MUSIC DATA ANALYSIS PROJECT USING SPARK

Table of Contents

1) Introduction & DATASET & Lookupfiles
2) Tools Used
3) Steps to Perform data analysis
   3.1) Start all daemons
   3.2) Job Scheduling
   3.3 Generating the data
   3.4) Populate lookup tables
   3.5) Data formatting
   3.6) Data Enrichment and Cleaning
   3.7)Data analysis
   3.8) Post Data Analyis

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.

Data set Description

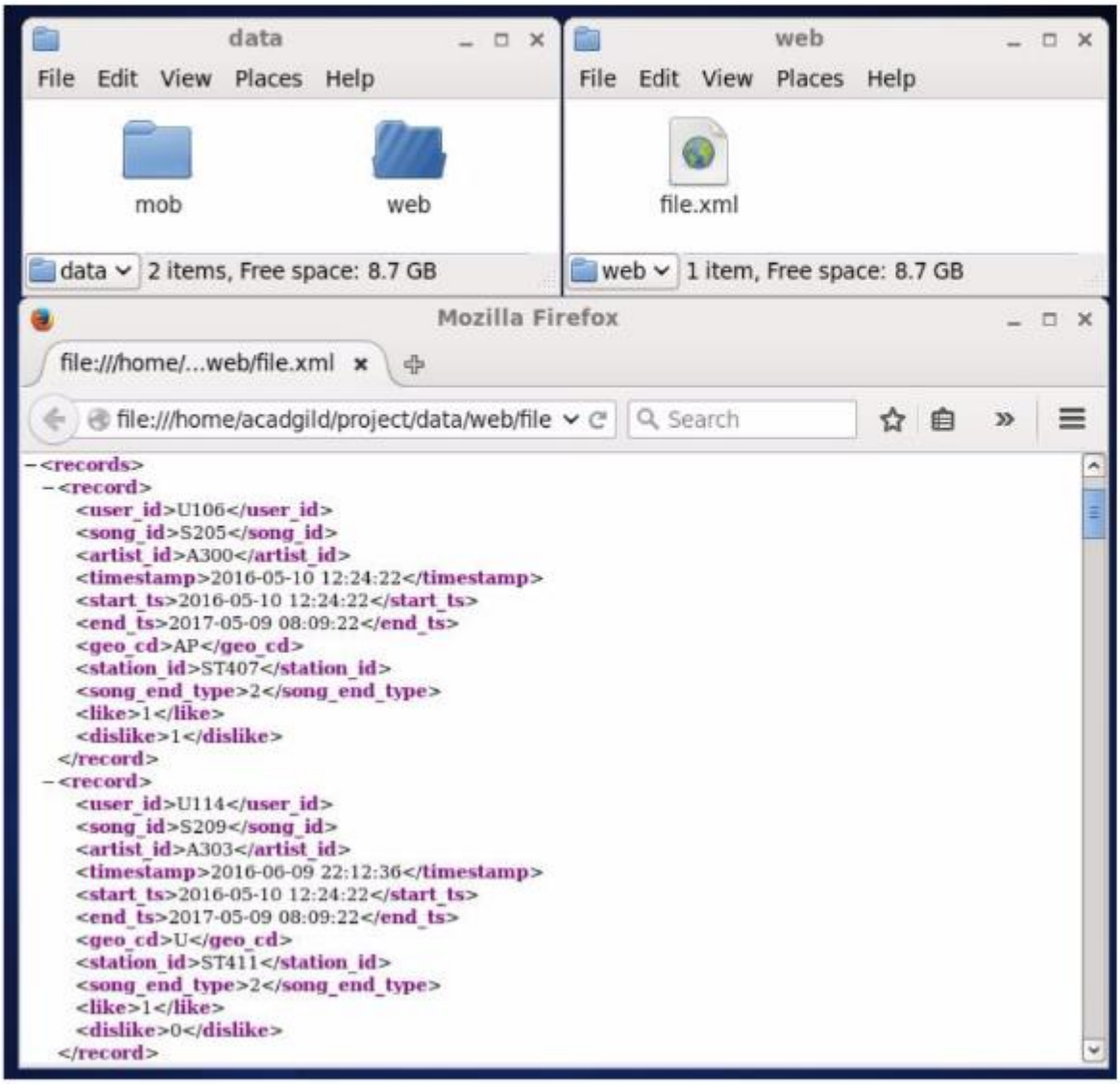
Link for data

https://drive.google.com/drive/folders/0B_P3pWagdIrrMjJGVlNsSUEtbG8?usp=sharing

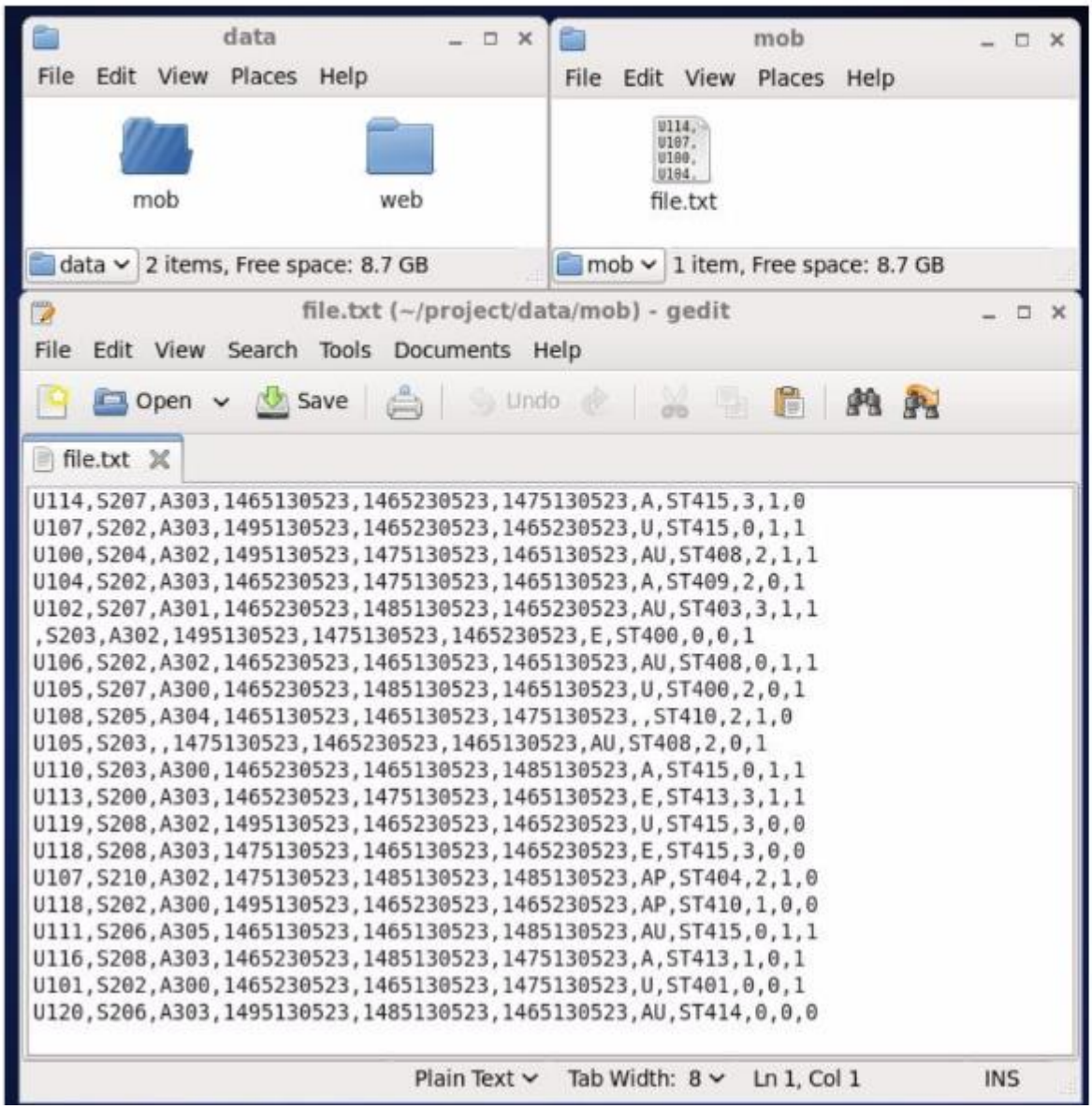
Column Name/Field Name	Column Description/Field Descrip
User_id	Unique identifier of every user
Song_id	Unique identifier of every song
Artist_id	Unique identifier of the lead artist
Timestamp	Timestamp when the record was
Start_ts	Start timestamp when the song st
End_ts	End timestamp when the song wa
Geo_cd	Can be 'A' for USA region, 'AP' for region, 'J' for Japan region, 'E' for 'AU' for australia region
Station_id	Unique identifier of the station fr 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 network error etc.
Like	0 means song was not liked 1 means song was liked
Dislike	0 means song was not disliked 1 means song was disliked

Data Files:

Below is the data coming from web applications, that reside in



Below is a sample of the data coming from mobile applications
csv format.



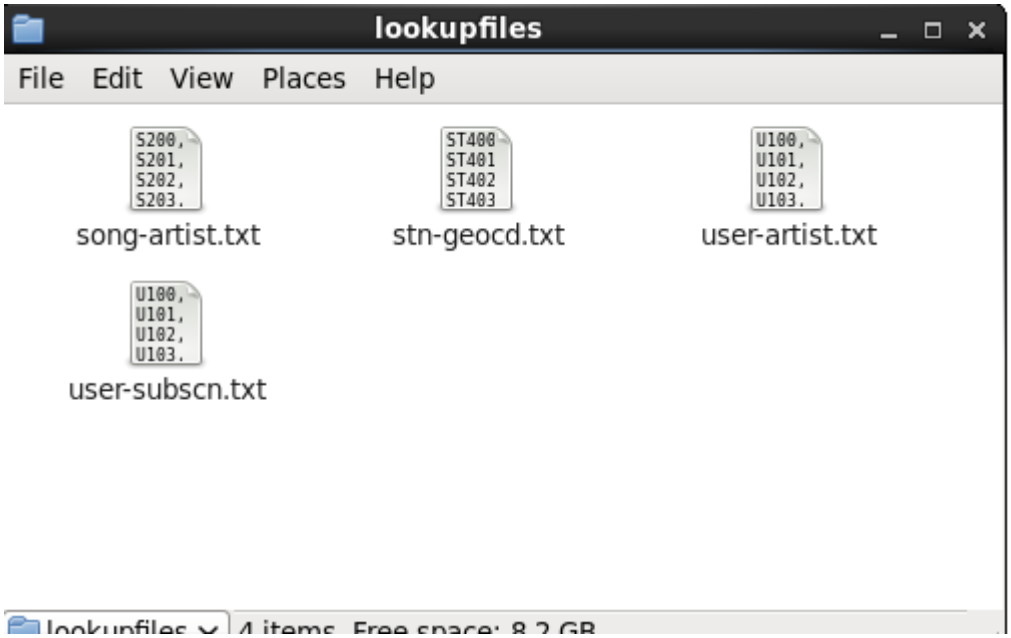
<u>Look-Up Tables Files</u>:

There are some existing lookup tables present in NoSQL Databases that play an important role in
data enrichment and analysis.

