artist_map
song_duration
station_geo_map
subscribed_users
top_10_songs_maxrevenue
top_10_stations
top_10_unsubscribed_users
users_artists

Time taken: 0.047 seconds, Fetched: 11 row(s)
hive> █
```

```
hive> select * from song_duration;
OK
U115    subscribed      S200    A300    -480116.76666666666   1
U108    subscribed      S200    A302    87193.78333333334   1
U120    subscribed      S201    A300    43405.55      1
U107    unsubscribed    S202    A304    -87193.78333333334   1
U101    subscribed      S202    A300    166666.6666666666   1
U110    unsubscribed    S202    A303    -436711.2166666667  1
U118    subscribed      S202    A300    0.0       1
U104    subscribed      S202    A303    -166666.6666666666   1
U102    subscribed      S203    A305    480116.76666666666   1
U113    subscribed      S203    A304    -43788.23333333333  1
U113    subscribed      S203    A303    436711.2166666667  1
U105    subscribed      S203    A303    -1666.666666666667  1
U113    subscribed      S203    A303    0.0       1
U103    unsubscribed    S204    A300    -480116.76666666666  1
U108    subscribed      S205    A304    166666.6666666666   1
U106    subscribed      S206    A300    -43788.23333333333  1
U120    subscribed      S206    A303    -333333.3333333333  1
U105    unsubscribed    S207    A300    -333333.3333333333  1
U116    subscribed      S208    A303    -166666.66666666666  1
U114    subscribed      S209    A303    523905.0      1
Time taken: 0.048 seconds, Fetched: 20 row(s)
hive> select * from top_10_stations;
OK
ST402    2      2      1
ST411    2      2      1
ST405    1      1      1
ST410    1      1      1
Time taken: 0.061 seconds, Fetched: 4 row(s)
hive> █
```

```

hive> select * from top_10_songs_maxrevenue;
OK
S209    A303      523905.0      1
S202    A300      166666.6666666666  1
S205    A304      166666.6666666666  1
S200    A302      87193.7833333334  1
S203    A303      0.0      1
S203    A304      -43788.2333333333  1
S206    A300      -43788.2333333333  1
S202    A304      -87193.7833333334  1
S206    A303      -333333.3333333333  1
S202    A303      -436711.2166666667  1
Time taken: 0.063 seconds, Fetched: 10 row(s)
hive> select * from connected_artist;
FAILED: SemanticException [Error 10001]: Line 1:14 Table not found 'connected_artist'
hive> select * from connected_artists;
OK
A300    6      7      1
A303    5      7      1
A304    3      3      1
A305    1      1      1
A302    1      1      1
Time taken: 0.062 seconds, Fetched: 5 row(s)
hive> select * from top_10_unsubscribed_users;
OK
Time taken: 0.057 seconds
hive> █

```

#### **g. Data Export to MySQL**

In this stage data will be exported from hive warehouse directory to MYSQL database using **data\_export.sh** command mentioned in the **DataAnalysis.sh** script.

```

#!/bin/bash
batchid=`cat /home/acadgild/project/logs/current-batch.txt`
LOGFILE=/home/acadgild/project/logs/log_batch_$batchid
echo "Running script for data analysis using spark..." >> $LOGFILE
chmod 775 /home/acadgild/project/Spark_Practice-1/out/artifacts/Spark_practice_1_jar/Spark_practice-1.jar
zip -d /home/acadgild/project/Spark_Practice-1/out/artifacts/Spark_practice_1_jar/Spark_practice-1.jar META-INF/*.DSA META-INF/*.RSA META-INF/*.SF
spark-submit \
--class Spark_analysis \
--master local[2] \
--driver-class-path /usr/local/hive/lib/hive-hbase-handler-0.14.0.jar:/usr/local/hbase/lib/* \
/home/acadgild/project/Spark_Practice-1/out/artifacts/Spark_practice_1_jar/Spark_practice-1.jar $batchid
spark-submit \
--class Spark_analysis_2 \
--master local[2] \
--driver-class-path /usr/local/hive/lib/hive-hbase-handler-0.14.0.jar:/usr/local/hbase/lib/* \
/home/acadgild/project/Spark_Practice-1/out/artifacts/Spark_practice_1_jar/Spark_practice-1.jar $batchid
echo "Exporting data to MySQL using sqoop export" >> $LOGFILE
sh /home/acadgild/project/scripts/data_export.sh
echo "Incrementing batchid..." >> $LOGFILE
batchid=`expr $batchid + 1`
echo -n $batchid > /home/acadgild/project/logs/current-batch.txt

```

