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.

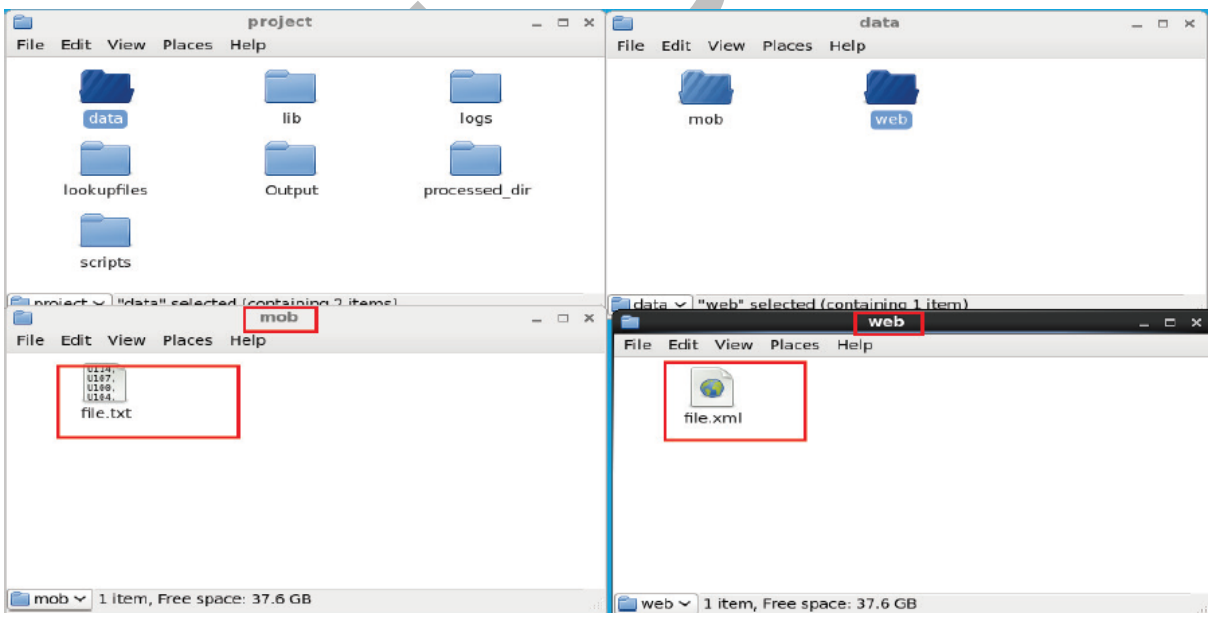
2. DATASET:

1. Data coming from web applications reside in /data/web and has xml format.

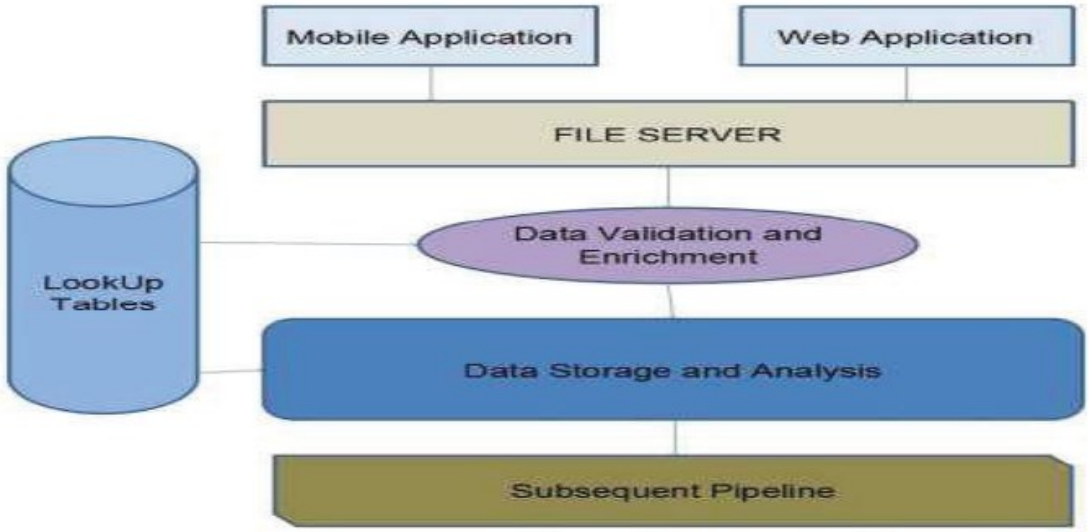
2. Data coming from mobile applications reside in /data/mob and has csv format.

3. Data present in lookup directory should be used in HBase.

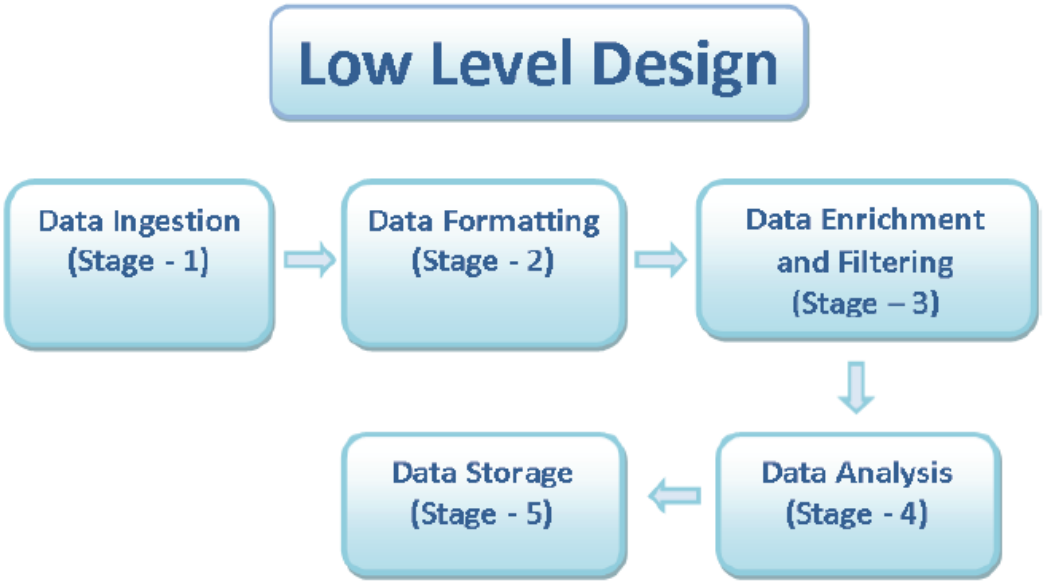
4. Flow of operations.



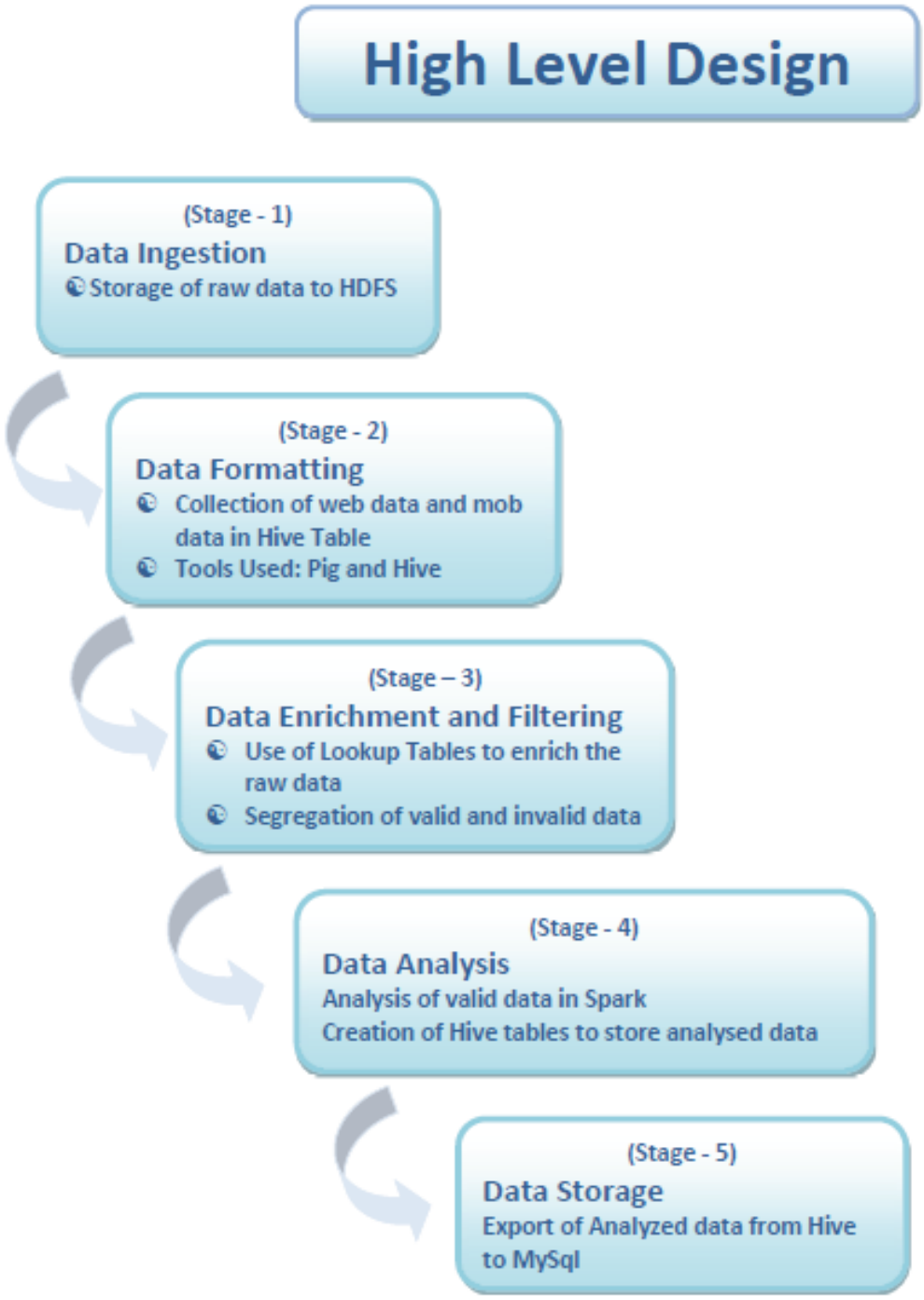
3. Flow of Operations:



4. Low Level Design:



5. High Level Design



Steps to perform Music Data Analysis:

1) Scheduling Crontab job

2) Launch all necessary daemons.

3) Populate look up tables into HBase.

4) Perform Data Formatting.

5) Perform Data Enrichment and Cleaning.

6) Perform Data Analysis.

Scheduling Crontab job :-

Open Crontab File using following command -

sudo crontab -e

Press I to enter into insert mode.

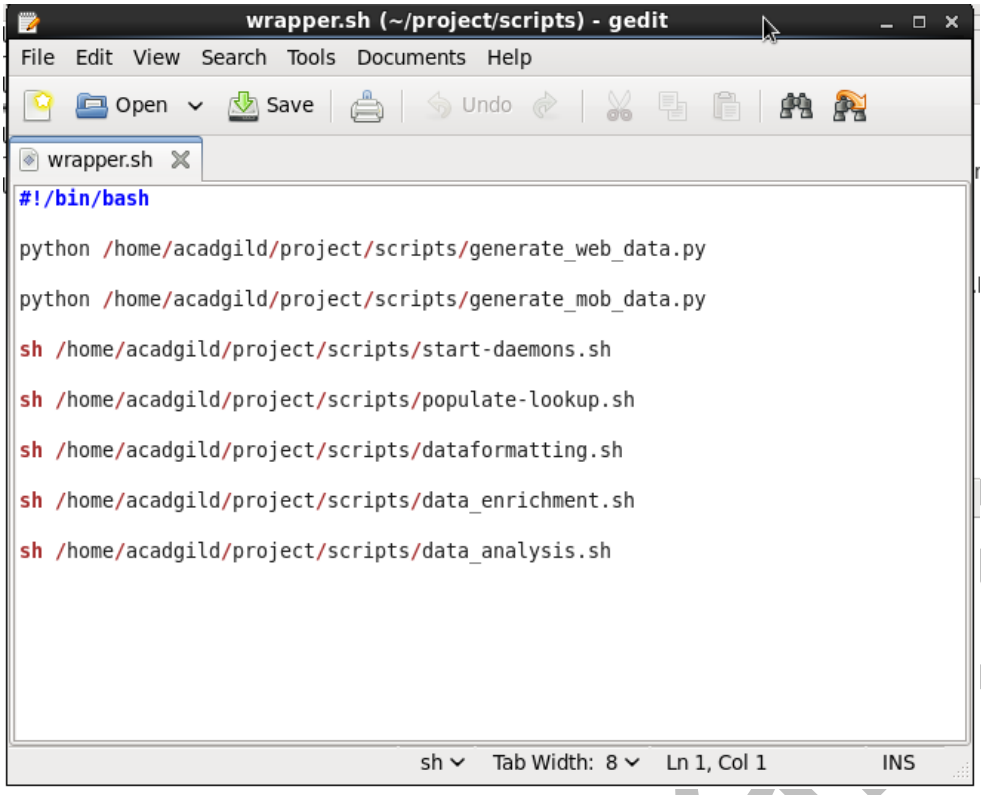
Insert the following command -

```
* */3 * * * /home/acadgild/project/scripts/wrapper.sh
```

Explaination : Crontab is used for Job Scheduling. In the -e mode, Crontab schedules execution of commands by a regular user.

The statement above runs the wrapper.sh shell script every 3 hours.

In the shell script wrapper.sh used above, all the processes needed to perform analysis on the Music Data is called once every 3 hours thereby creating a new batch. This is the job scheduling.



