

ACADGILD
**BIG DATA HADOOP & SPARK
TRAINING**

Final PROJECT

Music Data Analysis

From :

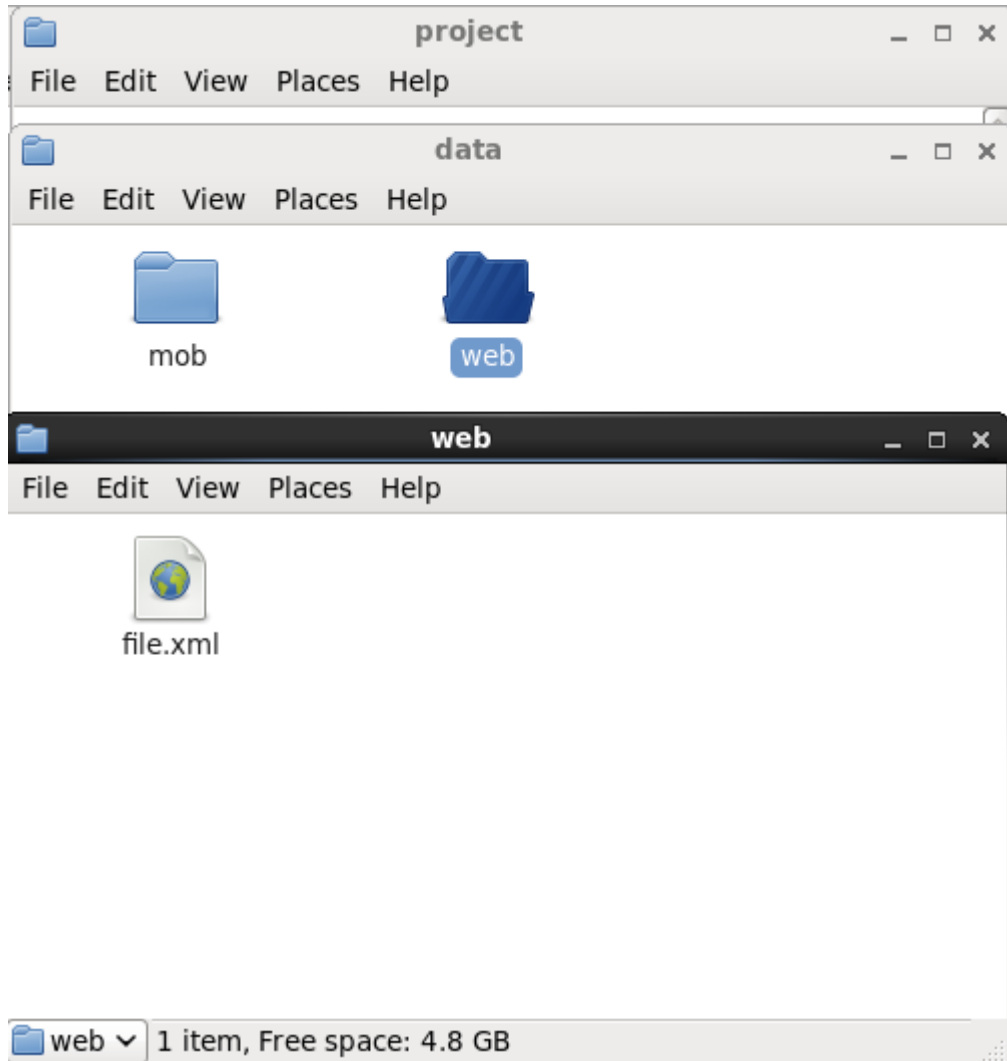
MOHAMMED FATHAULLA

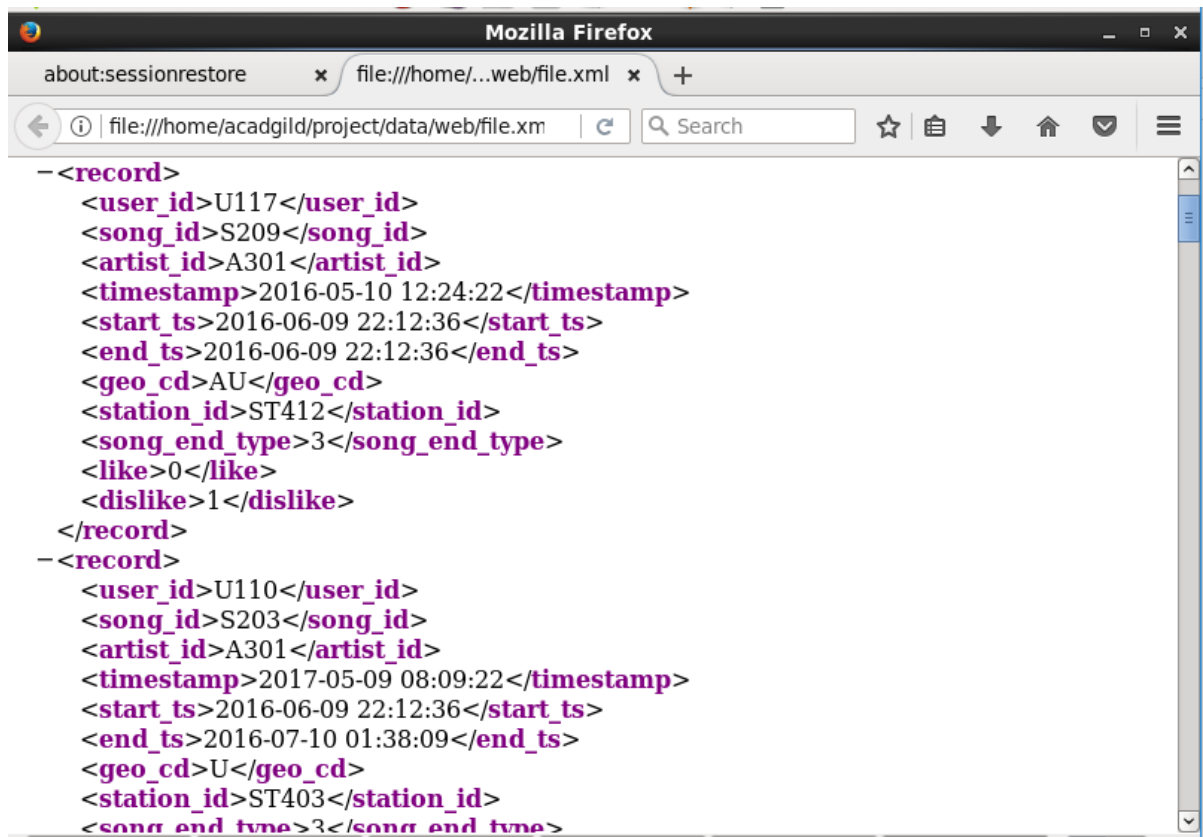
Data Description

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 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 /data/web and has **xml** format.

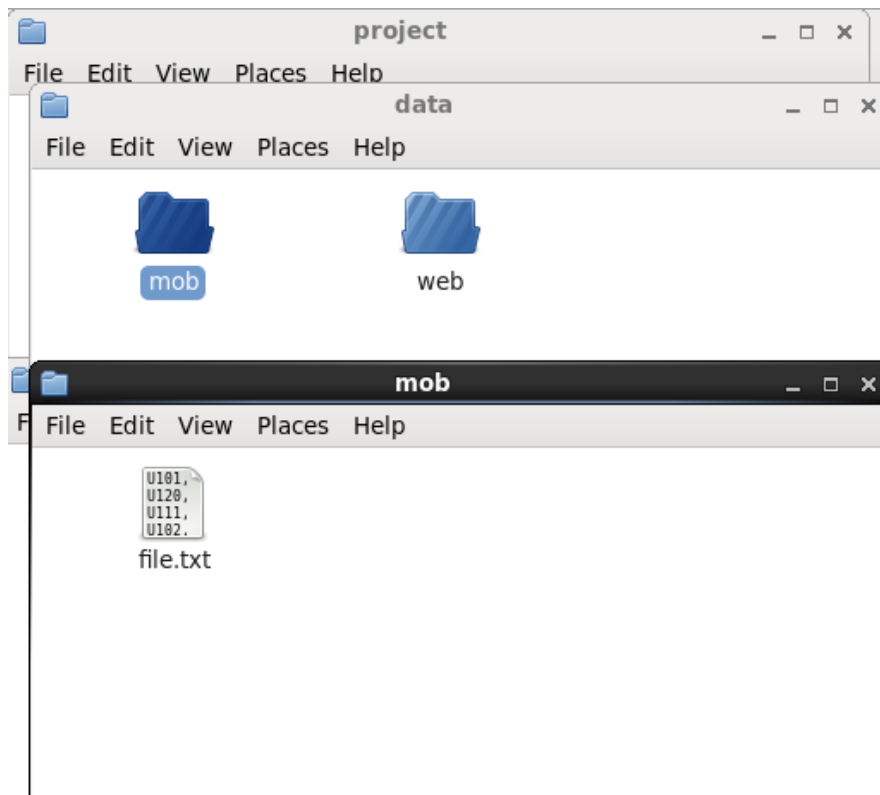




The screenshot shows a Mozilla Firefox browser window with the address bar displaying `file:///home/...web/file.xml`. The main content area shows the XML data from the file `file:///home/acadgild/project/data/web/file.xml`. The XML contains two records, each with various attributes like `user_id`, `song_id`, `artist_id`, `timestamp`, `start_ts`, `end_ts`, `geo_cd`, `station_id`, `song_end_type`, `like`, and `dislike`.

```
-<record>
  <user_id>U117</user_id>
  <song_id>S209</song_id>
  <artist_id>A301</artist_id>
  <timestamp>2016-05-10 12:24:22</timestamp>
  <start_ts>2016-06-09 22:12:36</start_ts>
  <end_ts>2016-06-09 22:12:36</end_ts>
  <geo_cd>AU</geo_cd>
  <station_id>ST412</station_id>
  <song_end_type>3</song_end_type>
  <like>0</like>
  <dislike>1</dislike>
</record>
-<record>
  <user_id>U110</user_id>
  <song_id>S203</song_id>
  <artist_id>A301</artist_id>
  <timestamp>2017-05-09 08:09:22</timestamp>
  <start_ts>2016-06-09 22:12:36</start_ts>
  <end_ts>2016-07-10 01:38:09</end_ts>
  <geo_cd>U</geo_cd>
  <station_id>ST403</station_id>
  <song_end_type>3</song_end_type>
```

Below is a sample of the data coming from mobile applications, that reside in `/data/mob` and has csv format



```
file.txt (~project/data/mob) - gedit
File Edit View Search Tools Documents Help
Open Save Undo
file.txt
U101,S204,A301,1465130523,1465230523,1475130523,U,ST407,2,1,0
U120,S200,A303,1495130523,1465230523,1465130523,A,ST407,1,0,1
U111,S200,A305,1495130523,1465130523,1475130523,U,ST407,1,1,0
U102,S203,A305,1475130523,1475130523,1475130523,AU,ST413,2,0,1
U111,S204,A302,1465130523,1475130523,1465230523,AU,ST415,0,1,0
,S209,A305,1465230523,1465230523,1475130523,AU,ST410,1,0,0
U116,S210,A303,1465230523,1465130523,1465230523,E,ST409,3,1,0
U108,S202,A301,1475130523,1475130523,1475130523,U,ST411,1,0,0
U108,S210,A302,1465230523,1465130523,1465130523,,ST400,0,1,1
U106,S206,,1495130523,1475130523,1485130523,AU,ST412,1,1,1
U120,S204,A305,1495130523,1465230523,1475130523,E,ST413,3,0,0
U119,S201,A304,1475130523,1475130523,1465230523,U,ST414,1,1,0
U111,S205,A301,1465230523,1465230523,1465230523,A,ST404,3,0,1
U107,S206,A305,1475130523,1475130523,1485130523,AU,ST406,2,0,1
U111,S210,A304,1495130523,1475130523,1485130523,AU,ST413,3,0,1
U115,S200,A300,1495130523,1485130523,1465230523,A,ST413,0,0,0
U111,S209,A304,1465230523,1475130523,1475130523,A,ST408,0,0,1
U120,S203,A304,1465130523,1485130523,1485130523,A,ST410,1,1,0
U112,S201,A305,1465230523,1465230523,1485130523,E,ST413,0,1,0
U111,S201,A300,1495130523,1465130523,1485130523,E,ST408,0,0,0
```