```

#!/bin/bash
batchid=`cat /home/acadgild/project/logs/current-batch.txt`
LOGFILE=/home/acadgild/project/logs/log_batch_$batchid
echo "Creating mysql tables if not present..." >> $LOGFILE
mysql -u root </home/acadgild/project/scripts/create_schema.sql
echo "Running sqoop job for data export..." >> $LOGFILE
sqoop export \
--connect jdbc:mysql://localhost/project \
--username 'root' \
--password \
--table 'top_10_stations' \
--export-dir '/user/hive/warehouse/project.db/top_10_stations/batchid=$batchid/part-00000' \
--input-fields-terminated-by ',' \
-m 1
sqoop export \
--connect jdbc:mysql://localhost/project \
--username 'root' \
--password \
--table 'song_duration' \
--export-dir '/user/hive/warehouse/project.db/song_duration/batchid=$batchid/part-00000' \
--input-fields-terminated-by ',' \
-m 1
sqoop export \
--connect jdbc:mysql://localhost/project \
--username 'root' \
--password \
--table 'connected_artists' \
--export-dir '/user/hive/warehouse/project.db/connected_artists/batchid=$batchid/part-00000' \
--input-fields-terminated-by ',' \
-m 1

```

```
sqoop export \
--connect jdbc:mysql://localhost/project \
--username 'root' \
--P \
--table 'top_10_songs_maxrevenue' \
--export-dir '/user/hive/warehouse/project.db/top_10_songs_maxrevenue/batchid=$batchid/part-00000' \
--input-fields-terminated-by ',' \
-m 1

sqoop export \
--connect jdbc:mysql://localhost/project \
--username 'root' \
--P \
--table 'top_10_unsubscribed_users' \
--export-dir '/user/hive/warehouse/project.db/top_10_unsubscribed_users/batchid=$batchid/part-00000' \
--input-fields-terminated-by ',' \
-m 1
```

Now after running above script we can see the data exported successfully into the MYSQL Database for all the 5 queries.

#### MYSQL:

```
mysql> use project;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> show tables;
+-----+
| Tables_in_project |
+-----+
| connected_artists |
| song_duration      |
| top_10_royalty_songs |
| top_10_stations    |
| top_10_unsubscribed_users |
+-----+
5 rows in set (0.00 sec)

mysql> select * from connected_artists;
+-----+-----+-----+
| artist_id | total_distinct_songs | user_count |
+-----+-----+-----+
| A300     | 6               | 7          |
| A303     | 5               | 7          |
| A304     | 3               | 3          |
| A305     | 1               | 1          |
| A302     | 1               | 1          |
+-----+-----+-----+
5 rows in set (0.05 sec)
```

```
mysql> select * from top_10_royalty_songs;
+-----+-----+
| song_id | artist_id | duration |
+-----+-----+
| S209   | A303     |      523905 |
| S202   | A300     | 166666.66666667 |
| S205   | A304     | 166666.66666667 |
| S200   | A302     | 87193.7833333333 |
| S203   | A303     |      0           |
| S203   | A304     | -43788.2333333333 |
| S206   | A300     | -43788.2333333333 |
| S202   | A304     | -87193.7833333333 |
| S206   | A303     | -333333.3333333333 |
| S202   | A303     | -436711.216666667 |
+-----+-----+
10 rows in set (0.01 sec)
```

```
mysql> select * from top_10_unsubscribed_users;
Empty set (0.00 sec)
```

```
mysql> select * from top_10_stations;
+-----+-----+-----+
| station_id | total_distinct_songs_played | distinct_user_count |
+-----+-----+-----+
| ST402      |          2 |          2 |
| ST411      |          2 |          2 |
| ST405      |          1 |          1 |
| ST410      |          1 |          1 |
+-----+-----+-----+
4 rows in set (0.02 sec)
```

---

```
mysql> select * from top_10_stations;
+-----+-----+-----+
| station_id | total_distinct_songs_played | distinct_user_count |
+-----+-----+-----+
| ST402      |          2 |          2 |
| ST411      |          2 |          2 |
| ST405      |          1 |          1 |
| ST410      |          1 |          1 |
+-----+-----+-----+
4 rows in set (0.02 sec)
```

```
mysql> select * from song_duration;
+-----+-----+-----+-----+
| user_id | user_type | song_id | artist_id | total_duration |
+-----+-----+-----+-----+
| U115    | subscribed | S200   | A300     | -480116.766666667 |
| U108    | subscribed | S200   | A302     | 87193.7833333333 |
| U120    | subscribed | S201   | A300     | 43405.55           |
| U107    | unsubscribed | S202   | A304     | -87193.7833333333 |
| U101    | subscribed | S202   | A300     | 166666.666666667 |
| U110    | unsubscribed | S202   | A303     | -436711.216666667 |
| U118    | subscribed | S202   | A300     |      0           |
| U104    | subscribed | S202   | A303     | -166666.666666667 |
| U102    | subscribed | S203   | A305     | 480116.766666667 |
| U113    | subscribed | S203   | A304     | -43788.2333333333 |
| U113    | subscribed | S203   | A303     | 436711.216666667 |
| U105    | subscribed | S203   | A303     | -1666.66666666667 |
| U113    | subscribed | S203   | A303     |      0           |
| U103    | unsubscribed | S204   | A300     | -480116.766666667 |
| U108    | subscribed | S205   | A304     | 166666.666666667 |
| U106    | subscribed | S206   | A300     | -43788.2333333333 |
| U120    | subscribed | S206   | A303     | -333333.3333333333 |
| U105    | unsubscribed | S207   | A300     | -333333.3333333333 |
| U116    | subscribed | S208   | A303     | -166666.666666667 |
| U114    | subscribed | S209   | A303     |      523905 |
+-----+-----+-----+-----+
20 rows in set (0.02 sec)
```