Implementation:

Starting all daemons in Hadoop:

· In this script we will start all the daemons that are required to start the services like hive, hbase, MySQL etc.

start-daemons.sh

```
#!/bin/bash

rm -r /home/acadgild/examples/music/logs
mkdir -p /home/acadgild/examples/music/logs

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

chmod 775 /home/acadgild/examples/music/logs/current-batch.txt
echo "After chmod"
batchid=`cat /home/acadgild/examples/music/logs/current-batch.txt`
echo "After batchid-->> "$batchid
LOGFILE=/home/acadgild/examples/music/logs/log_batch_$batchid

echo "Starting daemons" >> $LOGFILE

start-all.sh
start-hbase.sh
mr-jobhistory-daemon.sh start historyserver

cat /home/acadgild/examples/music/logs/current-batch.txt
```

· Terminal with all the daemons started screenshot

```
[acadgild@localhost music]$ sh music_project_master.sh
Preparing to execute python scripts to generate data...
Data Generated Successfully !
Starting the daemons....
After chmod
After batchid-->> 1
This script is Deprecated. Instead use start-dfs.sh and start-yarn.sh
18/10/30 08:50:55 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes wher
e applicable
Starting namenodes on [localhost]
localhost: starting namenode, logging to /home/acadgild/install/hadoop/hadoop-2.6.5/logs/hadoop-acadgild-namenode-localhost.localdomain
.out
localhost: starting datanode, logging to /home/acadgild/install/hadoop/hadoop-2.6.5/logs/hadoop-acadgild-datanode-localhost.localdomain
.out
Starting secondary namenodes [0.0.0.0]
0.0.0.0: starting secondarynamenode, logging to /home/acadgild/install/hadoop/hadoop-2.6.5/logs/hadoop-acadgild-secondarynamenode-local
host.localdomain.out
18/10/30 08:51:30 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes wher
e applicable
starting yarn daemons
starting resourcemanager, logging to /home/acadgild/install/hadoop/hadoop-2.6.5/logs/yarn-acadgild-resourcemanager-localhost.localdomai
n.out
localhost: starting nodemanager, logging to /home/acadgild/install/hadoop/hadoop-2.6.5/logs/yarn-acadgild-nodemanager-localhost.localdo
main.out
localhost: starting zookeeper, logging to /home/acadgild/install/hbase/hbase-1.2.6/logs/hbase-acadgild-zookeeper-localhost.localdomain.
out
starting master, logging to /home/acadgild/install/hbase/hbase-1.2.6/logs/hbase-acadgild-master-localhost.localdomain.out
starting regionserver, logging to /home/acadgild/install/hbase/hbase-1.2.6/logs/hbase-acadgild-1-regionserver-localhost.localdomain.out
starting historyserver, logging to /home/acadgild/install/hadoop/hadoop-2.6.5/logs/mapred-acadgild-historyserver-localhost.localdomain.
out
119520 DataNode
20129 NodeManager
21090 JobHistoryServer
20771 HQuorumPeer
21140 Jps
19780 SecondaryNameNode
20009 ResourceManager

20009 ResourceManager
20873 HMaster
20973 HRegionServer
19358 NameNode
All hadoop daemons started !
Upload the look up tables now in Hbase...
Done with data population in look up tables !
Lets do some data formatting now....
data formatting complete !
Creating hive tables on top of hbase tables for data enrichment and filtering...
Hive table with Hbase Mapping Complete !
Let us do data enrichment as per the requirement...
Data Enrichment Complete
Lets run some use cases now...
USE CASES COMPLETE !!
You have mail in /var/spool/mail/acadgild
[acadgild@localhost music]$
```

Generating data for analysis:

```
[acadgild@localhost data]$ ll
total 8
drwxrwxr-x. 2 acadgild acadgild 4096 Oct 30 08:50 mob
drwxrwxr-x. 2 acadgild acadgild 4096 Oct 30 08:50 web
[acadgild@localhost data]$ cd mob/
[acadgild@localhost mob]$ ll
total 4
-rw-rw-r--. 1 acadgild acadgild 1240 Oct 30 08:50 file.txt
[acadgild@localhost mob]$ cat file.txt
U109,S210,A301,1465130523,1475130523,1465130523,U,ST405,0,0,0
U111,S209,A301,1475130523,1475130523,1465230523,AU,ST402,0,0,1
U116,S207,A300,1495130523,1485130523,1485130523,AU,ST400,1,1,1
U103,S204,A302,1475130523,1465130523,1465230523,A,ST412,3,0,0
U117,S200,A301,1465230523,1485130523,1475130523,AU,ST414,2,0,1
,S209,A303,1495130523,1485130523,1485130523,A,ST405,2,1,0
U108,S203,A300,1495130523,1475130523,1475130523,AP,ST405,1,0,1
U120,S207,A303,1495130523,1465230523,1465230523,AP,ST401,0,0,1
U109,S205,A300,1475130523,1465130523,1465130523,,ST412,0,1,1
U117,S210,,1475130523,1465130523,1465230523,E,ST401,0,1,0
U103,S209,A305,1475130523,1485130523,1475130523,E,ST402,3,1,0
U119,S201,A305,1475130523,1465130523,1475130523,AU,ST413,0,0,0
U105,S205,A301,1465230523,1475130523,1465130523,A,ST408,2,1,0
U102,S209,A303,1495130523,1465130523,1465230523,AU,ST401,1,0,0
U117,S205,A305,1475130523,1465230523,1485130523,U,ST411,1,0,1
U100,S203,A301,1495130523,1465130523,1475130523,A,ST401,0,1,1
U116,S205,A301,1465230523,1465130523,1475130523,U,ST407,2,0,1
U100,S203,A304,1475130523,1475130523,1485130523,AU,ST415,3,1,1
U101,S204,A303,1465230523,1475130523,1485130523,AU,ST413,2,1,1
U100,S208,A301,1465230523,1465230523,1465130523,A,ST404,0,0,1
You have mail in /var/spool/mail/acadgild
[acadgild@localhost mob]$ pwd
/home/acadgild/examples/music/data/mob
[acadgild@localhost mob]$
```

```
[acadgild@localhost web]$ xmllint --format file.xml
<?xml version="1.0"?>
<records>
  <record>
    <user_id>U113</user_id>
    <song_id>S205</song_id>
    <artist_id>A304</artist_id>
    <timestamp>2016-05-10 12:24:22</timestamp>
    <start_ts>2017-05-09 08:09:22</start_ts>
    <end_ts>2016-06-09 22:12:36</end_ts>
    <geo_cd>A</geo_cd>
    <station_id>ST401</station_id>
    <song_end_type>3</song_end_type>
    <like>0</like>
    <dislike>1</dislike>
  </record>
  <record>
    <user_id>U114</user_id>
    <song_id>S209</song_id>
    <artist_id>A303</artist_id>
    <timestamp>2016-06-09 22:12:36</timestamp>
    <start_ts>2016-06-09 22:12:36</start_ts>
    <end_ts>2017-05-09 08:09:22</end_ts>
    <geo_cd>AU</geo_cd>
    <station_id>ST415</station_id>
    <song_end_type>0</song_end_type>
    <like>0</like>
    <dislike>0</dislike>
  </record>
  <record>
    <user_id>U113</user_id>
    <song_id>S201</song_id>
    <artist_id>A300</artist_id>
    <timestamp>2017-05-09 08:09:22</timestamp>
    <start_ts>2016-05-10 12:24:22</start_ts>
    <end_ts>2016-05-10 12:24:22</end_ts>
    <geo_cd>E</geo_cd>
```

The start-daemon.sh 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.

```
[acadgild@localhost logs]$ ll
total 8
-rwxrwxr-x. 1 acadgild acadgild  1 Oct 30 08:50 current-batch.txt
-rw-rw-r--. 1 acadgild acadgild 17 Oct 30 08:50 log_batch_1
[acadgild@localhost logs]$ cat current-batch.txt
1[acadgild@localhost logs]$ pwd
/home/acadgild/examples/music/logs
[acadgild@localhost logs]$
```

Lookup Tables creation in HBASE:

By using the populate-lookup.sh script we will create lookup tables in Hbase. These tables must be used in data formatting, data

enrichment and analysis stage.

Table Name Description	Table Name Description
Station_Geo_Map Contains mapping of a geo_cd with station_id	Station_Geo_Map Contains mapping of a geo_cd with station_id
Subscribed_Users Contains user_id, subscription_start_date and	Subscribed_Users Contains user_id, subscription_start_date and
subscription_end_date.	subscription_end_date.
Contains details only for subscribed users	Contains details only for subscribed users

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 Script.

populate-lookup.sh

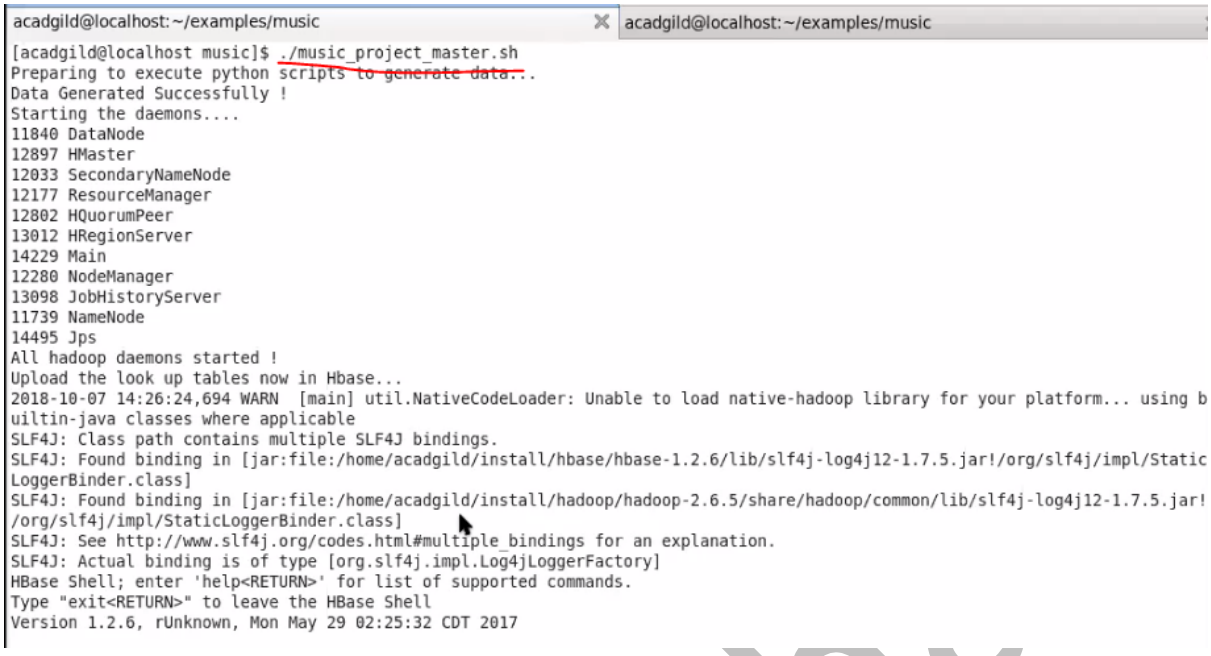
```
#!/bin/bash

batchid=`cat /home/acadgild/examples/music/logs/current-batch.txt`

LOGFILE=/home/acadgild/examples/music/logs/log_batch_$batchid

echo "Creating LookUp Tables" >> $LOGFILE

echo "disable 'station-geo-map'" | hbase shell
echo "drop 'station-geo-map'" | hbase shell
echo "disable 'subscribed-users'" | hbase shell
echo "drop 'subscribed-users'" | hbase shell
echo "disable 'song-artist-map'" | hbase shell
echo "drop 'song-artist-map'" | hbase shell

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


echo "Populating LookUp Tables" >> $LOGFILE

file="/home/acadgild/examples/music/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/acadgild/examples/music/lookupfiles/song-artist.txt"
while IFS= read -r line
```

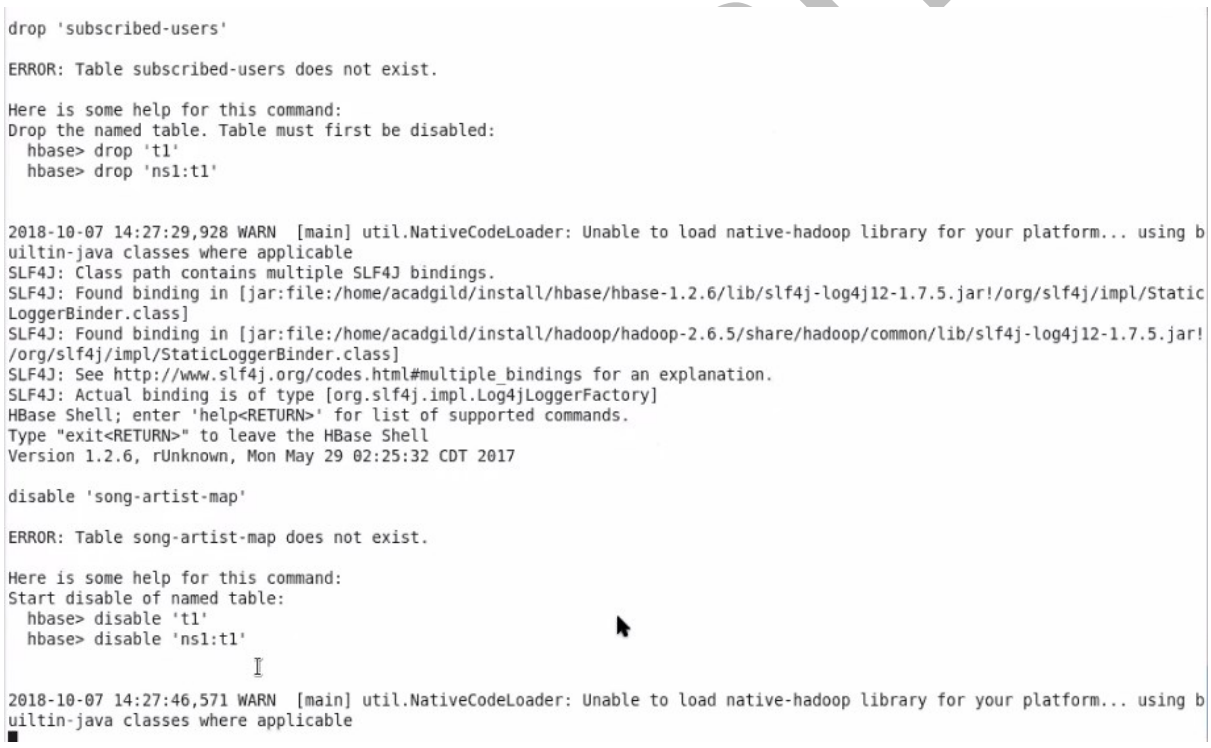
```
file="/home/acadgild/examples/music/lookupfiles/song-artist.txt"
while IFS= read -r line
do
 songid=`echo $line | cut -d',' -f1`
 artistid=`echo $line | cut -d',' -f2`
 echo "put 'song-artist-map', '$songid', 'artist:artistid', '$artistid'" | hbase shell
done <"$file"


file="/home/acadgild/examples/music/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"

#hive -f /home/acadgild/examples/music/user-artist.hql
```

music_project_master.sh | start-daemons.sh | current-batch.txt | log_batch_1 | populate-lookup.sh

```
python /home/acadgild/examples/music/generate_mob_data.py

echo "Data Generated Successfully !"

# Call Stop start daemon scripts to start hadoop daemons

echo "Starting the daemons...."
# sh start-daemons.sh

# run jps commands to check the daemons

jps

echo "All hadoop daemons started !"

echo "Upload the look up tables now in Hbase..."

sh populate-lookup.sh

echo "Done with data population in look up tables !"

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

#sh data_enrichment_filtering_schema.sh

echo "Lets do some data formatting now...."

#sh dataformatting.sh

echo "data formatting complete !"

echo "Hive table with Hbase Mapping Complete !"
```

Running project_master.sh to create the hbase tables & populate those tables with the files that was generated.





