Assignment Submission-Final Project (Music Data Analysis)

1. Project Description:

A leading music-catering company is planning to analyze large amount of data received from varieties of sources, namely mobile app and website to track the behavior 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.

1.1 Fields present in the data files

Data files contain below fields.

Column Name/Field Name	Column Description/Field Description
User_id	Unique identifier of every user
Song_id	Unique identifier of every song
Artist_id	Unique identifier of the lead artist of the song
Timestamp	Timestamp when the record was generated
Start_ts	Start timestamp when the song started to play
End_ts	End timestamp when the song was stopped
Geo_cd	Can be 'A' for USA region, 'AP' for asia pacific region,'J' for Japan region, 'E' for europe and 'AU' for australia region
Station_id	Unique identifier of the station from where the song was played
Song_end_type	How the song was terminated. 0 means completed successfully 1 means song was skipped 2 means song was paused 3 means other type of failure like device issue, network error etc.
Like	0 means song was not likedsong was played 1 means song was liked
Dislike	0 means song was not disliked 1 means song was disliked

1.2 LookUp Tables

There is some existing look up tables present in **NoSQL** databases. They play an important role in data enrichment and analysis.

Table Name	Description
Station_Geo_Map	Contains mapping of a geo_cd with station_id
Subscribed_Users	Contains user_id, subscription_start_date and subscription_end_date. Contains details only for subscribed users
Song_Artist_Map	Contains mapping of song_id with artist_id alongwith royalty associated with each play of the song
User_Artist_Map	Contains an array of artist_id(s) followed by a user_id

1.3 Data Ingestion and Initial Validation

Below is the link for datasets.

https://drive.google.com/drive/folders/0B_P3pWagdIrrMjJGVlNsSUEtbG8

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 files come every 3 hours.
4. All the timestamp fields in data coming from web application is of the format YYYY-MM-DD HH:MM:SS.
5. All the timestamp fields in data coming from mobile application is a long integer interpreted as UNIX timestamps.
6. Finally, all timestamps must have the format of a long integer to be interpreted as UNIX timestamps.
7. If both like and dislike are 1, consider that record to be invalid.
8. If any of the fields from User_id, Song_id, Timestamp, Start_ts, End_ts, Geo_cd is NULL or absent, consider that record to be invalid.
9. If Song_end_type is NULL or absent, treat it to be 3.
10. Create a temporary identfier for all the data files received in the last 3 hours (may be an integer batch_id which is auto incremented or a string obtained after combining current date and current hour, to keep track of valid and invalid records per batch).

1.4 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.

NULL or absent field	Look up field	Look up table (Table from which record can be updated)
Geo_cd	Station_id	Station_Geo_Map
Artist_id	Song_id	Song_Artist_Map

1.5 Data Analysis

It is not only the data which is important, rather it is the insight it can be used to generate important. Once we have made the data ready for analysis, we have to perform below analysis on a daily basis.

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 presents 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 that 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.

PROJECT IMPLEMENTATION

1. Data creation:

We have generated data through python scripts. Those python scripts are :

generate_web_data.py
generate_mob_data.py

Data coming from web applications reside in **/home/acadgild/examples/music/data/web** and has xml format and that coming from mobile applications reside in **/home/acadgild/examples/music/data/mob** and has text format.

The batch file **"music_project_master.sh"** does data creation through python scripts. Please find below script which is part of **music_project_master.sh:**

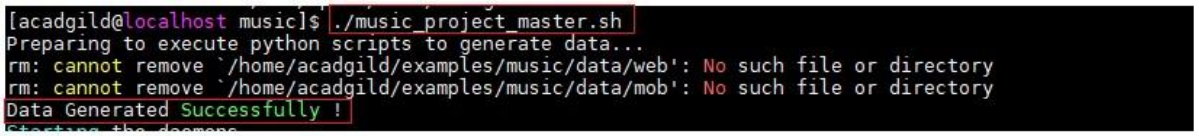
```
**************************************************************************
***

# Create data

echo "Preparing to execute python scripts to generate data..."

rm -r /home/acadgild/examples/music/data/web


rm -r /home/acadgild/examples/music/data/mob

mkdir -p /home/acadgild/examples/music/data/web

mkdir -p /home/acadgild/examples/music/data/mob


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


echo "Data Generated Successfully !"


**************************************************************************
***
```

So here it will first remove web and mob directories, if they are present already inside directory

/home/acadgild/examples/music/data.

Then it will create web and mob directories inside directory

/home/acadgild/examples/music/data.



2. Start Hadoop Daemons:

Created a batch file "**start-daemon.sh**".Through this batch file, these daemons are started. Please find below :

```
**************************************************************************
***

#!/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


**************************************************************************
```

Here, it will first remove logs directory, if they are present inside directory **/home/acadgild/examples/music/.**

Then it will create **logs** directory inside directory **/home/acadgild/examples/music/.**

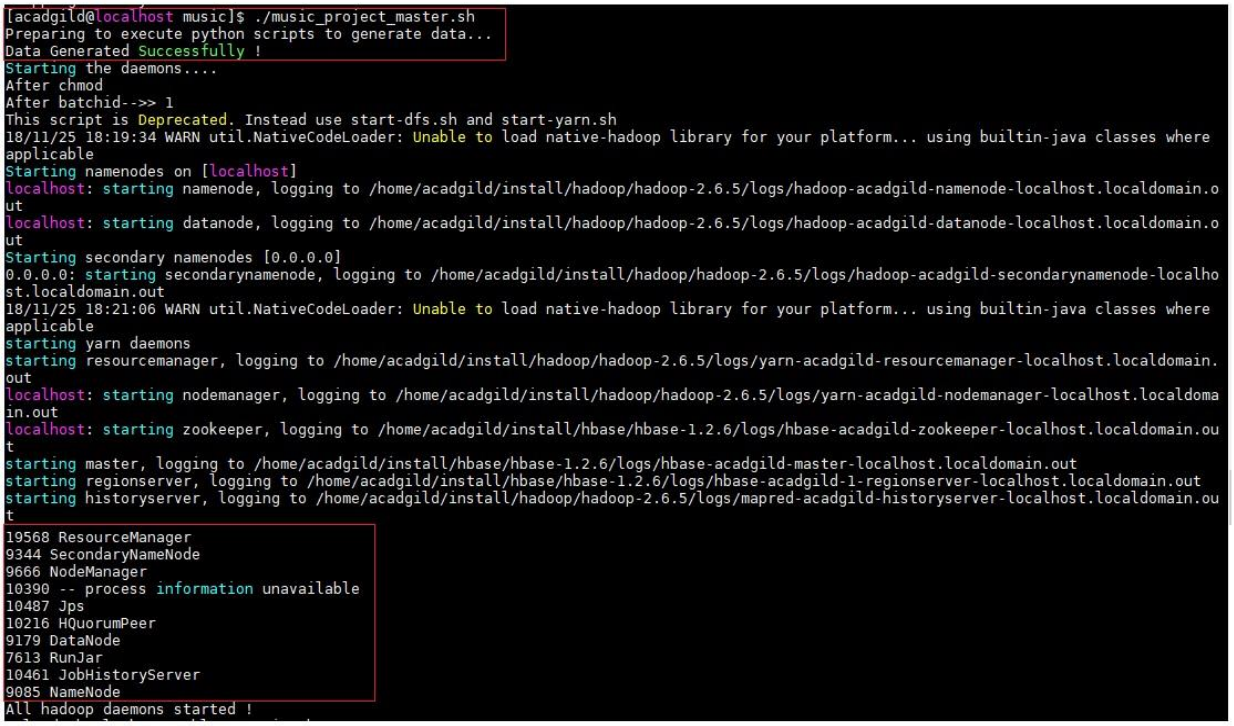
After this, it will search for **current-batch.txt** file inside directory: **/home/acadgild/examples/music/logs.**

If it is present, then message will be present as "Batch File Found" else it will create **current-batch.txt** file inside directory: **/home/acadgild/examples/music/logs** with content as **'1'.**

After this required permissions would be given for this file.

“batched” would be content of **current-batch.txt** file. i.e **1** . After this, log_batch_1 file as Logfile would be created inside directory **/home/acadgild/examples/music/logs/.**

“**start-daemon.sh**” batch file will start though **music_project_master.sh** batch file.



3. Populate Look up tables in HBase:

By using the **“populate-lookup.sh”** script, we will create below lookup tables in HBase. These tables we are using for Data formatting, Data enrichment and Analysis stage.