This data is present in lookup directory and loaded in HBase

Table Name	Description
Station_Geo_Map	Contains mapping of a
Subscribed_Users	Contains user_id, subs subscription_end_date Contains details only f
Song_Artist_Map	Contains mapping of s alongwith royalty asso the song
User_Artist_Map	Contains an array of ar user_id

Data Present in lookup files are



song-artist

Columns: song_id, artist_id

song-artist.txt

```
S200,A300
S201,A301
S202,A302
S203,A303
S204,A304
S205,A301
S206,A302
S207,A303
S208,A304
S209,A305
```

stn-geocd

Columns: station_id, geo_cd

stn-geocd.txt

```
ST400,A
ST401,AU
ST402,AP
ST403,J
ST404,E
ST405,A
ST406,AU
ST407,AP
ST408,E
ST409,E
ST410,A
ST411,A
ST412,AP
ST413,J
ST414,E
```

user-artist

Columns: user_id, artists_array

user-artist.txt

```
U100,A300&A301&A302
U101,A301&A302
U102,A302
U103,A303&A301&A302
U104,A304&A301
U105,A305&A301&A302
U106,A301&A302
U107,A302
U108,A300&A303&A304
U109,A301&A303
U110,A302&A301
U111,A303&A301
U112,A304&A301
U113,A305&A302
U114,A300&A301&A302
```

user-subscn

Columns: user_id, subscn_start_dt, subscn_end_dt

user-subscn.txt

```
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
```

Tools Used

PIG

HIVE

HBASE

SPARKSQL

Steps to Perform Data Analysis

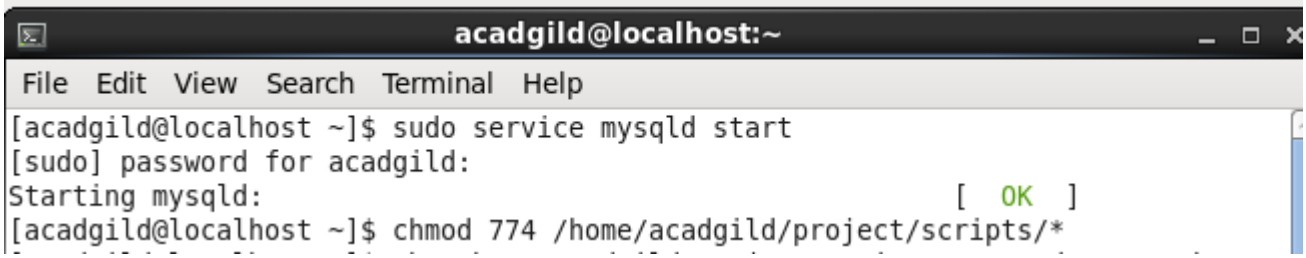
Step 1: Launch all necessary daemons
Step 2: Start Job Scheduling (using Crontab)
Step3 : Generate data from web and mobile in /dat/web and /data/mob
Step 4: Populate Look-Up tables (i.e. Load all data to HBase)
Step 5: Perform Data Formatting (using Pig and Hive)
Step 6: Perform Data Enrichment and Cleaning (using Hive)
Step 7: Perform Data Analysis (using Spark)

Step 1: Launch all necessary daemons

Launch the Mysql Service (needed for Hive)

Give permissions to scripts folder in project, so we are able to run scripts from the bash shell.

Run the shell script start-daemons.sh



```
acadgild@localhost:~
File  Edit  View  Search  Terminal  Help
[acadgild@localhost ~]$ sh  /home/acadgild/project/scripts/start-daemons.sh
This script is Deprecated. Instead use start-dfs.sh and start-yarn.sh
17/10/26 21:46:29 WARN util.NativeCodeLoader: Unable to load native-hadoop libra
ry for your platform... using builtin-java classes where applicable
Starting namenodes on [localhost]
localhost: starting namenode, logging to /usr/local/hadoop-2.6.0/logs/hadoop-aca
dgild-namenode-localhost.localdomain.out
localhost: starting datanode, logging to /usr/local/hadoop-2.6.0/logs/hadoop-aca
dgild-datanode-localhost.localdomain.out
Starting secondary namenodes [0.0.0.0]
0.0.0.0: starting secondarynamenode, logging to /usr/local/hadoop-2.6.0/logs/had
oop-acadgild-secondarynamenode-localhost.localdomain.out
17/10/26 21:46:59 WARN util.NativeCodeLoader: Unable to load native-hadoop libra
ry for your platform... using builtin-java classes where applicable
starting yarn daemons
starting resourcemanager, logging to /usr/local/hadoop-2.6.0/logs/yarn-acadgild-
resourcemanager-localhost.localdomain.out
localhost: starting nodemanager, logging to /usr/local/hadoop-2.6.0/logs/yarn-ac
adgild-nodemanager-localhost.localdomain.out
starting master, logging to /usr/local/hbase/logs/hbase-acadgild-master-localhos
t.localdomain.out
starting historyserver, logging to /usr/local/hadoop-2.6.0/logs/mapred-acadgild-
historyserver-localhost.localdomain.out
[acadgild@localhost ~]$
```

In the shell script start-daemons.sh used above, we perform the following operations:

a) Check if any batch file is present or not if not then create one
b) Create a log file
c) Starting all daemons like namenode, datanode ,hbase, historyserver

start-daemons.sh

```
#!/bin/bash

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

chmod 775 /home/acadgild/project/logs/current-batch.txt
batchid=cat `/home/acadgild/project/logs/current-batch.txt`
LOGFILE=/home/acadgild/project/logs/log_batch_$batchid

echo "Starting daemons" >> $LOGFILE

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

Step 2 : Job Scheduling

We will schedule the job for every 3 hours as data is generated after every 3 hours so we write a script wrapper.sh which contains all the process for data analysis

```
[acadgild@localhost ~]$ sudo crontab -e
[sudo] password for acadgild:
no crontab for root - using an empty one
crontab: installing new crontab
[acadgild@localhost ~]$
```

acadgild@localhost:~

File Edit View Search Terminal Help

```
* */3 * * * /home/acadgild/project/scripts/wrapper.sh
~
~
~
~
~
~
~
~
-- INSERT --
```

wrapper.sh

```bash
#!/bin/bash

python /home/acadgild/project/scripts/generate_web_data.py

python /home/acadgild/project/scripts/generate_mob_data.py

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

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

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

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

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

Step 3 : Generate Data from mobile as well as web

Mobile Data

generate_mob_data.py

```python
from random import randint
from random import choice

file = open("/home/acadgild/project/data/mob/file.txt", "w")
count = 20

while (count > 0):
        geo_cd_list=["A", "E", "AU", "AP", "U"]
        song_end_type_list=["0","1","2","3"]
        timestamp_list=["1465230523", "1465130523", "1475130523", "1495130523"]
        start_ts_list=["1465230523", "1465130523", "1475130523", "1485130523"]
        end_ts_list=["1465230523", "1465130523", "1475130523", "1485130523"]

        if (count%15 == 0):
                user_id = ""
        else:
                user_id = "U" + str(randint(100,120))

        song_id = "S" + str(randint(200,210))

        if (count%11 == 0):
                artist_id = ""
        else:
                artist_id = "A" + str(randint(300,305))

        timestamp = choice(timestamp_list)
        start_ts = choice(start_ts_list)
        end_ts = choice(end_ts_list)

        if (count%12 == 0):
                geo_cd = ""
        else:
                geo_cd = choice(geo_cd_list)
```

```
        station_id = "ST" + str(randint(400,415))
        song_end_type = choice(song_end_type_list)
        like = str(randint(0,1))
        dislike = str(randint(0,1))

        file.write("%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s\n" % (user_id, song_id, artist_id, timestamp,
geo_cd, station_id, song_end_type, like, dislike))

        count = count-1

file.close()
```

Web data

generate_web_data.py

```
from random import randint
from random import choice

file = open("/home/acadgild/project/data/web/file.xml", "w")
count = 20

file.write("<records>\n")

while (count > 0):
        geo_cd_list=["A", "E", "AU", "AP", "U"]
        song_end_type_list=["0","1","2","3"]
        timestamp_list=["2016-05-10 12:24:22", "2016-06-09 22:12:36", "2016-07-10 01:38:09", "2017-05-09 08:09
        start_ts_list=["2016-05-10 12:24:22", "2016-06-09 22:12:36", "2016-07-10 01:38:09", "2017-05-09 08:09:
        end_ts_list=["2016-05-10 12:24:22", "2016-06-09 22:12:36", "2016-07-10 01:38:09", "2017-05-09 08:09:22

        if (count%15 == 0):
                user_id = ""
        else:
                user_id = "U" + str(randint(100,120))

        song_id = "S" + str(randint(200,210))

        if (count%11 == 0):
                artist_id = ""
        else:
                artist_id = "A" + str(randint(300,305))

        timestamp = choice(timestamp_list)
```

```
        start_ts = choice(start_ts_list)
        end_ts = choice(end_ts_list)

        if (count%12 == 0):
                geo_cd = ""
        else:
                geo_cd = choice(geo_cd_list)

        station_id = "ST" + str(randint(400,415))
        song_end_type = choice(song_end_type_list)
        like = str(randint(0,1))
        dislike = str(randint(0,1))
        file.write("<record>\n")
        file.write("<user_id>%s</user_id>\n" % (user_id))
        file.write("<song_id>%s</song_id>\n" % (song_id))
        file.write("<artist_id>%s</artist_id>\n" % (artist_id))
        file.write("<timestamp>%s</timestamp>\n" % (timestamp))
        file.write("<start_ts>%s</start_ts>\n" % (start_ts))
        file.write("<end_ts>%s</end_ts>\n" % (end_ts))
        file.write("<geo_cd>%s</geo_cd>\n" % (geo_cd))
        file.write("<station_id>%s</station_id>\n" % (station_id))
        file.write("<song_end_type>%s</song_end_type>\n" % (song_end_type))
        file.write("<like>%s</like>\n" % (like))
        file.write("<dislike>%s</dislike>\n" % (dislike))
        file.write("</record>\n")

        count = count-1

file.write("</records>")
file.close()
```