```

mysql> select user_type,sum(total_duration) from song_duration where total_duration >=0 group by user_type;
+-----+-----+
| user_type | sum(total_duration) |
+-----+-----+
| subscribed | 1904665.65 |
+-----+
1 row in set (0.00 sec)

```

```

#!/bin/bash
batchid=`cat /home/acadgild/project/logs/current-batch.txt`
LOGFILE=/home/acadgild/project/logs/log_batch_$batchid
echo "Running script for data analysis using spark..." >> $LOGFILE
chmod 775 /home/acadgild/project/Spark_Practice-1/out/artifacts/Spark_practice_1_jar/Spark_practice-1.jar
zip -d /home/acadgild/project/Spark_Practice-1/out/artifacts/Spark_practice_1_jar/Spark_practice-1.jar META-INF/*.DSA META-INF/*.RSA META-INF/*.SF
spark-submit \
--class Spark_analysis \
--master local[2] \
--driver-class-path /usr/local/hive/lib/hive-hbase-handler-0.14.0.jar:/usr/local/hbase/lib/* \
/home/acadgild/project/Spark_Practice-1/out/artifacts/Spark_practice_1_jar/Spark_practice-1.jar $batchid
spark-submit \
--class Spark_analysis_2 \
--master local[2] \
--driver-class-path /usr/local/hive/lib/hive-hbase-handler-0.14.0.jar:/usr/local/hbase/lib/* \
/home/acadgild/project/Spark_Practice-1/out/artifacts/Spark_practice_1_jar/Spark_practice-1.jar $batchid
echo "Exporting data to MySQL using sqoop export" >> $LOGFILE
sh /home/acadgild/project/scripts/data_export.sh
echo "Incrementing batchid..." >> $LOGFILE
batchid=`expr $batchid + 1`
echo -n $batchid > /home/acadgild/project/logs/current-batch.txt

```

Now after exporting data into MySQL batchid will be incremented to 2 means one batch of data operations is successfully completed and new batch of data will be loaded for the analysis after every 3 hours.

```
[acadgild@localhost scripts]$ cat /home/acadgild/project/logs/current-batch.txt
2[acadgild@localhost scripts]$
```

We can check logs to track the behaviour of the operations we have done on the data and overcome failures in the pipeline and we can see the batchid incremented value in current-batch.txt.

```

Starting daemons...
Creating Lookup Tables...
Populating Lookup Tables...
Creating hive tables on top hbase tables for data enrichment and filtering...
Placing data files from local to HDFS...
Running pig script for data formatting...
Running hive script for formatted data load...
Running hive script for data enrichment and filtering...
Copying valid and invalid records in local file system...
Deleting older valid and invalid records from local file system...
Running script for data analysis using spark...
Creating mysql tables if not present...
Exporting data to MySQL using sqoop export...
Running sqoop job for data export...
Incrementing batchid...

```



Now we have wrapped all the scripts inside one single script file which has been scheduled to run at the periodic interval of every 3 hours.

```
#!/bin/bashexit

#All the below scripts will work based on the data provided by acadgild as data/web/file.xml and data/mob/file.txt

sh /home/acadgild/project/scripts/start-daemon.sh

sh /home/acadgild/project/scripts/populate-lookup.sh

sh /home/acadgild/project/scripts/data_enrichment_filtering_schema.sh

sh /home/acadgild/project/scripts/dataformatting.sh

sh /home/acadgild/project/scripts/data_enrichment.sh

sh /home/acadgild/project/scripts/data_analysis.sh
```

#### **h. Job Scheduling**

Creating Crontab to schedule above `wrapper.sh` script to run for every 3 hour interval.

```
[acadgild@localhost ~]$ cd project  
[acadgild@localhost project]$ crontab -e
```

```
#do this for every 3 hours
* */3 * * * date >> /home/acadgild/project/scripts/wrapper.sh >> /home/acadgild/project/
scripts/jobScheduling.log
```

-- INSERT --

```
[acadgild@localhost project]$ crontab -e
no crontab for acadgild - using an empty one
crontab: installing new crontab
[acadgild@localhost project]$ crontab -l
* */3 * * * /home/acadgild/project/scripts/wrapper.sh >> /home/acadgild/project/
scripts/jobScheduling.log
[acadgild@localhost project]$ █
```