Table Name	Description
Station_Geo_Map	Contains mapping of a geo_cd with station_id
Subscribed_Users	Contains user_id, subscription_start_date and subscription_end_date. Contains details only for subscribed users
Song_Artist_Map	Contains mapping of song_id with artist_id alongwith royalty associated with each play of the song
User_Artist_Map	Contains an array of artist_id(s) followed by a user_id

The “**populate-lookup.sh**” shell script creates the above lookup tables in the HBase and populate the data into the lookup tables from the dataset files.

```
**************************************************************

******

#!/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
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"

**********************************************************************

******
```

Below screen shots shows the tables creation and population of the data in the HBase. Here we are executing **populate-lookup.sh** via music_project_master.sh batch file.

```
drop 'subscribed-users'
0 row(s) in 3.9400 seconds

2018-11-25 22:05:08,393 WARN  [main] util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java
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/StaticLoggerBinder
.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/i
mpl/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

disable 'song-artist-map'
0 row(s) in 5.0760 seconds

2018-11-25 22:06:01,334 WARN  [main] util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java
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/StaticLoggerBinder
.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/i
mpl/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

drop 'song-artist-map'
0 row(s) in 4.2220 seconds

2018-11-25 22:06:55,326 WARN  [main] util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java
classes where applicable
```

```
create 'station-geo-map', 'geo'
0 row(s) in 4.9870 seconds

Hbase::Table - station-geo-map
2018-11-25 22:07:51,435 WARN  [main] util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java
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/StaticLoggerBinder
.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/i
mpl/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

create 'subscribed-users', 'subscn'
0 row(s) in 4.1690 seconds

Hbase::Table - subscribed-users
2018-11-25 22:08:46,537 WARN  [main] util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java
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/StaticLoggerBinder
.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/i
mpl/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

create 'song-artist-map', 'artist'
0 row(s) in 3.8860 seconds
```

Please find the tables with the values stored by script - populate-lookup.sh

```
hbase(main):002:0> scan 'song-artist-map'
ROW                              COLUMN+CELL
 S200                            column=artist:artistid, timestamp=1532277052808, value=A300
 S201                            column=artist:artistid, timestamp=1532277063975, value=A301
 S202                            column=artist:artistid, timestamp=1532277074927, value=A302
 S203                            column=artist:artistid, timestamp=1532277085940, value=A303
 S204                            column=artist:artistid, timestamp=1532277096508, value=A304
 S205                            column=artist:artistid, timestamp=1532277107380, value=A301
 S206                            column=artist:artistid, timestamp=1532277117916, value=A302
 S207                            column=artist:artistid, timestamp=1532277128708, value=A303
 S208                            column=artist:artistid, timestamp=1532277139626, value=A304
9 row(s) in 0.2440 seconds
```

```
hbase(main):003:0> scan 'station-geo-map'
ROW                                COLUMN+CELL
 ST400                             column=geo:geo_cd, timestamp=1532276901571, value=A
 ST401                             column=geo:geo_cd, timestamp=1532276912108, value=AU
 ST402                             column=geo:geo_cd, timestamp=1532276922831, value=AP
 ST403                             column=geo:geo_cd, timestamp=1532276933380, value=J
 ST404                             column=geo:geo_cd, timestamp=1532276944269, value=E
 ST405                             column=geo:geo_cd, timestamp=1532276954714, value=A
 ST406                             column=geo:geo_cd, timestamp=1532276966054, value=AU
 ST407                             column=geo:geo_cd, timestamp=1532276976538, value=AP
 ST408                             column=geo:geo_cd, timestamp=1532276987193, value=E
 ST409                             column=geo:geo_cd, timestamp=1532276998216, value=E
 ST410                             column=geo:geo_cd, timestamp=1532277009161, value=A
 ST411                             column=geo:geo_cd, timestamp=1532277020083, value=A
 ST412                             column=geo:geo_cd, timestamp=1532277030853, value=AP
 ST413                             column=geo:geo_cd, timestamp=1532277041902, value=J
14 row(s) in 0.1260 seconds
```

```
hbase(main):004:0> scan 'subscribed-users'
ROW                              COLUMN+CELL
 U100                            column=subscn:enddt, timestamp=1532277161513, value=1465130523
 U100                            column=subscn:startdt, timestamp=1532277150572, value=1465230523
 U101                            column=subscn:enddt, timestamp=1532277183558, value=1475130523
 U101                            column=subscn:startdt, timestamp=1532277172591, value=1465230523
 U102                            column=subscn:enddt, timestamp=1532277205001, value=1475130523
 U102                            column=subscn:startdt, timestamp=1532277194398, value=1465230523
 U103                            column=subscn:enddt, timestamp=1532277226044, value=1475130523
 U103                            column=subscn:startdt, timestamp=1532277215490, value=1465230523
 U104                            column=subscn:enddt, timestamp=1532277248517, value=1475130523
 U104                            column=subscn:startdt, timestamp=1532277237099, value=1465230523
 U105                            column=subscn:enddt, timestamp=1532277270534, value=1475130523
 U105                            column=subscn:startdt, timestamp=1532277259547, value=1465230523
 U106                            column=subscn:enddt, timestamp=1532277292198, value=1485130523
 U106                            column=subscn:startdt, timestamp=1532277281420, value=1465230523
 U107                            column=subscn:enddt, timestamp=1532277313425, value=1455130523
 U107                            column=subscn:startdt, timestamp=1532277302798, value=1465230523
 U108                            column=subscn:enddt, timestamp=1532277334643, value=1465230623
 U108                            column=subscn:startdt, timestamp=1532277323818, value=1465230523
 U109                            column=subscn:enddt, timestamp=1532277356273, value=1475130523
 U109                            column=subscn:startdt, timestamp=1532277345393, value=1465230523
 U110                            column=subscn:enddt, timestamp=1532277378160, value=1475130523
 U110                            column=subscn:startdt, timestamp=1532277367228, value=1465230523
 U111                            column=subscn:enddt, timestamp=1532277399844, value=1475130523
 U111                            column=subscn:startdt, timestamp=1532277389179, value=1465230523
 U112                            column=subscn:enddt, timestamp=1532277421027, value=1475130523
 U112                            column=subscn:startdt, timestamp=1532277410399, value=1465230523
 U113                            column=subscn:enddt, timestamp=1532277442510, value=1485130523
 U113                            column=subscn:startdt, timestamp=1532277431728, value=1465230523
14 row(s) in 0.1340 seconds
```

4. **Creating Hive Tables on the top of HBase:**

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 script will run the **"create_hive_hbase_lookup.hql"** which will create the HIVE external tables.

data_enrichment_filtering_schema.sh script :

```
**********************************************************************

#!/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
```

"create_hive_hbase_lookup.hql" script :

```
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");


**********************************************************
******
```

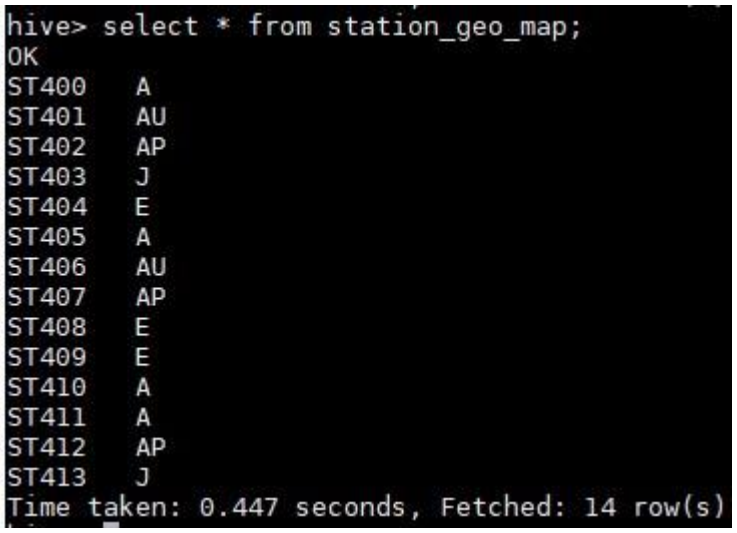
We are running **data_enrichment_filtering_schema.sh** script through the execution of **music_project_master.sh** script.

The below screenshot we can see tables are getting created in hive by running the "**data_enrichement_filtering_schema.sh** file".

```
hive> use project;
OK
Time taken: 5.868 seconds
hive> show tables;
OK
song_artist_map
station_geo_map
subscribed_users
users_artists
Time taken: 0.329 seconds, Fetched: 4 row(s)
```

```
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
Time taken: 4.118 seconds, Fetched: 9 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
Time taken: 0.604 seconds, Fetched: 14 row(s)
```



5. **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.
   dataformatting.sh script :

```
************************************************************
******

#!/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 formatting..." >> $LOGFILE


spark-submit --packages com.databricks:spark-xml_2.10:0.4.1 \
--class DataFormatting \
--master local[2] \
/home/acadgild/examples/music/MusicDataAnalysis/target/scala-
   2.11/musicdataanalysis_2.11-1.0.jar $batchid


************************************************************
******
```

DataFormatting.scala

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

object DataFormatting {
  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 project.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 ','
                              """

   val load_mob_data = s"""LOAD DATA LOCAL INPATH 'file:///home/acadgild/examples/music/data/mob/file.txt'
                          INTO TABLE project.formatted_input PARTITION (batchid='$batchId')"""

   val load_web_data = s"""INSERT INTO project.formatted_input
                          PARTITION(batchid='$batchId')
                          SELECT user_id,
                          song_id,
                          artist_id,
                          unix_timestamp(timestamp,'yyyy-MM-dd HH:mm:ss') AS timestamp,
                          unix_timestamp(start_ts,'yyyy-MM-dd HH:mm:ss') AS start_ts,
                          unix_timestamp(end_ts,'yyyy-MM-dd HH:mm:ss') AS end_ts,
                          geo_cd,
                          station_id,
                          song_end_type,
                          like,
                          dislike
                          FROM web_data
                          """


  try {
       val xmlData = sqlContext.read.format("com.databricks.spark.xml").option("rowTag", "record").load("file:///home/acadgild/examples/music/data/web/file.xml"
)
       xmlData.createOrReplaceTempView("web_data")

       sqlContext.sql(create_hive_table)
       sqlContext.sql(load_mob_data)
       sqlContext.sql(load_web_data)
     }
    catch{
     case e: Exception=>e.printStackTrace()
    }
 }
}
```

We have build.sbt file inside **MusicDataAnalysis** folder to create jar file :