Step 4 : Populate the lookup table

In this we create a table on hbase song-artist, stn-geocd and user-subscn
with their column families.For every lookup data file, read each line, extract the columns (comma separated) and add
the data as rows to the corresponding HBase tables created above and then run the hive script user-artist.hql. This will populate a hive table with the data in the
lookup data file user-artist. This is because this file has an array column that is difficult to populate in HBase After the data is stored in hive table we will store it in local file system for data analysis

populate-lookup.sh

```
#!/bin/bash

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

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

echo "Creating LookUp Tables" >> $LOGFILE

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


echo "Populating LookUp Tables" >> $LOGFILE

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


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

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

hive -f /home/acadgild/project/scripts/user-artist.hql
```

user-artist.hql

```
CREATE DATABASE IF NOT EXISTS project;

USE project;

CREATE TABLE users_artists
(
user_id STRING,
artists_array ARRAY<STRING>
)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
COLLECTION ITEMS TERMINATED BY '&';

LOAD DATA LOCAL INPATH '/home/acadgild/project/lookupfiles/user-artist.txt'
OVERWRITE INTO TABLE users_artists;

INSERT OVERWRITE LOCAL DIRECTORY '/home/acadgild/project/exporteddata/userartists'
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
STORED AS TEXTFILE
SELECT user_id,artists FROM users_artists LATERAL VIEW explode(artists_array) a AS artists;
```

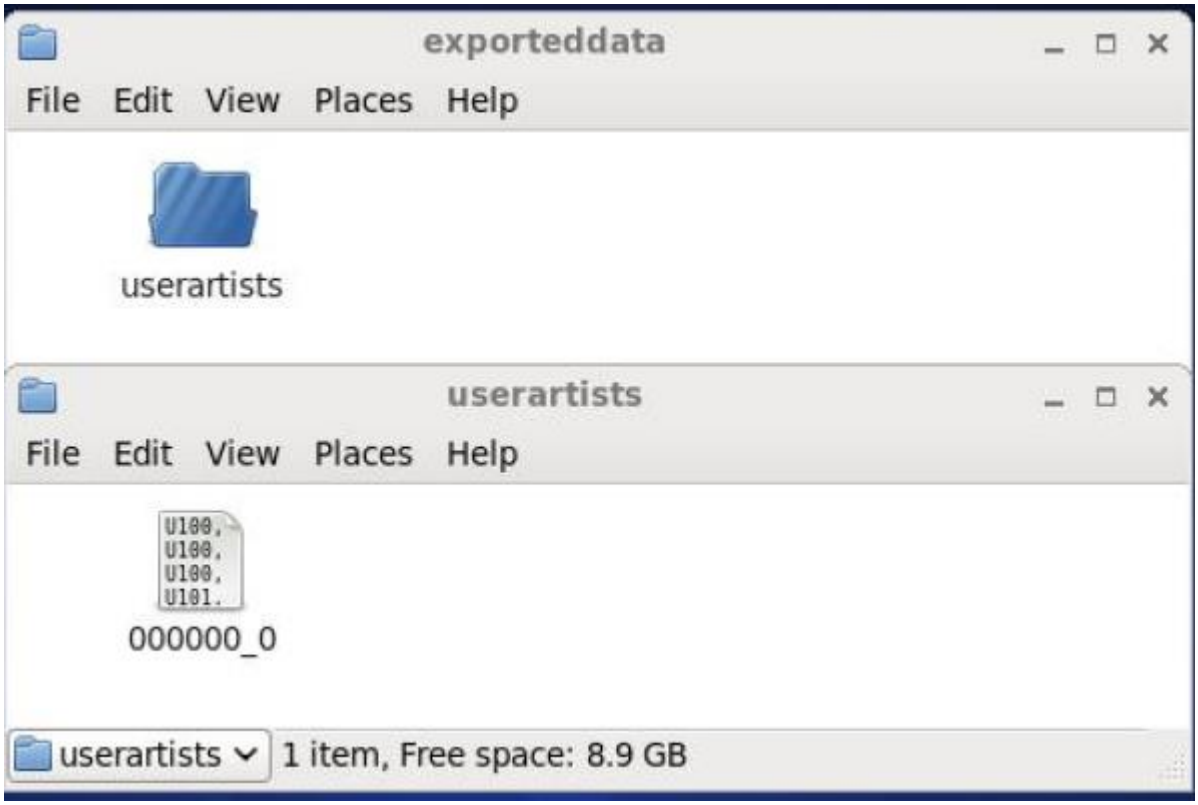
```
[acadgild@localhost ~]$ sh  /home/acadgild/project/scripts/populate-lookup.sh
2017-10-26 22:12:58,692 INFO  [main] Configuration.deprecation: hadoop.native.li
b is deprecated. Instead, use io.native.lib.available
HBase Shell; enter 'help<RETURN>' for list of supported commands.
Type "exit<RETURN>" to leave the HBase Shell
Version 0.98.14-hadoop2, r4e4aabb93b52f1b0fef6b66edd06ec8923014dec, Tue Aug 25 2
2:35:44 PDT 2015

create 'station-geo-map', 'geo'
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/local/hbase/lib/slf4j-log4j12-1.6.4.jar!/
org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/local/hadoop-2.6.0/share/hadoop/common/li
b/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
2017-10-26 22:13:02,619 WARN  [main] util.NativeCodeLoader: Unable to load nativ
e-hadoop library for your platform... using builtin-java classes where applicabl
e
0 row(s) in 3.0760 seconds

Hbase::Table - station-geo-map
2017-10-26 22:13:11,043 INFO  [main] Configuration.deprecation: hadoop.native.li
b is deprecated. Instead, use io.native.lib.available
HBase Shell; enter 'help<RETURN>' for list of supported commands.
Type "exit<RETURN>" to leave the HBase Shell
Version 0.98.14-hadoop2, r4e4aabb93b52f1b0fef6b66edd06ec8923014dec, Tue Aug 25 2
2:35:44 PDT 2015

create 'subscribed-users', 'subscn'
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/local/hbase/lib/slf4j-log4j12-1.6.4.jar!/
org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/local/hadoop-2.6.0/share/hadoop/common/li
```

```
File  Edit  View  Search  Terminal  Help
/usr/local/hive/bin/hive-config.sh: line 1: `# Licensed to the Apache Software Foundation (ASF) under one or m

Logging initialized using configuration in jar:file:/usr/local/hive/lib/hive-common-0.14.0.jar!/hive-log4j.pro
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/local/hive/lib/hive-jdbc-0.14.0-standalone.jar!/org/slf4j/impl/StaticLo
s]
SLF4J: Found binding in [jar:file:/usr/local/hadoop-2.6.0/share/hadoop/common/lib/slf4j-log4j12-1.7.5.jar!/org
ticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
OK
Time taken: 0.747 seconds
OK
Time taken: 0.047 seconds
OK
Time taken: 0.848 seconds
Loading data to table project.users_artists
Table project.users_artists stats: [numFiles=1, numRows=0, totalSize=240, rawDataSize=0]
OK
Time taken: 1.795 seconds
Query ID = acadgild_20171026230606_4f44fec0-d133-4e26-928f-d0c5b3c19b5c
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks is set to 0 since there's no reduce operator
Starting Job = job_1509034627624_0001, Tracking URL = http://localhost:8088/proxy/application_1509034627624_00
Kill Command = /home/acadgild/hadoop-2.6.0/bin/hadoop job  -kill job_1509034627624_0001
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 0
2017-10-26 23:06:52,003 Stage-1 map = 0%,  reduce = 0%
2017-10-26 23:07:04,447 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 1.43 sec
MapReduce Total cumulative CPU time: 1 seconds 430 msec
Ended Job = job_1509034627624_0001
Copying data to local directory /home/acadgild/project/exporteddata/userartist
Copying data to local directory /home/acadgild/project/exporteddata/userartist
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1   Cumulative CPU: 1.43 sec   HDFS Read: 476 HDFS Write: 330 SUCCESS
Total MapReduce CPU Time Spent: 1 seconds 430 msec
OK
Time taken: 39.95 seconds
[acadgild@localhost ~]$
```

```
hbase(main):002:0> scan 'song-artist-map'
ROW                    COLUMN+CELL
 S200                  column=artist:artistid, timestamp=1509038870966, value=A30
                       0
 S201                  column=artist:artistid, timestamp=1509038883270, value=A30
                       1
 S202                  column=artist:artistid, timestamp=1509038894947, value=A30
                       2
 S203                  column=artist:artistid, timestamp=1509038907274, value=A30
                       3
 S204                  column=artist:artistid, timestamp=1509038919750, value=A30
                       4
 S205                  column=artist:artistid, timestamp=1509038931528, value=A30
                       1
 S206                  column=artist:artistid, timestamp=1509038943640, value=A30
                       2
 S207                  column=artist:artistid, timestamp=1509038955863, value=A30
                       3
 S208                  column=artist:artistid, timestamp=1509038967968, value=A30
                       4
 S209                  column=artist:artistid, timestamp=1509038980618, value=A30
                       5
10 row(s) in 0.2460 seconds
```

```
hbase(main):006:0> scan 'station-geo-map'
ROW                                    COLUMN+CELL
 ST400                                 column=geo:geo_cd, timestamp=1509038684085, value=A
 ST401                                 column=geo:geo_cd, timestamp=1509038698412, value=AU
 ST402                                 column=geo:geo_cd, timestamp=1509038712134, value=AP
 ST403                                 column=geo:geo_cd, timestamp=1509038723447, value=J
 ST404                                 column=geo:geo_cd, timestamp=1509038735105, value=E
 ST405                                 column=geo:geo_cd, timestamp=1509038746942, value=A
 ST406                                 column=geo:geo_cd, timestamp=1509038759230, value=AU
 ST407                                 column=geo:geo_cd, timestamp=1509038771375, value=AP
 ST408                                 column=geo:geo_cd, timestamp=1509038784891, value=E
 ST409                                 column=geo:geo_cd, timestamp=1509038796404, value=E
 ST410                                 column=geo:geo_cd, timestamp=1509038808016, value=A
 ST411                                 column=geo:geo_cd, timestamp=1509038821948, value=A
 ST412                                 column=geo:geo_cd, timestamp=1509038834754, value=AP
 ST413                                 column=geo:geo_cd, timestamp=1509038847322, value=J
 ST414                                 column=geo:geo_cd, timestamp=1509038858669, value=E
15 row(s) in 0.1210 seconds
```

```
hbase(main):003:0> scan 'subscribed-users'
ROW                                    COLUMN+CELL
 U100                                  column=subscn:enddt, timestamp=1509039003710, value=1465130523
 U100                                  column=subscn:startdt, timestamp=1509038992074, value=1465230523
 U101                                  column=subscn:enddt, timestamp=1509039027634, value=1475130523
 U101                                  column=subscn:startdt, timestamp=1509039015171, value=1465230523
 U102                                  column=subscn:enddt, timestamp=1509039051557, value=1475130523
 U102                                  column=subscn:startdt, timestamp=1509039039666, value=1465230523
 U103                                  column=subscn:enddt, timestamp=1509039074407, value=1475130523
 U103                                  column=subscn:startdt, timestamp=1509039063055, value=1465230523
 U104                                  column=subscn:enddt, timestamp=1509039098603, value=1475130523
 U104                                  column=subscn:startdt, timestamp=1509039086038, value=1465230523
 U105                                  column=subscn:enddt, timestamp=1509039134492, value=1475130523
 U105                                  column=subscn:startdt, timestamp=1509039116056, value=1465230523
 U106                                  column=subscn:enddt, timestamp=1509039160947, value=1485130523
 U106                                  column=subscn:startdt, timestamp=1509039147884, value=1465230523
 U107                                  column=subscn:enddt, timestamp=1509039187775, value=1455130523
 U107                                  column=subscn:startdt, timestamp=1509039174366, value=1465230523
 U108                                  column=subscn:enddt, timestamp=1509039212405, value=1465230623
 U108                                  column=subscn:startdt, timestamp=1509039200128, value=1465230523
 U109                                  column=subscn:enddt, timestamp=1509039237756, value=1475130523
 U109                                  column=subscn:startdt, timestamp=1509039225103, value=1465230523
 U110                                  column=subscn:enddt, timestamp=1509039261975, value=1475130523
 U110                                  column=subscn:startdt, timestamp=1509039250056, value=1465230523
 U111                                  column=subscn:enddt, timestamp=1509039286003, value=1475130523
 U111                                  column=subscn:startdt, timestamp=1509039274138, value=1465230523
 U112                                  column=subscn:enddt, timestamp=1509039313069, value=1475130523
 U112                                  column=subscn:startdt, timestamp=1509039301456, value=1465230523
 U113                                  column=subscn:enddt, timestamp=1509039340223, value=1485130523
 U113                                  column=subscn:startdt, timestamp=1509039325918, value=1465230523
 U114                                  column=subscn:enddt, timestamp=1509039367183, value=1468130523
 U114                                  column=subscn:startdt, timestamp=1509039353440, value=1465230523
15 row(s) in 0.2080 seconds
```