By using the shell scripting, I created 4 lookup tables in Hbase NoSQL Database.

Now we can see the lookup tables in Hbase shell terminal as shown below:

```
[acadgild@localhost music]$ hbase shell
2018-10-30 09:44:08,523 WARN  [main] util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-jav
a classes where applicable
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/acadgild/install/hbase/hbase-1.2.6/lib/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBind
er.class]
SLF4J: Found binding in [jar:file:/home/acadgild/install/hadoop/hadoop-2.6.5/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.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
HBase Shell; enter 'help<RETURN>' for list of supported commands.
Type "exit<RETURN>" to leave the HBase Shell
Version 1.2.6, rUnknown, Mon May 29 02:25:32 CDT 2017

hbase(main):001:0> list
TABLE
TRANSACTIONS
bulktable
clicks
customer
dept_tbl
employee
htest
people
song-artist-map
station-geo-map
subscribed-users
t1
12 row(s) in 0.6050 seconds

=> ["TRANSACTIONS", "bulktable", "clicks", "customer", "dept_tbl", "employee", "htest", "people", "song-artist-map", "station-geo-map",
 "subscribed-users", "t1"]
hbase(main):002:0> scan "song-artist-map"
ROW                           COLUMN+CELL
 S200                         column=artist:artistid, timestamp=1538316428029, value=A300
 S201                         column=artist:artistid, timestamp=1538316447297, value=A301
 S202                         column=artist:artistid, timestamp=1538316466192, value=A302
```

Now we can see the lookup tables in Hbase shell terminal as shown below:

```
=> ["TRANSACTIONS", "bulktable", "clicks", "customer", "dept_tbl", "employee", "htest", "people", "song-artist-map", "station-geo-map"
 "subscribed-users", "t1"]
hbase(main):009:0> scan "song-artist-map"
ROW                           COLUMN+CELL
 S200                         column=artist:artistid, timestamp=1538316428029, value=A300
 S201                         column=artist:artistid, timestamp=1538316447297, value=A301
 S202                         column=artist:artistid, timestamp=1538316466192, value=A302
 S203                         column=artist:artistid, timestamp=1538316484112, value=A303
 S204                         column=artist:artistid, timestamp=1538316502211, value=A304
 S205                         column=artist:artistid, timestamp=1538316523126, value=A301
 S206                         column=artist:artistid, timestamp=1538316541157, value=A302
 S207                         column=artist:artistid, timestamp=1538316560475, value=A303
 S208                         column=artist:artistid, timestamp=1538316579065, value=A304
 S209                         column=artist:artistid, timestamp=1538316598447, value=A305
10 row(s) in 0.0790 seconds

hbase(main):010:0> scan "station-geo-map"
ROW                           COLUMN+CELL
 ST400                        column=geo:geo_cd, timestamp=1538316137799, value=A
 ST401                        column=geo:geo_cd, timestamp=1538316157614, value=AU
 ST402                        column=geo:geo_cd, timestamp=1538316178049, value=AP
 ST403                        column=geo:geo_cd, timestamp=1538316197421, value=J
 ST404                        column=geo:geo_cd, timestamp=1538316216997, value=E
 ST405                        column=geo:geo_cd, timestamp=1538316236577, value=A
 ST406                        column=geo:geo_cd, timestamp=1538316261731, value=AU
 ST407                        column=geo:geo_cd, timestamp=1538316281989, value=AP
 ST408                        column=geo:geo_cd, timestamp=1538316300049, value=E
 ST409                        column=geo:geo_cd, timestamp=1538316318283, value=E
 ST410                        column=geo:geo_cd, timestamp=1538316336485, value=A
 ST411                        column=geo:geo_cd, timestamp=1538316355175, value=A
 ST412                        column=geo:geo_cd, timestamp=1538316372864, value=AP
 ST413                        column=geo:geo_cd, timestamp=1538316391314, value=J
 ST414                        column=geo:geo_cd, timestamp=1538316409064, value=E
15 row(s) in 0.0900 seconds

hbase(main):011:0>
```

```
hbase(main):011:0> scan "subscribed-users"
ROW                                   COLUMN+CELL
 U100                                 column=subscn:enddt, timestamp=1538316637003, value=1465130523
 U100                                 column=subscn:startdt, timestamp=1538316617571, value=1465230523
 U101                                 column=subscn:enddt, timestamp=1538316672734, value=1475130523
 U101                                 column=subscn:startdt, timestamp=1538316654507, value=1465230523
 U102                                 column=subscn:enddt, timestamp=1538316710842, value=1475130523
 U102                                 column=subscn:startdt, timestamp=1538316690609, value=1465230523
 U103                                 column=subscn:enddt, timestamp=1538316747290, value=1475130523
 U103                                 column=subscn:startdt, timestamp=1538316728339, value=1465230523
 U104                                 column=subscn:enddt, timestamp=1538316785143, value=1475130523
 U104                                 column=subscn:startdt, timestamp=1538316766279, value=1465230523
 U105                                 column=subscn:enddt, timestamp=1538316820854, value=1475130523
 U105                                 column=subscn:startdt, timestamp=1538316802813, value=1465230523
 U106                                 column=subscn:enddt, timestamp=1538316861414, value=1485130523
 U106                                 column=subscn:startdt, timestamp=1538316839943, value=1465230523
 U107                                 column=subscn:enddt, timestamp=1538316897273, value=1455130523
 U107                                 column=subscn:startdt, timestamp=1538316879775, value=1465230523
 U108                                 column=subscn:enddt, timestamp=1538316936435, value=1465230623
 U108                                 column=subscn:startdt, timestamp=1538316915598, value=1465230523
 U109                                 column=subscn:enddt, timestamp=1538316974613, value=1475130523
 U109                                 column=subscn:startdt, timestamp=1538316955809, value=1465230523
 U110                                 column=subscn:enddt, timestamp=1538317011292, value=1475130523
 U110                                 column=subscn:startdt, timestamp=1538316992978, value=1465230523
 U111                                 column=subscn:enddt, timestamp=1538317048768, value=1475130523
 U111                                 column=subscn:startdt, timestamp=1538317030665, value=1465230523
 U112                                 column=subscn:enddt, timestamp=1538317084945, value=1475130523
 U112                                 column=subscn:startdt, timestamp=1538317066869, value=1465230523
 U113                                 column=subscn:enddt, timestamp=1538317121102, value=1485130523
 U113                                 column=subscn:startdt, timestamp=1538317102485, value=1465230523
 U114                                 column=subscn:enddt, timestamp=1538317157864, value=1468130523
 U114                                 column=subscn:startdt, timestamp=1538317139087, value=1465230523
15 row(s) in 0.2660 seconds

hbase(main):012:0>
```

Now Lookup table creation is completed. So now we need to link theses lookup tables in hive using the Hbase Storage Handler.

Creating Hive External tables with actual data in Hbase:

In this stage with the help of Hbase storage handler & SerDe properties we are creating the hive external tables by matching the columns of Hbase tables to hive tables.

data_enrichment_filtering_schema.sh | create_hive_hbase_lookup.hql

```
#!/bin/bash

batchid=`cat /home/acadgild/examples/music/logs/current-batch.txt`
LOGFILE=/home/acadgild/examples/music/logs/log_batch_$batchid

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

hive -f /home/acadgild/examples/music/create_hive_hbase_lookup.hql
```

data_enrichment_filtering_schema.sh | 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");
```

Run the data_enrichment by uncommenting it within master file as shown below:

music_project_master.sh | data_enrichment_filtering_schema.sh | create_hive_hbase_lookup.hql

```
python /home/acadgild/examples/music/generate_mob_data.py

echo "Data Generated Successfully !"

# Call Stop start daemon scripts to start hadoop daemons

echo "Starting the daemons...."
# sh start-daemons.sh

# run jps commands to check the daemons

jps

echo "All hadoop daemons started !"

echo "Upload the look up tables now in Hbase..."

#sh populate-lookup.sh

echo "Done with data population in look up tables !"

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

sh data_enrichment_filtering_schema.sh

echo "Lets do some data formatting now...."

#sh dataformatting.sh

echo "data formatting complete !"

echo "Hive table with Hbase Mapping Complete !"
```

File Edit View Search Terminal Tabs Help

acadgild@localhost:~/e... | acadgild@localhost:~/e... | acadgild@localhost:~/e... | acadgild@localhost:~/e... | acadgild@localhost:~/e...

```
[acadgild@localhost music]$ ./music_project_master.sh
Preparing to execute python scripts to generate data...
Data Generated Successfully !
Starting the daemons....
10656 DataNode
13156 RunJar
10532 NameNode
15365 HRegionServer
30759 Jps
14471 ZooKeeperMain
22953 RunJar
8495 SparkSubmit
10993 ResourceManager
11095 NodeManager
15256 HMaster
11480 JobHistoryServer
15673 Main
15161 HQuorumPeer
10847 SecondaryNameNode
All hadoop daemons started !
Upload the look up tables now in Hbase...
Done with data population in look up tables !
Creating hive tables on top of hbase tables for data enrichment and filtering...
```

```
acadgild@localhost:~/e... | acadgild@localhost:~/e... | acadgild@localhost:~/e... | acadgild@localhost:~/e... | acadgild@localhost:~/e...
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFactory]

Logging initialized using configuration in jar:file:/home/acadgild/install/hive/apache-hive-2.3.2-bin/lib/hive-common-2.3.2.
jar!/hive-log4j2.properties Async: true
OK
Time taken: 11.608 seconds
OK
Time taken: 0.081 seconds
OK
Time taken: 0.545 seconds
OK
Time taken: 0.109 seconds
OK
Time taken: 0.135 seconds
Lets do some data formatting now....
data formatting complete !
Hive table with Hbase Mapping Complete !
Let us do data enrichment as per the requirement...
Data Enrichment Complete
Lets run some use cases now...
USE CASES COMPLETE !!
You have new mail in /var/spool/mail/acadgild
```

In the below screenshot we can see tables getting created in hive by running the data_enrichement_filtering_schema.sh file.

```
3. Acad | 4. Acad | 5. Acad | 6. Acad
[acadgild@localhost ~]$ hive
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/acadgild/install/hive/apache-hive-2.3.2-bin/lib/log4j-slf4j-impl-2.6.2.jar!/org/slf4j/impl/Stat
icLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/acadgild/install/hadoop/hadoop-2.6.5/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.
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFactory]

Logging initialized using configuration in jar:file:/home/acadgild/install/hive/apache-hive-2.3.2-bin/lib/hive-common-2.3.2.jar!/hive-l
og4j2.properties Async: true
Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execution engine (i.e. s
park, tez) or using Hive 1.X releases.
hive> show databases;
OK
acadgilddb
custom
default
project
simplidb
test
Time taken: 18.14 seconds, Fetched: 6 row(s)
hive> use project;
OK
Time taken: 0.063 seconds
hive> show tables;
OK
enriched_data
formatted_input
song_artist_map
station_geo_map
subscribed_users
Time taken: 1.113 seconds, Fetched: 5 row(s)
hive>
```

```
hive> select * from song_artist_map;
OK
S200    A300
S201    A301
S202    A302
S203    A303
S204    A304
S205    A301
S206    A302
S207    A303
S208    A304
S209    A305
Time taken: 7.641 seconds, Fetched: 10 row(s)
```

```
hive> select * from subscribed_users;
OK
U100    1465230523      1465130523
U101    1465230523      1475130523
U102    1465230523      1475130523
U103    1465230523      1475130523
U104    1465230523      1475130523
U105    1465230523      1475130523
U106    1465230523      1485130523
U107    1465230523      1455130523
U108    1465230523      1465230623
U109    1465230523      1475130523
U110    1465230523      1475130523
U111    1465230523      1475130523
U112    1465230523      1475130523
U113    1465230523      1485130523
U114    1465230523      1468130523
Time taken: 1.253 seconds, Fetched: 15 row(s)
```

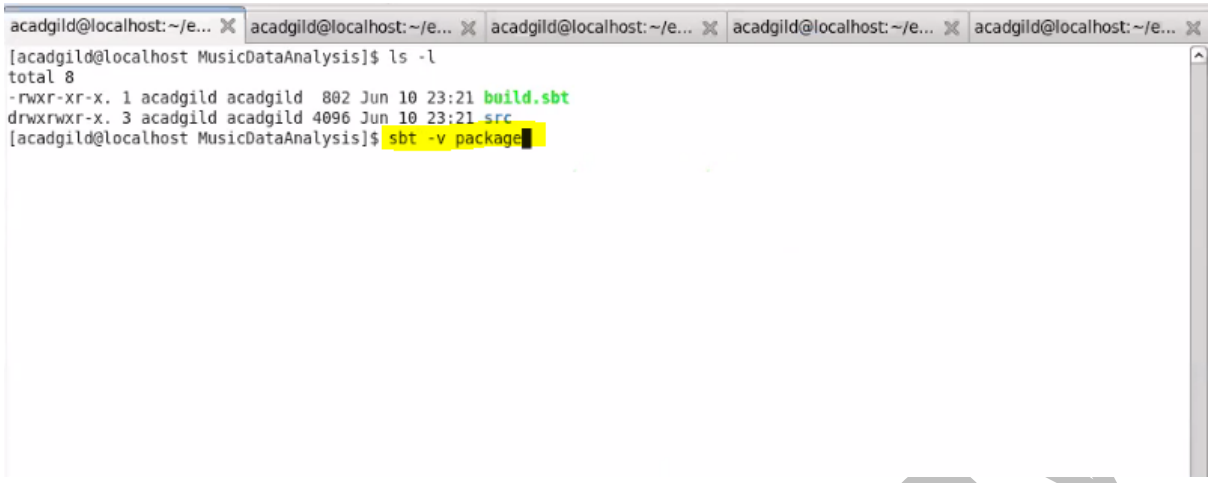
Data Formatting:

In this stage we are merging the data coming from both web applications and mobile applications and create a common table for analysing purpose and create partitioned data based on batchid, since we are running this scripts for every 3 hours.

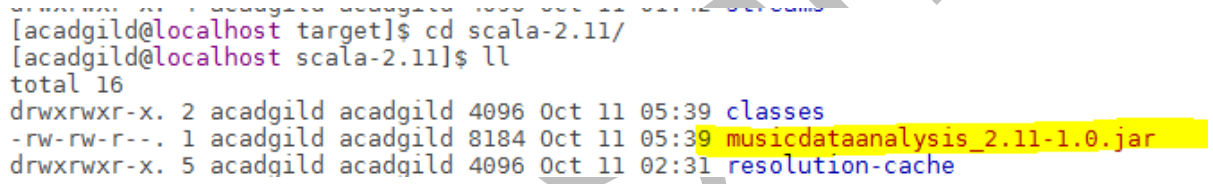
Scala programs related to data lies in the location below:

```
[acadgild@localhost target]$ cd ..
[acadgild@localhost MusicDataAnalysis]$ ll
total 24
-rw-rw-r--. 1 acadgild acadgild  802 Jun 16 00:40 build.sbt
-rw-rw-r--. 1 acadgild acadgild 6148 Jun 16 00:40 _DS_Store
drwxrwxr-x. 3 acadgild acadgild 4096 Oct 11 01:41 project
drwxrwxr-x. 3 acadgild acadgild 4096 Oct 11 01:06 src
drwxrwxr-x. 4 acadgild acadgild 4096 Oct 11 01:52 target
[acadgild@localhost MusicDataAnalysis]$ cd src/
[acadgild@localhost src]$ ll
total 12
-rw-rw-r--. 1 acadgild acadgild 6148 Jun 16 00:40 _DS_Store
drwxrwxr-x. 3 acadgild acadgild 4096 Oct 11 01:06 main
[acadgild@localhost src]$ cd main/
[acadgild@localhost main]$ LL
-bash: LL: command not found
[acadgild@localhost main]$ ll
total 12
-rw-rw-r--. 1 acadgild acadgild 6148 Jun 16 00:41 _DS_Store
drwxrwxr-x. 2 acadgild acadgild 4096 Oct 11 01:06 scala
[acadgild@localhost main]$ cd scala/
[acadgild@localhost scala]$ ll
total 16
-rw-rw-r--. 1 acadgild acadgild 4814 Jun 16 00:41 DataAnalysis.scala
-rw-rw-r--. 1 acadgild acadgild 3264 Jun 16 00:41 DataEnrichment.scala
-rw-rw-r--. 1 acadgild acadgild 2613 Oct 14 21:55 DataFormatting.scala
[acadgild@localhost scala]$ pwd
/home/acadgild/examples/music/MusicDataAnalysis/src/main/scala
[acadgild@localhost scala]$
```

Command to create jar file in verbose mode:



jar file got created



Running the master script to perform the formatting

Running DataFormatting class within jar using **SparkSubmit.**

```
acadgild@localhost:~/e... | acadgild@localhost:~/e... | acadgild@localhost:~/e... | acadgild@localhost:~/e... | acadgild@localhost:~/e...

        |                 |          modules          ||   artifacts   |
        |      conf       | number| search|dwnlded|evicted|| number|dwnlded|
        ---------------------------------------------------------------------
        |      default    |   1   |   0   |   0   |   0   ||   1   |   0   |
        ---------------------------------------------------------------------
:: retrieving :: org.apache.spark#spark-submit-parent
        confs: [default]
        0 artifacts copied, 1 already retrieved (0kB/41ms)
18/10/14 04:56:41 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classe
s where applicable
18/10/14 04:56:41 WARN Utils: Your hostname, localhost.localdomain resolves to a loopback address: 127.0.0.1; using 192.168.
1.12 instead (on interface eth15)
18/10/14 04:56:41 WARN Utils: Set SPARK_LOCAL_IP if you need to bind to another address
18/10/14 04:56:44 WARN Utils: Service 'SparkUI' could not bind on port 4040. Attempting port 4041.
18/10/14 04:57:03 ERROR KeyProviderCache: Could not find uri with key [dfs.encryption.key.provider.uri] to create a keyProvi
der !!
data formatting complete !
Hive table with Hbase Mapping Complete !
Let us do data enrichment as per the requirement...
Data Enrichment Complete
Lets run some use cases now...
USE CASES COMPLETE !!
You have new mail in /var/spool/mail/acadgild
```

Let's look at the merged table that got created during this formatting.

```
hive> select * from formatted_input;
OK
U115    S209    A304    1465130523      1465130523      1465230523      E       ST413   3       1       0       1
U109    S204    A300    1495130523      1485130523      1465230523      AP      ST410   1       1       1       1
U109    S209    A302    1465230523      1475130523      1465130523      AP      ST402   2       0       1       1
U107    S210    A302    1475130523      1475130523      1475130523      AP      ST406   3       1       1       1
U118    S209    A301    1465130523      1465230523      1465130523      A       ST408   2       1       0       1
        S203    A303    1465230523      1465130523      1485130523      A       ST407   2       0       1       1
U107    S206    A302    1475130523      1485130523      1485130523      E       ST415   3       0       1       1
U115    S201    A301    1465230523      1465130523      1485130523      U       ST412   1       1       1       1
U117    S208    A301    1475130523      1465230523      1465130523              ST411   2       0       0       1
U116    S205            1465230523      1465230523      1465230523      AP      ST414   3       0       1       1
U104    S202    A304    1475130523      1465230523      1475130523      AU      ST414   3       1       0       1
U103    S202    A303    1465230523      1465130523      1465230523      AU      ST403   2       0       1       1
U105    S202    A302    1465230523      1475130523      1485130523      A       ST413   1       1       0       1
U117    S207    A301    1465130523      1485130523      1465230523      A       ST413   3       0       1       1
U113    S201    A305    1475130523      1485130523      1465130523      AU      ST414   2       0       0       1
U117    S205    A302    1465230523      1475130523      1465130523      AU      ST410   0       0       0       1
U116    S207    A300    1475130523      1465230523      1465230523      AP      ST401   3       1       0       1
U119    S204    A305    1495130523      1465230523      1485130523      A       ST410   0       1       1       1
U117    S206    A303    1465230523      1465130523      1465130523      E       ST415   0       1       1       1
U108    S205    A302    1465230523      1465230523      1465230523      U       ST400   0       0       0       1
U110    S206    A301    1465490556      1494297562      1468094889      A       ST406   3       0       1       1
U102    S202    A301    1465490556      1468094889      1465490556      AP      ST414   1       1       0       1
U107    S203    A301    1468094889      1462863262      1468094889      E       ST414   0       1       1       1
U113    S205    A300    1494297562      1465490556      1494297562      AU      ST408   1       0       1       1
U117    S206    A304    1465490556      1494297562      1468094889      U       ST401   3       1       1       1
NULL    S208    A300    1462863262      1465490556      1494297562      E       ST410   0       0       0       1
U120    S207    A303    1462863262      1465490556      1465490556      AU      ST406   2       1       0       1
U102    S206    A301    1494297562      1462863262      1468094889      A       ST407   1       0       1       1
U100    S209    A304    1462863262      1468094889      1468094889      NULL    ST410   3       0       0       1
U116    S205    NULL    1462863262      1494297562      1494297562      A       ST409   0       1       0       1
U109    S207    A302    1468094889      1468094889      1494297562      U       ST403   0       0       1       1
U112    S210    A303    1468094889      1462863262      1465490556      AU      ST408   1       1       0       1
U111    S205    A302    1465490556      1465490556      1465490556      A       ST409   0       0       0       1
U103    S200    A304    1462863262      1468094889      1462863262      AP      ST415   0       1       1       1
U106    S204    A300    1465490556      1468094889      1494297562      E       ST411   0       1       0       1
```

· In the above screenshot we can see the formatted input data with some null values in user_id, aritist_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.

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

1. If any of like or dislike is NULL or absent, consider it as 0.

2. 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.

3. 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.

data_enrichment.sh

```
#!/bin/bash

batchid=`cat /home/acadgild/examples/music/logs/current-batch.txt`
LOGFILE=/home/acadgild/examples/music/logs/log_batch_$batchid
VALIDDIR=/home/acadgild/project/processed_dir/valid/batch_$batchid
INVALIDDIR=/home/acadgild/project/processed_dir/invalid/batch_$batchid

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

spark-submit --class DataEnrichment \
--master local[2] \
--jars /home/acadgild/install/hive/apache-hive-2.3.2-bin/lib/hive-hbase-handler-2.3.2.jar,/home/acadgild/install/hive/
apache-hive-
2.3.2-bin/lib/hbase-client-1.1.1.jar,/home/acadgild/install/hive/apache-hive-2.3.2-bin/lib/hbase-common-
1.1.1.jar,/home/acadgild/install/hive/apache-hive-2.3.2-bin/lib/hbase-hadoop-compat-1.1.1.jar,/home/acadgild/install/hive/
apache-hive-
2.3.2-bin/lib/hbase-server-1.1.1.jar,/home/acadgild/install/hive/apache-hive-2.3.2-bin/lib/hbase-protocol-
1.1.1.jar,/home/acadgild/install/hive/apache-hive-2.3.2-bin/lib/zookeeper-3.4.6.jar,/home/acadgild/install/hive/apache-hive-
2.3.2-bin/lib/guava-14.0.1.jar,/home/acadgild/install/hive/apache-hive-2.3.2-bin/lib/htrace-core-3.1.0-incubating.jar \
/home/acadgild/examples/music/MusicDataAnalysis/target/scala-2.11/musicdataanalysis_2.11-1.0.jar $batchid

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
```

Running data enrichment class using master file.

```
[acadgild@localhost music]$ sh music_project_master.sh
Preparing to execute python scripts to generate data...
Data Generated Successfully !
Starting the daemons....
After chmod
After batchid-->> 1
This script is Deprecated. Instead use start-dfs.sh and start-yarn.sh
18/10/30 08:50:55 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes wher
e applicable
Starting namenodes on [localhost]
```

/home/acadgild/examples/music/music_project_master.sh - acadgild@192.168.56.102 - Editor - WinSCP

Encoding Color

```
echo "Data Generated Successfully !"

# Call Stop start daemon scripts to start hadoop daemons

echo "Starting the daemons...."
#sh start-daemons.sh

# run jps commands to check the daemons

#jps

#echo "All hadoop daemons started !"

#echo "Upload the look up tables now in Hbase..."

#sh populate-lookup.sh

#echo "Done with data population in look up tables !"

#echo "Creating hive tables on top of hbase tables for data enrichment and filtering..."

#sh data_enrichment_filtering_schema.sh

#echo "Hive table with Hbase Mapping Complete !"

#echo "Lets do some data formatting now...."

#sh dataformatting.sh

#echo "data formatting complete !"

#echo "Let us do data enrichment as per the requirement..."

sh data_enrichment.sh

#echo "Data Enrichment Complete"

echo "Lets run some use cases now..."

#sh data_analysis.sh

echo "USE CASES COMPLETE !!"
```

/home/acadgild/examples/music/MusicDataAnalysis/src/main/scala/DataEnrichment.scala - acadgild@192.168.56.102 - Editor - WinSCP

Encoding Color

```
import org.apache.spark.{SparkConf, SparkContext}
import org.apache.spark.sql

object DataEnrichment {
  def main(args: Array[String]): Unit = {
    val conf = new SparkConf().setAppName("Data Formatting")
    val sc = new SparkContext(conf)
    val sqlContext = new org.apache.spark.sql.hive.HiveContext(sc)
    val batchId = args(0)
    val create_hive_table = """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
                            """

    val load_data = s"""INSERT OVERWRITE TABLE enriched_data
                        PARTITION (batchid, status)
                        SELECT
                        i.user_id,
                        i.song_id,
                        sa.artist_id,
                        i.timestamp,
                        i.start_ts,
                        i.end_ts,
                        sg.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
```

Line: 1/80 Column: 1 Character: 105 (0x69) Encoding: 1252 (ANSI - La

```
acadgild@localhost:~/e... | acadgild@localhost:~/e... | acadgild@localhost:~/e... | acadgild@localhost:~/e... | acadgild@localhost:~/e...

        |                |          modules          ||   artifacts   |
        |      conf      | number| search|dwnlded|evicted|| number|dwnlded|
        ---------------------------------------------------------------------
        |      default   |   1   |   0   |   0   |   0   ||   1   |   0   |
        ---------------------------------------------------------------------
:: retrieving :: org.apache.spark#spark-submit-parent
        confs: [default]
        0 artifacts copied, 1 already retrieved (0kB/41ms)
18/10/14 04:56:41 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classe
s where applicable
18/10/14 04:56:41 WARN Utils: Your hostname, localhost.localdomain resolves to a loopback address: 127.0.0.1; using 192.168.
1.12 instead (on interface eth15)
18/10/14 04:56:41 WARN Utils: Set SPARK_LOCAL_IP if you need to bind to another address
18/10/14 04:56:44 WARN Utils: Service 'SparkUI' could not bind on port 4040. Attempting port 4041.
18/10/14 04:57:03 ERROR KeyProviderCache: Could not find uri with key [dfs.encryption.key.provider.uri] to create a keyProvi
der !!
data formatting complete !
Hive table with Hbase Mapping Complete !
Let us do data enrichment as per the requirement...
Data Enrichment Complete
Lets run some use cases now...
USE CASES COMPLETE !!
You have new mail in /var/spool/mail/acadgild
```

In the above step Data_Enrichment is completed.

Let's have a look at the data enrichment table that got created.