```
[acadgild@localhost music]$ cd MusicDataAnalysis
[acadgild@localhost MusicDataAnalysis]$ ls -ls
total 8
4 -rw-rw-r--. 1 acadgild acadgild  802 Dec  1 18:34 build.sbt
4 drwxrwxr-x. 3 acadgild acadgild 4096 Dec  1 18:34 src
```

Below is the command to create jar file in verbose mode:

```
[acadgild@localhost MusicDataAnalysis]$ sbt -v package
[process_args] java_version = '1.8'
# Executing command line:
java
-Xms1024m
-Xmx1024m
-XX:ReservedCodeCacheSize=128m
-XX:MaxMetaspaceSize=256m
-jar
/usr/share/sbt/bin/sbt-launch.jar
package

Getting org.scala-sbt sbt 1.0.4  (this may take some time)...
downloading https://repo1.maven.org/maven2/org/scala-sbt/sbt/1.0.4/sbt-1.0.4.jar ...
        [SUCCESSFUL ] org.scala-sbt#sbt;1.0.4!sbt.jar (910ms)
downloading https://repo1.maven.org/maven2/org/scala-lang/scala-library/2.12.4/scala-library-2.12.4.jar ...
        [SUCCESSFUL ] org.scala-lang#scala-library;2.12.4!scala-library.jar (22703ms)
downloading https://repo1.maven.org/maven2/org/scala-sbt/main_2.12/1.0.4/main_2.12-1.0.4.jar ...
        [SUCCESSFUL ] org.scala-sbt#main_2.12;1.0.4!main_2.12.jar (6609ms)
downloading https://repo1.maven.org/maven2/org/scala-sbt/logic_2.12/1.0.4/logic_2.12-1.0.4.jar ...
        [SUCCESSFUL ] org.scala-sbt#logic_2.12;1.0.4!logic_2.12.jar (632ms)
downloading https://repo1.maven.org/maven2/org/scala-sbt/actions_2.12/1.0.4/actions_2.12-1.0.4.jar ...
        [SUCCESSFUL ] org.scala-sbt#actions_2.12;1.0.4!actions_2.12.jar (1258ms)
downloading https://repo1.maven.org/maven2/org/scala-sbt/main-settings_2.12/1.0.4/main-settings_2.12-1.0.4.jar ...
        [SUCCESSFUL ] org.scala-sbt#main-settings_2.12;1.0.4!main-settings_2.12.jar (2525ms)
```

Below is the location of Jar file which gets created under **/MusicDataAnalysis/target/scala-2.11 :**

```
[acadgild@localhost MusicDataAnalysis]$ ls -ls
total 16
4 -rw-rw-r--. 1 acadgild acadgild  802 Dec  1 18:34 build.sbt
4 drwxrwxr-x. 3 acadgild acadgild 4096 Dec  1 18:52 project
4 drwxrwxr-x. 3 acadgild acadgild 4096 Dec  1 18:34 src
4 drwxrwxr-x. 4 acadgild acadgild 4096 Dec  1 18:58 target
You have new mail in /var/spool/mail/acadgild
[acadgild@localhost MusicDataAnalysis]$ cd target
[acadgild@localhost target]$ ls -ls
total 8
4 drwxrwxr-x. 4 acadgild acadgild 4096 Dec  1 19:12 scala-2.11
4 drwxrwxr-x. 4 acadgild acadgild 4096 Dec  1 18:53 streams
[acadgild@localhost target]$ cd scala-2.11
[acadgild@localhost scala-2.11]$ ls -ls
total 16
4 drwxrwxr-x. 2 acadgild acadgild 4096 Dec  1 19:12 classes
8 -rw-rw-r--. 1 acadgild acadgild 8183 Dec  1 19:12 musicdataanalysis_2.11-1.0.jar
4 drwxrwxr-x. 5 acadgild acadgild 4096 Dec  1 19:10 resolution-cache
```

Scala programs related to data lies in the location below:

```
[acadgild@localhost MusicDataAnalysis]$ ls -ls
total 16
4 -rw-rw-r--. 1 acadgild acadgild  802 Dec  1 18:34 build.sbt
4 drwxrwxr-x. 3 acadgild acadgild 4096 Dec  1 18:52 project
4 drwxrwxr-x. 3 acadgild acadgild 4096 Dec  1 18:34 src
4 drwxrwxr-x. 4 acadgild acadgild 4096 Dec  1 18:58 target
[acadgild@localhost MusicDataAnalysis]$ cd src
[acadgild@localhost src]$ ls -ls
total 4
4 drwxrwxr-x. 3 acadgild acadgild 4096 Dec  1 18:34 main
[acadgild@localhost src]$ cd main
[acadgild@localhost main]$ ls -ls
total 4
4 drwxrwxr-x. 2 acadgild acadgild 4096 Dec  1 18:40 scala
[acadgild@localhost main]$ cd scala
[acadgild@localhost scala]$ ls -ls
total 20
8 -rw-rw-r--. 1 acadgild acadgild 4814 Dec  1 18:34 DataAnalysis.scala
4 -rw-rw-r--. 1 acadgild acadgild 3264 Dec  1 18:34 DataEnrichment.scala
4 -rw-rw-r--. 1 acadgild acadgild 2620 Dec  1 18:40 DataFormatting.scala
```

We are executing master script which internally calls **dataformatting.sh** which performs data formatting :

```
[acadgild@localhost music]$ ./music_project_master.sh
Preparing to execute python scripts to generate data...
Data Generated Successfully !
Starting the daemons....
13921 ResourceManager
14722 HRegionServer
14019 NodeManager
14791 JobHistoryServer
14631 HMaster
14571 HQuorumPeer
29232 RunJar
17489 Main
24311 RunJar
13692 SecondaryNameNode
30268 Jps
13502 DataNode
13407 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....
Ivy Default Cache set to: /home/acadgild/.ivy2/cache
The jars for the packages stored in: /home/acadgild/.ivy2/jars
:: loading settings :: url = jar:file:/home/acadgild/install/spark/spark-2.2.1-bin-hadoop2.7/jars/ivy-2.4.0.jar!/org/apache/ivy/core/sett
ings/ivysettings.xml
com.databricks#spark-xml_2.10 added as a dependency
:: resolving dependencies :: org.apache.spark#spark-submit-parent;1.0
        confs: [default]
        found com.databricks#spark-xml_2.10;0.4.1 in central
:: resolution report :: resolve 1398ms :: artifacts dl 34ms
        :: modules in use:
        com.databricks#spark-xml_2.10;0.4.1 from central in [default]
        ---------------------------------------------------------------------
        |                  |            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/93ms)
18/12/01 20:15:29 INFO spark.SparkContext: Running Spark version 2.2.1
18/12/01 20:15:31 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where
applicable
18/12/01 20:15:33 WARN util.Utils: Your hostname, localhost.localdomain resolves to a loopback address: 127.0.0.1; using 192.168.0.102 in
stead (on interface eth15)
18/12/01 20:15:33 WARN util.Utils: Set SPARK_LOCAL_IP if you need to bind to another address
18/12/01 20:15:33 INFO spark.SparkContext: Submitted application: Data Formatting
18/12/01 20:15:33 INFO spark.SecurityManager: Changing view acls to: acadgild
18/12/01 20:15:33 INFO spark.SecurityManager: Changing modify acls to: acadgild
18/12/01 20:15:33 INFO spark.SecurityManager: Changing view acls groups to:
18/12/01 20:15:33 INFO spark.SecurityManager: Changing modify acls groups to:
18/12/01 20:15:33 INFO spark.SecurityManager: SecurityManager: authentication disabled; ui acls disabled; users  with view permissions: S
et(acadgild); groups with view permissions: Set(); users  with modify permissions: Set(acadgild); groups with modify permissions: Set()
18/12/01 20:15:36 INFO util.Utils: Successfully started service 'sparkDriver' on port 35422.
18/12/01 20:15:36 INFO spark.SparkEnv: Registering MapOutputTracker
```