000000_0

```
U100,A300
U100,A301
U100,A302
U101,A301
U101,A302
U102,A302
U103,A303
U103,A301
U103,A302
U104,A304
U104,A301
U105,A305
U105,A301
U105,A302
U106,A301
U106,A302
U107,A302
U108,A300
U108,A303
U108,A304
U109,A301
U109,A303
U110,A302
U110,A301
U111,A303
U111,A301
U112,A304
U112,A301
U113,A305
U113,A302
U114,A300
U114,A301
U114,A302
```

Step 5 :Data formatting

In this we will convert xml file to csv using pig and load the 2 files web and mob into hive table for enrichment

dataformatting.sh

```
#!/bin/bash

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

echo "Placing data files from local to HDFS..." >> $LOGFILE

hdfs dfs -rm -r /user/acadgild/project/batch${batchid}/web/
hdfs dfs -rm -r /user/acadgild/project/batch${batchid}/formattedweb/
hdfs dfs -rm -r /user/acadgild/project/batch${batchid}/mob/

hdfs dfs -mkdir -p /user/acadgild/project/batch${batchid}/web/
hdfs dfs -mkdir -p /user/acadgild/project/batch${batchid}/mob/

hdfs dfs -put /home/acadgild/project/data/web/* /user/acadgild/project/batch${batchid}/web/
hdfs dfs -put /home/acadgild/project/data/mob/* /user/acadgild/project/batch${batchid}/mob/

echo "Running pig script for data formatting..." >> $LOGFILE

pig -param batchid=$batchid /home/acadgild/project/scripts/dataformatting.pig

echo "Running hive script for formatted data load..." >> $LOGFILE

hive -hiveconf batchid=$batchid -f /home/acadgild/project/scripts/formatted_hive_load.hql
```

Dataformatting.pig

Stores the formatted data to a folder in the HDFS called formattedweb

dataformatting.pig

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

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

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

B = FOREACH A GENERATE TRIM(XPath(x, 'record/user_id')) AS user_id,
    TRIM(XPath(x, 'record/song_id')) AS song_id,
    TRIM(XPath(x, 'record/artist_id')) AS artist_id,
    ToUnixTime(ToDate(TRIM(XPath(x, 'record/timestamp')),'yyyy-MM-dd HH:mm:ss')) AS t
    ToUnixTime(ToDate(TRIM(XPath(x, 'record/start_ts')),'yyyy-MM-dd HH:mm:ss')) AS st
    ToUnixTime(ToDate(TRIM(XPath(x, 'record/end_ts')),'yyyy-MM-dd HH:mm:ss')) AS end_
    TRIM(XPath(x, 'record/geo_cd')) AS geo_cd,
    TRIM(XPath(x, 'record/station_id')) AS station_id,
    TRIM(XPath(x, 'record/song_end_type')) AS song_end_type,
    TRIM(XPath(x, 'record/like')) AS like,
    TRIM(XPath(x, 'record/dislike')) AS dislike;

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

formatted_hive_load.hql

Combines the data from mob and formattedweb to make one data-set and stores it partitioned by

batchid.

formatted_hive_load.hql

```
USE project;

CREATE TABLE IF NOT EXISTS formatted_input
(
User_id STRING,
Song_id STRING,
Artist_id STRING,
Timestamp STRING,
Start_ts STRING,
End_ts STRING,
Geo_cd STRING,
Station_id STRING,
Song_end_type INT,
Like INT,
Dislike INT
)
PARTITIONED BY
(batchid INT)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ',';

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

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

acadgild@localhost:~

File Edit View Search Terminal Help

```
17/10/26 23:30:28 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using
asses where applicable
17/10/26 23:30:29 INFO fs.TrashPolicyDefault: Namenode trash configuration: Deletion interval = 0 minutes, Emp
 0 minutes.
Deleted /user/acadgild/project/batch1/formattedweb
17/10/26 23:30:31 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using
asses where applicable
17/10/26 23:30:32 INFO fs.TrashPolicyDefault: Namenode trash configuration: Deletion interval = 0 minutes, Emp
 0 minutes.
Deleted /user/acadgild/project/batch1/mob
17/10/26 23:30:34 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using
asses where applicable
17/10/26 23:30:36 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using
asses where applicable
17/10/26 23:30:39 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using
asses where applicable
17/10/26 23:30:43 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using
asses where applicable
2017-10-26 23:30:48,546 INFO  [main] pig.ExecTypeProvider: Trying ExecType : LOCAL
2017-10-26 23:30:48,552 INFO  [main] pig.ExecTypeProvider: Trying ExecType : MAPREDUCE
2017-10-26 23:30:48,552 INFO  [main] pig.ExecTypeProvider: Picked MAPREDUCE as the ExecType
2017-10-26 23:30:48,685 [main] INFO  org.apache.pig.Main - Apache Pig version 0.14.0 (r1640057) compiled Nov 1
5
2017-10-26 23:30:48,686 [main] INFO  org.apache.pig.Main - Logging error messages to: /home/acadgild/pig_15090
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/local/hbase/lib/slf4j-log4j12-1.6.4.jar!/org/slf4j/impl/StaticLoggerBin
SLF4J: Found binding in [jar:file:/usr/local/hadoop-2.6.0/share/hadoop/common/lib/slf4j-log4j12-1.7.5.jar!/org
ticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
2017-10-26 23:30:49,105 [main] WARN  org.apache.hadoop.util.NativeCodeLoader - Unable to load native-hadoop li
platform... using builtin-java classes where applicable
2017-10-26 23:30:49,484 [main] INFO  org.apache.pig.impl.util.Utils - Default bootup file /home/acadgild/.pigb
2017-10-26 23:30:49,750 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - mapred.job.tracker is
tead, use mapreduce.jobtracker.address
2017-10-26 23:30:49,752 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is dep
d, use fs.defaultFS
2017-10-26 23:30:49,752 [main] INFO  org.apache.pig.backend.hadoop.executionengine.HExecutionEngine - Connecti
le system at: hdfs://localhost:9000
2017-10-26 23:30:49,761 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - mapred.used.genericopt
```

```
.
2017-10-26 23:31:30,114 [main] INFO  org.apache.pig.Main - Pig script completed in 41 seconds and 726 millisec
/usr/local/hive/bin/hive-config.sh: line 1: syntax error near unexpected token `('
/usr/local/hive/bin/hive-config.sh: line 1: `# Licensed to the Apache Software Foundation (ASF) under one or m

Logging initialized using configuration in jar:file:/usr/local/hive/lib/hive-common-0.14.0.jar!/hive-log4j.pro
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/local/hive/lib/hive-jdbc-0.14.0-standalone.jar!/org/slf4j/impl/StaticLo
s]
SLF4J: Found binding in [jar:file:/usr/local/hadoop-2.6.0/share/hadoop/common/lib/slf4j-log4j12-1.7.5.jar!/org
ticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
OK
Time taken: 0.767 seconds
OK
Time taken: 1.045 seconds
Loading data to table project.formatted_input partition (batchid=1)
Partition project.formatted_input{batchid=1} stats: [numFiles=1, numRows=0, totalSize=1241, rawDataSize=0]
OK
Time taken: 2.349 seconds
Loading data to table project.formatted_input partition (batchid=1)
Partition project.formatted_input{batchid=1} stats: [numFiles=2, numRows=0, totalSize=2480, rawDataSize=0]
OK
Time taken: 1.205 seconds
[acadgild@localhost ~]$
```

acadgild@localhost:~

File Edit View Search Terminal Help

```
[acadgild@localhost ~]$ hdfs dfs -ls project
17/10/26 23:34:12 WARN util.NativeCodeLoader: Unable to load native-hadoop libra
ry for your platform... using builtin-java classes where applicable
Found 1 items
drwxr-xr-x   - acadgild supergroup          0 2017-10-26 23:31 project/batch1
[acadgild@localhost ~]$ hdfs dfs -ls project/batch1
17/10/26 23:34:20 WARN util.NativeCodeLoader: Unable to load native-hadoop libra
ry for your platform... using builtin-java classes where applicable
Found 3 items
drwxr-xr-x   - acadgild supergroup          0 2017-10-26 23:31 project/batch1/fo
rmattedweb
drwxr-xr-x   - acadgild supergroup          0 2017-10-26 23:31 project/batch1/mo
b
drwxr-xr-x   - acadgild supergroup          0 2017-10-26 23:30 project/batch1/we
b
[acadgild@localhost ~]$
```