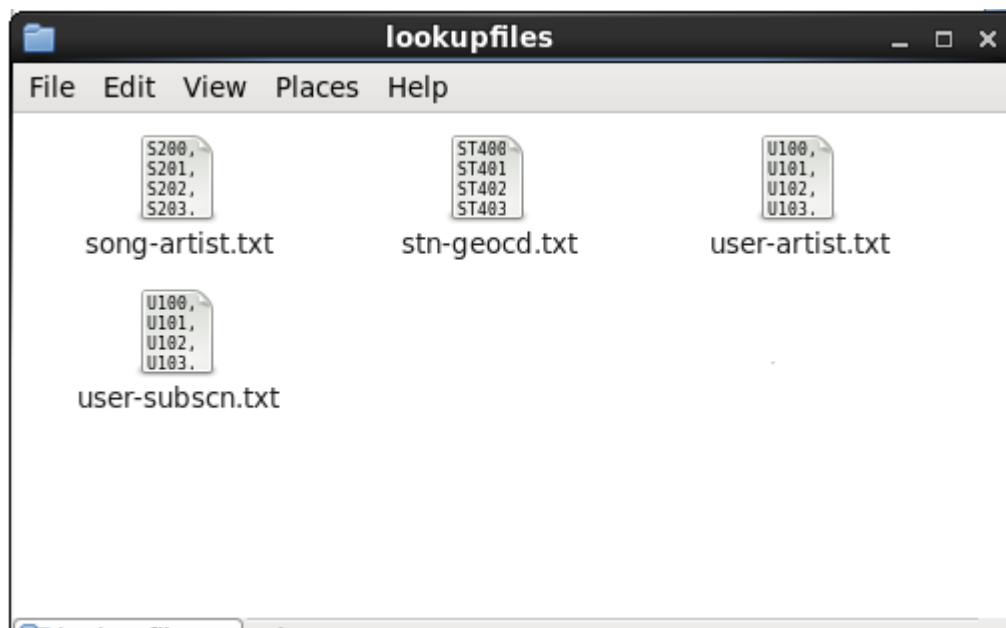
Look-Up Tables Files:

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.

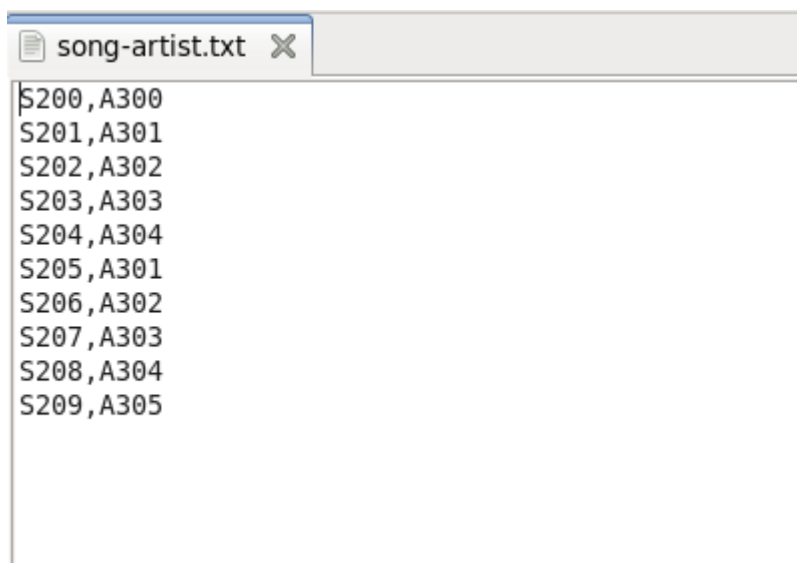
Table Name	Description
Station_Geo_Map	Contains mapping of a geo_cd with station_id
Subscribed_Users	Contains user_id, subscription_start_date and subscription_end_date. Contains details only for subscribed users
Song_Artist_Map	Contains mapping of song_id with artist_id alongwith royalty associated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

Below are the data to be present in the lookup tables. There are 4 tables:



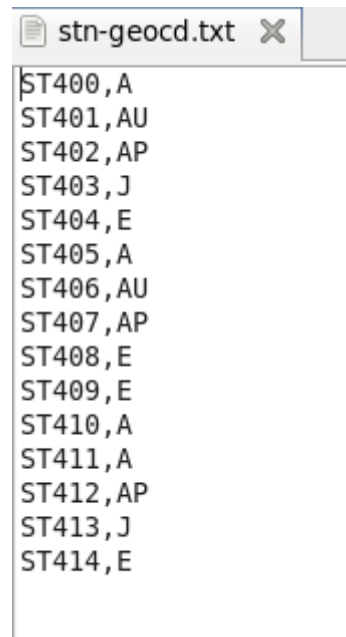
song-artist

Columns: **song_id**, **artist_id**



stn-geocd

Columns: **station_id**, **geo_cd**

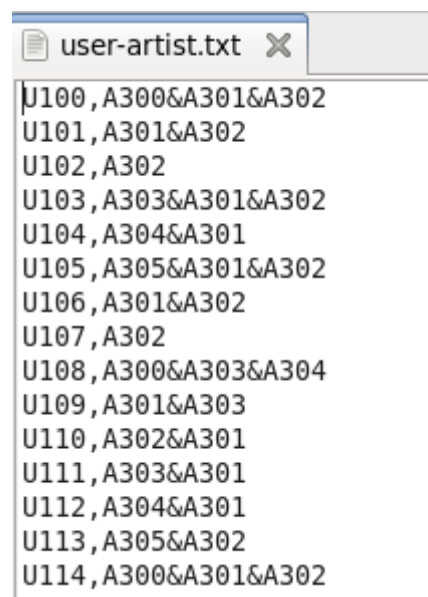


```
stn-geocd.txt X
```

station_id	geo_cd
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**

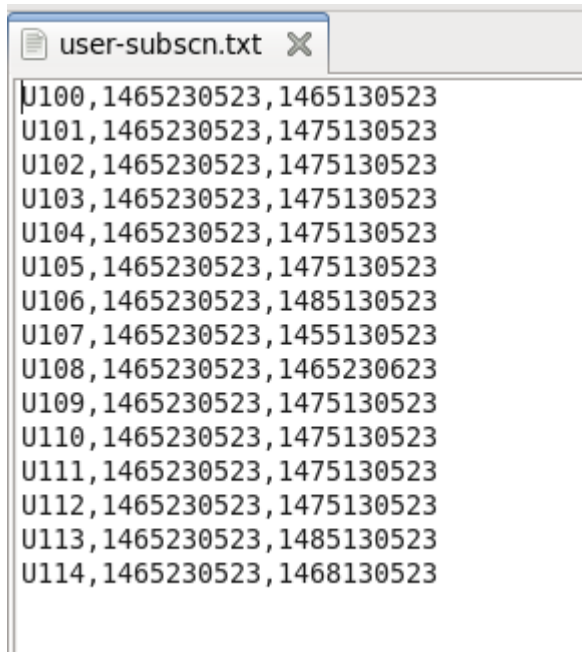


```
user-artist.txt X
```

user_id	artists_array
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**



```
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
```

Steps to perform data analysis on the Music Data:

Step 1: Launch all necessary daemons

Step 2: Start Job Scheduling (using Crontab)

Step 3: Populate Look-Up tables (i.e. Load all data to HBase)

Step 4: Perform Data Formatting (using Pig and Hive)

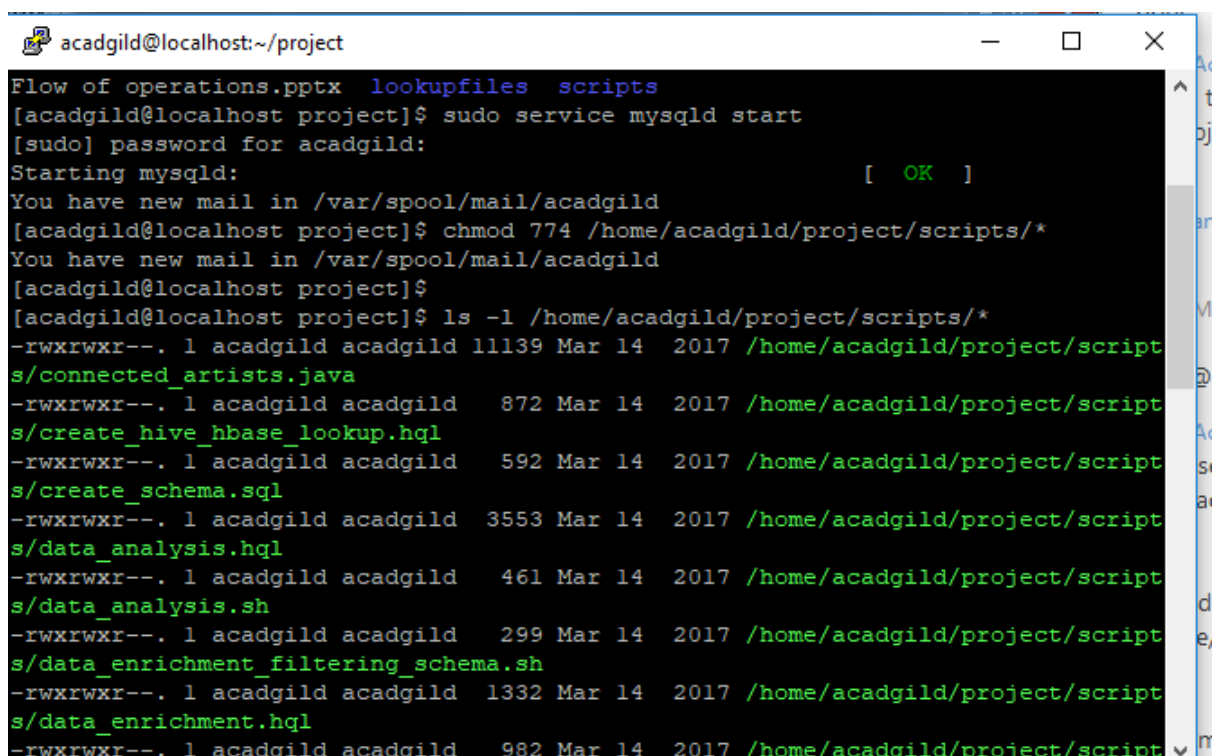
Step 5: Perform Data Enrichment and Cleaning (using Hive)

Step 6: Perform Data Analysis (using Spark)

Step 1:

Launch all necessary daemons

- Launch the Mysql Service (needed for Hive)
`sudo service mysqld start`
- Give permissions to scripts folder in project, so we are able to run scripts from the bash shell.
`chmod 774 /home/acadgild/project/scripts/*`
`ls -l /home/acadgild/project/scripts/*`



```
acadgild@localhost:~/project
Flow of operations.pptx  lookupfiles  scripts
[acadgild@localhost project]$ sudo service mysqld start
[sudo] password for acadgild:
Starting mysqld: [ OK ]
You have new mail in /var/spool/mail/acadgild
[acadgild@localhost project]$ chmod 774 /home/acadgild/project/scripts/*
You have new mail in /var/spool/mail/acadgild
[acadgild@localhost project]$
[acadgild@localhost project]$ ls -l /home/acadgild/project/scripts/*
-rwxrwxr--. 1 acadgild acadgild 11139 Mar 14 2017 /home/acadgild/project/scripts/connected_artists.java
-rwxrwxr--. 1 acadgild acadgild 872 Mar 14 2017 /home/acadgild/project/scripts/create_hive_hbase_lookup.hql
-rwxrwxr--. 1 acadgild acadgild 592 Mar 14 2017 /home/acadgild/project/scripts/create_schema.sql
-rwxrwxr--. 1 acadgild acadgild 3553 Mar 14 2017 /home/acadgild/project/scripts/data_analysis.hql
-rwxrwxr--. 1 acadgild acadgild 461 Mar 14 2017 /home/acadgild/project/scripts/data_analysis.sh
-rwxrwxr--. 1 acadgild acadgild 299 Mar 14 2017 /home/acadgild/project/scripts/data_enrichment_filtering_schema.sh
-rwxrwxr--. 1 acadgild acadgild 1332 Mar 14 2017 /home/acadgild/project/scripts/data_enrichment.hql
-rwxrwxr--. 1 acadgild acadgild 982 Mar 14 2017 /home/acadgild/project/scripts/data_enrichment.hql
```