```
18/12/01 20:17:20 INFO metastore.HiveMetaStore: 0: get_database: project
18/12/01 20:17:20 INFO HiveMetaStore.audit: ugi=acadgild        ip=unknown-ip-addr      cmd=get_database: project
18/12/01 20:17:20 INFO metastore.HiveMetaStore: 0: get_table : db=project tbl=formatted_input
18/12/01 20:17:20 INFO HiveMetaStore.audit: ugi=acadgild        ip=unknown-ip-addr      cmd=get_table : db=project tbl=formatted_input
18/12/01 20:17:20 INFO metastore.HiveMetaStore: 0: get_table : db=project tbl=formatted_input
18/12/01 20:17:20 INFO HiveMetaStore.audit: ugi=acadgild        ip=unknown-ip-addr      cmd=get_table : db=project tbl=formatted_input
18/12/01 20:17:20 INFO parser.CatalystSqlParser: Parsing command: int
18/12/01 20:17:20 INFO parser.CatalystSqlParser: Parsing command: string
18/12/01 20:17:20 INFO parser.CatalystSqlParser: Parsing command: string
18/12/01 20:17:20 INFO parser.CatalystSqlParser: Parsing command: string
18/12/01 20:17:20 INFO parser.CatalystSqlParser: Parsing command: string
18/12/01 20:17:21 INFO parser.CatalystSqlParser: Parsing command: string
18/12/01 20:17:21 INFO parser.CatalystSqlParser: Parsing command: string
18/12/01 20:17:21 INFO parser.CatalystSqlParser: Parsing command: string
18/12/01 20:17:21 INFO parser.CatalystSqlParser: Parsing command: int
18/12/01 20:17:21 INFO parser.CatalystSqlParser: Parsing command: int
18/12/01 20:17:21 INFO parser.CatalystSqlParser: Parsing command: int
18/12/01 20:17:21 INFO spark.SparkContext: Invoking stop() from shutdown hook
18/12/01 20:17:21 INFO server.AbstractConnector: Stopped Spark@15dd1d35{HTTP/1.1,[http/1.1]}{0.0.0.0:4040}
18/12/01 20:17:21 INFO ui.SparkUI: Stopped Spark web UI at http://192.168.0.102:4040
18/12/01 20:17:21 INFO spark.MapOutputTrackerMasterEndpoint: MapOutputTrackerMasterEndpoint stopped!
18/12/01 20:17:21 INFO memory.MemoryStore: MemoryStore cleared
18/12/01 20:17:21 INFO storage.BlockManager: BlockManager stopped
18/12/01 20:17:21 INFO storage.BlockManagerMaster: BlockManagerMaster stopped
18/12/01 20:17:21 INFO scheduler.OutputCommitCoordinator$OutputCommitCoordinatorEndpoint: OutputCommitCoordinator stopped!
18/12/01 20:17:21 INFO spark.SparkContext: Successfully stopped SparkContext
18/12/01 20:17:21 INFO util.ShutdownHookManager: Shutdown hook called
18/12/01 20:17:21 INFO util.ShutdownHookManager: Deleting directory /tmp/spark-3a3b142c-3ae0-4ae8-8bbb-4a3289f36a9b
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 new mail in /var/spool/mail/acadgild
```

Below hive table **formatted_input** gets created which contains all data which gets merged from web and mobile applications (file.txt and file.xml) :

```
hive> show tables;
OK
formatted_input
song_artist_map
station_geo_map
subscribed_users
Time taken: 0.221 seconds, Fetched: 4 row(s)
hive> select * from formatted_input;
OK
U120    S203    A302    1495130523      1475130523      1465230523      AP      ST410   3       0       1       1
U106    S203    A303    1495130523      1465130523      1485130523      AU      ST403   0       1       0       1
U119    S204    A302    1475130523      1485130523      1475130523      E       ST403   0       0       1       1
U108    S200    A301    1475130523      1485130523      1485130523      U       ST410   2       0       1       1
U115    S202    A305    1475130523      1475130523      1465130523      A       ST403   2       0       0       1
        S206    A304    1495130523      1485130523      1475130523      AU      ST404   1       1       1       1
U101    S202    A300    1495130523      1475130523      1485130523      AU      ST406   3       0       1       1
U105    S208    A301    1465230523      1465230523      1475130523      U       ST400   3       1       1       1
U101    S201    A302    1465230523      1465130523      1475130523              ST412   1       0       0       1
U112    S203            1465130523      1465130523      1475130523      E       ST406   0       1       1       1
U110    S209    A303    1495130523      1475130523      1475130523      U       ST406   0       1       0       1
U100    S207    A300    1475130523      1485130523      1485130523      E       ST413   1       1       1       1
U103    S202    A301    1465130523      1475130523      1485130523      A       ST404   1       1       1       1
U109    S203    A301    1465130523      1485130523      1485130523      E       ST415   1       1       0       1
U102    S204    A301    1465230523      1485130523      1475130523      E       ST411   0       0       0       1
U111    S200    A303    1495130523      1465230523      1465230523      E       ST404   2       0       0       1
U107    S205    A301    1465130523      1475130523      1465230523      AU      ST409   1       1       0       1
U114    S210    A302    1465130523      1465230523      1475130523      A       ST409   0       0       1       1
U109    S200    A301    1465230523      1485130523      1485130523      AP      ST407   0       0       0       1
U110    S200    A300    1465230523      1485130523      1475130523      AP      ST404   1       1       1       1
U105    S205    A300    1465490556      1462863262      1462863262      E       ST407   0       1       1       1
U100    S205    A304    1468094889      1468094889      1465490556      AU      ST415   2       0       1       1
U100    S203    A302    1462863262      1468094889      1465490556      A       ST403   0       0       0       1
U119    S202    A304    1462863262      1465490556      1462863262      A       ST408   3       1       1       1
U114    S210    A305    1494297562      1468094889      1465490556      AP      ST409   2       1       0       1
NULL    S202    A304    1462863262      1462863262      1465490556      A       ST415   0       1       1       1
U109    S204    A300    1468094889      1494297562      1494297562      AU      ST403   3       0       1       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.

6. Data Enrichment and Cleaning:

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.

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.

```
**********************************************************
*

#!/bin/bash


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


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


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/acadgild/examples/music/processed_dir/ -mtime +7 -exec rm
{} \;




**********************************************************
*
**
```

We have executed data_enrichment.sh script by calling music_project_master.sh batch file as shown below:

```
[acadgild@localhost music]$ ./music_project_master.sh
Preparing to execute python scripts to generate data...
Data Generated Successfully !
Starting the daemons....
15888 RunJar
4528 HMaster
3890 NodeManager
3298 NameNode
5011 RunJar
3555 SecondaryNameNode
3417 DataNode
4714 JobHistoryServer
4635 HRegionServer
4460 HQuorumPeer
17007 Jps
3791 ResourceManager
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...
18/12/02 15:25:32 INFO spark.SparkContext: Running Spark version 2.2.1
18/12/02 15:25:35 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where
applicable
18/12/02 15:25:36 WARN util.Utils: Your hostname, localhost.localdomain resolves to a loopback address: 127.0.0.1; using 192.168.0.100 in
stead (on interface eth15)
18/12/02 15:25:36 WARN util.Utils: Set SPARK_LOCAL_IP if you need to bind to another address
18/12/02 15:25:37 INFO spark.SparkContext: Submitted application: Data Formatting
18/12/02 15:25:37 INFO spark.SecurityManager: Changing view acls to: acadgild
18/12/02 15:25:37 INFO spark.SecurityManager: Changing modify acls to: acadgild
18/12/02 15:25:37 INFO spark.SecurityManager: Changing view acls groups to:
18/12/02 15:25:37 INFO spark.SecurityManager: Changing modify acls groups to:
18/12/02 15:25:37 INFO spark.SecurityManager: SecurityManager: authentication disabled; ui acls disabled; users  with view permissions: S
et(acadgild); groups with view permissions: Set(); users  with modify permissions: Set(acadgild); groups with modify permissions: Set()
18/12/02 15:25:40 INFO util.Utils: Successfully started service 'sparkDriver' on port 36466.
```

```
18/12/02 15:28:44 INFO HiveMetaStore.audit: ugi=acadgild        ip=unknown-ip-addr      cmd=get_database: project
18/12/02 15:28:44 INFO metastore.HiveMetaStore: 0: get_table : db=project tbl=enriched_data
18/12/02 15:28:44 INFO HiveMetaStore.audit: ugi=acadgild        ip=unknown-ip-addr      cmd=get_table : db=project tbl=enriched_data
18/12/02 15:28:44 INFO metastore.HiveMetaStore: 0: get_table : db=project tbl=enriched_data
18/12/02 15:28:44 INFO HiveMetaStore.audit: ugi=acadgild        ip=unknown-ip-addr      cmd=get_table : db=project tbl=enriched_data
18/12/02 15:28:44 INFO parser.CatalystSqlParser: Parsing command: int
18/12/02 15:28:44 INFO parser.CatalystSqlParser: Parsing command: string
18/12/02 15:28:44 INFO parser.CatalystSqlParser: Parsing command: string
18/12/02 15:28:44 INFO parser.CatalystSqlParser: Parsing command: string
18/12/02 15:28:44 INFO parser.CatalystSqlParser: Parsing command: string
18/12/02 15:28:44 INFO parser.CatalystSqlParser: Parsing command: string
18/12/02 15:28:44 INFO parser.CatalystSqlParser: Parsing command: string
18/12/02 15:28:44 INFO parser.CatalystSqlParser: Parsing command: string
18/12/02 15:28:44 INFO parser.CatalystSqlParser: Parsing command: string
18/12/02 15:28:44 INFO parser.CatalystSqlParser: Parsing command: string
18/12/02 15:28:44 INFO parser.CatalystSqlParser: Parsing command: int
18/12/02 15:28:44 INFO parser.CatalystSqlParser: Parsing command: int
18/12/02 15:28:44 INFO parser.CatalystSqlParser: Parsing command: int
18/12/02 15:28:44 INFO spark.SparkContext: Invoking stop() from shutdown hook
18/12/02 15:28:44 INFO server.AbstractConnector: Stopped Spark@33ba7d11{HTTP/1.1,[http/1.1]}{0.0.0.0:4040}
18/12/02 15:28:44 INFO ui.SparkUI: Stopped Spark web UI at http://192.168.0.100:4040
18/12/02 15:28:45 INFO spark.MapOutputTrackerMasterEndpoint: MapOutputTrackerMasterEndpoint stopped!
18/12/02 15:28:45 INFO memory.MemoryStore: MemoryStore cleared
18/12/02 15:28:45 INFO storage.BlockManager: BlockManager stopped
18/12/02 15:28:45 INFO storage.BlockManagerMaster: BlockManagerMaster stopped
18/12/02 15:28:45 INFO scheduler.OutputCommitCoordinator$OutputCommitCoordinatorEndpoint: OutputCommitCoordinator stopped!
18/12/02 15:28:45 INFO spark.SparkContext: Successfully stopped SparkContext
18/12/02 15:28:45 INFO util.ShutdownHookManager: Shutdown hook called
18/12/02 15:28:45 INFO util.ShutdownHookManager: Deleting directory /tmp/spark-c8a1afe1-ed1c-4784-a537-66d2b0ea2e65
18/12/02 15:28:53 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where
applicable
18/12/02 15:29:06 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where
applicable
Data Enrichment Complete
Lets run some use cases now...
USE CASES COMPLETE !!
You have new mail in /var/spool/mail/acadgild
[acadgild@localhost music]$
```