```
acadgild@localhost:~
File  Edit  View  Search  Terminal  Help
hive> use project;
OK
Time taken: 0.462 seconds
hive> show tables;
OK
formatted_input
users_artists
Time taken: 0.176 seconds, Fetched: 2 row(s)
hive> select * from formatted_input;
OK
U117    S204    A301    1495130523      1465130523      1475130523      A       S
T402    0       1       0       1
U115    S203    A305    1465230523      1465130523      1475130523      AP      S
T409    0       1       0       1
U117    S208    A305    1465130523      1465130523      1465130523      AP      S
T407    3       0       1       1
U111    S206    A303    1465230523      1485130523      1465130523      U       S
T414    1       0       0       1
U119    S207    A301    1465230523      1475130523      1485130523      AU      S
T408    1       1       1       1
        S209    A301    1465230523      1465230523      1485130523      U       S
T411    3       0       1       1
U112    S207    A302    1465230523      1465230523      1475130523      AU      S
T410    0       1       1       1
U118    S203    A304    1475130523      1465130523      1465230523      U       S
T403    0       0       0       1
U101    S204    A301    1475130523      1485130523      1485130523              S
T411    2       0       1       1
U103    S207            1465230523      1465130523      1465130523      A       S
T400    1       1       1       1
U113    S202    A300    1465130523      1475130523      1475130523      U       S
T415    1       1       0       1
U104    S206    A303    1495130523      1465130523      1475130523      U       S
T401    1       1       1       1
U113    S207    A305    1495130523      1465130523      1485130523      AU      S
T402    0       0       1       1
U101    S206    A305    1465130523      1465230523      1465230523      AP      S
T415    3       0       0       1
U110    S202    A303    1495130523      1465130523      1465130523      AP      S
```

Step 6 :Perform Data Enrichment and Cleaning

The data enrichment is carried out in two steps:
Create lookup tables in Hive and import the data from the HBase lookup tables to them. This is done by shell script data_enrichment_filtering_schema.sh
Perform the data enrichment to the data in formatted_input using the lookup tables. This is done by shell script data_enrichment.sh

1) data_enrichment_filtering_schema.sh

Below is the shell script data_enrichment_filtering_schema.sh where the following operations are performed:
Run the hive script create_hive_hbase_lookup.hql. This will create the lookup tables in Hive and import the data from the HBase lookup tables to the Hive lookup tables.

data_enrichment_filtering_schema.sh

```
#!/bin/bash

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

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

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

create_hive_hbase_lookup.hql

Create Hive lookup tables and save lookup table subscribed_users to Local FS.

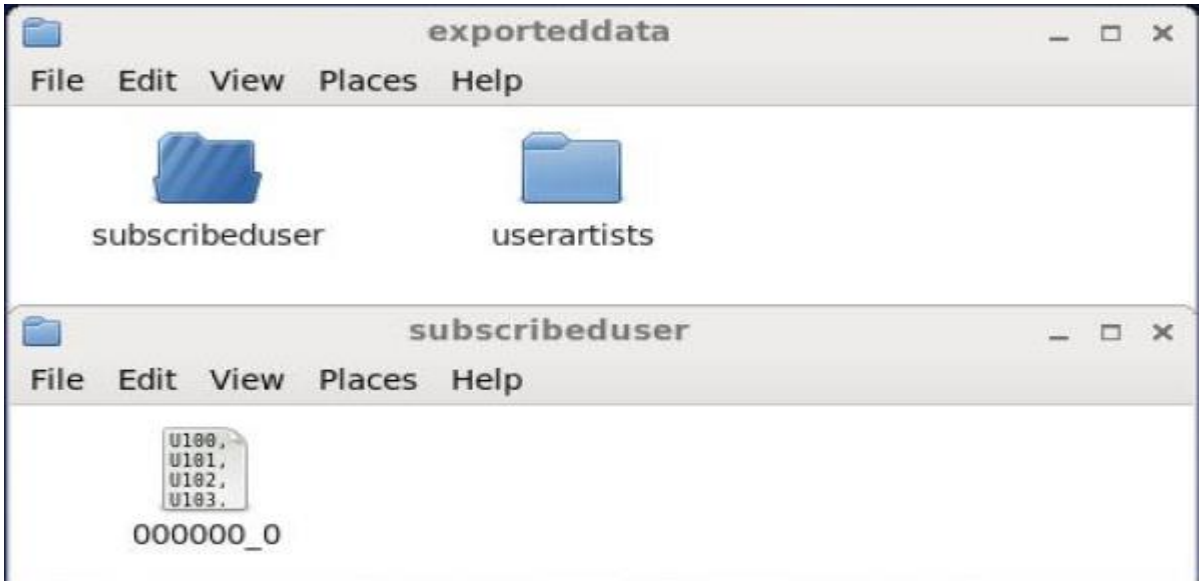
create_hive_hbase_lookup.hql

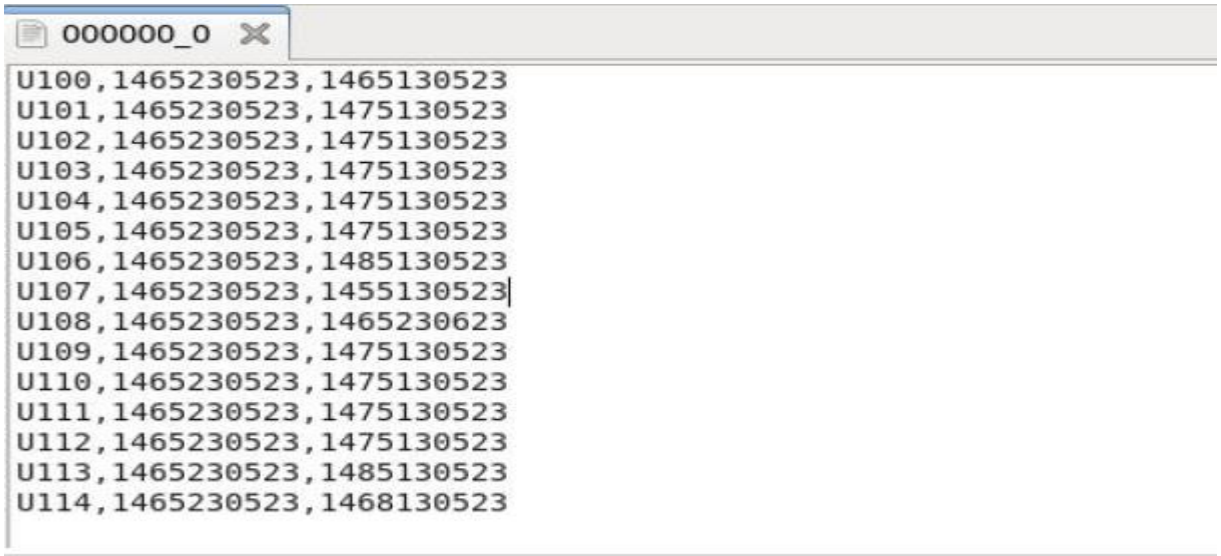
```
USE project;
create external table if not exists station_geo_map
(
station_id String,
geo_cd string
)
STORED BY 'org.apache.hadoop.hive.hbase.HBaseStorageHandler'
with serdeproperties
("hbase.columns.mapping"=":key,geo:geo_cd")
tblproperties("hbase.table.name"="station-geo-map");

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");
```

Plain Text

```
acadgild@localhost:~
File  Edit  View  Search  Terminal  Help
[acadgild@localhost ~]$ sh  /home/acadgild/project/scripts/data_enrichment_filtering_schema.sh
/usr/local/hive/bin/hive-config.sh: line 1: syntax error near unexpected token `('
/usr/local/hive/bin/hive-config.sh: line 1: `# Licensed to the Apache Software Foundation (ASF) under one or more'

Logging initialized using configuration in jar:file:/usr/local/hive/lib/hive-common-0.14.0.jar!/hive-log4j.properties
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/local/hive/lib/hive-jdbc-0.14.0-standalone.jar!/org/slf4j/impl/StaticLoggerBinder.clas
s]
SLF4J: Found binding in [jar:file:/usr/local/hadoop-2.6.0/share/hadoop/common/lib/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/Sta
ticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
OK
Time taken: 0.836 seconds
OK
Time taken: 0.988 seconds
OK
Time taken: 0.072 seconds
Query ID = acadgild_20171026234747_e7caedbc-00be-4646-8a5c-c3448db1fe46
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks is set to 0 since there's no reduce operator
Starting Job = job_1509034627624_0004, Tracking URL = http://localhost:8088/proxy/application_1509034627624_0004/
Kill Command = /home/acadgild/hadoop-2.6.0/bin/hadoop job  -kill job_1509034627624_0004
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 0
2017-10-26 23:47:42,134 Stage-1 map = 0%,  reduce = 0%
2017-10-26 23:47:56,464 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 2.67 sec
MapReduce Total cumulative CPU time: 2 seconds 670 msec
Ended Job = job_1509034627624_0004
Copying data to local directory /home/acadgild/project/exporteddata/subscribeduser
Copying data to local directory /home/acadgild/project/exporteddata/subscribeduser
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1   Cumulative CPU: 2.67 sec   HDFS Read: 276 HDFS Write: 405 SUCCESS
Total MapReduce CPU Time Spent: 2 seconds 670 msec
OK
Time taken: 37.744 seconds
OK
Time taken: 0.076 seconds
[acadgild@localhost ~]$
```

```
hive> use project;
OK
Time taken: 0.422 seconds
hive> show tables;
OK
formatted_input
song_artist_map
station_geo_map
subscribed_users
users_artists
Time taken: 0.179 seconds, Fetched: 5 row(s)
hive>
```





2) Data enrichmet.sh

Run the hive script data_enrichment.hql. This will create a Hive table enriched_data that will hold the data that is enriched and partitioned based on given rules as pass or fail (status) and batchid. Add logs to the Log File signifying that the valid and invalid outputs are being
recorded in their respective folders.Copy the data from the pass and fail folders (valid & invalid) in the Hive warehouse to
the Local FS.

data_enrichment.sh

```bash
#!/bin/bash

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

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

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

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

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

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

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

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

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

For the data enrichment, a table enriched_data is created and the table is overwritten with the
result of the below operations:
a)The data in the formatted_input table is joined with the lookup tables station_geo_map and song_artist_map to fill in the data gaps that can be obtained by said tables.
and
b)The same data is then filtered by the rules given above and partitioned by status (pass or fail)
& batchid.
The data of the enriched_data table is then stored in a folder in the Local FS

data_enrichment.hql

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

USE project;

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