```
hive> show tables;
OK
enriched_data
formatted_input
song_artist_map
station_geo_map
subscribed_users
Time taken: 56.171 seconds, Fetched: 5 row(s)
hive> select * from enriched_data;
OK
```

· 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

```
U100    S209    A305    1468094889      1462863262      1494297562      AP      ST412   0       1       1       1       fail
U117    S206    A302    1465490556      1494297562      1468094889      AU      ST401   3       1       1       1       fail
U107    S206    A302    1475130523      1485130523      1485130523      NULL    ST415   3       0       1       1       fail
U117    S206    A302    1465230523      1465130523      1465130523      NULL    ST415   0       1       1       1       fail
NULL    S208    A304    1462863262      1465490556      1494297562      A       ST410   0       0       0       1       fail
U117    S208    A304    1475130523      1465230523      1465130523      A       ST411   2       0       0       1       fail
U112    S210    NULL    1468094889      1462863262      1465490556      E       ST408   1       1       0       1       fail
U107    S210    NULL    1475130523      1475130523      1475130523      AU      ST406   3       1       1       1       fail
U103    S200    A300    1462863262      1468094889      1462863262      NULL    ST415   0       1       1       1       fail
U107    S203    A303    1468094889      1462863262      1468094889      E       ST414   0       1       1       1       fail
        S203    A303    1465230523      1465130523      1485130523      AP      ST407   2       0       1       1       fail
U116    S203    A303    1494297562      1465490556      1494297562      NULL    ST415   2       0       0       1       fail
U113    S201    A301    1475130523      1485130523      1465130523      E       ST414   2       0       0       1       pass
U109    S207    A303    1468094889      1468094889      1494297562      J       ST403   0       0       1       1       pass
U116    S207    A303    1475130523      1465230523      1465230523      AU      ST401   3       1       0       1       pass
U117    S207    A303    1465130523      1485130523      1465230523      J       ST413   3       0       1       1       pass
U120    S207    A303    1462863262      1465490556      1465490556      AU      ST406   2       1       0       1       pass
U104    S202    A302    1475130523      1465230523      1475130523      E       ST414   3       1       0       1       pass
U102    S202    A302    1465490556      1468094889      1465490556      E       ST414   1       1       0       1       pass
U103    S202    A302    1465230523      1465130523      1465230523      J       ST403   2       0       1       1       pass
U105    S202    A302    1465230523      1475130523      1485130523      J       ST413   1       1       0       1       pass
U106    S204    A304    1465490556      1468094889      1494297562      A       ST411   0       1       0       1       pass
U109    S209    A305    1465230523      1475130523      1465130523      AP      ST402   2       0       1       1       pass
U107    S209    A305    1462863262      1494297562      1494297562      A       ST405   3       0       1       1       pass
U118    S209    A305    1465130523      1465230523      1465130523      E       ST408   2       1       0       1       pass
U115    S209    A305    1465130523      1465130523      1465230523      J       ST413   3       1       0       1       pass
U102    S206    A302    1494297562      1462863262      1468094889      AP      ST407   1       0       1       1       pass
U110    S206    A302    1465490556      1494297562      1468094889      AU      ST406   3       0       1       1       pass
U102    S208    A304    1462863262      1468094889      1462863262      E       ST414   1       1       0       1       pass
U108    S205    A301    1465230523      1465230523      1465230523      A       ST400   0       0       0       1       pass
U116    S205    A301    1465230523      1465230523      1465230523      E       ST414   3       0       1       1       pass
U116    S205    A301    1462863262      1494297562      1494297562      E       ST409   0       1       0       1       pass
U111    S205    A301    1465490556      1465490556      1465490556      E       ST409   0       0       0       1       pass
U117    S205    A301    1465230523      1475130523      1465130523      A       ST410   0       0       0       1       pass
U113    S205    A301    1494297562      1465490556      1494297562      E       ST408   1       0       1       1       pass
Time taken: 56.951 seconds, Fetched: 40 row(s)
```

At the end script will automatically divide the records based on status pass & fail and dump the result into processed_dir folder with valid and invalid folders as shown below:

```
[acadgild@localhost batch_1]$ cd ..
[acadgild@localhost invalid]$ cd ..
[acadgild@localhost processed_dir]$ ll
total 8
drwxrwxr-x. 3 acadgild acadgild 4096 Oct 11 06:43 invalid
drwxrwxr-x. 3 acadgild acadgild 4096 Oct 11 06:43 valid
[acadgild@localhost processed_dir]$ pwd
/home/acadgild/project/processed_dir
[acadgild@localhost processed_dir]$ cd valid/
[acadgild@localhost valid]$ ll
total 4
drwxrwxr-x. 2 acadgild acadgild 4096 Oct 11 07:06 batch_1
[acadgild@localhost valid]$ cd batch_1/
[acadgild@localhost batch_1]$ ll
total 64
-rw-r--r--. 1 acadgild acadgild 1020 Oct 11 06:58 part-00020-26d8d5c7-3eda-4ab8-bbb5-aee692588258.c000
-rw-r--r--. 1 acadgild acadgild 1020 Oct 11 07:06 part-00020-bf543a70-2e3d-4519-a148-25c5553b0fe1.c000
-rw-r--r--. 1 acadgild acadgild 1225 Oct 11 06:58 part-00033-26d8d5c7-3eda-4ab8-bbb5-aee692588258.c000
-rw-r--r--. 1 acadgild acadgild 1225 Oct 11 07:06 part-00033-bf543a70-2e3d-4519-a148-25c5553b0fe1.c000
-rw-r--r--. 1 acadgild acadgild 1181 Oct 11 06:58 part-00057-26d8d5c7-3eda-4ab8-bbb5-aee692588258.c000
-rw-r--r--. 1 acadgild acadgild 1181 Oct 11 07:06 part-00057-bf543a70-2e3d-4519-a148-25c5553b0fe1.c000
-rw-r--r--. 1 acadgild acadgild 1034 Oct 11 06:58 part-00087-26d8d5c7-3eda-4ab8-bbb5-aee692588258.c000
-rw-r--r--. 1 acadgild acadgild 1034 Oct 11 07:06 part-00087-bf543a70-2e3d-4519-a148-25c5553b0fe1.c000
-rw-r--r--. 1 acadgild acadgild 1214 Oct 11 06:58 part-00095-26d8d5c7-3eda-4ab8-bbb5-aee692588258.c000
-rw-r--r--. 1 acadgild acadgild 1214 Oct 11 07:06 part-00095-bf543a70-2e3d-4519-a148-25c5553b0fe1.c000
-rw-r--r--. 1 acadgild acadgild 1158 Oct 11 06:58 part-00107-26d8d5c7-3eda-4ab8-bbb5-aee692588258.c000
-rw-r--r--. 1 acadgild acadgild 1158 Oct 11 07:06 part-00107-bf543a70-2e3d-4519-a148-25c5553b0fe1.c000
-rw-r--r--. 1 acadgild acadgild 1024 Oct 11 06:58 part-00160-26d8d5c7-3eda-4ab8-bbb5-aee692588258.c000
-rw-r--r--. 1 acadgild acadgild 1024 Oct 11 07:06 part-00160-bf543a70-2e3d-4519-a148-25c5553b0fe1.c000
-rw-r--r--. 1 acadgild acadgild 1260 Oct 11 06:58 part-00165-26d8d5c7-3eda-4ab8-bbb5-aee692588258.c000
-rw-r--r--. 1 acadgild acadgild 1260 Oct 11 07:06 part-00165-bf543a70-2e3d-4519-a148-25c5553b0fe1.c000
[acadgild@localhost batch_1]$
```

```
drwxrwxr-x. 2 acadgild acadgild 4096 Oct 11 07:06 batch_1
[acadgild@localhost invalid]$ cd
[acadgild@localhost ~]$ cd -
/home/acadgild/project/processed_dir/invalid
[acadgild@localhost invalid]$ cd batch_1/
[acadgild@localhost batch_1]$ ll
total 64
-rw-r--r--. 1 acadgild acadgild 1008 Oct 11 06:58 part-00020-26d8d5c7-3eda-4ab8-bbb5-aee692588258.c000
-rw-r--r--. 1 acadgild acadgild 1008 Oct 11 07:06 part-00020-bf543a70-2e3d-4519-a148-25c5553b0fe1.c000
-rw-r--r--. 1 acadgild acadgild 1167 Oct 11 06:58 part-00087-26d8d5c7-3eda-4ab8-bbb5-aee692588258.c000
-rw-r--r--. 1 acadgild acadgild 1167 Oct 11 07:06 part-00087-bf543a70-2e3d-4519-a148-25c5553b0fe1.c000
-rw-r--r--. 1 acadgild acadgild 1133 Oct 11 06:58 part-00095-26d8d5c7-3eda-4ab8-bbb5-aee692588258.c000
-rw-r--r--. 1 acadgild acadgild 1133 Oct 11 07:06 part-00095-bf543a70-2e3d-4519-a148-25c5553b0fe1.c000
-rw-r--r--. 1 acadgild acadgild 1191 Oct 11 06:58 part-00107-26d8d5c7-3eda-4ab8-bbb5-aee692588258.c000
-rw-r--r--. 1 acadgild acadgild 1191 Oct 11 07:06 part-00107-bf543a70-2e3d-4519-a148-25c5553b0fe1.c000
-rw-r--r--. 1 acadgild acadgild 1166 Oct 11 06:58 part-00160-26d8d5c7-3eda-4ab8-bbb5-aee692588258.c000
-rw-r--r--. 1 acadgild acadgild 1166 Oct 11 07:06 part-00160-bf543a70-2e3d-4519-a148-25c5553b0fe1.c000
-rw-r--r--. 1 acadgild acadgild 1140 Oct 11 06:58 part-00161-26d8d5c7-3eda-4ab8-bbb5-aee692588258.c000
-rw-r--r--. 1 acadgild acadgild 1140 Oct 11 07:06 part-00161-bf543a70-2e3d-4519-a148-25c5553b0fe1.c000
-rw-r--r--. 1 acadgild acadgild 1016 Oct 11 06:58 part-00177-26d8d5c7-3eda-4ab8-bbb5-aee692588258.c000
-rw-r--r--. 1 acadgild acadgild 1016 Oct 11 07:06 part-00177-bf543a70-2e3d-4519-a148-25c5553b0fe1.c000
-rw-r--r--. 1 acadgild acadgild 1214 Oct 11 06:58 part-00199-26d8d5c7-3eda-4ab8-bbb5-aee692588258.c000
-rw-r--r--. 1 acadgild acadgild 1214 Oct 11 07:06 part-00199-bf543a70-2e3d-4519-a148-25c5553b0fe1.c000
[acadgild@localhost batch_1]$
```

· Enrichment phase is executed successfully by applying all the rules of enrichment.

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.

· I used spark-submit for Data Analysis.


data_analysis.sh

```
#!/bin/bash

batchid=`cat /home/acadgild/examples/music/logs/current-batch.txt`
LOGFILE=/home/acadgild/examples/music/logs/log_batch_$batchid

echo "Running script for data analysis..." >> $LOGFILE

spark-submit --class DataAnalysis --master local[2] \
--jars /home/acadgild/install/hive/apache-hive-2.3.2-bin/lib/hive-hbase-handler-2.3.2.jar,/home/acadgild/install/hive/apache-
hive-2.3.2-bin/lib/hbase-client-1.1.1.jar,/home/acadgild/install/hive/apache-hive-2.3.2-bin/lib/hbase-common-1.1.1.jar,/home/
acadgild/install/hive/apache-hive-2.3.2-bin/lib/hbase-hadoop-compat-1.1.1.jar,/home/acadgild/install/hive/apache-hive-2.3.2-
bin/lib/hbase-server-1.1.1.jar,/home/acadgild/install/hive/apache-hive-2.3.2-bin/lib/hbase-protocol-1.1.1.jar,/home/acadgild/
install/hive/apache-hive-2.3.2-bin/lib/zookeeper-3.4.6.jar,/home/acadgild/install/hive/apache-hive-2.3.2-bin/lib/
guava-14.0.1.jar,/home/acadgild/install/hive/apache-hive-2.3.2-bin/lib/htrace-core-3.1.0-incubating.jar/home/acadgild/project/
scripts/MusicDataAnalysis/target/scala-2.11/musicdataanalysis_2.11-1.0.jar $batchid

sh /home/acadgild/examples/music/data_export.sh

echo "Incrementing batchid..." >> $LOGFILE

batchid=`expr $batchid + 1`
echo -n $batchid > /home/acadgild/examples/music/logs/current-batch.txt
```

Problem Statement:-

1. Determine top 10 station_id(s) where maximum number of songs were played, which were liked by unique users.

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.

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

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.

5. Determine top 10 unsubscribed users who listened to the songs for the longest duration.

Spark Source Code:

I have created one Scala file for creating tables as per query wise.

DataAnalysis.scala

```
object DataAnalysis {
  def main(args: Array[String]): Unit = {
    val conf = new SparkConf().setAppName("Data Analysis")
    val sc = new SparkContext(conf)
    val sqlContext = new org.apache.spark.sql.hive.HiveContext(sc)
    val batchId = args(0)


val create_top_10_stations = """CREATE TABLE IF NOT EXISTS 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"""

val load_top_10_stations = s"""INSERT OVERWRITE TABLE top_10_stations
PARTITION(batchid='$batchId')
SELECT
station_id,
COUNT(DISTINCT song_id) AS total_distinct_songs_played,
COUNT(DISTINCT user_id) AS distinct_user_count
FROM enriched_data
WHERE status='pass'
AND batchid='$batchId'
AND like=1
GROUP BY station_id
ORDER BY total_distinct_songs_played DESC
LIMIT 10"""
```

DataAnalysis.scala

```

val create_users_behaviour = """CREATE TABLE IF NOT EXISTS users_behaviour
(
user_type STRING,
duration INT
)
PARTITIONED BY (batchid INT)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
STORED AS TEXTFILE"""

val load_users_behaviour = s"""INSERT OVERWRITE TABLE users_behaviour
PARTITION(batchid='$batchId')
SELECT
CASE WHEN (su.user_id IS NULL OR CAST(ed.timestamp AS DECIMAL(20,0)) > CAST(su.subscn_end_dt AS DECIMAL(20,0))) THEN
'UNSUBSCRIBED'
WHEN (su.user_id IS NOT NULL AND CAST(ed.timestamp AS DECIMAL(20,0)) <= CAST(su.subscn_end_dt AS DECIMAL(20,0))) THEN
'SUBSCRIBED'
END AS user_type,
SUM(ABS(CAST(ed.end_ts AS DECIMAL(20,0))-CAST(ed.start_ts AS DECIMAL(20,0)))) AS duration
FROM enriched_data ed
LEFT OUTER JOIN subscribed_users su
ON ed.user_id=su.user_id
WHERE ed.status='pass'
AND ed.batchid='$batchId'
GROUP BY CASE WHEN (su.user_id IS NULL OR CAST(ed.timestamp AS DECIMAL(20,0)) > CAST(su.subscn_end_dt AS DECIMAL(20,0)))
THEN 'UNSUBSCRIBED'
WHEN (su.user_id IS NOT NULL AND CAST(ed.timestamp AS DECIMAL(20,0)) <= CAST(su.subscn_end_dt AS DECIMAL(20,0))) THEN
'SUBSCRIBED' END"""
```

DataAnalysis.scala