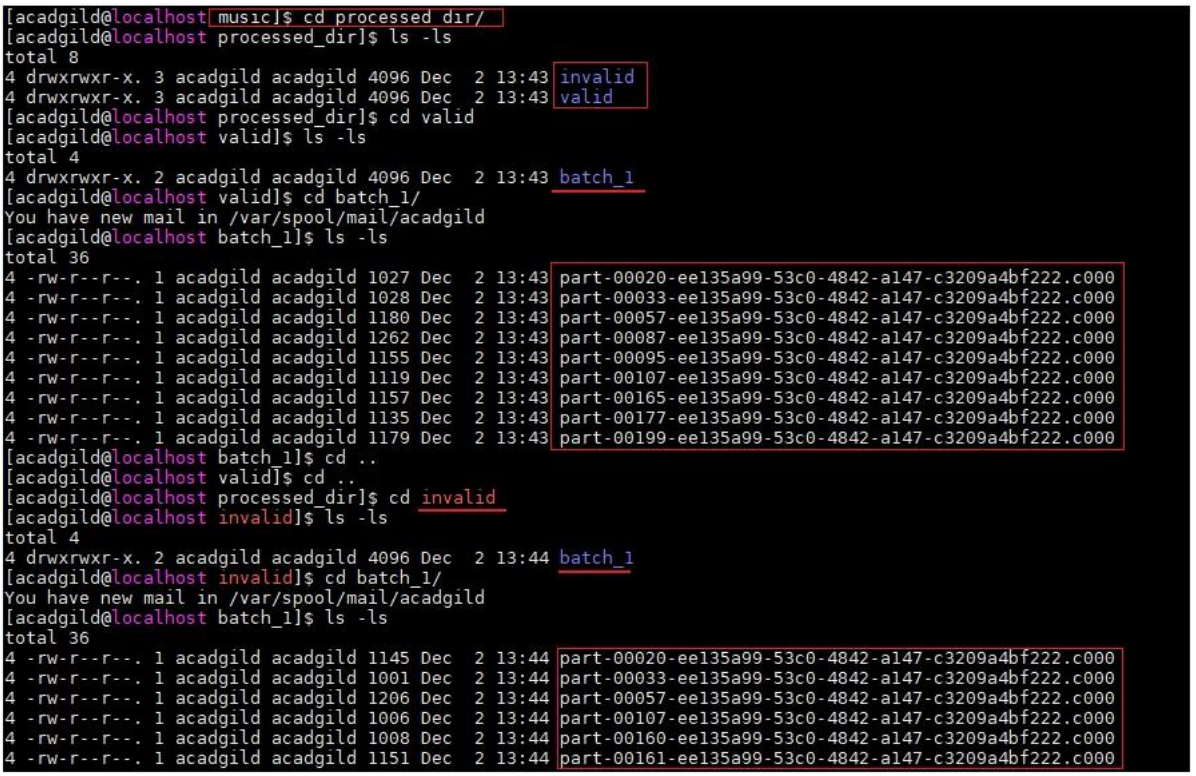
Data enrichment tables:

```
hive> show databases;
OK
default
project
Time taken: 4.22 seconds, Fetched: 2 row(s)
hive> use project;
OK
Time taken: 0.116 seconds
hive> show tables;
OK
enriched_data
formatted_input
song_artist_map
station_geo_map
subscribed_users
Time taken: 0.251 seconds, Fetched: 5 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

```
hive> select * from enriched_data;
OK
U111    S201    A301    1465490556      1494297562      1465490556      J       ST403   1       1       1       1       fail
U101    S201    A301    1465230523      1465130523      1475130523      AP      ST412   1       0       0       1       fail
U100    S207    A303    1475130523      1485130523      1485130523      J       ST413   1       1       1       1       fail
U103    S202    A302    1465130523      1475130523      1485130523      E       ST404   1       1       1       1       fail
U119    S202    A302    1462863262      1465490556      1462863262      E       ST408   3       1       1       1       fail
NULL    S202    A302    1462863262      1462863262      1465490556      NULL    ST415   0       1       1       1       fail
        S206    A302    1495130523      1485130523      1475130523      E       ST404   1       1       1       1       fail
U105    S208    A304    1465230523      1465230523      1475130523      A       ST400   3       1       1       1       fail
U114    S210    NULL    1465130523      1465230523      1475130523      E       ST409   0       0       1       1       fail
U114    S210    NULL    1494297562      1468094889      1465490556      E       ST409   2       1       0       1       fail
U108    S205    A301    1462863262      1468094889      1465490556      A       ST410   1       1       1       1       fail
U105    S205    A301    1465490556      1462863262      1462863262      AP      ST407   0       1       1       1       fail
U100    S205    A301    1468094889      1468094889      1465490556      NULL    ST415   2       0       1       1       fail
U110    S200    A300    1465230523      1485130523      1475130523      E       ST404   1       1       1       1       fail
U113    S203    A303    1465490556      1465490556      1468094889      AP      ST407   0       0       0       1       fail
U109    S203    A303    1465130523      1485130523      1485130523      NULL    ST415   1       1       0       1       fail
U114    S203    A303    1494297562      1462863262      1468094889      NULL    ST415   3       1       0       1       fail
U112    S203    A303    1465130523      1465130523      1475130523      AU      ST406   0       1       1       1       fail
U106    S201    A301    1468094889      1462863262      1462863262      J       ST403   2       0       1       1       pass
U106    S207    A303    1494297562      1494297562      1468094889      E       ST404   3       0       1       1       pass
U117    S202    A302    1462863262      1465490556      1465490556      E       ST404   0       1       0       1       pass
U115    S202    A302    1475130523      1475130523      1465130523      J       ST403   2       0       0       1       pass
U101    S202    A302    1495130523      1475130523      1485130523      AU      ST406   3       0       1       1       pass
U102    S204    A304    1494297562      1462863262      1465490556      E       ST414   3       1       0       1       pass
U119    S204    A304    1475130523      1485130523      1475130523      J       ST403   0       0       1       1       pass
U109    S204    A304    1468094889      1494297562      1494297562      J       ST403   3       0       1       1       pass
U103    S204    A304    1462863262      1465490556      1465490556      A       ST410   3       0       1       1       pass
U102    S204    A304    1465230523      1485130523      1475130523      A       ST411   0       0       0       1       pass
U104    S209    A305    1465490556      1462863262      1494297562      AP      ST407   0       0       1       1       pass
U110    S209    A305    1495130523      1475130523      1475130523      AU      ST406   0       1       0       1       pass
U116    S206    A302    1465490556      1462863262      1468094889      E       ST409   0       1       0       1       pass
U118    S206    A302    1465490556      1465490556      1462863262      A       ST411   1       0       1       1       pass
U107    S205    A301    1465130523      1475130523      1465230523      E       ST409   1       1       0       1       pass
U104    S205    A301    1462863262      1468094889      1468094889      E       ST409   2       0       0       1       pass
```

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:



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

7. Data Analysis :

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

Data_analysis.sh script file **:**

```
************************************************************
*


#!/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/examples/music/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
```

DataAnalysis.scala Program:

DataAnalysis.scala program:

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

import org.apache.spark.sql


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)




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

```
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"""


// Problem 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.



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"""

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



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"""



//Problem 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.


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"""
```

```
//Problem 5: Determine top 10 unsubscribed users who
listened to the songs for the longest duration.