```
INSERT OVERWRITE TABLE enriched_data
PARTITION (batchid, status)
SELECT
i.user_id,
i.song_id,
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
OR i.geo_cd IS NULL
OR i.user_id=''
OR i.song_id=''
OR i.timestamp=''
OR i.start_ts=''
OR i.end_ts=''
OR i.geo_cd=''
OR sg.geo_cd IS NULL
OR sg.geo_cd=''
OR sa.artist_id IS NULL
OR sa.artist_id='', 'fail', 'pass') AS status
FROM formatted_input i
LEFT OUTER JOIN station_geo_map sg ON i.station_id = sg.station_id
LEFT OUTER JOIN song_artist_map sa ON i.song_id = sa.song_id
WHERE i.batchid=${hiveconf:batchid};
```

```
INSERT OVERWRITE LOCAL DIRECTORY '/home/acadgild/project/exporteddata/enricheddata'
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
STORED AS TEXTFILE
SELECT * FROM enriched_data;
```

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

[acadgild@localhost ~]$ sh  /home/acadgild/project/scripts/data_enrichment.sh
/usr/local/hive/bin/hive-config.sh: line 1: syntax error near unexpected token `('
/usr/local/hive/bin/hive-config.sh: line 1: `# Licensed to the Apache Software Foundation (ASF) und

Logging initialized using configuration in jar:file:/usr/local/hive/lib/hive-common-0.14.0.jar!/hiv
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/local/hive/lib/hive-jdbc-0.14.0-standalone.jar!/org/slf4j/im
s]
SLF4J: Found binding in [jar:file:/usr/local/hadoop-2.6.0/share/hadoop/common/lib/slf4j-log4j12-1.7
ticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
OK
Time taken: 0.75 seconds
OK
Time taken: 0.955 seconds
Query ID = acadgild_20171027000606_53486740-5ec3-498f-a862-3c998ff9bed7
Total jobs = 2
Launching Job 1 out of 2
Number of reduce tasks not specified. Estimated from input data size: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1509034627624_0007, Tracking URL = http://localhost:8088/proxy/application_15090
Kill Command = /home/acadgild/hadoop-2.6.0/bin/hadoop job  -kill job_1509034627624_0007
Hadoop job information for Stage-1: number of mappers: 3; number of reducers: 1
2017-10-27 00:06:46,135 Stage-1 map = 0%,  reduce = 0%
2017-10-27 00:07:14,784 Stage-1 map = 67%,  reduce = 0%, Cumulative CPU 3.18 sec
2017-10-27 00:07:17,007 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 7.13 sec
2017-10-27 00:07:26,316 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 9.14 sec
MapReduce Total cumulative CPU time: 9 seconds 140 msec
Ended Job = job_1509034627624_0007
Launching Job 2 out of 2
Number of reduce tasks not specified. Estimated from input data size: 1
In order to change the average load for a reducer (in bytes):
```

```
acadgild@localhost:~
File  Edit  View  Search  Terminal  Help
Ended Job = job_1509034627624_0008
Loading data to table project.enriched_data partition (batchid=null, status=null)
         Time taken for load dynamic partitions : 1314
        Loading partition {batchid=1, status=pass}
        Loading partition {batchid=1, status=fail}
         Time taken for adding to write entity : 6
Partition project.enriched_data{batchid=1, status=fail} stats: [numFiles=1, numRows=20, totalSize
Partition project.enriched_data{batchid=1, status=pass} stats: [numFiles=1, numRows=20, totalSize
MapReduce Jobs Launched:
Stage-Stage-1: Map: 3  Reduce: 1   Cumulative CPU: 9.14 sec   HDFS Read: 3242 HDFS Write: 3088 SU
Stage-Stage-2: Map: 2  Reduce: 1   Cumulative CPU: 7.61 sec   HDFS Read: 3768 HDFS Write: 3040 SU
Total MapReduce CPU Time Spent: 16 seconds 750 msec
OK
Time taken: 113.461 seconds
Query ID = acadgild_20171027000808_b628ef63-ee10-46b8-af0e-1bf4398bec53
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks is set to 0 since there's no reduce operator
Starting Job = job_1509034627624_0009, Tracking URL = http://localhost:8088/proxy/application_150
Kill Command = /home/acadgild/hadoop-2.6.0/bin/hadoop job  -kill job_1509034627624_0009
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 0
2017-10-27 00:08:30,653 Stage-1 map = 0%,  reduce = 0%
2017-10-27 00:08:41,137 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 1.65 sec
MapReduce Total cumulative CPU time: 1 seconds 650 msec
Ended Job = job_1509034627624_0009
Copying data to local directory /home/acadgild/project/exporteddata/enricheddata
Copying data to local directory /home/acadgild/project/exporteddata/enricheddata
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1   Cumulative CPU: 1.65 sec   HDFS Read: 4611 HDFS Write: 2759 SUCCESS
Total MapReduce CPU Time Spent: 1 seconds 650 msec
OK
Time taken: 25.823 seconds
17/10/27 00:08:44 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platf
asses where applicable
get: `/home/acadgild/project/processed_dir/valid/batch_1/000000_0': File exists
17/10/27 00:08:47 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platf
asses where applicable
get: `/home/acadgild/project/processed_dir/invalid/batch_1/000000_0': File exists
[acadgild@localhost ~]$
```

```
hive> SHOW TABLES;
OK
enriched_data
formatted_input
song_artist_map
station_geo_map
subscribed_users
users_artists
Time taken: 0.035 seconds, Fetched: 6 row(s)
hive> SELECT * FROM enriched_data;
OK
U113    S200    A300    1465230523      1475130523      1465130523      J       ST413   3
U100    S200    A300    1494297562      1494297562      1494297562      A       ST410   3
U120    S201    A301    1494297562      1465490556      1468094889      A       ST410   3
U107    S202    A302    1495130523      1465230523      1465230523      NULL    ST415   0
U103    S202    A302    1465490556      1465490556      1465490556      NULL    ST415   2
U106    S202    A302    1465230523      1465130523      1465130523      E       ST408   0
U109    S203    A303    1462863262      1494297562      1468094889      A       ST405   1
        S203    A303    1495130523      1475130523      1465230523      A       ST400   0
U110    S203    A303    1465230523      1465130523      1485130523      NULL    ST415   0
U111    S204    A304    1465490556      1465490556      1468094889      A       ST410   3
U113    S204    A304    1494297562      1494297562      1465490556      NULL    ST415   3
U100    S204    A304    1495130523      1475130523      1465130523      E       ST408   2
U106    S205    A301    1462863262      1462863262      1494297562      AP      ST407   2
U108    S205    A301    1465130523      1465130523      1475130523      A       ST410   2
U111    S206    A302    1465130523      1465130523      1485130523      NULL    ST415   0
U114    S207    A303    1465130523      1465230523      1475130523      NULL    ST415   3
U102    S207    A303    1465230523      1485130523      1465230523      J       ST403   3
        S208    A304    1465490556      1494297562      1465490556      A       ST411   1
U118    S208    A304    1475130523      1465130523      1465230523      NULL    ST415   3
U119    S208    A304    1495130523      1465230523      1465230523      NULL    ST415   3
U101    S208    A304    1462863262      1468094889      1462863262      E       ST408   0
U107    S210    NULL    1475130523      1485130523      1485130523      E       ST404   2
U115    S200    A300    1465490556      1494297562      1465490556      E       ST404   3
U108    S200    A300    1468094889      1462863262      1468094889      E       ST414   0
U107    S202    A302    1494297562      1468094889      1462863262      E       ST409   0
U101    S202    A302    1465230523      1465130523      1475130523      AU      ST401   0
```

000000_0 (~/project/exporteddata/enricheddata) - gedit

File Edit View Search Tools Documents Help

Open Save Undo

000000_0

```
U113,S200,A300,1465230523,1475130523,1465130523,J,ST413,3,1,1,1,fail
U100,S200,A300,1494297562,1494297562,1494297562,A,ST410,3,1,1,1,fail
U120,S201,A301,1494297562,1465490556,1468094889,A,ST410,3,0,1,1,fail
U107,S202,A302,1495130523,1465230523,1465230523,\N,ST415,0,1,1,1,fail
U103,S202,A302,1465490556,1465490556,1465490556,\N,ST415,2,1,1,1,fail
U106,S202,A302,1465230523,1465130523,1465130523,E,ST408,0,1,1,1,fail
U109,S203,A303,1462863262,1494297562,1468094889,A,ST405,1,1,1,1,fail
,S203,A303,1495130523,1475130523,1465230523,A,ST400,0,0,1,1,fail
U110,S203,A303,1465230523,1465130523,1485130523,\N,ST415,0,1,1,1,fail
U111,S204,A304,1465490556,1465490556,1468094889,A,ST410,3,1,1,1,fail
U113,S204,A304,1494297562,1494297562,1465490556,\N,ST415,3,0,1,1,fail
U100,S204,A304,1495130523,1475130523,1465130523,E,ST408,2,1,1,1,fail
```

Step 7 : Data Analysis Using Spark

Get the batch id number from the batch file and get the Log File for the batch using the

batch id. This will be log_batch_1 and add logs to the Log File signifying that the data analysis is being performed using Spark and that the result is being exported to the Local FS.

Run the spark script data_analysis.scala. This will perform the data analysis required

in the problem statement given and save the result to the Local FS.

 Add logs to the Log File signifying that the data analysis has completed and that the batch is

being incremented. Here from 1 to 2

 Get batchid number from batch file and increment the batchid by 1

data-analysis.sh

```
#!/bin/bash

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

echo "Running spark script for analysis" >>$LOGFILE
echo "Exportng data to local fs" >>$LOGFILE

cat /home/acadgild/project/scripts/data_analysis.scala | spark-shell

echo "Activities completed" >>$LOGFILE
echo "Incrementing batchid" >>$LOGFILE

batchid=`expr $batchid + 1`

echo -n $batchid > /home/acadgild/project/logs/current-batch.txt
```

*data_analysis.scala

```
import org.apache.spark.sql.DataFrame

import org.apache.spark.sql.functions._


val batid = sc.textFile("/home/acadgild/project/logs/current-batch.txt").map(x=>x.toInt).toDF().firs

val musicdata= sc.textFile("/home/acadgild/project/exporteddata/enricheddata/000000_0")

case class music_schema
( user_id:String,song_id:String,artist_id:String,timestamp:String,start_ts:String,end_ts:String,geo_
String,station_id:String,song_end_type:Int,like:Int,dislike:Int,batchid :Int,status:String)



val music_rowrdd = musicdata.map(r=>r.split(",")).map(r=>music_schema(r(0),r(1),r(2),r(3),r(4),r(5),
(9).toInt,r(10).toInt,r(11).toInt,r(12)))toDF

 music_rowrdd.registerTempTable("music_data")



val subscriber_data= sc.textFile("/home/acadgild/project/exporteddata/subscribeduser/000000_0")

case class subscriber_schema (user_id : String, start_dt:String,end_dt:String)


val subscriber_rowrdd = subscriber_data.map(r=>r.split(",")).map(r=>subscriber_schema(r(0),r(1),r(2)


 subscriber_rowrdd.registerTempTable("subscribed_users")
```