```
val create_connected_artists = """CREATE TABLE IF NOT EXISTS connected_artists
(
artist_id STRING,
user_count INT
)
PARTITIONED BY (batchid INT)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
STORED AS TEXTFILE"""

val load_connected_artists = s"""INSERT OVERWRITE TABLE connected_artists
PARTITION(batchid='$batchId')
SELECT
ua.artist_id,
COUNT(DISTINCT ua.user_id) AS user_count
FROM
(
SELECT user_id, artist_id FROM users_artists
LATERAL VIEW explode(artists_array) artists AS artist_id
) ua
INNER JOIN
(
SELECT artist_id, song_id, user_id
FROM enriched_data
WHERE status='pass'
AND batchid='$batchId'
) ed
ON ua.artist_id=ed.artist_id
AND ua.user_id=ed.user_id
GROUP BY ua.artist_id
ORDER BY user_count DESC
LIMIT 10"""
```

DataAnalysis.scala

```
val create_top_10_royalty_songs = """CREATE TABLE IF NOT EXISTS top_10_royalty_songs
(
song_id STRING,
duration INT
)
PARTITIONED BY (batchid INT)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
STORED AS TEXTFILE"""

val load_top_10_royalty_songs = s"""INSERT OVERWRITE TABLE top_10_royalty_songs
PARTITION(batchid='$batchId')
SELECT song_id,
SUM(ABS(CAST(end_ts AS DECIMAL(20,0))-CAST(start_ts AS DECIMAL(20,0)))) AS duration
FROM enriched_data
WHERE status='pass'
AND batchid='$batchId'
AND (like=1 OR song_end_type=0)
GROUP BY song_id
ORDER BY duration DESC
LIMIT 10"""
```

DataAnalysis.scala

```
val create_top_10_unsubscribed_users = """CREATE TABLE IF NOT EXISTS top_10_unsubscribed_users
(
user_id STRING,
duration INT
)
PARTITIONED BY (batchid INT)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
STORED AS TEXTFILE"""

val load_top_10_unsubscribed_users = s"""INSERT OVERWRITE TABLE top_10_unsubscribed_users
PARTITION(batchid='$batchId')
SELECT
ed.user_id,
SUM(ABS(CAST(ed.end_ts AS DECIMAL(20,0))-CAST(ed.start_ts AS DECIMAL(20,0)))) AS duration
FROM enriched_data ed
LEFT OUTER JOIN subscribed_users su
ON ed.user_id=su.user_id
WHERE ed.status='pass'
AND ed.batchid='$batchId'
AND (su.user_id IS NULL OR (CAST(ed.timestamp AS DECIMAL(20,0)) > CAST(su.subscn_end_dt AS DECIMAL(20,0))))
GROUP BY ed.user_id
ORDER BY duration DESC
LIMIT 10"""


    try {
         sqlContext.sql("SET hive.auto.convert.join=false")
         sqlContext.sql("USE project")
         sqlContext.sql(create_top_10_stations)
         sqlContext.sql(load_top_10_stations)
         sqlContext.sql(create_users_behaviour)
```

```
ORDER BY duration DESC
LIMIT 10"""


    try {
         sqlContext.sql("SET hive.auto.convert.join=false")
         sqlContext.sql("USE project")
         sqlContext.sql(create_top_10_stations)
         sqlContext.sql(load_top_10_stations)
         sqlContext.sql(create_users_behaviour)
         sqlContext.sql(load_users_behaviour)
         sqlContext.sql(create_connected_artists)
         sqlContext.sql(load_connected_artists)
         sqlContext.sql(create_top_10_royalty_songs)
         sqlContext.sql(load_top_10_royalty_songs)
         sqlContext.sql(create_top_10_unsubscribed_users)
         sqlContext.sql(load_top_10_unsubscribed_users)
       }
      catch{
       case e: Exception=>e.printStackTrace()
      }
 }
}
```

Running master file to run analysis.sh.

/home/acadgild/examples/music/music_project_master.sh - acadgild@192.168.56.102 - Editor - WinSCP

Encoding Color

```
python /home/acadgild/examples/music/generate_web_data.py
python /home/acadgild/examples/music/generate_mob_data.py

echo "Data Generated Successfully !"

# Call Stop start daemon scripts to start hadoop daemons

echo "Starting the daemons...."
#sh start-daemons.sh

# run jps commands to check the daemons

#jps

#echo "All hadoop daemons started !"

#echo "Upload the look up tables now in Hbase..."

#sh populate-lookup.sh

#echo "Done with data population in look up tables !"

#echo "Creating hive tables on top of hbase tables for data enrichment and filtering..."

#sh data_enrichment_filtering_schema.sh

#echo "Hive table with Hbase Mapping Complete !"

#echo "Lets do some data formatting now...."

#sh dataformatting.sh

#echo "data formatting complete !"

#echo "Let us do data enrichment as per the requirement..."

#sh data_enrichment.sh

#echo "Data Enrichment Complete"

echo "Lets run some use cases now..."

sh data_analysis.sh

echo "USE CASES COMPLETE !!"
```

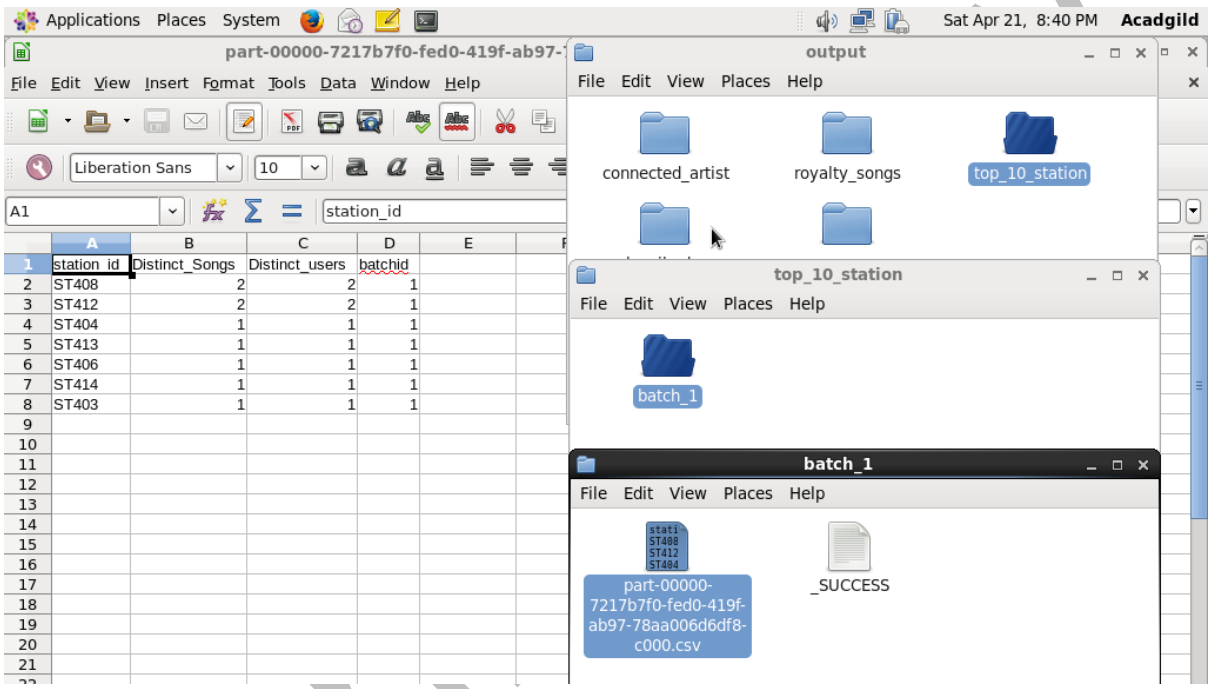
Problem 1: Determine top 10 station_id(s) where maximum number of songs were played, which were liked by unique users.

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

Solution:

```
//************************************************************************
//                              PROBLEM 1
//************************************************************************

val prob_1  = spark.sql("select station_id, count(distinct song_id) Distinct_Songs," +
  " count(distinct user_id) Distinct_users, batchid from music_data " +
  s"where status='fail' and batchid = $batch_id " +
  s"and u_like=1 group by station_id,batchid order by Distinct_Songs " +
  s"DESC limit 10").toDF("station_id","Distinct_Songs","Distinct_users","batchid")
```

```
prob_1.write.format("csv").option("header","true").save(s"/home/acadgild/project/output/top_10_station/batch_$batch_id")
```

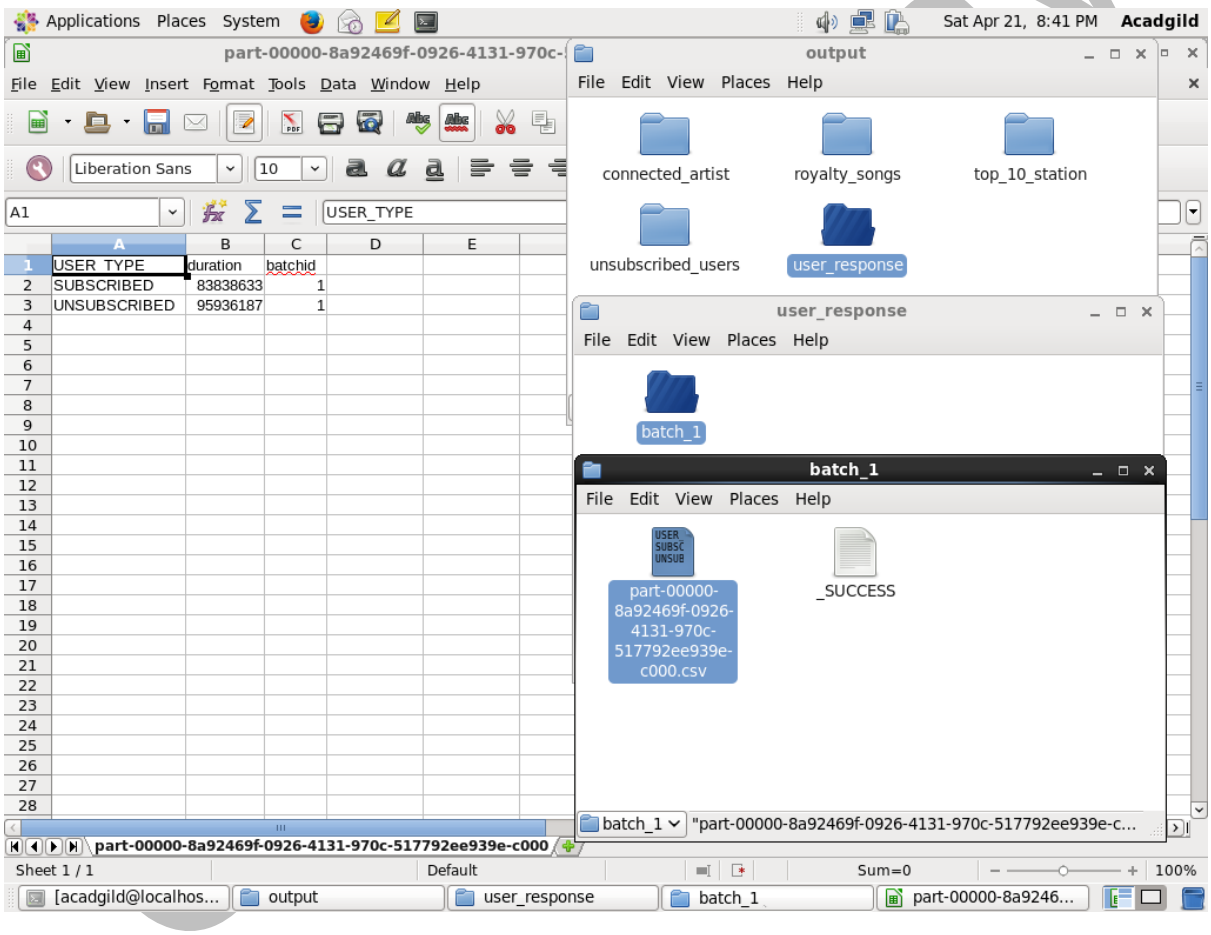


	A	B	C	D	E
1	station_id	Distinct_Songs	Distinct_users	batchid	
2	ST408	2	2	1	
3	ST412	2	2	1	
4	ST404	1	1	1	
5	ST413	1	1	1	
6	ST406	1	1	1	
7	ST414	1	1	1	
8	ST403	1	1	1	

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.