- Run the shell script start-daemons.sh
`sh /home/acadgild/project/scripts/start-daemons.sh`

```

acadgild@localhost:~
[acadgild@localhost ~]$ sudo service mysqld start
[sudo] password for acadgild:
Starting mysqld: [ OK ]
[acadgild@localhost ~]$ chmod 774 /home/acadgild/project/scripts/*
You have new mail in /var/spool/mail/acadgild
[acadgild@localhost ~]$ sh /home/acadgild/project/scripts/start-daemons.sh
Batch File Found!
This script is Deprecated. Instead use start-dfs.sh and start-yarn.sh
18/05/31 20:05:08 WARN util.NativeCodeLoader: Unable to load native-hadoop libra
ry for your platform... using builtin-java classes where applicable
Starting namenodes on [localhost]
localhost: namenode running as process 3393. Stop it first.
localhost: datanode running as process 3488. Stop it first.
Starting secondary namenodes [0.0.0.0]
0.0.0.0: secondarynamenode running as process 3618. Stop it first.
18/05/31 20:05:18 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Starting yarn daemons
resourcemanager running as process 3787. Stop it first.
localhost: nodemanager running as process 3889. Stop it first.
localhost: starting zookeeper, logging to /home/acadgild/install/hbase/hbase-1.2.6/logs/hbase-acadgild-zookeeper-localhost.localdomain.out
starting master, logging to /home/acadgild/install/hbase/hbase-1.2.6/logs/hbase-acadgild-master-localhost.localdomain.out
starting regionserver, logging to /home/acadgild/install/hbase/hbase-1.2.6/logs/hbase-acadgild-l-regionserver-localhost.localdomain.out
historyserver running as process 4335. Stop it first.
[acadgild@localhost ~]$

```

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

```

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

```

- Check if a file **current-batch.txt** has been created or not, If already created, print **Batch File Found!** else create the file and add 1 to it to signify batch 1.

```

populate-lookup.sh start-daemon.sh current-batch.txt
1

```

- Give permissions to the file, so that we are able to modify it on the run.
- Get the batch id number from the batch file created above and create a Log File for the batch using the batch id. This will be log_batch_1.
Through out the course of the analysis process this log file will document the tasks that are performed for the Music Data Analysis.
- Add a log to the Log File signifying that the all necessary daemons have been started



- Start the dfs, yarn, hbase and jobhistory daemons.

Step 2:

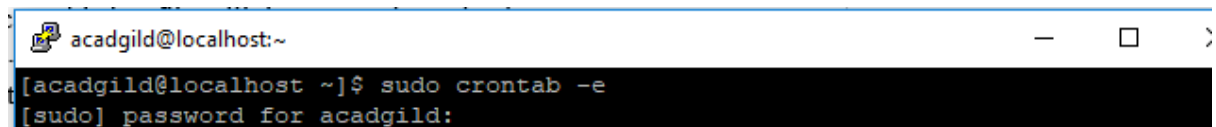
Start Job Scheduling

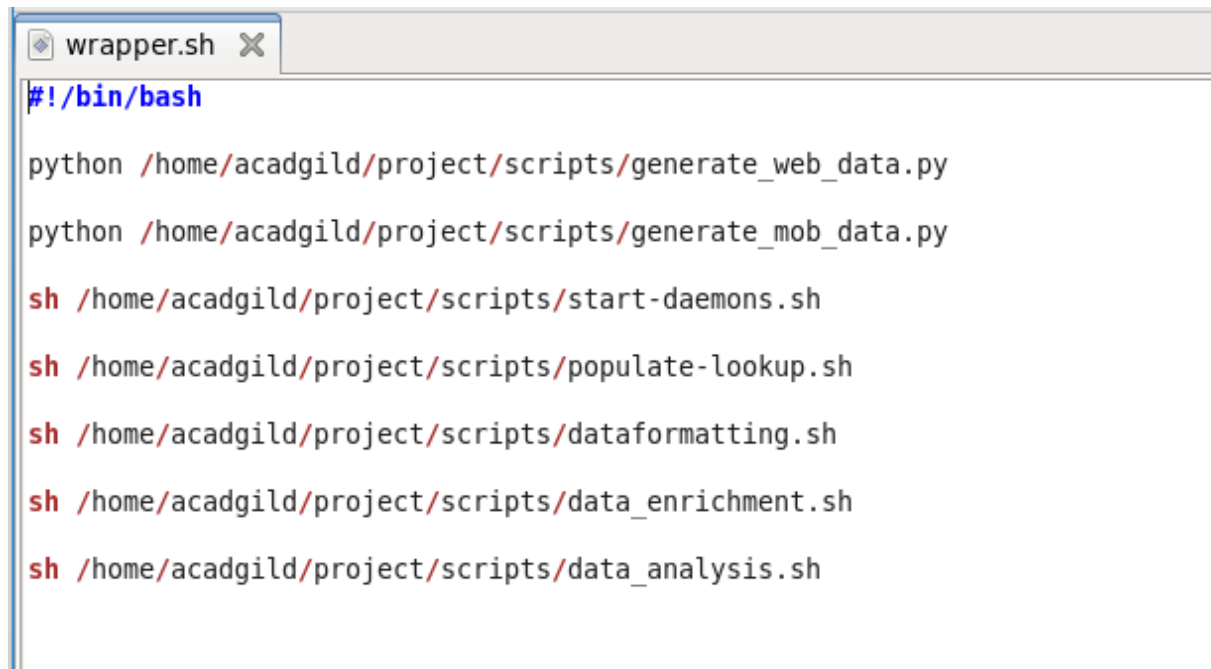
- Open the crontab file and insert the statement:

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

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

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



A terminal window titled 'wrapper.sh' with a close button. The prompt is '#!/bin/bash'. The following commands are executed in sequence:

```
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:

Populate Look-Up tables

Below is the shell script populate-lookup.sh that is used to load the data for the lookup tables into HBase tables.

The following operations are performed:

- 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
- Add logs to the Log File signifying that the lookup tables are being created and populated
- Create the HBase tables for the lookup data files: 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
- 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.

```
populate-lookup.sh X
#!/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"
```

```
populate-lookup.sh X
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
```

After the data in user-artist is loaded in the Hive Table users-artists, it is then saved as a text file as below (for data analysis using spark)

```
user-artist.hql X
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;
```

Below is a view of the execution of the above:

```
acadgild@localhost:~/project
[acadgild@localhost project]$ sh /home/acadgild/project/scripts/populate-lookup.sh
2018-05-26 12:13:43,804 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/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
HBase Shell; enter 'help<RETURN>' for list of supported commands.
Type "exit<RETURN>" to leave the HBase Shell
Version 1.2.6, rUnknown, Mon May 29 02:25:32 CDT 2017

create 'station-geo-map', 'geo'
0 row(s) in 6.5530 seconds

Hbase::Table - station-geo-map
```

acadgild@localhost:~/project

```
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

put 'station-geo-map', 'ST402', 'geo:geo_cd', 'AP'
0 row(s) in 1.2300 seconds

2018-05-26 12:16:48,240 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/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
HBase Shell; enter 'help<RETURN>' for list of supported commands.
Type "exit<RETURN>" to leave the HBase Shell
Version 1.2.6, rUnknown, Mon May 29 02:25:32 CDT 2017

put 'station-geo-map', 'ST403', 'geo:geo_cd', 'J'
```

acadgild@localhost:~/project

```
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/acadgild/install/hive/apache-hive-2.3.2-bin/lib/log4j-slf4j-impl-2.6.2.jar!/org/slf4j/impl/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/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFactory]

Logging initialized using configuration in jar:file:/home/acadgild/install/hive/apache-hive-2.3.2-bin/lib/hive-common-2.3.2.jar!/hive-log4j2.properties Async: true
OK
Time taken: 21.131 seconds
OK
Time taken: 0.077 seconds
OK
Time taken: 2.596 seconds
Loading data to table project.users_artists
OK
Time taken: 4.603 seconds
You have new mail in /var/spool/mail/acadgild
[acadgild@localhost project]$
```


Output of the above (HBase):

```
acadgild@localhost:~  
[acadgild@localhost ~]$ hbase shell  
2018-05-26 12:39:14,791 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/impl/StaticLoggerBinder.class]  
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.  
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]  
HBase Shell; enter 'help<RETURN>' for list of supported commands.  
Type "exit<RETURN>" to leave the HBase Shell  
Version 1.2.6, rUnknown, Mon May 29 02:25:32 CDT 2017  
  
hbase(main):001:0>   
  
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/impl/StaticLoggerBinder.class]  
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.  
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]  
HBase Shell; enter 'help<RETURN>' for list of supported commands.  
Type "exit<RETURN>" to leave the HBase Shell  
Version 1.2.6, rUnknown, Mon May 29 02:25:32 CDT 2017  
  
hbase(main):001:0> list  
TABLE  
acadgilddb.transactions_hbase  
ht  
htest  
song-artist-map  
station-geo-map  
subscribed-users  
6 row(s) in 1.0280 seconds  
  
=> ["acadgilddb.transactions_hbase", "ht", "htest", "song-artist-map", "station-geo-map", "subscribed-users"]  
hbase(main):002:0> 
```

acadgild@localhost:~

hbase(main):008:0> scan 'song-artist-map'

ROW	COLUMN+CELL
S200	column=artist:artistid, timestamp=1527317516991, value=A300
S201	column=artist:artistid, timestamp=1527317537800, value=A301
S202	column=artist:artistid, timestamp=1527317557428, value=A302
S203	column=artist:artistid, timestamp=1527317576629, value=A303
S204	column=artist:artistid, timestamp=1527317595865, value=A304
S205	column=artist:artistid, timestamp=1527317616442, value=A301
S206	column=artist:artistid, timestamp=1527317636831, value=A302
S207	column=artist:artistid, timestamp=1527317656371, value=A303
S208	column=artist:artistid, timestamp=1527317677738, value=A304
S209	column=artist:artistid, timestamp=1527317698867, value=A305

10 row(s) in 0.6090 seconds

hbase(main):009:0> █

acadgild@localhost:~

hbase(main):012:0* scan 'subscribed-users'

ROW	COLUMN+CELL
U100	column=subscn:enddt, timestamp=1527317735605, value=1465130523
U100	column=subscn:startdt, timestamp=1527317717956, value=1465230523
U101	column=subscn:enddt, timestamp=1527317772597, value=1475130523
U101	column=subscn:startdt, timestamp=1527317753524, value=1465230523
U102	column=subscn:enddt, timestamp=1527317809861, value=1475130523
U102	column=subscn:startdt, timestamp=1527317790425, value=1465230523
U103	column=subscn:enddt, timestamp=1527317853219, value=1475130523
U103	column=subscn:startdt, timestamp=1527317834148, value=1465230523
U104	column=subscn:enddt, timestamp=1527317896888, value=1475130523
U104	column=subscn:startdt, timestamp=1527317873897, value=1465230523
U105	column=subscn:enddt, timestamp=1527317945888, value=1475130523
U105	column=subscn:startdt, timestamp=1527317922227, value=1465230523
U106	column=subscn:enddt, timestamp=1527317989852, value=1485130523
U106	column=subscn:startdt, timestamp=1527317967214, value=1465230523
U107	column=subscn:enddt, timestamp=1527318032972, value=1455130523
U107	column=subscn:startdt, timestamp=1527318011441, value=1465230523
U108	column=subscn:enddt, timestamp=1527318076352, value=1465230623
U108	column=subscn:startdt, timestamp=1527318054061, value=1465230523
U109	column=subscn:enddt, timestamp=1527318120227, value=1475130523
U109	column=subscn:startdt, timestamp=1527318100534, value=1465230523
U110	column=subscn:enddt, timestamp=1527318162036, value=1475130523
U110	column=subscn:startdt, timestamp=1527318140950, value=1465230523
U111	column=subscn:enddt, timestamp=1527318210478, value=1475130523
U111	column=subscn:startdt, timestamp=1527318186178, value=1465230523
U112	column=subscn:enddt, timestamp=1527318256247, value=1475130523
U112	column=subscn:startdt, timestamp=1527318234738, value=1465230523
U113	column=subscn:enddt, timestamp=1527318296103, value=1485130523
U113	column=subscn:startdt, timestamp=1527318276730, value=1465230523
U114	column=subscn:enddt, timestamp=1527318337339, value=1468130523
U114	column=subscn:startdt, timestamp=1527318319097, value=1465230523

15 row(s) in 0.2630 seconds

hbase(main):013:0> █

Output of the above (Hive)