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)

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()

}

}

}


**********************************************************
*
```

We are executing **Data_analysis.sh** script by running **music_project_master.sh** script file.

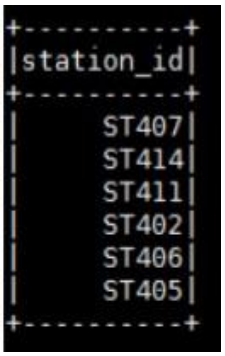
```
[acadgild@localhost music]$ ./music_project_master.sh
Preparing to execute python scripts to generate data...
Data Generated Successfully !
Starting the daemons....
5152 NodeManager
15505 RunJar
4610 RunJar
4691 HRegionServer
3923 NameNode
4531 HQuorumPeer
4580 HMaster
4084 DataNode
4999 ResourceManager
4344 SecondaryNameNode
3835 JobHistoryServer
17325 Jps
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...
18/12/09 15:40:57 INFO spark.SparkContext: Running Spark version 2.2.1
18/12/09 15:41:03 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where
applicable
18/12/09 15:41:06 WARN util.Utils: Your hostname, localhost.localdomain resolves to a loopback address: 127.0.0.1; using 192.168.0.100 in
stead (on interface eth15)
18/12/09 15:41:06 WARN util.Utils: Set SPARK_LOCAL_IP if you need to bind to another address
18/12/09 15:41:07 INFO spark.SparkContext: Submitted application: Data Analysis
18/12/09 15:41:07 INFO spark.SecurityManager: Changing view acls to: acadgild
18/12/09 15:41:07 INFO spark.SecurityManager: Changing modify acls to: acadgild
18/12/09 15:41:07 INFO spark.SecurityManager: Changing view acls groups to:
```

```
18/12/09 15:45:58 INFO parser.CatalystSqlParser: Parsing command: string
18/12/09 15:45:58 INFO metastore.HiveMetaStore: 0: get_table : db=project tbl=top_10_unsubscribed_users
18/12/09 15:45:58 INFO HiveMetaStore.audit: ugi=acadgild        ip=unknown-ip-addr      cmd=get_table : db=project tbl=top_10_unsubscribe
d_users
18/12/09 15:45:58 INFO parser.CatalystSqlParser: Parsing command: int
18/12/09 15:45:58 INFO parser.CatalystSqlParser: Parsing command: string
18/12/09 15:45:58 INFO parser.CatalystSqlParser: Parsing command: int
18/12/09 15:45:59 INFO common.FileUtils: Creating directory if it doesn't exist: hdfs://localhost:8020/user/hive/warehouse/project.db/top
_10_unsubscribed_users/.hive-staging_hive_2018-12-09_15-45-59_012_7240998144222739209-1
18/12/09 15:45:59 INFO output.FileOutputCommitter: File Output Committer Algorithm version is 1
18/12/09 15:45:59 INFO datasources.SQLHadoopMapReduceCommitProtocol: Using output committer class org.apache.hadoop.mapreduce.lib.output.
FileOutputCommitter
18/12/09 15:45:59 INFO aggregate.HashAggregateExec: spark.sql.codegen.aggregate.map.twolevel.enable is set to true, but current version o
f codegened fast hashmap does not support this aggregate.
18/12/09 15:45:59 INFO spark.ContextCleaner: Cleaned shuffle 9
18/12/09 15:45:59 INFO spark.ContextCleaner: Cleaned shuffle 10
18/12/09 15:45:59 INFO spark.ContextCleaner: Cleaned accumulator 439
18/12/09 15:45:59 INFO spark.ContextCleaner: Cleaned accumulator 431
18/12/09 15:45:59 INFO spark.ContextCleaner: Cleaned accumulator 429
18/12/09 15:45:59 INFO spark.ContextCleaner: Cleaned accumulator 438
18/12/09 15:45:59 INFO spark.ContextCleaner: Cleaned accumulator 433
18/12/09 15:45:59 INFO spark.ContextCleaner: Cleaned accumulator 428
18/12/09 15:45:59 INFO spark.ContextCleaner: Cleaned accumulator 437
18/12/09 15:45:59 INFO spark.ContextCleaner: Cleaned accumulator 432
18/12/09 15:45:59 INFO spark.ContextCleaner: Cleaned accumulator 427
18/12/09 15:45:59 INFO storage.BlockManagerInfo: Removed broadcast_21_piece0 on 192.168.0.100:45515 in memory (size: 13.4 KB, free: 413.8
 MB)
18/12/09 15:45:59 INFO spark.ContextCleaner: Cleaned accumulator 434
18/12/09 15:45:59 INFO spark.ContextCleaner: Cleaned accumulator 435
18/12/09 15:45:59 INFO storage.BlockManagerInfo: Removed broadcast_19_piece0 on 192.168.0.100:45515 in memory (size: 24.9 KB, free: 413.8
 MB)
18/12/09 15:45:59 INFO spark.ContextCleaner: Cleaned accumulator 430
18/12/09 15:45:59 INFO spark.ContextCleaner: Cleaned accumulator 436
```

```
Warning: /home/acadgild/install/sqoop/sqoop-1.4.6.bin__hadoop-2.0.4-alpha/../hcatalog does not exist! HCatalog jobs will fail.
Please set $HCAT_HOME to the root of your HCatalog installation.
Warning: /home/acadgild/install/sqoop/sqoop-1.4.6.bin__hadoop-2.0.4-alpha/../accumulo does not exist! Accumulo imports will fail.
Please set $ACCUMULO_HOME to the root of your Accumulo installation.
18/12/09 16:21:09 INFO sqoop.Sqoop: Running Sqoop version: 1.4.6
18/12/09 16:21:09 WARN tool.BaseSqoopTool: Setting your password on the command-line is insecure. Consider using -P instead.
18/12/09 16:21:10 INFO manager.MySQLManager: Preparing to use a MySQL streaming resultset.
18/12/09 16:21:10 INFO tool.CodeGenTool: Beginning code generation
Sun Dec 09 16:21:12 IST 2018 WARN: Establishing SSL connection without server's identity verification is not recommended. According to My
SQL 5.5.45+, 5.6.26+ and 5.7.6+ requirements SSL connection must be established by default if explicit option isn't set. For compliance w
ith existing applications not using SSL the verifyServerCertificate property is set to 'false'. You need either to explicitly disable SSL
 by setting useSSL=false, or set useSSL=true and provide truststore for server certificate verification.
18/12/09 16:21:19 INFO manager.SqlManager: Executing SQL statement: SELECT t.* FROM `top_10_stations` AS t LIMIT 1
18/12/09 16:21:19 INFO manager.SqlManager: Executing SQL statement: SELECT t.* FROM `top_10_stations` AS t LIMIT 1
18/12/09 16:21:19 INFO orm.CompilationManager: HADOOP_MAPRED_HOME is /home/acadgild/install/hadoop/hadoop-2.6.5
Note: /tmp/sqoop-acadgild/compile/2e7e412a7c900121d658a47478733800/top_10_stations.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
18/12/09 16:21:32 INFO orm.CompilationManager: Writing jar file: /tmp/sqoop-acadgild/compile/2e7e412a7c900121d658a47478733800/top_10_stat
ions.jar
18/12/09 16:21:32 INFO mapreduce.ExportJobBase: Beginning export of top_10_stations
SLF4J: Class path contains multiple SLF4J bindings.
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/i
mpl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/acadgild/install/hbase/hbase-1.2.6/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]
18/12/09 16:21:33 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where
applicable
18/12/09 16:21:34 INFO Configuration.deprecation: mapred.jar is deprecated. Instead, use mapreduce.job.jar
18/12/09 16:21:41 INFO Configuration.deprecation: mapred.reduce.tasks.speculative.execution is deprecated. Instead, use mapreduce.reduce.
speculative
18/12/09 16:21:41 INFO Configuration.deprecation: mapred.map.tasks.speculative.execution is deprecated. Instead, use mapreduce.map.specul
ative
18/12/09 16:21:41 INFO Configuration.deprecation: mapred.map.tasks is deprecated. Instead, use mapreduce.job.maps
```

Solution1:

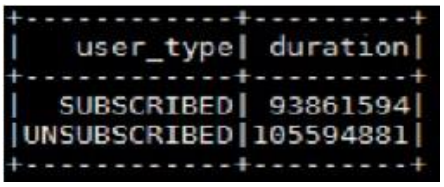
Determine top 10 station_id(s) where maximum number of songs were played, which were liked by unique users.



station_id
ST407
ST414
ST411
ST402
ST406
ST405

Solution2:

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	duration
SUBSCRIBED	93861594
UNSUBSCRIBED	105594881

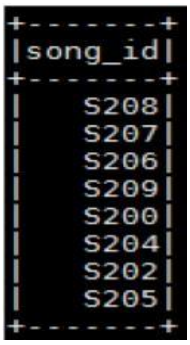
Solution3:

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

artist_id
A303
A302
A300

Solution4:

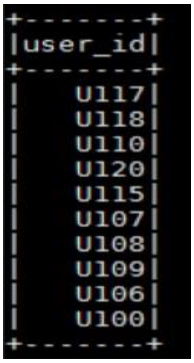
Determine top 10 songs who have generated the maximum revenue. Royalty applies to a song only if it was liked or was completed successfully or both



song_id
S208
S207
S206
S209
S200
S204
S202
S205

Solution5:

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



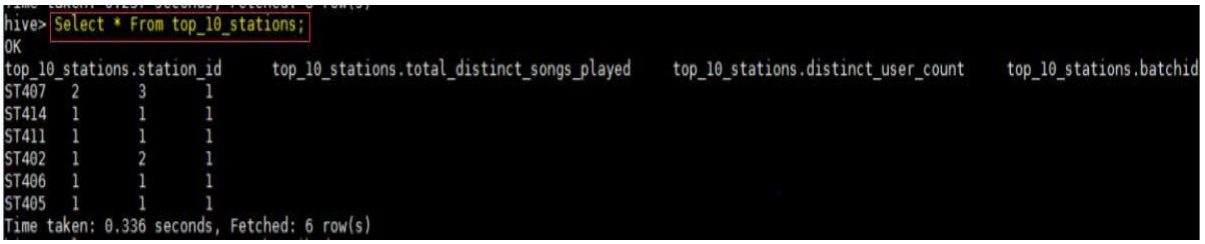
user_id
U117
U118
U110
U120
U115
U107
U108
U109
U106
U100

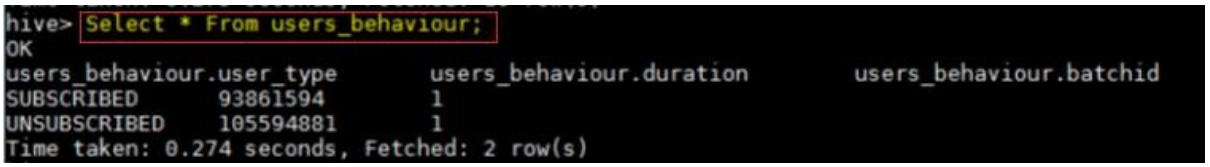
We could see below that all tables have also been created in the Hive :

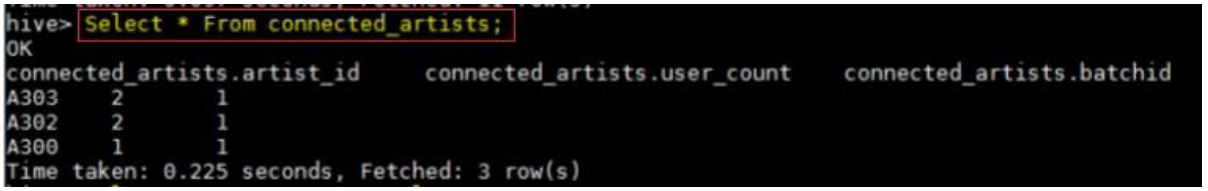
```
hive> use project;
OK
Time taken: 0.098 seconds
hive> show tables;
OK
connected_artists
enriched_data
formatted_input
song_artist_map
station_geo_map
subscribed_users
top_10_royalty_songs
top_10_stations
top_10_unsubscribed_users
users_artists
users_behaviour
Time taken: 0.407 seconds, Fetched: 11 row(s)
hive>
```

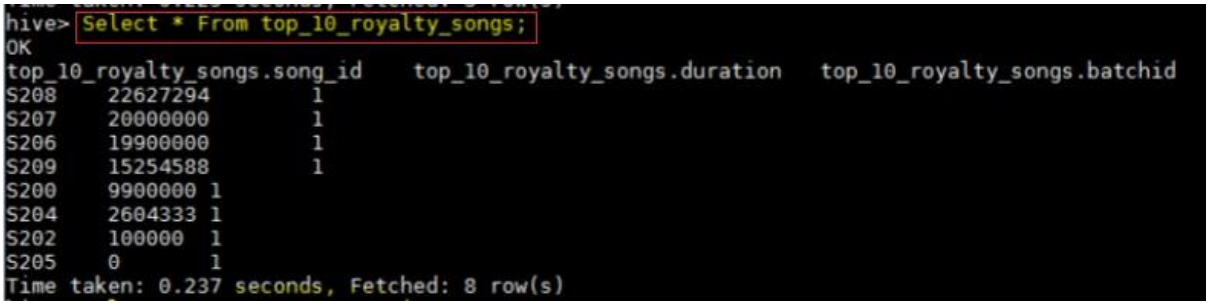
The data analysis result is shown in the Hive tables below in the screen shot:

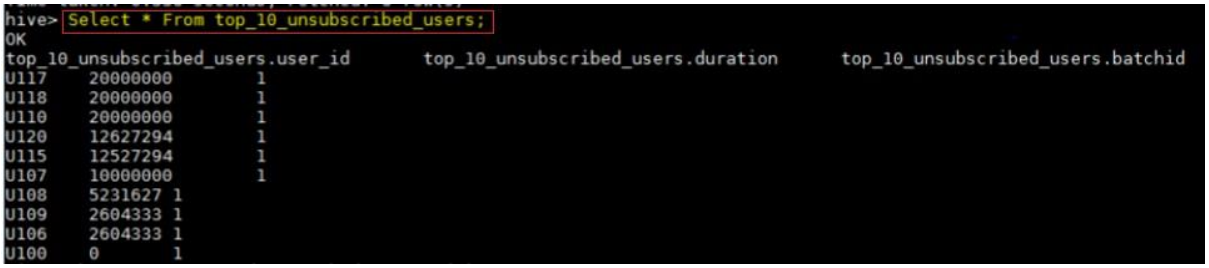
Below is the output of **top_10_stations** table: Below is the output of **users_behaviour** table: Below is the output of **connected_artists** table:











Now we need to export all the data to the MYSQL using sqoop, by executing **data_export.sh** script file:

By using **data_export.sh** script file, we are going to export the data from the hive tables into mysql using Sqoop export.

data_export.sh

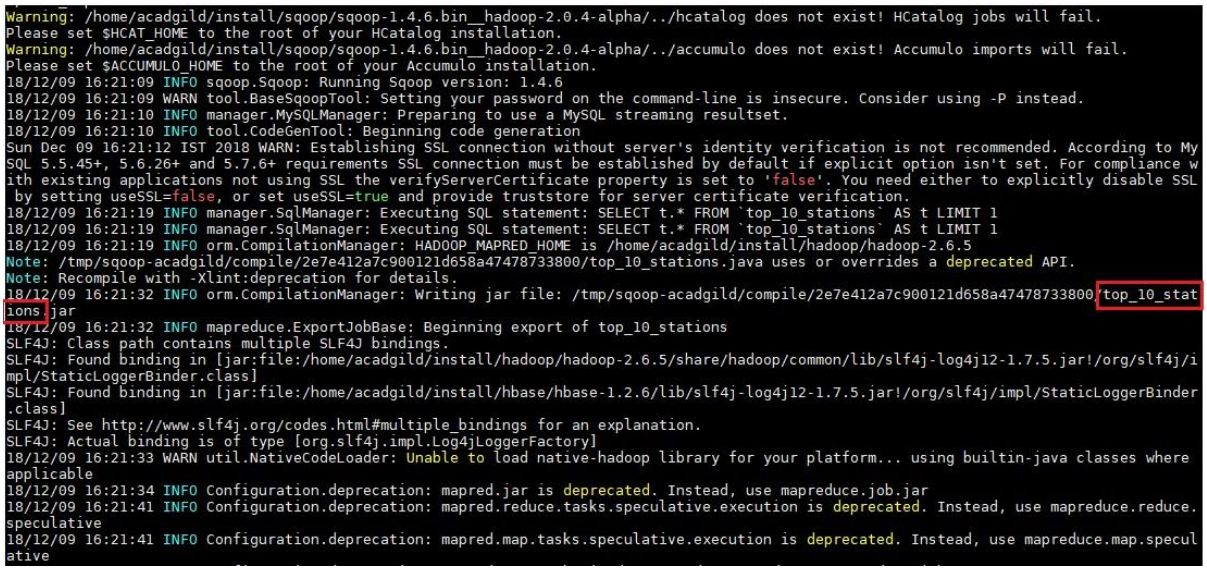
```
#!/bin/bash

#This script is not working.
#Either change table to text or use STRING as type of partitioned column

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

echo "Creating mysql tables if not present...AnkithTest" >> $LOGFILE

mysql -u "root" "-pRoot@123" < /home/acadgild/examples/music/create_schema.sql

echo "Running sqoop job for data export...AnkithTest" >> $LOGFILE

sqoop export --connect jdbc:mysql://localhost/project --username root --password Root@123 --table top_10_stations --export-dir /user/hive/warehouse/project.db/
/top_10_stations/batchid=$batchid --input-fields-terminated-by ',' -m 1

sqoop export --connect jdbc:mysql://localhost/project --username root --password Root@123 --table users_behaviour  --export-dir /user/hive/warehouse/project.db/
/users_behaviour/batchid=$batchid --input-fields-terminated-by ',' -m 1

sqoop export --connect jdbc:mysql://localhost/project --username root --password Root@123 --table connected_artists --export-dir /user/hive/warehouse/project.db/
/connected_artists/batchid=$batchid --input-fields-terminated-by ',' -m 1

sqoop export --connect jdbc:mysql://localhost/project --username root --password Root@123 --table top_10_royalty_songs --export-dir /user/hive/warehouse/project.db/
/top_10_royalty_songs/batchid=$batchid --input-fields-terminated-by ',' -m 1

sqoop export --connect jdbc:mysql://localhost/project --username root --password Root@123 --table top_10_unsubscribed_users --export-dir /user/hive/warehouse/
/project.db/top_10_unsubscribed_users/batchid=$batchid --input-fields-terminated-by ',' -m 1

```

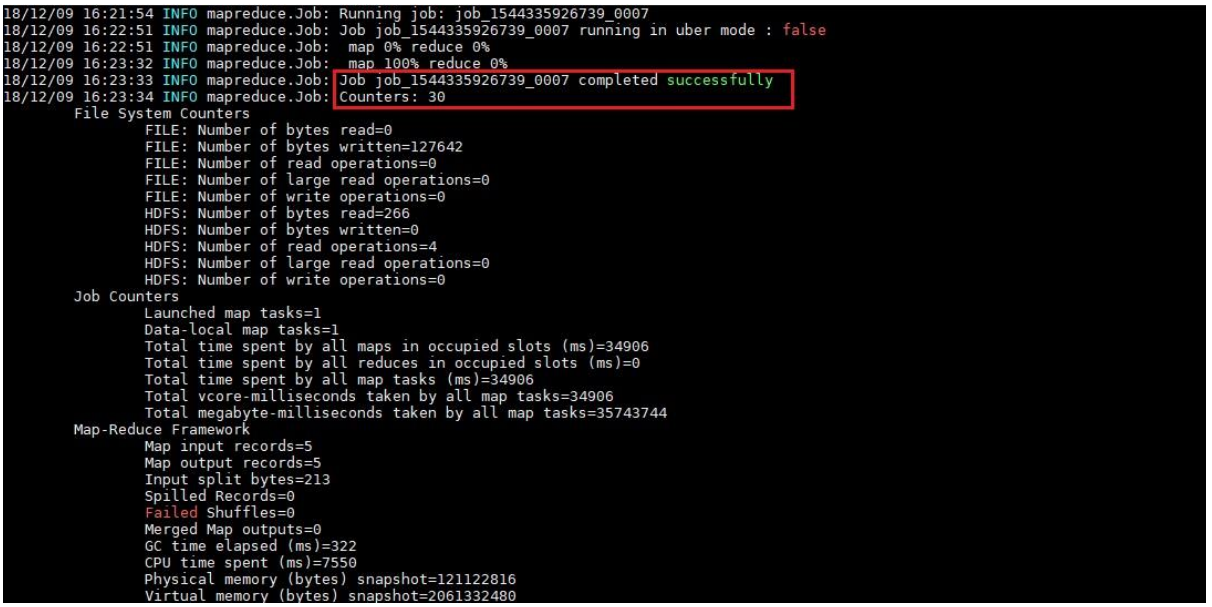
create_schema.sql

```
CREATE DATABASE IF NOT EXISTS project;

USE project;

CREATE TABLE IF NOT EXISTS top_10_stations
(
station_id VARCHAR(50),
total_distinct_songs_played INT,
distinct_user_count INT
);

CREATE TABLE IF NOT EXISTS users_behaviour
(
user_type VARCHAR(50),
duration BIGINT
);

CREATE TABLE IF NOT EXISTS connected_artists
(
artist_id VARCHAR(50),
user_count INT
);

CREATE TABLE IF NOT EXISTS top_10_royalty_songs
(
song_id VARCHAR(50),
duration BIGINT
);

CREATE TABLE IF NOT EXISTS top_10_unsubscribed_users
(
user_id VARCHAR(50),
duration BIGINT
);

commit;
```

Below we could see that data exported successfully into the MYSQL Database for all the 5 queries:

The sqoop export command exported the tables from the hive and it stored in the Mysql. The below screen shot show the successful Sqoop export from hive to mysql. The data stored in the Mysql is shown in below screenshots:





```
                Total committed heap usage (bytes)=62980096
        File Input Format Counters
                Bytes Read=0
        File Output Format Counters
                Bytes Written=0
18/12/09 16:23:34 INFO mapreduce.ExportJobBase: Transferred 266 bytes in 112.788 seconds (2.3584 bytes/sec)
18/12/09 16:23:34 INFO mapreduce.ExportJobBase: Exported 5 records.
Warning: /home/acadgild/install/sqoop/sqoop-1.4.6.bin__hadoop-2.0.4-alpha/../hcatalog does not exist! HCatalog jobs will fail.
Please set $HCAT_HOME to the root of your HCatalog installation.
Warning: /home/acadgild/install/sqoop/sqoop-1.4.6.bin__hadoop-2.0.4-alpha/../accumulo does not exist! Accumulo imports will fail.
Please set $ACCUMULO_HOME to the root of your Accumulo installation.
18/12/09 16:23:44 INFO sqoop.Sqoop: Running Sqoop version: 1.4.6
18/12/09 16:23:44 WARN tool.BaseSqoopTool: Setting your password on the command-line is insecure. Consider using -P instead.
18/12/09 16:23:46 INFO manager.MySQLManager: Preparing to use a MySQL streaming resultset.
18/12/09 16:23:46 INFO tool.CodeGenTool: Beginning code generation
Sun Dec 09 16:23:47 IST 2018 WARN: Establishing SSL connection without server's identity verification is not recommended. According to My
SQL 5.5.45+, 5.6.26+ and 5.7.6+ requirements SSL connection must be established by default if explicit option isn't set. For compliance w
ith existing applications not using SSL the verifyServerCertificate property is set to 'false'. You need either to explicitly disable SSL
 by setting useSSL=false, or set useSSL=true and provide truststore for server certificate verification.
18/12/09 16:23:53 INFO manager.SqlManager: Executing SQL statement: SELECT t.* FROM `song_duration` AS t LIMIT 1
18/12/09 16:23:53 INFO manager.SqlManager: Executing SQL statement: SELECT t.* FROM `song_duration` AS t LIMIT 1
18/12/09 16:23:53 INFO orm.CompilationManager: HADOOP_MAPRED_HOME is /home/acadgild/install/hadoop/hadoop-2.6.5
Note: /tmp/sqoop-acadgild/compile/b6353bd5b73c4f37c3f559414c34192f/song_duration.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
18/12/09 16:24:18 INFO orm.CompilationManager: Writing jar file: /tmp/sqoop-acadgild/compile/b6353bd5b73c4f37c3f559414c34192f/song_durati
on.jar
18/12/09 16:24:18 INFO mapreduce.ExportJobBase: Beginning export of song_duration
SLF4J: Class path contains multiple SLF4J bindings.
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/i
mpl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/acadgild/install/hbase/hbase-1.2.6/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.
```

The **project** database had been exported from hive (HDFS) and the below screen shot shows all tables:

```
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 (0.02 sec)
```

Output from **top_10_stations** table in mysql is shown below:

```
mysql> Select * From top_10_stations;
+------------+-------------------------------+----------------------+
| station_id | total_distinct_songs_played   | distinct_user_count  |
+------------+-------------------------------+----------------------+
| ST407      |                             2 |                    3 |
| ST414      |                             1 |                    1 |
| ST411      |                             1 |                    1 |
| ST402      |                             1 |                    2 |
| ST406      |                             1 |                    1 |
| ST405      |                             1 |                    1 |
+------------+-------------------------------+----------------------+
6 rows in set (0.00 sec)
```

Output from **users_behaviour** table in mysql is shown below:

```
mysql> Select * From users_behaviour;
+--------------+-----------+
| user_type    | duration  |
+--------------+-----------+
| SUBSCRIBED   |  93861594 |
| UNSUBSCRIBED | 105594881 |
+--------------+-----------+
2 rows in set (0.00 sec)
```

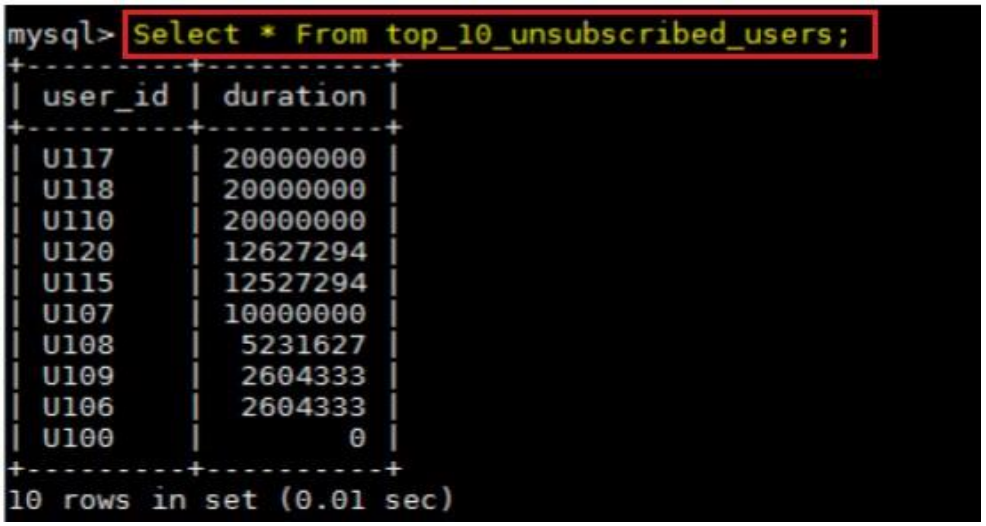
Output from **connected_artists** table in mysql is shown below:

```
mysql> Select * From connected_artists;
+-----------+------------+
| artist_id | user_count |
+-----------+------------+
| A303      |          2 |
| A302      |          2 |
| A300      |          1 |
+-----------+------------+
3 rows in set (0.00 sec)
```

Output from **top_10_royalty_songs** table in mysql is shown below:

```
mysql> Select * From top_10_royalty_songs;
+---------+----------+
| song_id | duration |
+---------+----------+
| S208    | 22627294 |
| S207    | 20000000 |
| S206    | 19900000 |
| S209    | 15254588 |
| S200    |  9900000 |
| S204    |  2604333 |
| S202    |   100000 |
| S205    |        0 |
+---------+----------+
8 rows in set (0.00 sec)
```

Output from **top_10_unsubscribed_users** table in mysql is shown below:



8. Job Scheduling

Now after exporting data into MySQL, **batchid** will be incremented to additional 1 means one batch of data operations is successfully completed and new batch of data will be loaded for the analysis after every 3 hours.

We can check logs to track the behaviour of the operations we have done on the data and overcome failures (if any) we could see the **batchid** gets incremented by 1 in **current-batch.txt**

Finally Jar file gets created as highlighted below :