```
.765469 ms
+----------+--------------------+
| user_type|total_song_duration|
+----------+--------------------+
|subscribed| 1904665.6500000001|
+----------+--------------------+
```

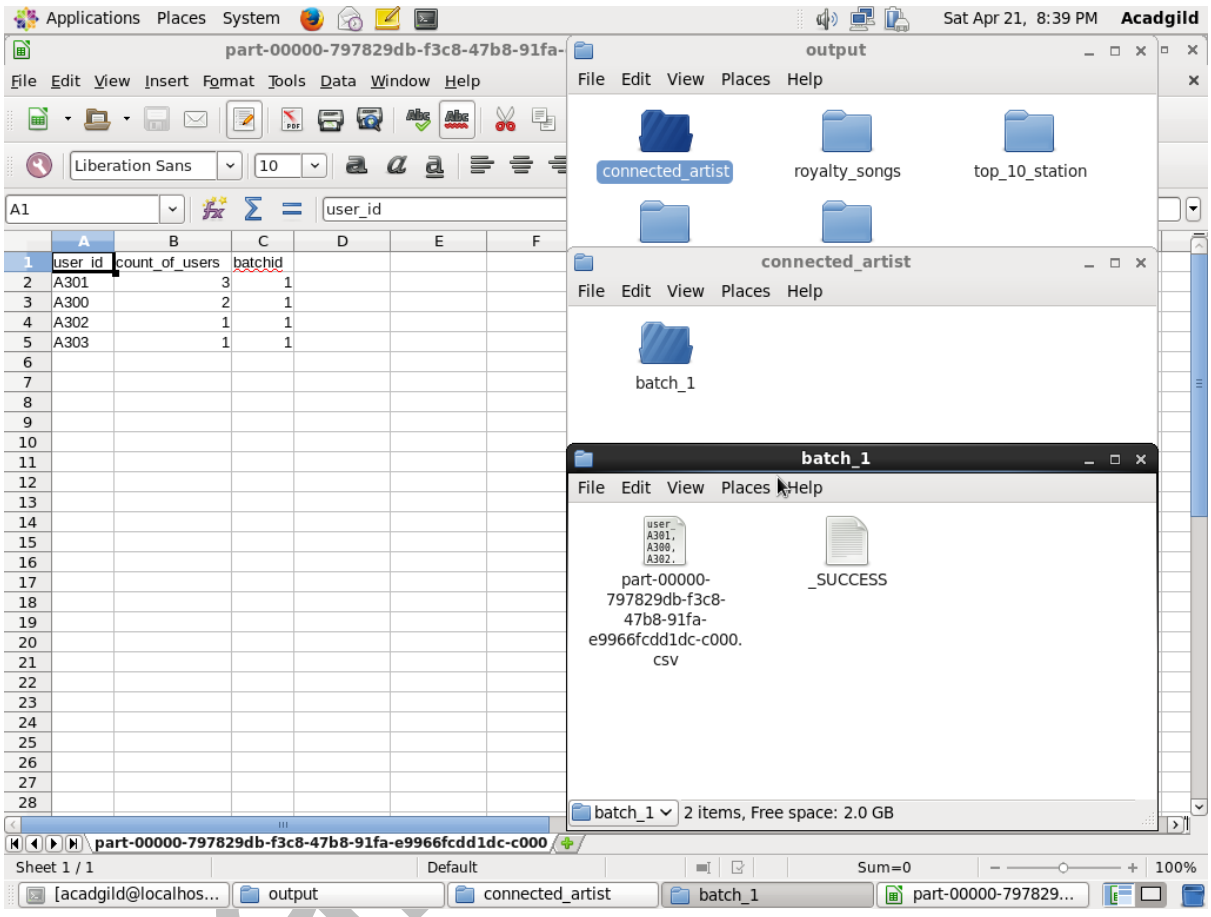
```
//*******************************************************************************
//                                  PROBLEM 2
//*******************************************************************************

val prob_2 = spark.sql("select CASE WHEN ( subusr.user_id is null or " +
  "cast(music.u_timestamp as decimal(20,0)) > cast(subusr.end_dt as decimal(20,0)) " +
  ") then 'UNSUBSCRIBED' WHEN (subusr.user_id is NOT null or " +
  "cast(music.u_timestamp as decimal(20,0)) <= cast(subusr.end_dt as decimal(20,0)) " +
  ") then 'SUBSCRIBED' END AS USER_TYPE, " +
  "SUM(ABS(CAST(music.end_ts as decimal(20,0)) - CAST(music.start_ts as decimal(20,0))))" +
  " as duration ,batchid" +
  " from music_data music left outer join " +
  "subscribed_user subusr on music.user_id = subusr.user_id " +
  s"where music.status = 'pass' and music.batchid = $batch_id " +
  s"group by music.batchid, CASE WHEN ( subusr.user_id is null or " +
  "cast(music.u_timestamp as decimal(20,0)) > cast(subusr.end_dt as decimal(20,0)) " +
  ") then 'UNSUBSCRIBED' WHEN (subusr.user_id is NOT null or " +
  "cast(music.u_timestamp as decimal(20,0)) <= cast(subusr.end_dt as decimal(20,0)) " +
  ") then 'SUBSCRIBED' END " ).toDF()

prob_2.repartition(1).write.format("csv").option("header","true").save(s"/home/acadgild/project/output/user_response/
batch_$batch_id")
```



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

```
: Removed TaskSet 5.0, whose tasks have all completed, from pool
+---------+
|artist_id|
+---------+
|     A300|
|     A303|
|     A304|
|     A305|
|     A302|
+---------+
```

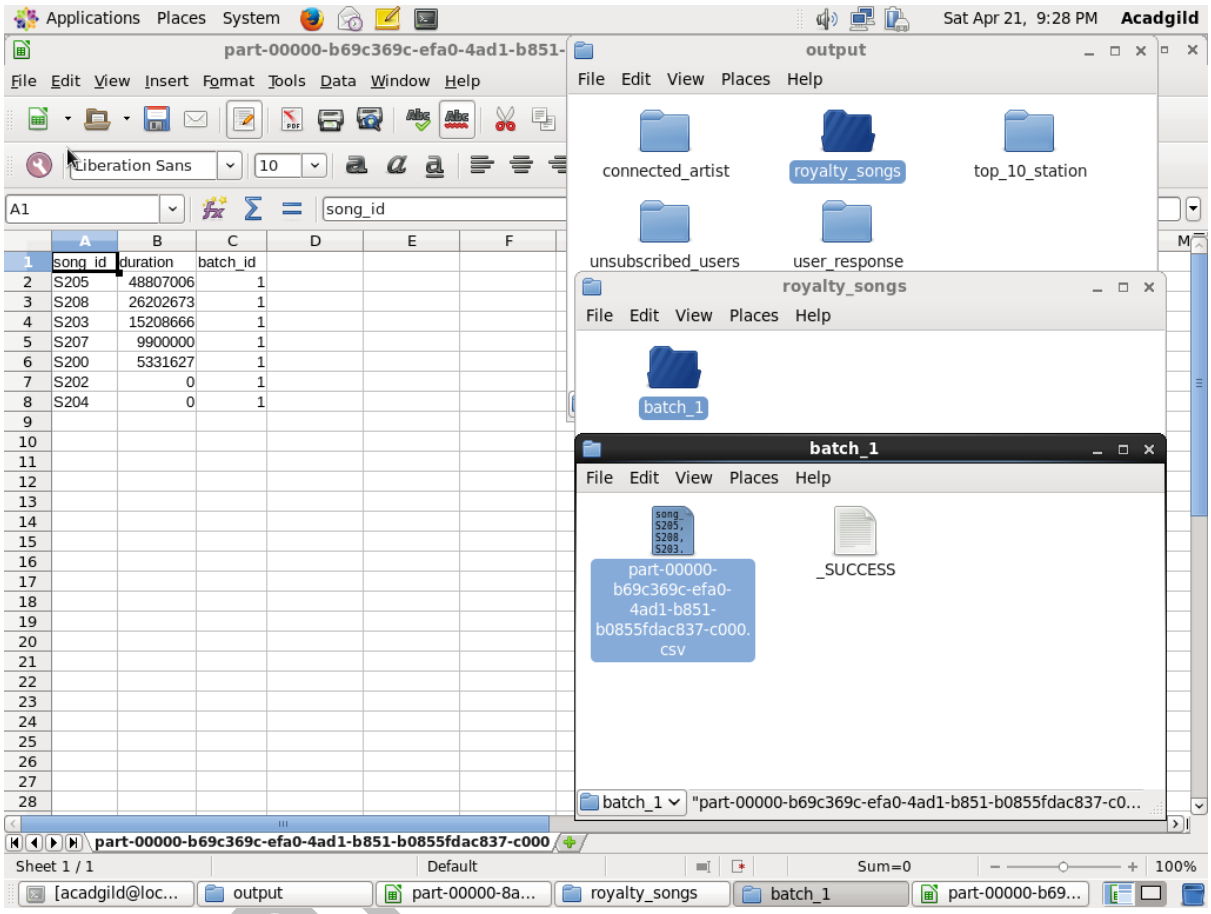
```
  //*************************************************************************
  //                              PROBLEM 3
  //*************************************************************************

  val prob_3 = spark.sql("select ua.artists, count(distinct ua.user_id) as count_of_users, " +
    "md.batchid from users_artists ua inner join (select artist_id,song_id,user_id,batchid" +
    s" from music_data where status = 'pass' and batchid = $batch_id)md " +
    s" on ua.artists = md.artist_id and ua.user_id = md.user_id group by ua.artists," +
    s" batchid order by count_of_users DESC limit 10").toDF("user_id","count_of_users","batchid")

    prob_3.write.format("csv").option("header","true").save(s"/home/acadgild/project/output/connected_artist/batch_
$batch_id")
```



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

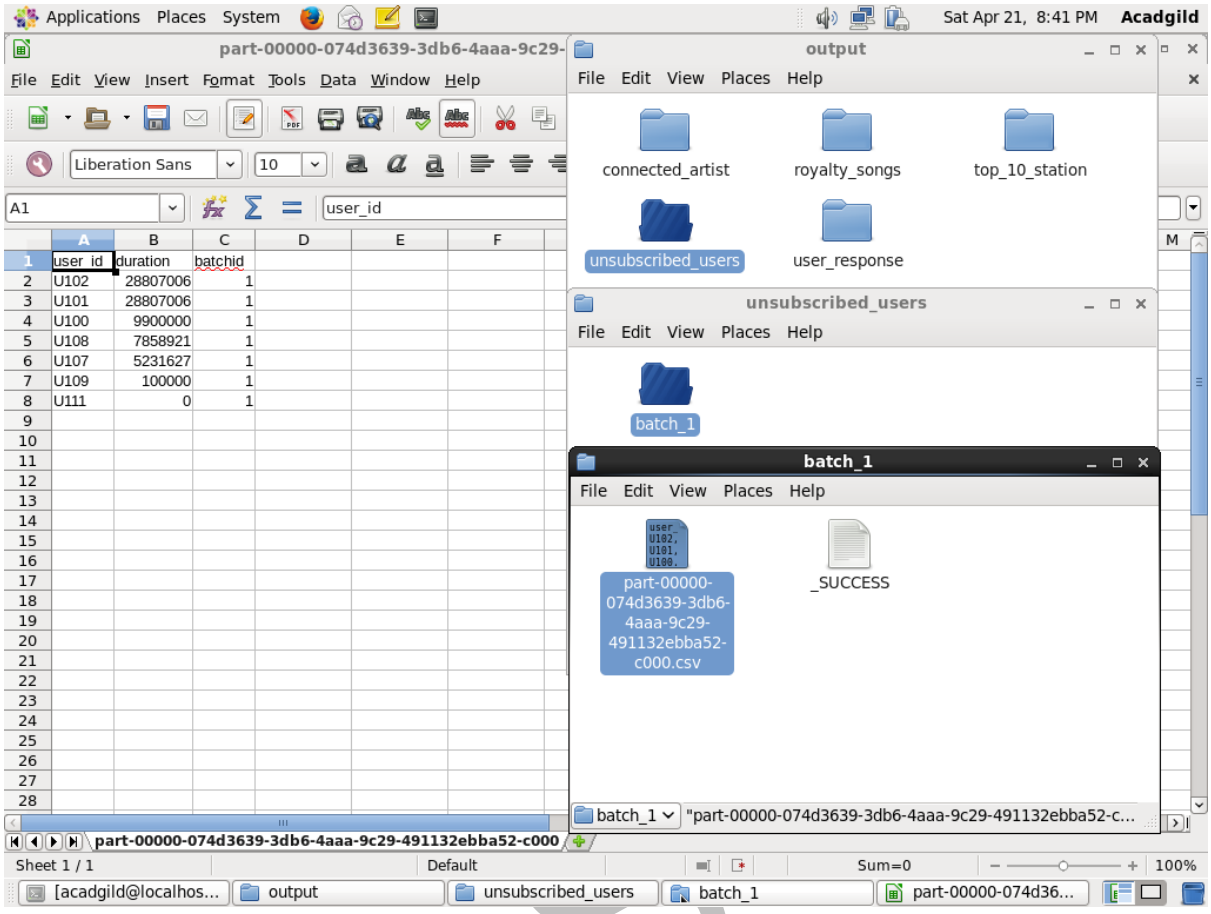
```
: Removed TaskSet 6.0, whose tasks have all completed, from pool
+-------+
|song_id|
+-------+
|   S209|
|   S202|
|   S205|
|   S200|
|   S203|
|   S203|
|   S206|
|   S202|
|   S206|
|   S202|
+-------+
```