```
acadgild@localhost:~  
[acadgild@localhost ~]$ hive  
SLF4J: Class path contains multiple SLF4J bindings.  
SLF4J: Found binding in [jar:file:/home/acadgild/install/hive/apache-hive-2.3.2-bin/lib/log4j-slf4j-impl-2.6.2.jar!/org/slf4j/impl/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/impl/StaticLoggerBinder.class]  
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.  
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFactory]  
  
Logging initialized using configuration in jar:file:/home/acadgild/install/hive/apache-hive-2.3.2-bin/lib/hive-common-2.3.2.jar!/hive-log4j2.properties Async: true  
Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execution engine (i.e. spark, tez) or using Hive 1.X releases.  
hive>
```

```
acadgild@localhost:~  
hive> use project;  
OK  
Time taken: 17.789 seconds  
hive> show tables;  
OK  
users_artists  
Time taken: 0.671 seconds, Fetched: 1 row(s)  
hive> select * from users_artists;  
OK  
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"]  
Time taken: 7.913 seconds, Fetched: 15 row(s)  
hive>
```

Step 4:

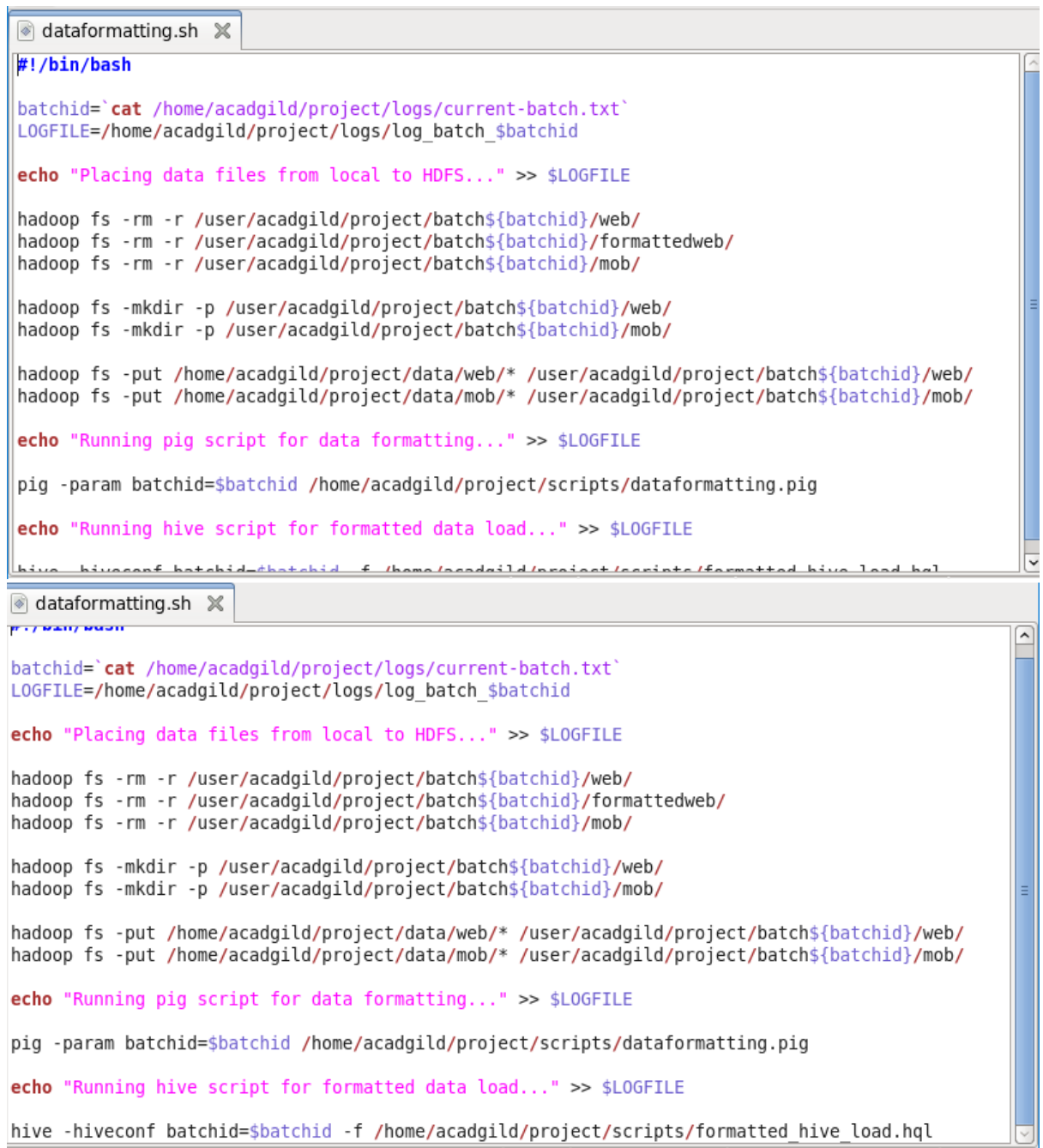
Perform Data Formatting

Below is the shell script dataformatting.sh that is used to:

- Format the web xml data using Pig to a csv format and
- Load the 2 data files, mob and web (formatted by Pig), to a Hive Table for data enrichment

The following operations are performed:

- 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
- Add logs to the Log File signifying that the data is placed in the HDFS and the running of the Pig and Hive scripts for data formatting and loading respectively.
- Delete, if they exist, folders for the mob, web and formattedweb. This is done in-case any old data remains because of execution failure.
- Create the above folders web and mob that were deleted above and move the data from the Local FS to the HDFS. The formattedweb folder is created in the Pig Script.
- Run the pig script dataformatting.pig. This will format the web data (stored in the web folder in the HDFS) in xml format to csv format and store it in the HDFS in the folder formattedweb.
- Run the hive script formatted_hive_load.hql. This will load the data in the mob
- folder and formattedweb folder in the HDFS to a table formatted_input in Hive which will be used for data enrichment later.



```
dataformatting.sh X
#!/bin/bash

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

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

hadoop fs -rm -r /user/acadgild/project/batch${batchid}/web/
hadoop fs -rm -r /user/acadgild/project/batch${batchid}/formattedweb/
hadoop fs -rm -r /user/acadgild/project/batch${batchid}/mob/

hadoop fs -mkdir -p /user/acadgild/project/batch${batchid}/web/
hadoop fs -mkdir -p /user/acadgild/project/batch${batchid}/mob/

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

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

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

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

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

dataformatting.sh X
#!/bin/bash

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

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

hadoop fs -rm -r /user/acadgild/project/batch${batchid}/web/
hadoop fs -rm -r /user/acadgild/project/batch${batchid}/formattedweb/
hadoop fs -rm -r /user/acadgild/project/batch${batchid}/mob/

hadoop fs -mkdir -p /user/acadgild/project/batch${batchid}/web/
hadoop fs -mkdir -p /user/acadgild/project/batch${batchid}/mob/

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

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

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

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

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

dataformatting.pig

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

```
dataformatting.pig X
REGISTER /home/acadgild/project/lib/piggybank.jar;

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

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

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

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

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 X
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});
```

Below is a view of the execution of the above:

```

acadgild@localhost:~/project
[acadgild@localhost project]$
[acadgild@localhost project]$ sh /home/acadgild/project/scripts/dataformatting.s
h
18/05/26 12:54:31 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
rm: /user/acadgild/project/batch1/web/: No such file or directory
18/05/26 12:54:40 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
rm: /user/acadgild/project/batch1/formattedweb/: No such file or directory
18/05/26 12:54:46 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
rm: /user/acadgild/project/batch1/mob/: No such file or directory
18/05/26 12:54:50 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
18/05/26 12:54:55 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
18/05/26 12:55:00 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
18/05/26 12:55:06 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
18/05/26 12:55:16 INFO pig.ExecTypeProvider: Trying ExecType : LOCAL
18/05/26 12:55:16 INFO pig.ExecTypeProvider: Trying ExecType : MAPREDUCE
18/05/26 12:55:16 INFO pig.ExecTypeProvider: Picked MAPREDUCE as the ExecType
2018-05-26 12:55:16,803 [main] INFO org.apache.pig.Main - Apache Pig version 0.16.0 (r1746530) compiled Jun 01 2016, 23:10:49
2018-05-26 12:55:16,803 [main] INFO org.apache.pig.Main - Logging error messages to: /home/acadgild/project/pig_1527319516781.log
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/impl/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]
2018-05-26 12:55:18,144 [main] WARN org.apache.hadoop.util.NativeCodeLoader - Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
2018-05-26 12:55:18,846 [main] INFO org.apache.pig.impl.util.Util - Default bootstrap file /home/acadgild/.pigbootstrap not found
2018-05-26 12:55:19,311 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - mapred.job.tracker is deprecated. Instead, use mapreduce.jobtracker.address
2018-05-26 12:55:19,312 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS
2018-05-26 12:55:19,312 [main] INFO org.apache.pig.backend.hadoop.executionengine.HExecutionEngine - Connecting to hadoop file system at: hdfs://localhost:8020
2018-05-26 12:55:20,785 [main] INFO org.apache.pig.PigServer - Pig Script ID for the session: PIG-dataformatting.pig-38bcee3f-eel8-4619-856f-b3795337ecd1
2018-05-26 12:55:20,785 [main] WARN org.apache.pig.PigServer - ATS is disabled since yarn.timeline-service.enabled set to false
2018-05-26 12:55:21,102 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS
2018-05-26 12:55:22,930 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS
2018-05-26 12:55:23,483 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS
2018-05-26 12:55:23,734 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - mapred.textoutputformat.separator is deprecated. Instead, use mapreduce.outputtextoutputformat.separator
2018-05-26 12:55:23,861 [main] INFO org.apache.pig.tools.pigstats.ScriptState - Pig features used in the script: UNKNOWN
2018-05-26 12:55:24,072 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS
2018-05-26 12:55:24,080 [main] INFO org.apache.pig.data.SchemaTupleBackend - Key [pig.schematuple] was not set... will not generate code.
2018-05-26 12:55:24,234 [main] INFO org.apache.pig.newplan.logical.optimizer.LogicalPlanOptimizer - (RULES ENABLED) - [AdaptForEach, ColumnMapKeyPrune, ConstantCalculator, GroupByConstParallelSetter, LimitOptimizer, LoadTypeCastInserter, MergeFilter, MergeForEach, PartitionFilterOptimizer, PredicatePushdownOptimizer, PushDownForEachFlatMap, PushUpFilter, SplitFilter, StreamTypeCastInserter]
2018-05-26 12:55:24,639 [main] INFO org.apache.pig.impl.util.SpillableMemoryManager - Selected heap (Tenured Gen) of size 699072512 to monitor. collectionUsageThreshold = 489350752, usageThreshold = 489350752