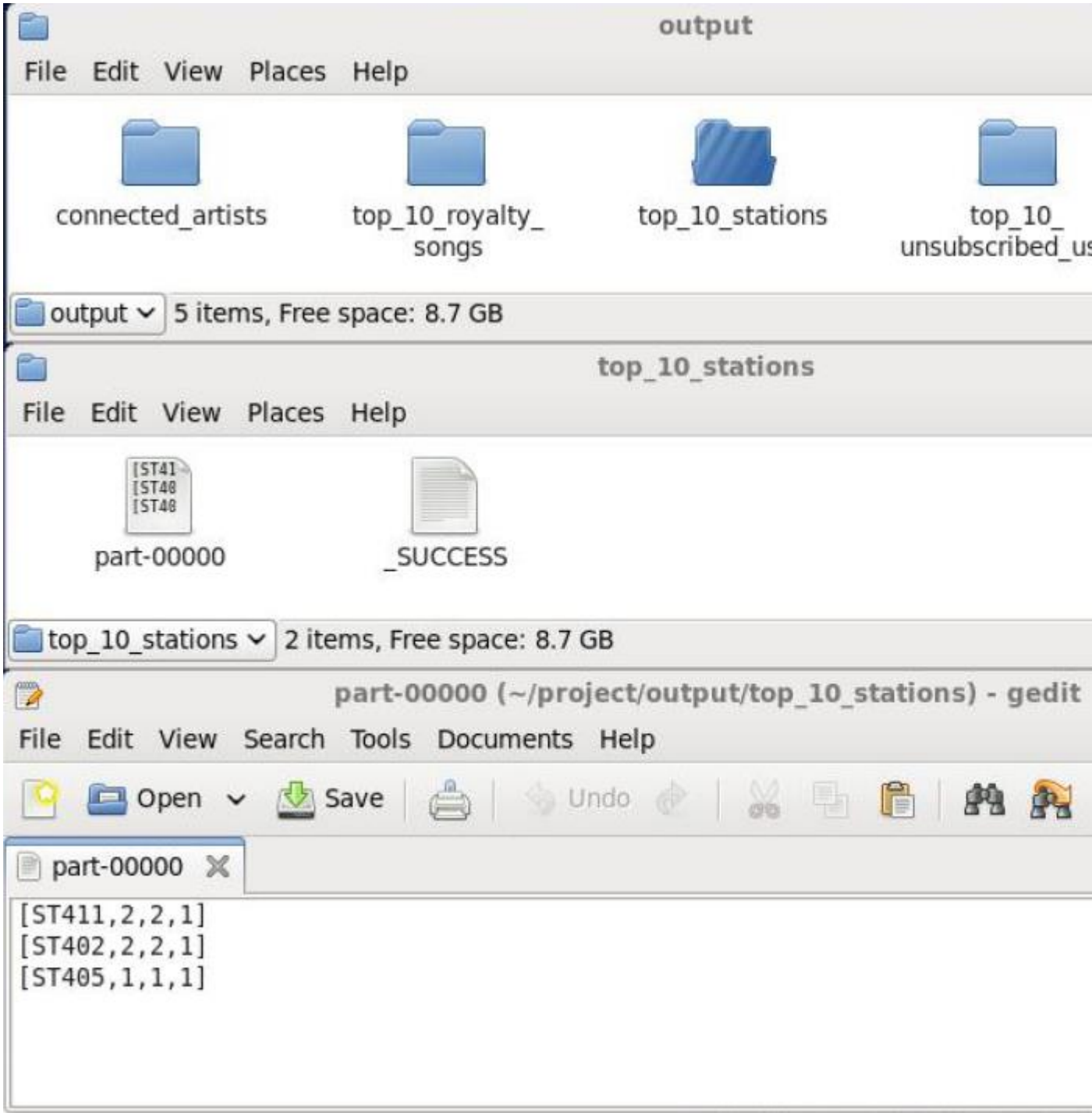
```
val artists_data= sc.textFile("/home/acadgild/project/exporteddata/userartists/000000_0")

case class artist_schema(user_id : String, artists:String)

val artist_rowrdd = artists_data.map(r=>r.split(",")).map(r=>artist_schema(r(0),r(1))).toDF



 artist_rowrdd.registerTempTable("user_artists")


val sqlContext = new org.apache.spark.sql.SQLContext(sc)
import sqlContext.implicits._




val top10stations = sqlContext.sql("Select station_id,COUNT(DISTINCT song_id) as total_distinct_son
user_id) as distinct_user,batchid From music_data WHERE status = 'pass' and batchid =$batid AND lik
station_id,batchid ORDER BY total_distinct_songs_played DESC LIMIT 10")

top10stations.rdd.saveASTextFile("/home/acadgild/project/output/top 10 stations")
```

```
val users_behaviour = sqlContext.sql("Select CASE WHEN (su.user_id IS NULL OR CAST(md.timestamp
(su.end_dt AS DECIMAL(20,0))) THEN 'UNSUBSCRIBED'WHEN (su.user_id IS NOT NULL AND CAST(md.timest
CAST(su.end_dt AS DECIMAL(20,0))) THEN 'SUBSCRIBED' END AS user_type,SUM(ABS(CAST(md.end_ts AS D
(md.start_ts AS DECIMAL(20,0)))) AS duration,batchid FROM music_data md LEFT OUTER JOIN subscrib
md.user_id=su.user_id WHERE md.status='pass'
AND md.batchid=$batid GROUP BY CASE WHEN (su.user_id IS NULL OR CAST(md.timestamp AS DECIMAL(20,
DECIMAL(20,0))) THEN 'UNSUBSCRIBED' WHEN (su.user_id IS NOT NULL AND CAST(md.timestamp AS DECIMA
AS DECIMAL(20,0))) THEN 'SUBSCRIBED' END,batchid")


users_behaviour.rdd.saveASTextFile("/home/acadgild/project/output/user_behaviour")


val connectd_artists = sqlContext.sql("Select ua.artists,COUNT(DISTINCT ua.user_id) as usercount
user_artists ua INNER JOIN(select artis_id,song_id,user_id,batchid from musc_data Where status =
$batid) md on ua.artists = md.artist_id and ua.user_id = md.user_id Group by ua.artists,batchid
LIMIT 10")

connectd_artists.rdd.saveASTextFile("/home/acadgild/project/output/connected_artists")

val top10_royality = sqlContext.sql("select song_id,SUM(ABS(CAST(end_ts as DECIMAL(20,0)) - CAST
(20,0))))) as duration,batchid,from music_data Where status - 'pass' and  batchid =$batid and (l
GROUP BY song_id,batchid,Order by duration DESC Limit 10

top10_royality.rdd.saveASTextFile("/home/acadgild/project/output/top10_royality")

val top10_unsubscibed = sqlContext.sql("select m.user_id,SUM(ABS(CAST(md.end_ts as DECIMAL(20,0)
DECIMAL(20,0))))) as duration from music_data md LEFT OUTER JOIN subscribed_users su on md.useri
md.status = 'pass' and batch id =$batid and (su.user_is IS NULL or CAST(md.timestamp AS DECIMAL(
DECIMAL(20,0))))) GROUP BY md.user_id ORDER BY duration DESC Limit 10

top10_unsubscibed.rdd.saveASTextFile("/home/acadgild/project/output/top 10 unsubscibed")
```

Problem Statement 1:

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

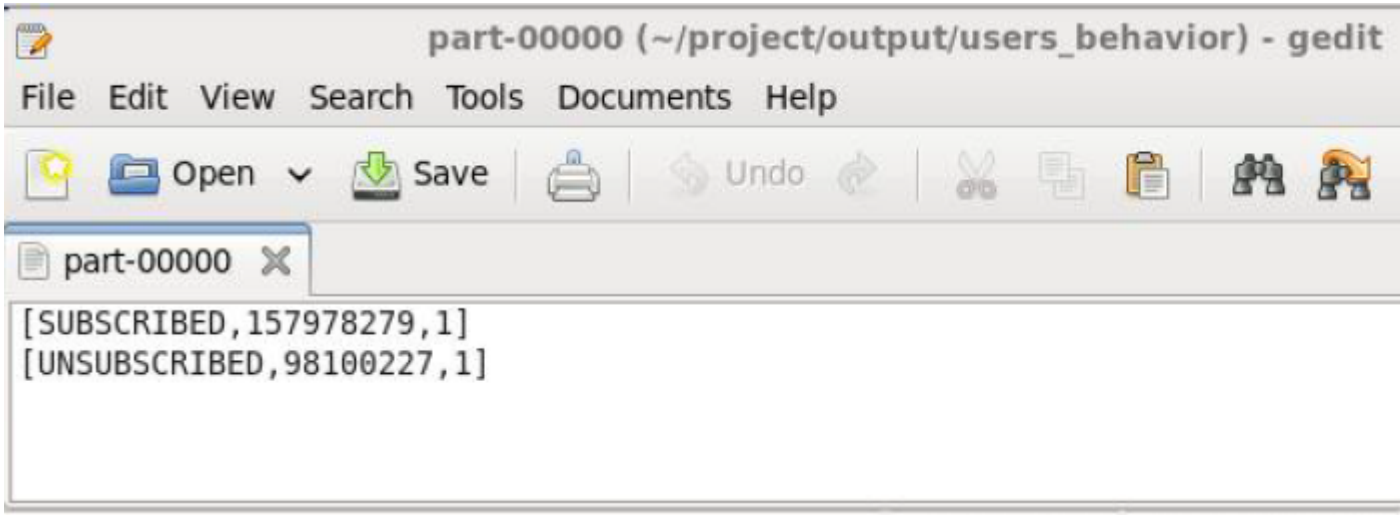
```
val top10stations = sqlContext.sql("Select station_id,COUNT(DISTINCT song_id) as total_
user_id) as distinct_user,batchid From music_data WHERE status = 'pass' and batchid =$b
station_id,batchid ORDER BY total_distinct_songs_played DESC LIMIT 10")

top10stations.rdd.saveASTextFile("/home/acadgild/project/output/top_10 stations")
```



<u>Problem Statement 2</u>:

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

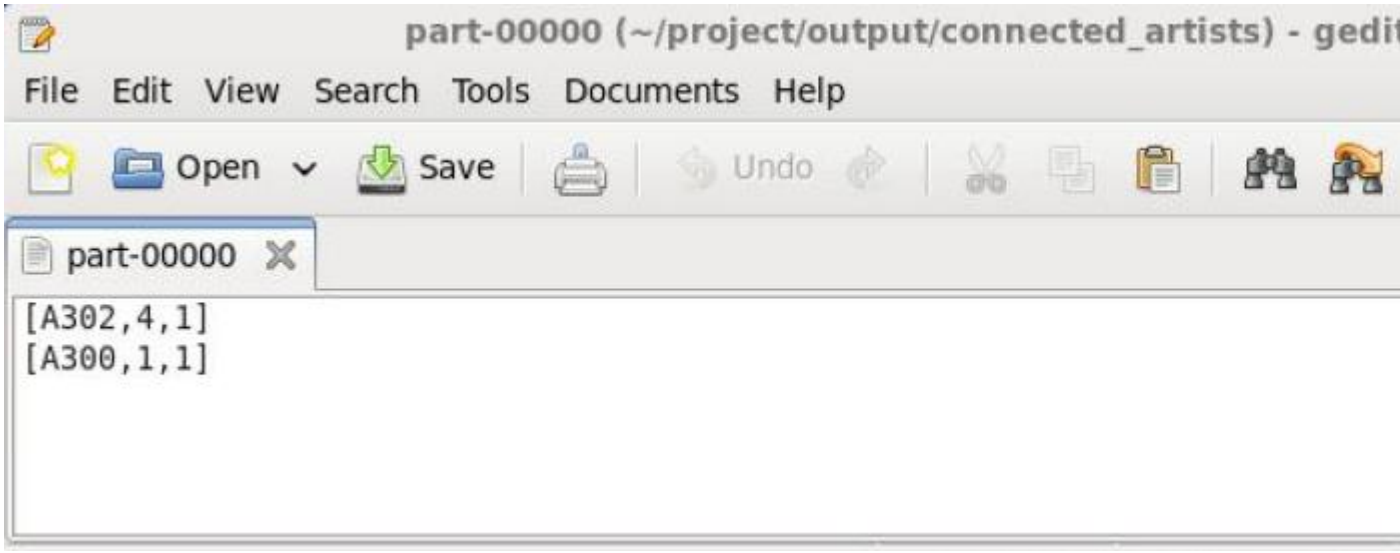
```
val users_behaviour = sqlContext.sql("Select CASE WHEN (su.user_id IS NULL OR CAST(md.t
(su.end_dt AS DECIMAL(20,0))) THEN 'UNSUBSCRIBED'WHEN (su.user_id IS NOT NULL AND CAST(
CAST(su.end_dt AS DECIMAL(20,0))) THEN 'SUBSCRIBED' END AS user_type,SUM(ABS(CAST(md.en
(md.start_ts AS DECIMAL(20,0)))) AS duration,batchid FROM music_data md LEFT OUTER JOIN
md.user_id=su.user_id WHERE md.status='pass'
AND md.batchid=$batid GROUP BY CASE WHEN (su.user_id IS NULL OR CAST(md.timestamp AS DE
DECIMAL(20,0))) THEN 'UNSUBSCRIBED' WHEN (su.user_id IS NOT NULL AND CAST(md.timestamp
AS DECIMAL(20,0))) THEN 'SUBSCRIBED' END,batchid")


users_behaviour.rdd.saveASTextFile("/home/acadgild/project/output/user_behaviour")
```



Problem Statement 3:

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

```
val connectd_artists = sqlContext.sql("Select ua.artists,COUNT(DISTINCT ua.user_id) as
user_artists ua INNER JOIN(select artis_id,song_id,user_id,batchid from musc_data Where
$batid) md on ua.artists = md.artist_id and ua.user_id = md.user_id Group by ua.artists
LIMIT 10")
```

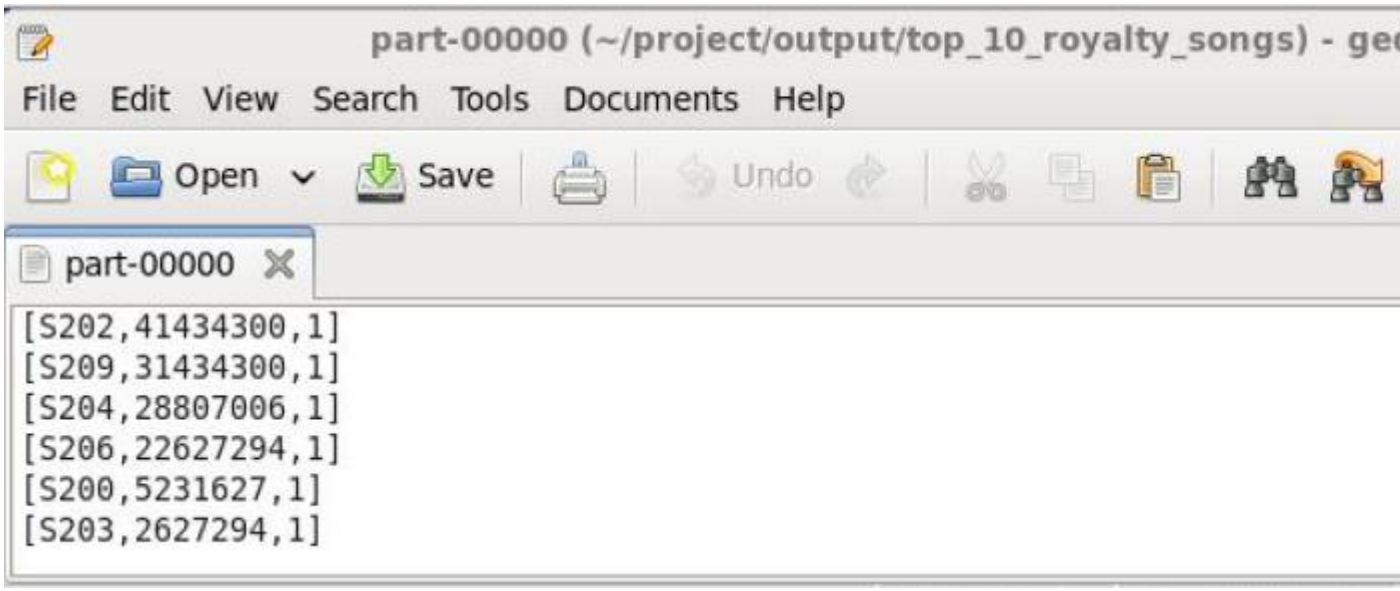


Problem Statement 4:

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

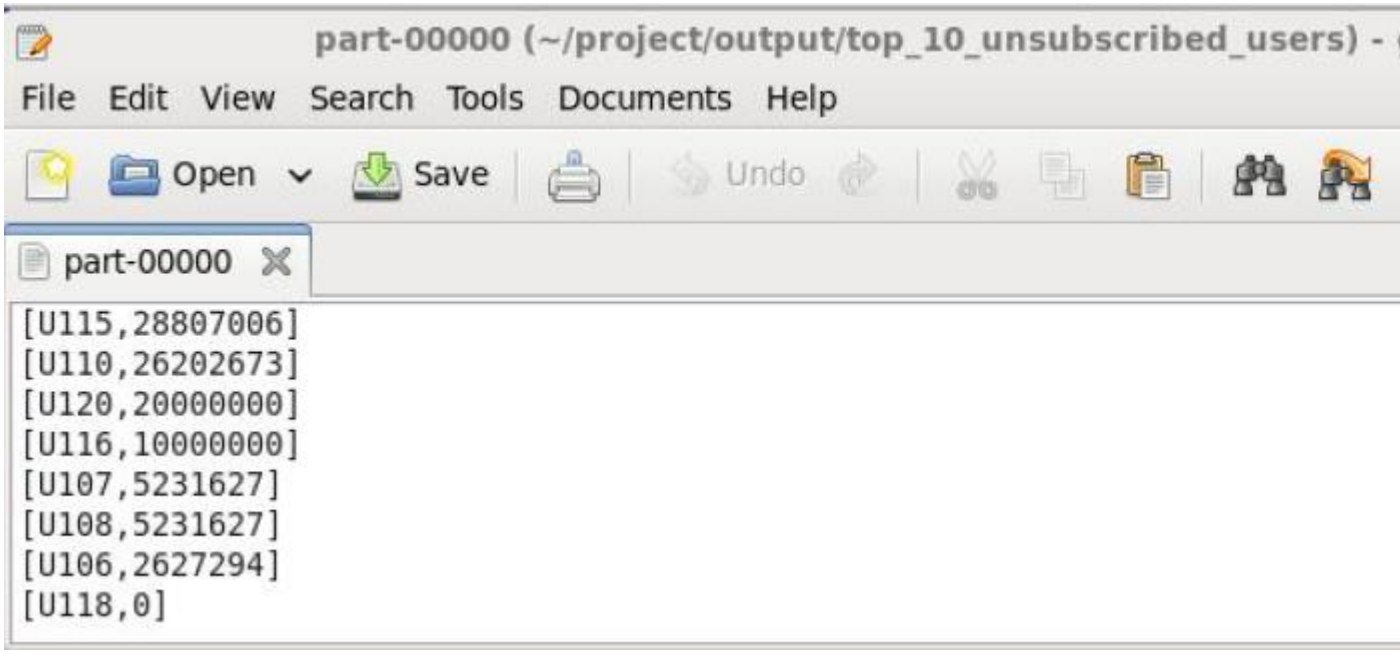
```
val top10_royality = sqlContext.sql("select song_id,SUM(ABS(CAST(end_ts as DECIMAL(20,0
(20,0))))) as duration,batchid,from music_data Where status - 'pass' and  batchid =$bat
GROUP BY song_id,batchid,Order by duration DESC Limit 10

top10_royality.rdd.saveASTextFile("/home/acadgild/project/output/top10_royality")
```



Problem Statement 5:

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

```
val top10_unsubscibed = sqlContext.sql("select m.user_id,SUM(ABS(CAST(md.end_ts as DECI
DECIMAL(20,0))))) as duration from music_data md LEFT OUTER JOIN subscribed_users su or
md.status = 'pass' and batch_id =$batid and (su.user_is IS NULL or CAST(md.timestamp AS
DECIMAL(20,0))))) GROUP BY md.user_id ORDER BY duration DESC Limit 10

top10_unsubscibed.rdd.saveASTextFile("/home/acadgild/project/output/top 10 unsubscibed"
```



Step 8 Post Analysis

Check the log files

log_batch_1

```
Starting daemons
Creating LookUp Tables
Populating LookUp Tables
Placing data files from local to HDFS...
Running pig script for data formatting...
Running hive script for formatted data load...
Creating hive tables on top of hbase tables for data enrichment and filte
Running hive script for data enrichment and filtering...
Copying valid and invalid records in local file system...
Deleting older valid and invalid records from local file system...
Running Spark Script for Data Analysis...
Exporting analyzed data to Local FS...
All Activities Complete...
Incrementing batchid...
```