```
//*************************************************************************
//                              PROBLEM 4
//*************************************************************************

val prob_4 = spark.sql("select song_id, " +
  "SUM(ABS(CAST(end_ts as decimal(20,0)) - " +
  " CAST(start_ts as decimal(20,0)))) as duration,batchid" +
  s" from music_data where status = 'pass' and batchid = $batch_id " +
  "and (u_like= 1 or song_end_type = 0) group by song_id, batchid " +
  "order by duration desc limit 10").toDF("song_id","duration","batch_id")

prob_4.write.format("csv").option("header","true").save(s"/home/acadgild/project/output/royalty_songs/batch_$batch_id")
```



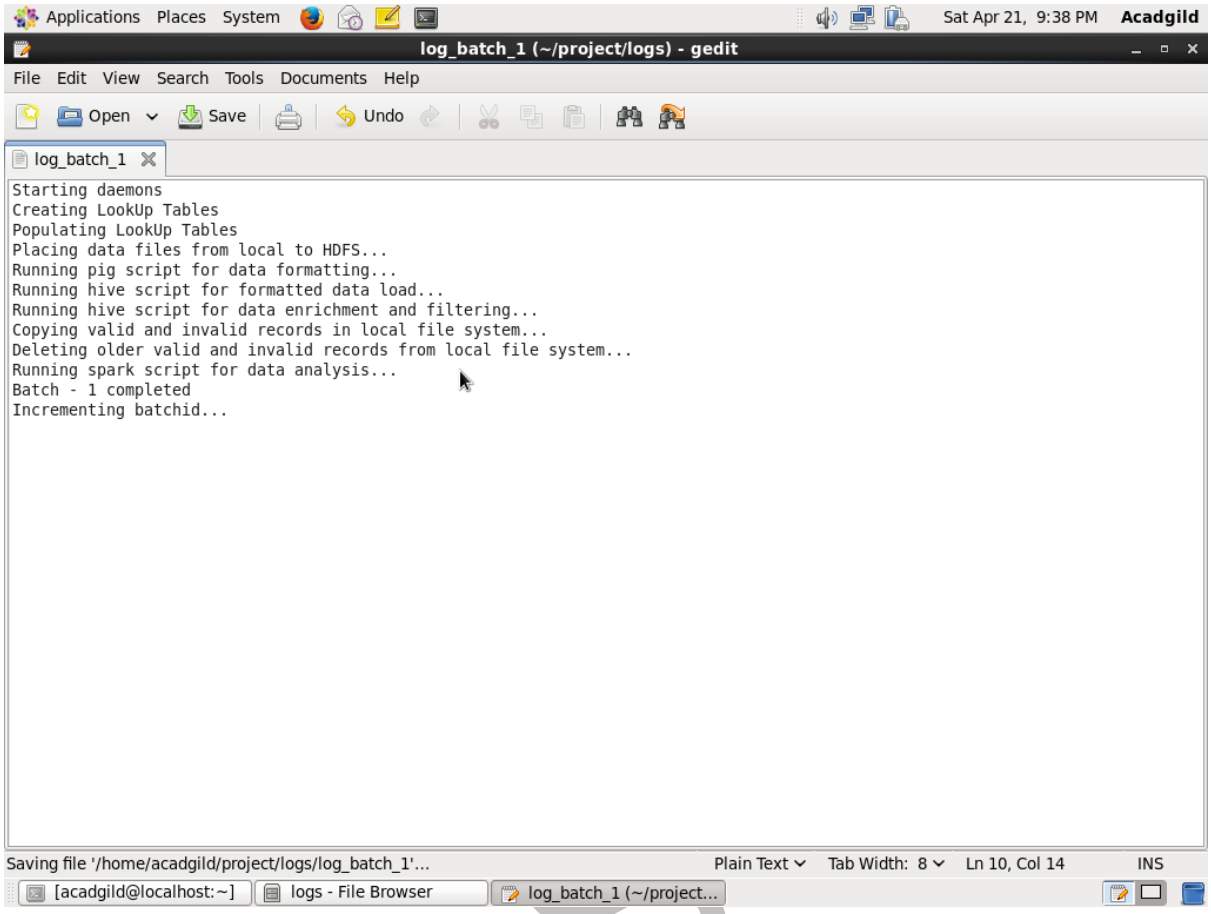
Query-5: Determine top 10 unsubscribed users who listened to the songs for the longest duration.

```
//*************************************************************************
//                              PROBLEM 5
//*************************************************************************

val prob_5 = spark.sql("select md.user_id, " +
  "SUM(ABS(CAST(end_ts as decimal(20,0)) - " +
  " CAST(start_ts as decimal(20,0)))) as duration,batchid" +
  s" from music_data md LEFT OUTER JOIN subscribed_user su " +
  s"on md.user_id = su.user_id where md.status = 'pass' and md.batchid = $batch_id " +
  "and (md.user_id IS NULL or (CAST(md.u_timestamp as DECIMAL(20,0)) > " +
  " cast(su.end_dt as DECIMAL(20,0))))" +
  " group by md.user_id, batchid " +
  "order by duration desc limit 10").toDF()

prob_5.write.format("csv").option("header","true").save(s"/home/acadgild/project/output/unsubscribed_users/batch_$batch_id")
```

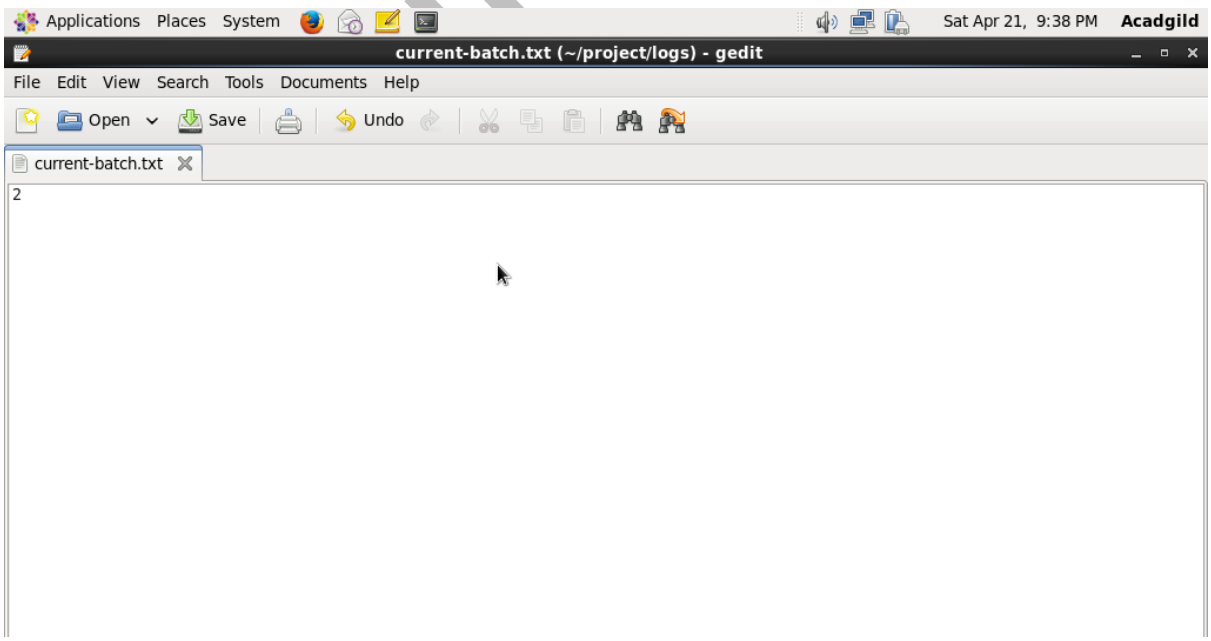


View of Log File after Data Analysis :

Incremented Batch



Incremented Batch id to 2



To check the output of Spark analysis in hive:

```
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>
```

[acadgild@l... | acadgild@l... | acadgild@l... | acad

```
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    166666.66666666666      1
U110    unsubscribed    S202    A303    -436711.2166666667      1
U118    subscribed      S202    A300    0.0     1
U104    subscribed      S202    A303    -166666.66666666666     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    -1666.6666666666667     1
U113    subscribed      S203    A303    0.0     1
U103    unsubscribed    S204    A300    -480116.76666666666     1
U108    subscribed      S205    A304    166666.66666666666      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    166666.66666666666      1
S205    A304    166666.66666666666      1
S200    A302    87193.78333333334       1
S203    A303    0.0     1
S203    A304    -43788.23333333333      1
S206    A300    -43788.23333333333      1
S202    A304    -87193.78333333334      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>
```

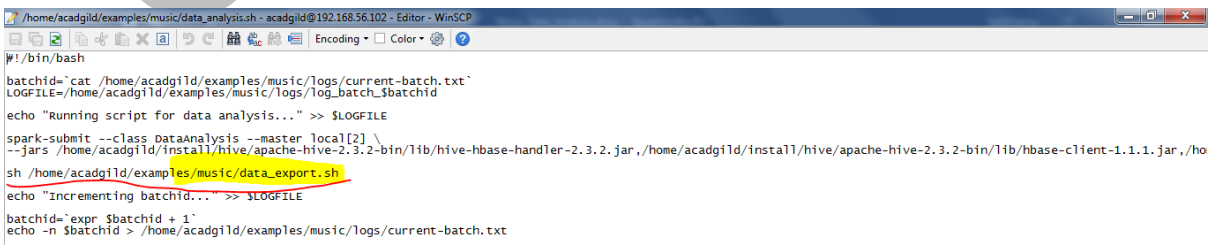

```
hive> select user_type,SUM(total_duration_in_minutes) from song_duration where tota
minutes) from song_duration where total_duration_in_minutes >=0 group by user_type;
Query ID = acadgild_20170911202020_14573a7d-939c-410f-aff0-0de67d02cbc2
Total jobs = 1
Launching Job 1 out of 1
Number of reduce 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_1505113626668_0001, Tracking URL = http://localhost:8088/proxy/a
Kill Command = /home/acadgild/hadoop-2.6.0/bin/hadoop job  -kill job_1505113626668
```

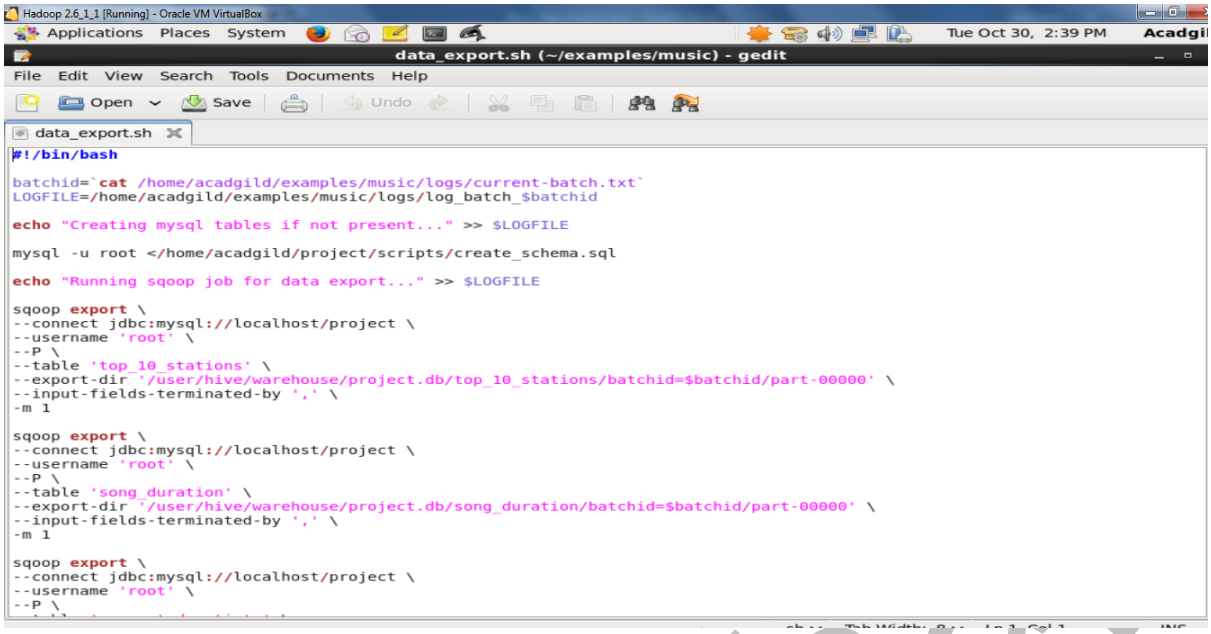
```
MapReduce Total cumulative CPU time: 4 seconds 850 msec
Ended Job = job_1505113626668_0001
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 4.85 sec   HDFS Read: 1102 HDFS Write: 30 SUCCESS
Total MapReduce CPU Time Spent: 4 seconds 850 msec
OK
subscribed      1904665.6500000001
Time taken: 30.85 seconds, Fetched: 1 row(s)
hive>
```

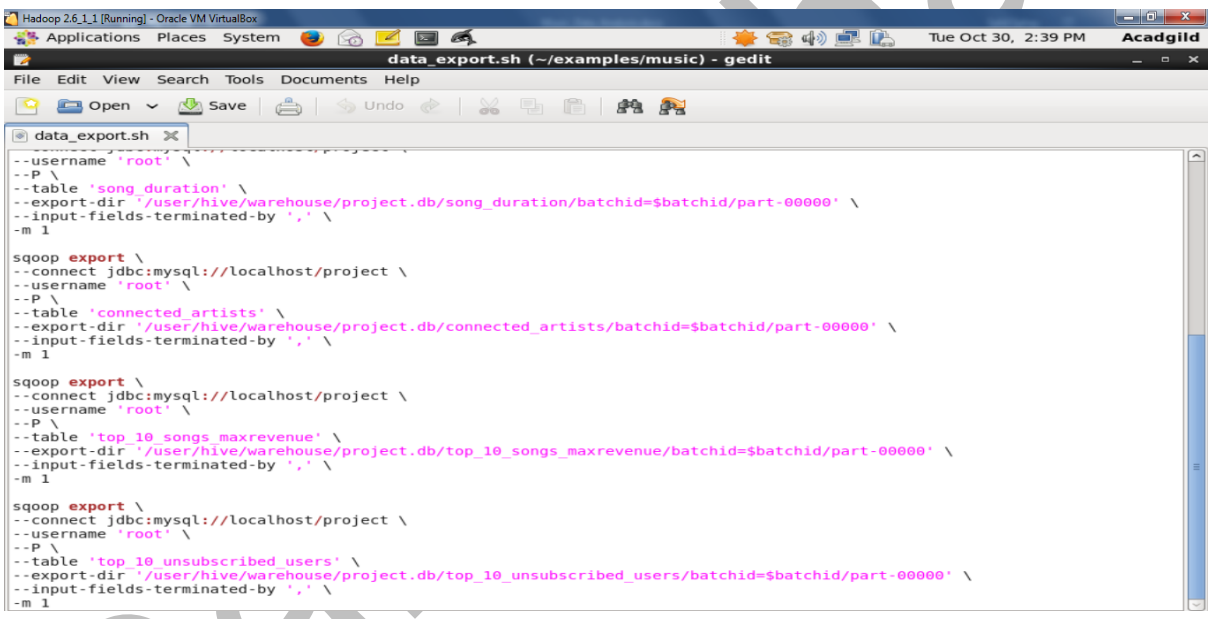
Now we have seen all the spark queries creating the tables for each query. So Data Analysis using Spark is executed successfully.

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.







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

```
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_royality_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)
```

File Edit View Search Terminal Help

```
mysql> select * from top_10_royality_songs;
+---------+-----------+-------------------+
| song_id | artist_id | duration          |
+---------+-----------+-------------------+
| S209    | A303      |            523905 |
| S202    | A300      |  166666.666666667 |
| S205    | A304      |  166666.666666667 |
| S200    | A302      |  87193.7833333333 |
| S203    | A303      |                 0 |
| S203    | A304      | -43788.2333333333 |
| S206    | A300      | -43788.2333333333 |
| S202    | A304      | -87193.7833333333 |
| S206    | A303      | -333333.333333333 |
| 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)
```

```
File  Edit  View  Search  Terminal  Help
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      | -333333.333333333 |
| U105    | unsubscribed | S207    | A300      | -333333.333333333 |
| 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)
```

Conclusion:

Perform Data Analysis for custom generated data by embedding all the scripts within a single script called **music_project_master.sh**

********** **End** **********