```

Below is a look into the HDFS:

```

acadgild@localhost:~/project
[acadgild@localhost project]$ hadoop fs -ls /user/acadgild/project/batch1/web/
18/05/26 13:00:26 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Found 1 items
-rw-r--r-- 1 acadgild supergroup          6717 2018-05-26 12:55 /user/acadgild/project/batch1/web/file.xml
[acadgild@localhost project]$
You have new mail in /var/spool/mail/acadgild
[acadgild@localhost project]$ hadoop fs -ls /user/acadgild/project/batch1/mob/
18/05/26 13:03:10 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Found 1 items
-rw-r--r-- 1 acadgild supergroup          1237 2018-05-26 12:55 /user/acadgild/project/batch1/mob/file.txt
[acadgild@localhost project]$ hadoop fs -ls /user/acadgild/project/batch1/formattedweb/
18/05/26 13:03:25 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Found 2 items
-rw-r--r-- 1 acadgild supergroup              0 2018-05-26 12:57 /user/acadgild/project/batch1/formattedweb/ SUCCESS
-rw-r--r-- 1 acadgild supergroup          1237 2018-05-26 12:57 /user/acadgild/project/batch1/formattedweb/part-m-00000
[acadgild@localhost project]$

```



```
hive> select * from formatted_input;
OK
U101      S204      A301      1465130523      1465230523      1475130523      U      S
T407      2          1          0          1
U120      S200      A303      1495130523      1465230523      1465130523      A      S
T407      1          0          1          1
U111      S200      A305      1495130523      1465130523      1475130523      U      S
T407      1          1          0          1
U102      S203      A305      1475130523      1475130523      1475130523      AU     S
T413      2          0          1          1
U111      S204      A302      1465130523      1475130523      1465230523      AU     S
T415      0          1          0          1
          S209      A305      1465230523      1465230523      1475130523      AU     S
T410      1          0          0          1
U116      S210      A303      1465230523      1465130523      1465230523      E      S
T409      3          1          0          1
U108      S202      A301      1475130523      1475130523      1475130523      U      S
T411      1          0          0          1
U108      S210      A302      1465230523      1465130523      1465130523      S
T400      0          1          1          1
U106      S206      A301      1495130523      1475130523      1485130523      AU     S
T412      1          1          1          1
U120      S204      A305      1495130523      1465230523      1475130523      E      S
T413      3          0          0          1
U119      S201      A304      1475130523      1475130523      1465230523      U      S
T414      1          1          0          1
U111      S205      A301      1465230523      1465230523      1465230523      A      S
T404      3          0          1          1
U107      S206      A305      1475130523      1475130523      1485130523      AU     S
T406      2          0          1          1
U111      S210      A304      1495130523      1475130523      1485130523      AU     S
T413      3          0          1          1
U115      S200      A300      1495130523      1485130523      1465230523      A      S
T413      0          0          0          1
U111      S209      A304      1465230523      1475130523      1475130523      A      S
T408      0          0          1          1
U120      S203      A304      1465130523      1485130523      1485130523      A      S
T410      1          1          0          1
U112      S201      A305      1465230523      1465230523      1485130523      E      S
T413      0          1          0          1
U111      S201      A300      1495130523      1465130523      1485130523      E      S
T408      0          0          0          1
```

```
acagild@localhost:~$
[acagild@localhost ~]$ hadoop fs -ls /user/hive/warehouse/project.db/formatted_input/batchid=1
18/06/01 20:46:19 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Found 3 items
-rwxr-xr-x 1 acagild supergroup 0 2018-06-01 20:40 /user/hive/warehouse/project.db/formatted_input/batchid=1/ SUCCESS
-rwxr-xr-x 1 acagild supergroup 1237 2018-06-01 20:39 /user/hive/warehouse/project.db/formatted_input/batchid=1/file.txt
-rwxr-xr-x 1 acagild supergroup 1237 2018-06-01 20:40 /user/hive/warehouse/project.db/formatted_input/batchid=1/part-m-00000
[acagild@localhost ~]$
```

Step 5:

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:

- 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
- Add logs to the Log File signifying that the Hive lookup tables are created from the HBase lookup tables.
- 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 X
#!/bin/bash

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

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

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

create_hive_hbase_lookup.hql

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

```
create_hive_hbase_lookup.hql X
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
```

```
create_hive_hbase_lookup.hql
(hbase.columns.mapping="":key,geocode);
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");

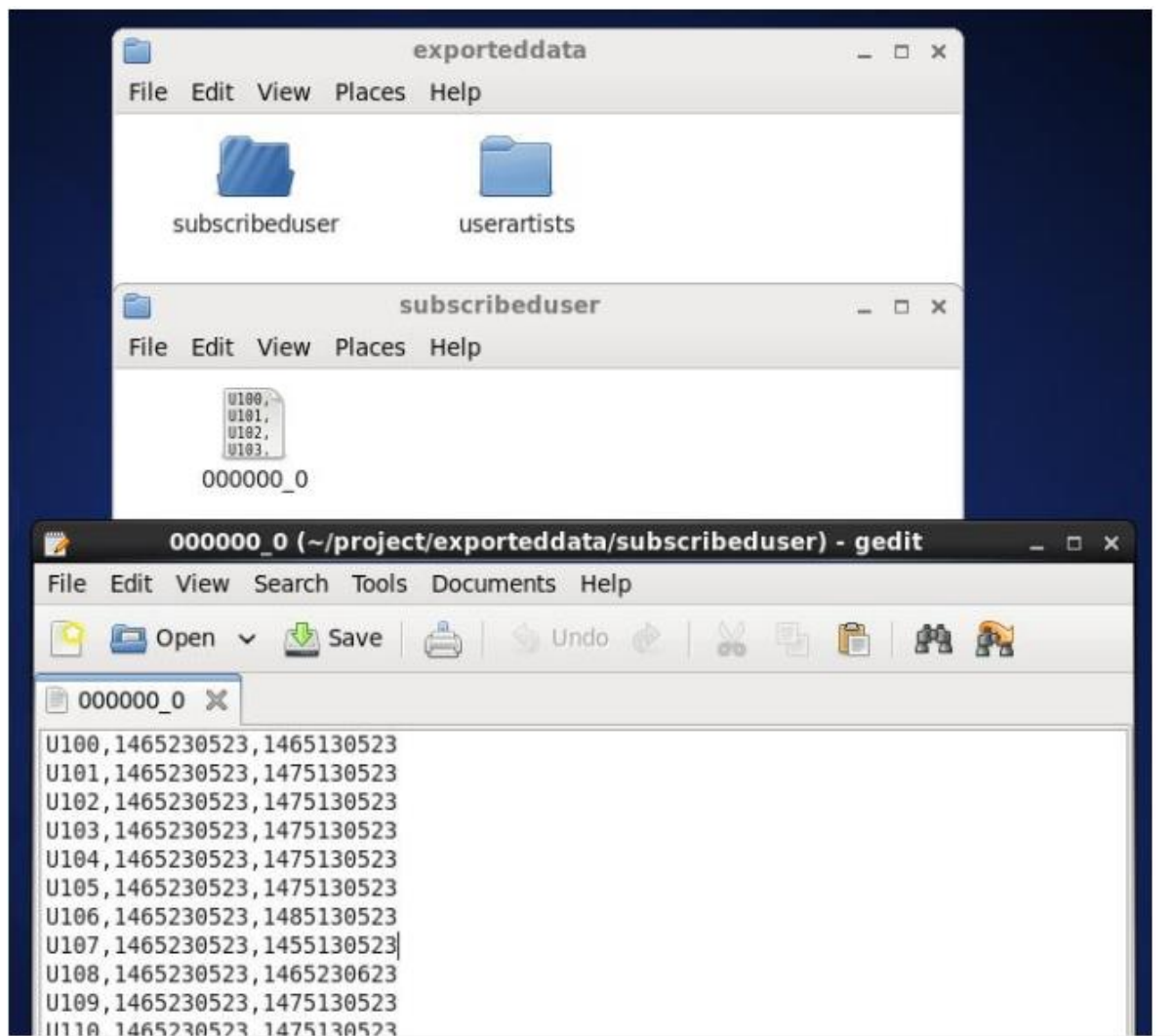
create external table if not exists song_artist_map
(
  song_id STRING,
  artist_id STRING
)
STORED BY 'org.apache.hadoop.hive.hbase.HBaseStorageHandler'
with serdeproperties
("hbase.columns.mapping"="":key,artist:artistid")
tblproperties("hbase.table.name"="song-artist-map");
```

Below is a view of the execution of the above:

```
acadgild@localhost:~
File Edit View Search Terminal Help
[acadgild@localhost ~]$ sh /home/acadgild/project/scripts/data_enrichment_filtering_schema.sh

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.class]
SLF4J: Found binding in [jar:file:/usr/local/hadoop-2.6.0/share/hadoop/common/lib/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
OK
Time taken: 1.466 seconds
OK
Time taken: 2.198 seconds
OK
Time taken: 0.297 seconds
Query ID = acadgild_20171006013838_b459e5fd-ce6e-44a4-b846-3a962417819a
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_1507230696879_0003, Tracking URL = http://localhost:8088/proxy/application_1507230696879_0003/
Kill Command = /home/acadgild/hadoop-2.6.0/bin/hadoop job -kill job_1507230696879_0003
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 0
2017-10-06 01:38:18,978 Stage-1 map = 0%, reduce = 0%
2017-10-06 01:38:29,735 Stage-1 map = 100%, reduce = 0%, Cumulative CPU 2.35 sec
MapReduce Total cumulative CPU time: 2 seconds 350 msec
Ended Job = job_1507230696879_0003
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: 2.35 sec HDFS Read: 276 HDFS Write: 405 SUCCESS
Total MapReduce CPU Time Spent: 2 seconds 350 msec
OK
```

Output of the saved table subscribed_users in the Local FS



Output in Hive:

The tables were created and populated as intended.



```

hive> SHOW TABLES;
OK
formatted_input
song_artist_map
station_geo_map
subscribed_users
users_artists
Time taken: 0.037 seconds, Fetched: 5 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
S209      A305
Time taken: 0.292 seconds, Fetched: 10 row(s)

```

```

hive> select * from station_geo_map;
OK
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
Time taken: 0.194 seconds, Fetched: 15 row(s)

```

```

hive> select * from subscribed_users;
OK
U100      1465230523      1465130523
U101      1465230523      1475130523
U102      1465230523      1475130523
U103      1465230523      1475130523
U104      1465230523      1475130523
U105      1465230523      1475130523
U106      1465230523      1485130523
U107      1465230523      1455130523
U108      1465230523      1465230623
U109      1465230523      1475130523
U110      1465230523      1475130523
U111      1465230523      1475130523
U112      1465230523      1475130523
U113      1465230523      1485130523
U114      1465230523      1468130523

```

2) data_enrichment.sh

Below is the shell script data_enrichment.sh where the following operations are performed:

- 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.
- Add logs to the Log File signifying that the data enrichment has begun.
- 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 X
#!/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

hdfs dfs -get /user/hive/warehouse/project.db/enriched_data/batchid=$batchid/status=pass/* $VALIDDIR
hdfs dfs -get /user/hive/warehouse/project.db/enriched_data/batchid=$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 {} \;
```

data_enrichment.hql

Rules for data enrichment

1. If any of like or dislike is **NULL** or *absent*, consider it as 0.
2. If fields like Geo_cd and Artist_id are **NULL** or *absent*, consult the lookup tables for fields Station_id and Song_id respectively to get the values of Geo_cd and Artist_id.
3. If corresponding lookup entry is not found, consider that record to be invalid.

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

For the data enrichment, a table **enriched_data** is created and the table is overwritten with the result of the below operations:

- 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
- 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 X
CREATE TABLE IF NOT EXISTS enriched_data
(
  User_id STRING,
  Song_id STRING,
  Artist_id STRING,
  Timestamp1 STRING,
  Start_ts STRING,
  End_ts STRING,
  Geo_cd STRING,
  Station_id STRING,
  Song_end_type INT,
  Like1 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,
i.artist_id,
i.timestamp1,
i.start_ts,
i.end_ts,
i.geo_cd,
i.station_id,
i.song_end_type,
i.like1,
i.dislike,
i.batchid,
IF((i.dislike IS NULL, 0, i.dislike) AS dislike,
i.batchid,
IF((i.like1=1 AND i.dislike=1)
OR i.user_id IS NULL
OR i.song_id IS NULL
OR i.timestamp1 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.timestamp1=''
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};

data_enrichment.hql X
IF (i.dislike IS NULL, 0, i.dislike) AS dislike,
i.batchid,
IF((i.like1=1 AND i.dislike=1)
OR i.user_id IS NULL
OR i.song_id IS NULL
OR i.timestamp1 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.timestamp1=''
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};
```


Output in Hive :

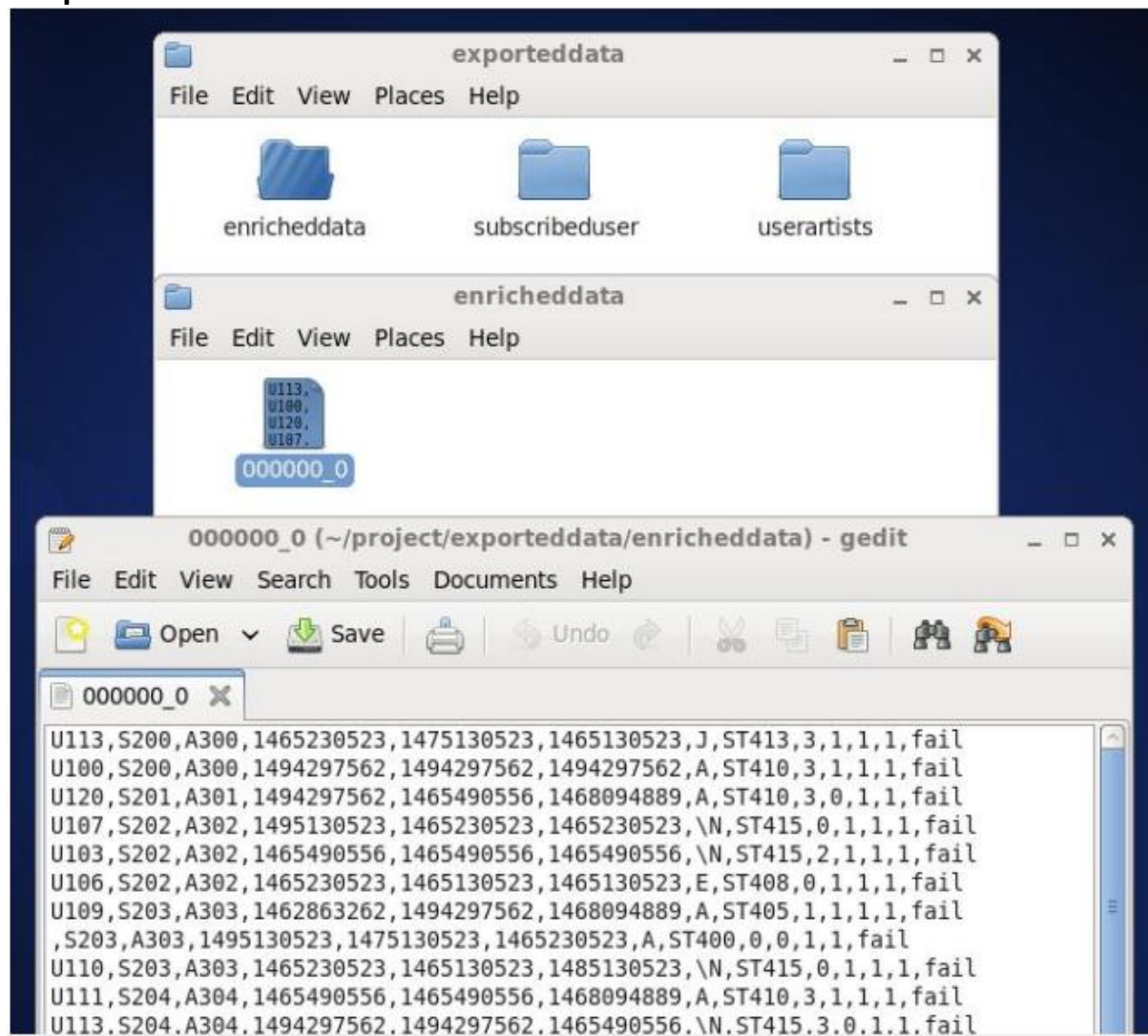
```
acadgild@localhost:~  
File Edit View Search Terminal Help  
hive> SHOW TABLES;  
OK  
enriched_data  
formatted_input  
song_artist_map  
station_geo_map  
subscribed_users  
users_artists  
Time taken: 0.035 seconds, Fetched: 6 row(s)  
hive> SELECT * FROM enriched_data;  
OK  
U113 S200 A300 1465230523 1475130523 1465130523 J ST413 3 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 NULL ST415 0 1 1 1 fail  
U103 S202 A302 1465490556 1465490556 1465490556 NULL 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 NULL 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 NULL ST415 3 0 1 1 fail  
U100 S204 A304 1495130523 1475130523 1465130523 E ST408 2 1 1 1 fail  
U106 S205 A301 1462863262 1462863262 1494297562 AP ST407 2 1 1 1 fail  
U108 S205 A301 1465130523 1465130523 1475130523 A ST410 2 1 0 1 fail  
U111 S206 A302 1465130523 1465130523 1485130523 NULL ST415 0 1 1 1 fail  
U114 S207 A303 1465130523 1465230523 1475130523 NULL ST415 3 1 0 1 fail  
U102 S207 A303 1465230523 1485130523 1465230523 J ST403 3 1 1 1 fail  
S208 A304 1465490556 1494297562 1465490556 A ST411 1 0 1 1 fail  
U118 S208 A304 1475130523 1465130523 1465230523 NULL ST415 3 0 0 1 fail
```

Output in Hive Warehouse:

Below is a view of the enriched_data in the Hive warehouse, partitioned by status and batched

```
[acadgild@localhost ~]$ hdfs dfs -ls /user/hive/warehouse/project.db/enriched_data/batchid=1  
17/10/06 01:55:06 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java cl  
asses where applicable  
Found 2 items  
drwxr-xr-x - acadgild supergroup 0 2017-10-06 01:51 /user/hive/warehouse/project.db/enriched_data/batchid=1/status  
=fail  
drwxr-xr-x - acadgild supergroup 0 2017-10-06 01:51 /user/hive/warehouse/project.db/enriched_data/batchid=1/status  
=pass  
[acadgild@localhost ~]$
```

Output saved in the Local FS:



Step 6:

Perform Data Analysis

Below is the shell script data_analysis.sh where the following operations are performed:

- 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
- 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 X
#!/bin/bash

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

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

echo "Running Spark Script for Data Analysis..." >> $LOGFILE
echo "Exporting analyzed data to Local FS..." >> $LOGFILE
cat /home/acadgild/project/scripts/data_analysis.scala | spark-shell
echo "All Activities Complete..." >> $LOGFILE
echo "Incrementing batchid..." >> $LOGFILE
batchid=`expr $batchid + 1`
echo -n $batchid > /home/acadgild/project/logs/current-batch.txt
```

Initialization:

- Import Row, DataFrame, Structure type and function dependencies needed to perform analysis.
- Get the batchid from the batch file and store it in the variable batchid
- Get the data that was exported and saved in the Local FS from the steps above i.e. enriched_data, subscribed_user and user_artists and perform the foll. on each of them
- Create the schema for the data
- Create a DataFrame from the schema and data
- Create a temporary table from the DataFrame created

```

data_analysis.scala
import org.apache.spark.sql.Row
import org.apache.spark.sql.DataFrame
import org.apache.spark.sql.types.{StructType, StructField, StringType, NumericType, IntegerType, ArrayType}

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

val batid = sc.textFile("/home/acadgild/project/logs/current-batch.txt").map(x => x.toInt).toDF().first.getInt(0)

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

val MDSchemaString =
"user_id:string,song_id:string,artist_id:string,timestamp:string,start_ts:string,end_ts:string,geo_cd:string,station_id:string"

val MDdataSchema = StructType(MDSchemaString.split(",").map(fieldInfo => StructField(fieldInfo.split(":")(0), if
(fieldInfo.split(":")(1).equals("string")) StringType else IntegerType, true)))

val MDrowRDD = data.map(_._split(",")).map(r => Row(r(0), r(1), r(2), r(3), r(4), r(5), r(6), r(7), r(8).toInt, r(9).toInt, r
(10).toInt, r(11).toInt, r(12)))

val MusicDataDF = spark.createDataFrame(MDrowRDD, MDdataSchema)

MusicDataDF.registerTempTable("Music_Data")

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

val SUSchemaString = "user_id:string,start_dt:string,end_dt:string"

val SUdataSchema = StructType(SUSchemaString.split(",").map(fieldInfo => StructField(fieldInfo.split(":")(0), if
(fieldInfo.split(":")(1).equals("string")) StringType else IntegerType, true)))

val SUrowRDD = data.map(_._split(",")).map(r => Row(r(0), r(1), r(2)))

val SubscribedUsersDF = spark.createDataFrame(SUrowRDD, SUdataSchema)

SubscribedUsersDF.registerTempTable("Music_SubscribedUsers")

//User Artists
val data = sc.textFile("/home/acadgild/project/exporteddata/userartists/000000_0")

val UASchemaString = "user_id:string,artists:string"

val UAdatSchema = StructType(UASchemaString.split(",").map(fieldInfo => StructField(fieldInfo.split(":")(0), if
(fieldInfo.split(":")(1).equals("string")) StringType else IntegerType, true)))

val UArowRDD = data.map(_._split(",")).map(r => Row(r(0), r(1)))

val UserArtistsDF = spark.createDataFrame(UArowRDD, UAdatSchema)

UserArtistsDF.registerTempTable("Music_UserArtists")

```

Problem Statement 1:

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

Code:

```

val Top10Stations = spark.sql(SELECT station_id,COUNT(DISTINCT
song_id) AS total_distinct_songs_played, COUNT(DISTINCT user_id) AS
distinct_user_count, batchid FROM Music_Data WHERE status='pass'
AND batchid=$batid AND like=1 GROUP BY station_id,batchid ORDER BY
total_distinct_songs_played DESC LIMIT 10");

```

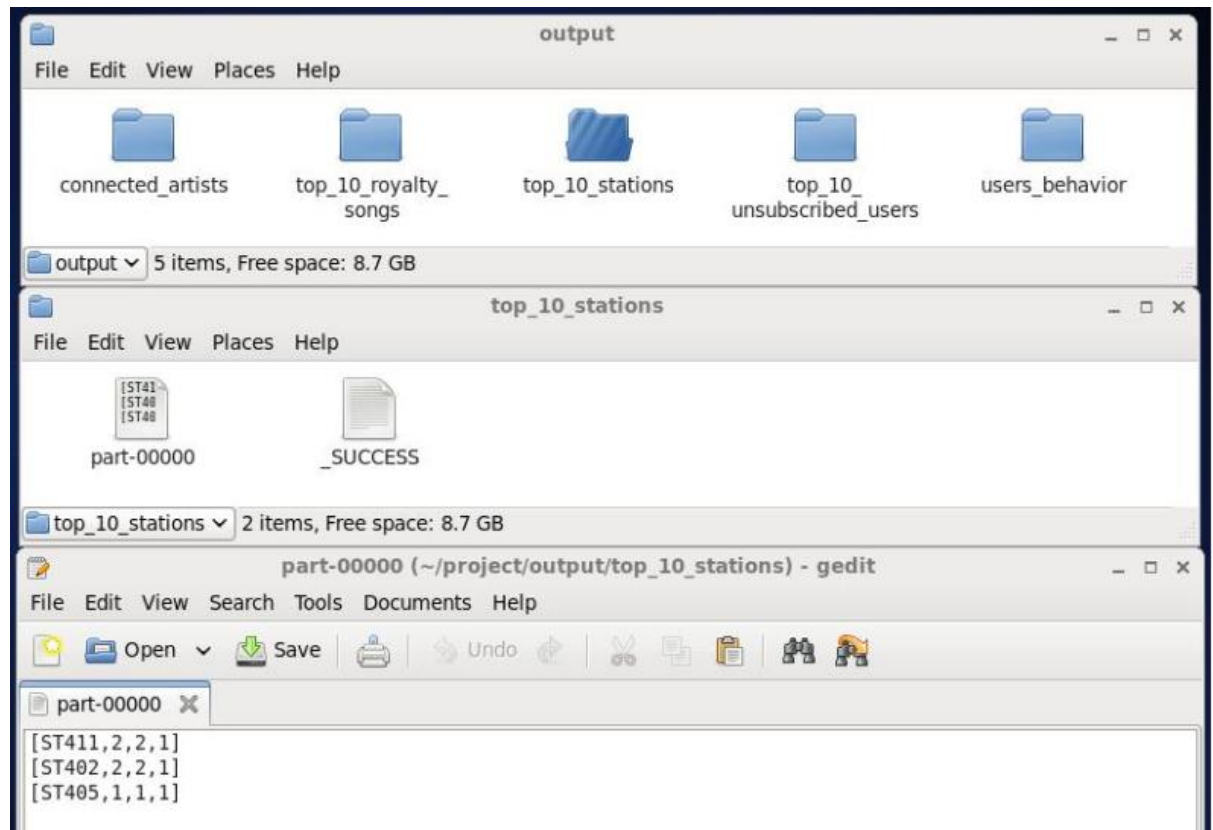
```

Top10Stations.rdd.saveAsTextFile("/home/acadgild/project/output/top_
10_stations")

```

```
val Top10Stations = spark.sql(s"SELECT station_id, COUNT(DISTINCT song_id) AS total_distinct_songs_played, COUNT(DISTINCT user_id) AS distinct_user_count, batchid FROM Music_Data WHERE status='pass' AND batchid=$batid AND like=1 GROUP BY station_id, batchid ORDER BY total_distinct_songs_played DESC LIMIT 10");
Top10Stations.rdd.saveAsTextFile("/home/acadgild/project/output/top_10_stations")
```

Output



Problem Statement 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.

Code:

```
val users_behavior = spark.sql(s"SELECT CASE WHEN (subusers.user_id IS NULL OR CAST(music.timestamp AS DECIMAL(20,0)) > CAST(subusers.end_dt AS DECIMAL(20,0))) THEN 'UNSUBSCRIBED' WHEN (subusers.user_id IS NOT NULL AND CAST(music.timestamp AS DECIMAL(20,0)) <= CAST(subusers.end_dt AS DECIMAL(20,0))) THEN 'SUBSCRIBED' END AS user_type, SUM(ABS(CAST(music.end_ts AS DECIMAL(20,0))-CAST(music.start_ts AS DECIMAL(20,0)))) AS duration,
```

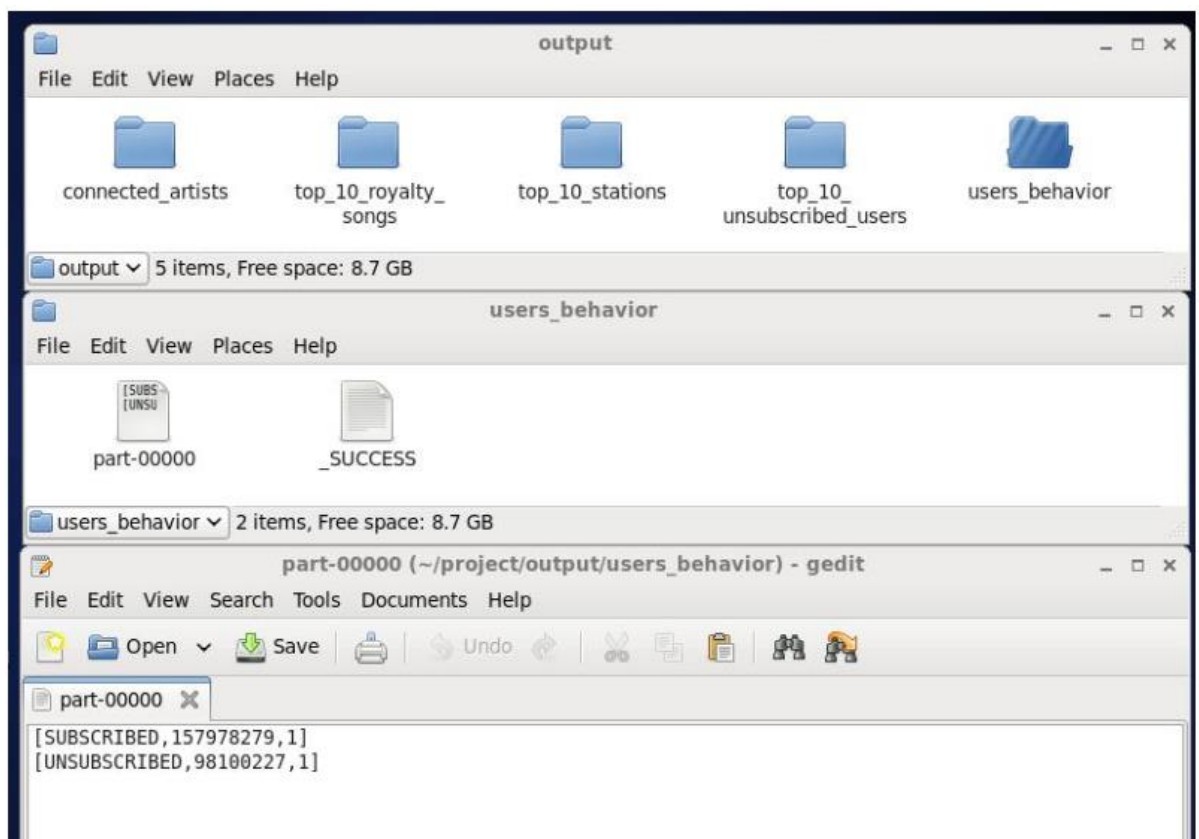
```
batchid FROM Music_Data music LEFT OUTER JOIN
Music_SubscribedUsers subusers ON music.user_id=subusers.user_id
WHERE music.status='pass' AND music.batchid=$batid GROUP BY CASE
WHEN (subusers.user_id IS NULL OR CAST(music.timestamp AS
DECIMAL(20,0)) > CAST(subusers.end_dt AS DECIMAL(20,0))) THEN
'UNSUBSCRIBED' WHEN (subusers.user_id IS NOT NULL AND
CAST(music.timestamp AS DECIMAL(20,0)) <= CAST(subusers.end_dt AS
DECIMAL(20,0))) THEN 'SUBSCRIBED' END,batchid")
```

```
users_behavior.rdd.saveAsTextFile("/home/acadgild/project/output/use
r_behavior")
```

```
val users_behavior = spark.sql(s"SELECT CASE WHEN (subusers.user_id IS NULL OR CAST(music.timestamp AS DECIMAL(20,0)) > CAST
(subusers.end_dt AS DECIMAL(20,0))) THEN 'UNSUBSCRIBED' WHEN (subusers.user_id IS NOT NULL AND CAST(music.timestamp AS
DECIMAL(20,0)) <= CAST(subusers.end_dt AS DECIMAL(20,0))) THEN 'SUBSCRIBED' END AS user_type, SUM(ABS(CAST(music.end_ts AS
DECIMAL(20,0))-CAST(music.start_ts AS DECIMAL(20,0)))) AS duration, batchid FROM Music_Data music LEFT OUTER JOIN
Music_SubscribedUsers subusers ON music.user_id=subusers.user_id WHERE music.status='pass' AND music.batchid=$batid GROUP
BY CASE WHEN (subusers.user_id IS NULL OR CAST(music.timestamp AS DECIMAL(20,0)) > CAST(subusers.end_dt AS DECIMAL(20,0)))
THEN 'UNSUBSCRIBED' WHEN (subusers.user_id IS NOT NULL AND CAST(music.timestamp AS DECIMAL(20,0)) <= CAST(subusers.end_dt
AS DECIMAL(20,0))) THEN 'SUBSCRIBED' END,batchid")

users_behavior.rdd.saveAsTextFile("/home/acadgild/project/output/users_behavior")
```

Output :



Problem Statement 3:

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

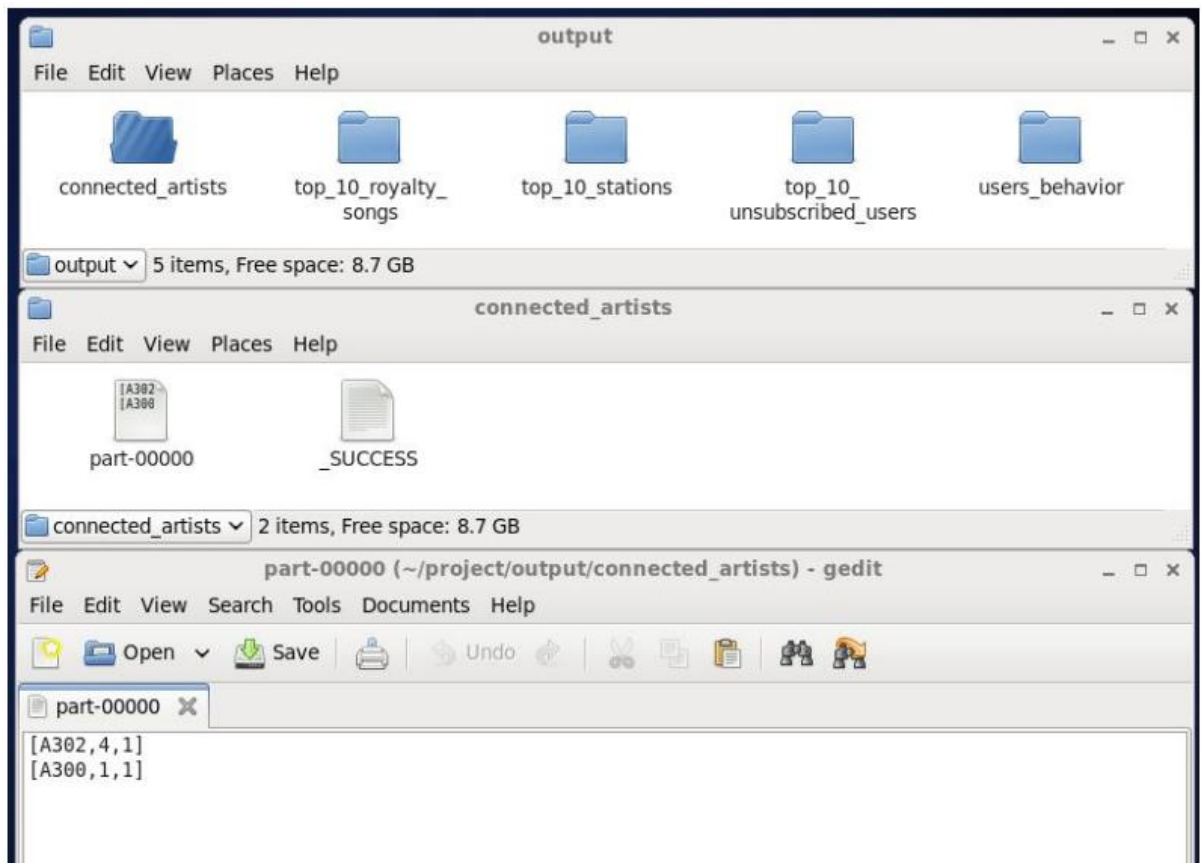
Code:

```
val connected_artists = spark.sql(s"SELECT ua.artists, COUNT(DISTINCT  
ua.user_id) AS user_count, md.batchid FROM Music_UserArtists ua  
INNER JOIN (SELECT artist_id, song_id, user_id, batchid FROM  
Music_Data WHERE status='pass' AND batchid=$batid ) md ON  
ua.artists=md.artist_id AND ua.user_id=md.user_id GROUP BY  
ua.artists, batchid ORDER BY user_count DESC LIMIT 10")
```

```
connected_artists.rdd.saveAsTextFile("/home/acadgild/project/output/c  
onected_artists")
```

```
val connected_artists = spark.sql(s"SELECT ua.artists, COUNT(DISTINCT ua.user_id) AS user_count, md.batchid FROM  
Music_UserArtists ua INNER JOIN ( SELECT artist_id, song_id, user_id, batchid FROM Music_Data WHERE status='pass' AND  
batchid=$batid ) md ON ua.artists=md.artist_id AND ua.user_id=md.user_id GROUP BY ua.artists, batchid ORDER BY user_count  
DESC LIMIT 10")  
  
connected_artists.rdd.saveAsTextFile("/home/acadgild/project/output/connected_artists")
```

Output:



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.

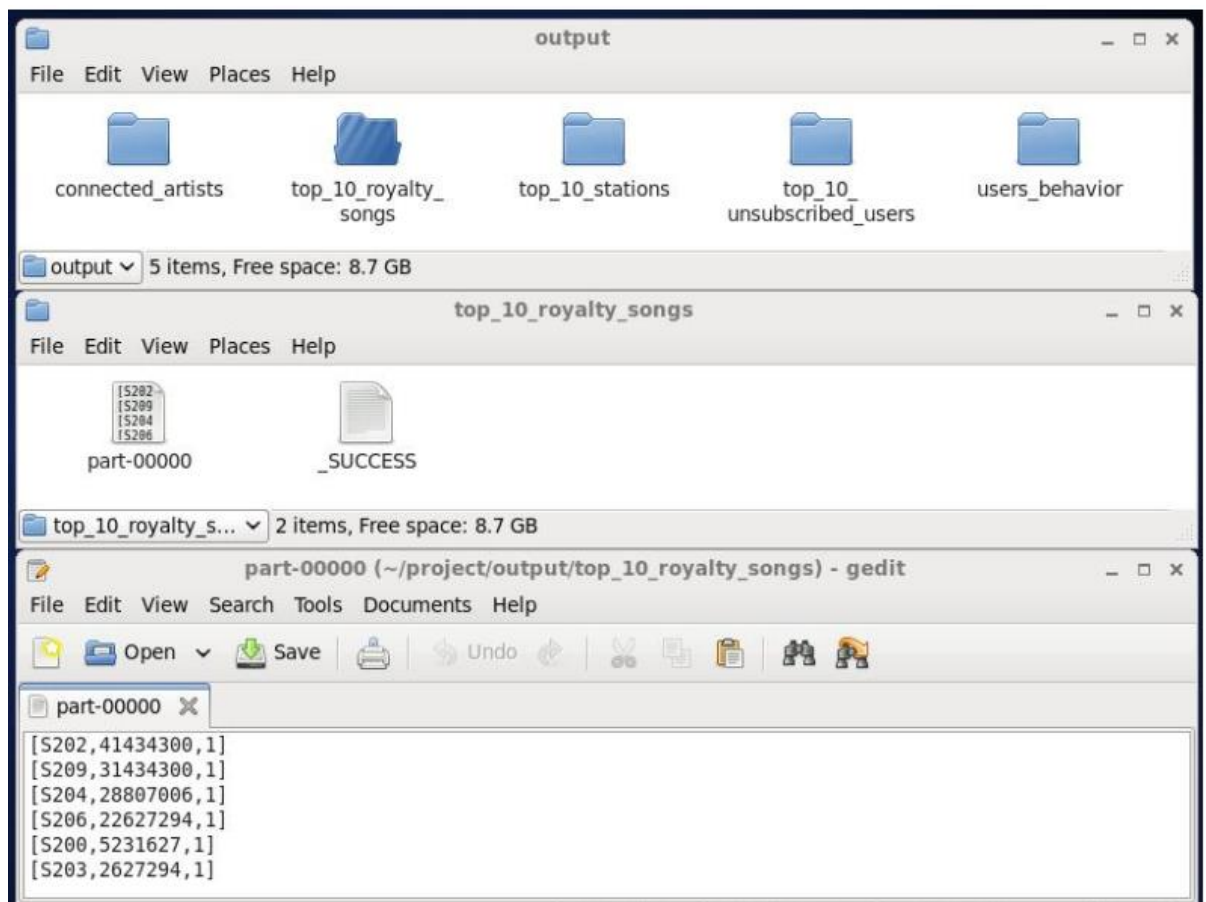
Code:

```
val top_10_royalty_songs = spark.sql(s"SELECT song_id,  
SUM(ABS(CAST(end_ts AS DECIMAL(20,0))-CAST(start_ts AS  
DECIMAL(20,0))) AS Duration, batchid FROM Music_Data WHERE  
status='pass' AND batchid=$batid AND (like=1 OR song_end_type=0)  
GROUP BY song_id,batchid ORDER BY duration DESC LIMIT 10")
```

```
top_10_royalty_songs.rdd.saveAsTextFile("/home/acadgild/project/out  
put/top_10_royalty_songs")
```

```
val top_10_royalty_songs = spark.sql(s"SELECT song_id, SUM(ABS(CAST(end_ts AS DECIMAL(20,0))-CAST(start_ts AS DECIMAL  
(20,0))) AS duration, batchid FROM Music_Data WHERE status='pass' AND batchid=$batid AND (like=1 OR song_end_type=0) GROUP  
BY song_id,batchid ORDER BY duration DESC LIMIT 10")  
top_10_royalty_songs.rdd.saveAsTextFile("/home/acadgild/project/output/top_10_royalty_songs")
```


Output:



Problem Statement 5:

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

Code:

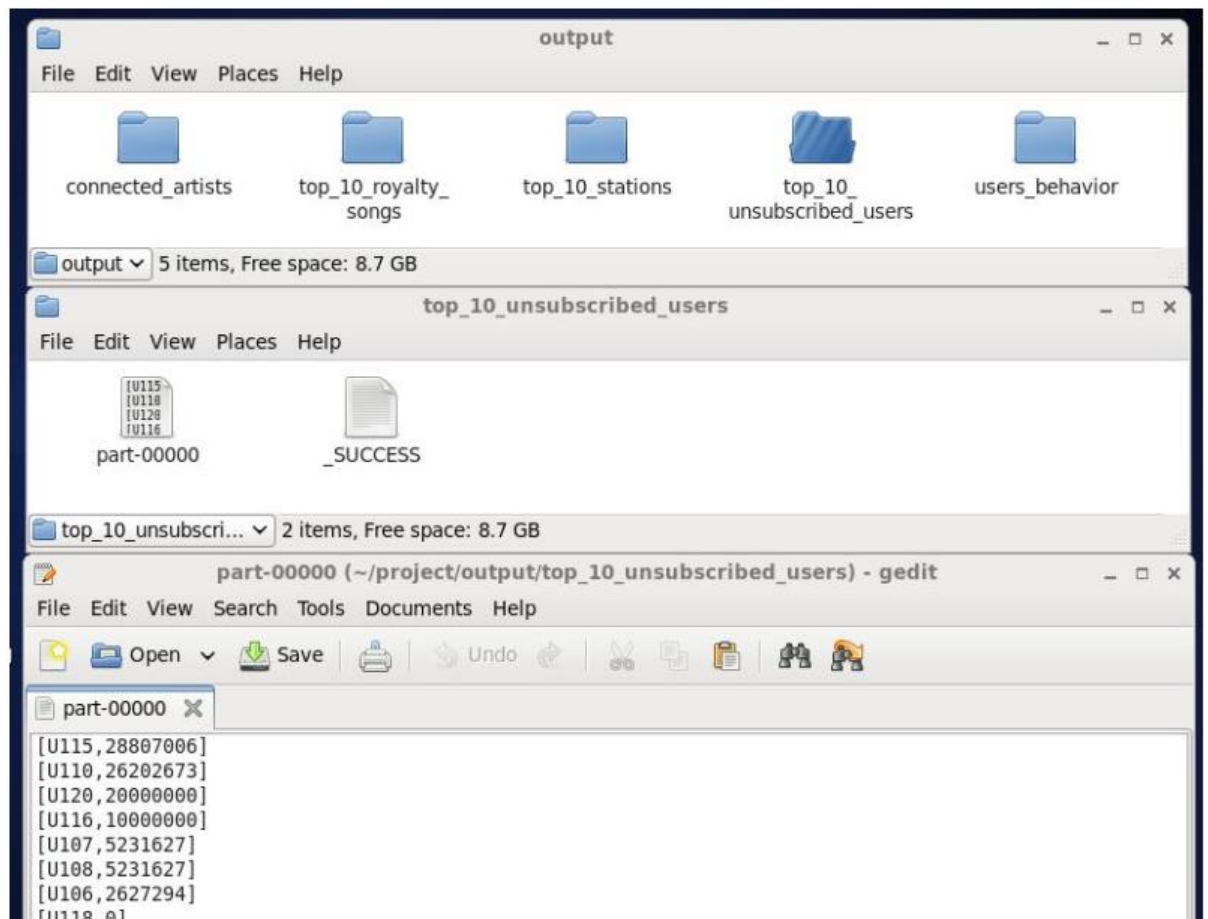
```
val top_10_unsubscribed_users = spark.sql(s"SELECT md.user_id,
SUM(ABS(CAST(ms.end_ts AS DECIMAL(20,0))-CAST(md.start_ts AS
DECIMAL(20,0)))) AS duration FROM Music_Data md LEFT OUTER JOIN
Music_SubscribedUsers su ON md.user_id=su.user_id WHERE
md.status='pass' AND md.batchid=$batid AND (su.user_id IS NULL OR
(CAST(md.timestamp AS DECIMAL(20,0)) > CAST(su.end_dt AS
DECIMAL(20,0)))) GROUP BY md.user_id ORDER BY duration DESC LIMIT
10")
```

```
top_10_unsubscribed_users.rdd.saveAsTextFile("/home/acadgild/projec
t/output/top_10_royalty_songs")
```

```
val top_10_unsubscribed_users = spark.sql(s"SELECT md.user_id, SUM(ABS(CAST(md.end_ts AS DECIMAL(20,0))-CAST(md.start_ts AS DECIMAL(20,0)))) AS duration FROM Music Data md LEFT OUTER JOIN Music SubscribedUsers su ON md.user_id=su.user_id WHERE md.status='pass' AND md.batchid=$batid AND (su.user_id IS NULL OR (CAST(md.timestamp AS DECIMAL(20,0)) > CAST(su.end_dt AS DECIMAL(20,0)))) GROUP BY md.user_id ORDER BY duration DESC LIMIT 10")

top_10_unsubscribed_users.rdd.saveAsTextFile("/home/acadgild/project/output/top_10_unsubscribed_users")
```

Output:



Below is a view of the execution of the above:

```
File Edit View Search Terminal Help
[acadgild@localhost ~]$ sh /home/acadgild/project/scripts/data_analysis.sh
Using Spark's default log4j profile: org/apache/spark/log4j-defaults.properties
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
17/10/06 02:04:57 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes
where applicable
17/10/06 02:04:57 WARN SparkConf:
SPARK_WORKER_INSTANCES was detected (set to '2').
This is deprecated in Spark 1.0+.

Please instead use:
- ./spark-submit with --num-executors to specify the number of executors
- Or set SPARK_EXECUTOR_INSTANCES
- spark.executor.instances to configure the number of instances in the spark config.

17/10/06 02:04:58 WARN Utils: Your hostname, localhost.localdomain resolves to a loopback address: 127.0.0.1; using 10.0.2.15
instead (on interface eth5)
17/10/06 02:04:58 WARN Utils: Set SPARK_LOCAL_IP if you need to bind to another address
17/10/06 02:05:11 WARN ObjectStore: Failed to get database global_temp, returning NoSuchObjectException
Spark context Web UI available at http://10.0.2.15:4040
Spark context available as 'sc' (master = local[*], app id = local-1507235699720).
Spark session available as 'spark'.
Welcome to

      / \
     / \
    /   \
   /     \
  /       \
 /         \
/           \
 \           /
  \         /
   \       /
    \     /
     \   /
      \ /

version 2.1.0

Using Scala version 2.11.8 (Java HotSpot(TM) 64-Bit Server VM, Java 1.8.0_65)
Type in expressions to have them evaluated.
Type :help for more information.

scala> :quit
[acadgild@localhost ~]$
```

Post Analysis:

A view of the log file post analysis.

```
log_batch_1 x
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 filtering...
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..|
```

The batchid is incremented from 1 to 2:

```
current-batch.txt x
p
```

A view of the log folder:

