FINAL PROJECT

MUSIC DATA ANALYSIS

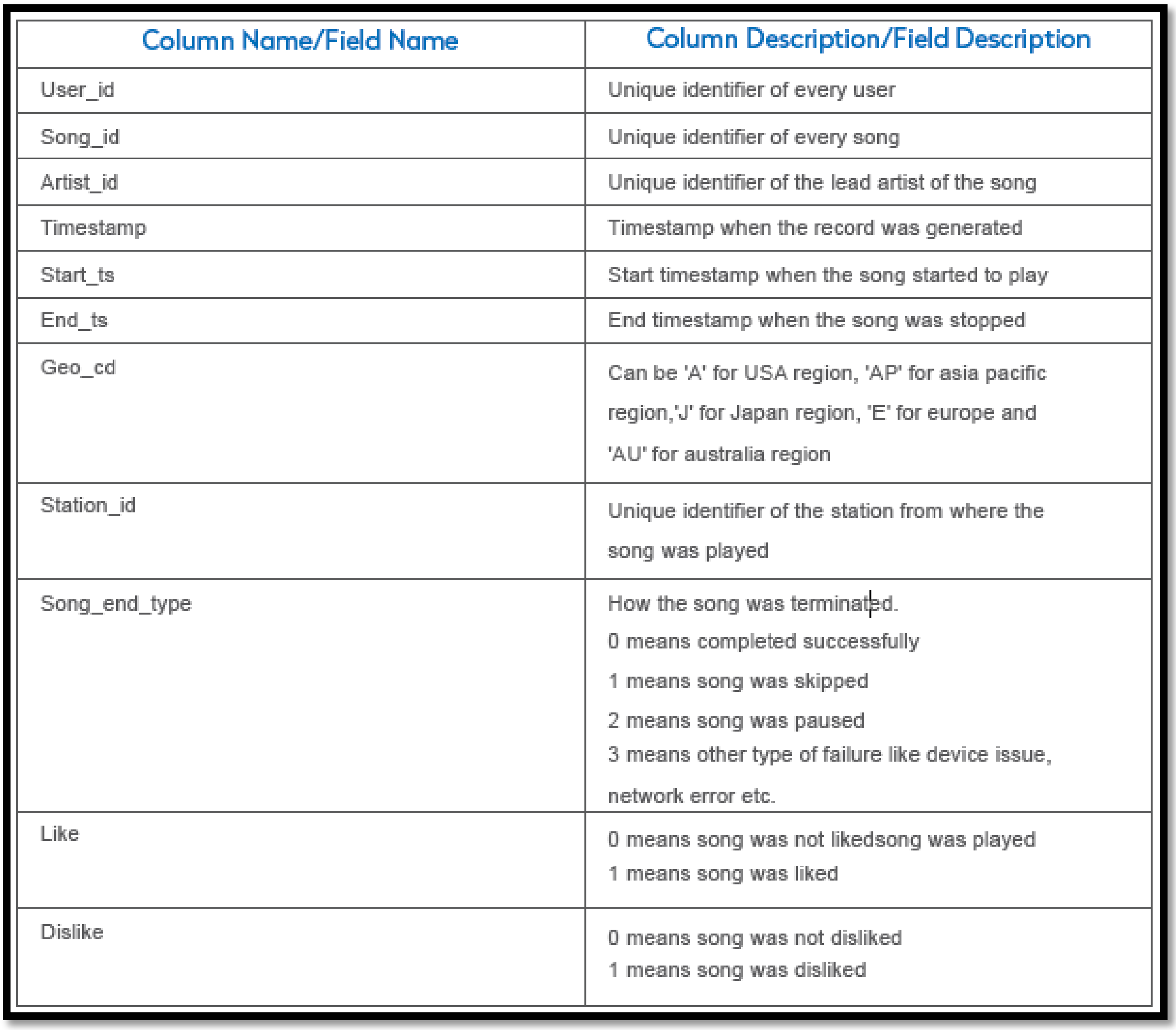
Table of Contents

SECTION – 1

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.

1.1 Fields present in the data files

Data files contain below fields.



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

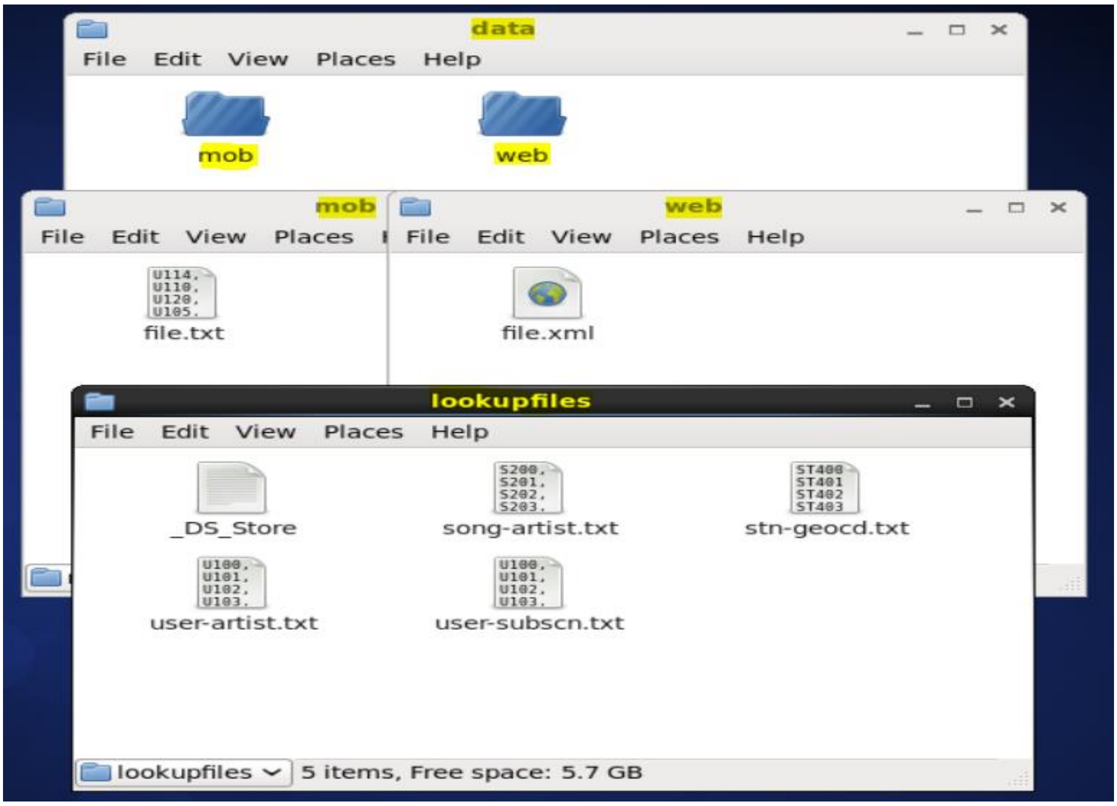
1.2 LookUp Tables

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

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

1.3 Dataset

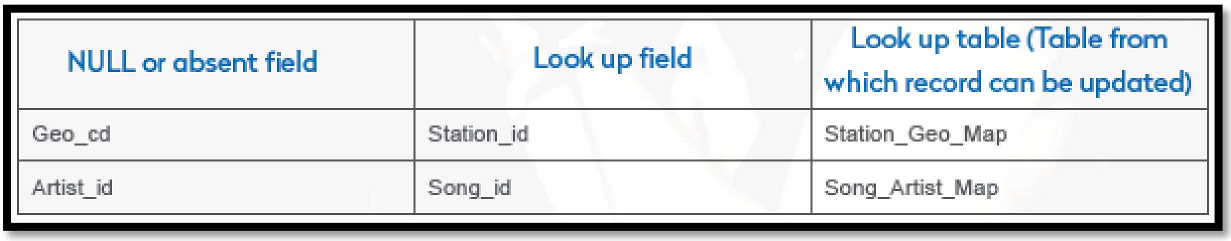
1. Data coming from web applications reside in /data/web and has xml format.
2. Data coming from mobile applications reside in /data/mob and has csv format.
3. Data present in lookup directory should be used in HBase.



1.4 Data Enrichment

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

1.5 Data Analysis

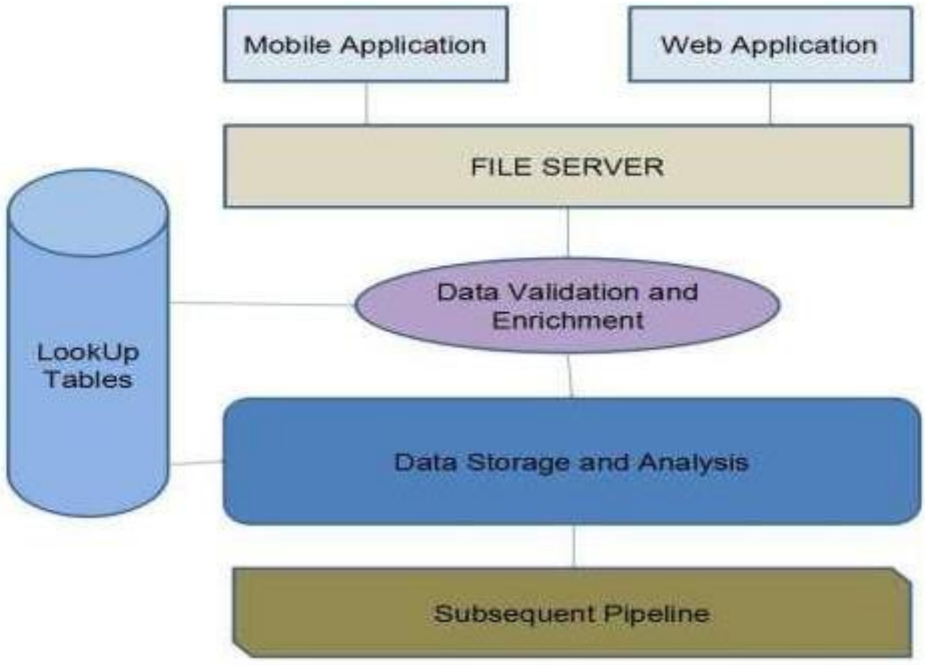
It is not only the data which is important, rather it is the insight it can be used to generate important. Once we have made the data ready for analysis, we have to perform below analysis on a daily basis.

1. Determine top 10 **station_id**(s) where maximum number of songs were played, which were liked by unique users.
2. Determine total duration of songs played by each type of user, where type of user can be **'subscribed'** or **'unsubscribed'**. An unsubscribed user is the one whose record is either not present in **Subscribed_users** lookup table or has **subscription_end_date** earlier than the timestamp of the song played by him.
3. Determine top 10 connected artists. Connected artists are those whose songs are most listened by the unique users who follow them.
4. Determine top 10 songs who have generated the maximum revenue. Royalty applies to a song only if it was liked or was completed successfully or both.
5. Determine top 10 unsubscribed users who listened to the songs for the longest duration.

1.6 Challenges and Optimizations

1. LookUp tables are in NoSQL databases. Integrate them with the actual data flow.
2. Try to make joins as less expensive as possible.
3. Data Cleaning, Validation, Enrichment, Analysis and Post Analysis have to be automated. Try using schedulers.
4. Appropriate logs have to maintain to track the behaviour and overcome failures in the pipeline.

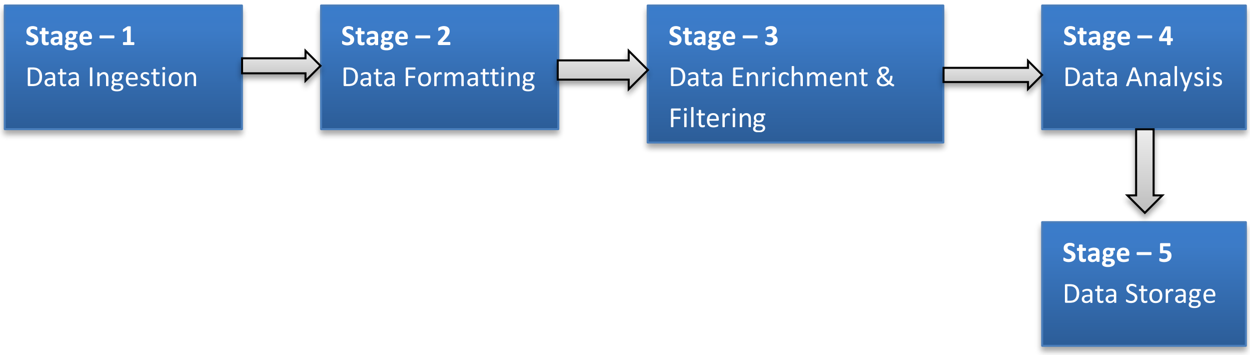
1.7 Flow of operation



This figure gives an overview of the project

SECTION - 2 – Design of Project

2.1 Low level design


Stage – 1
Data Ingestion

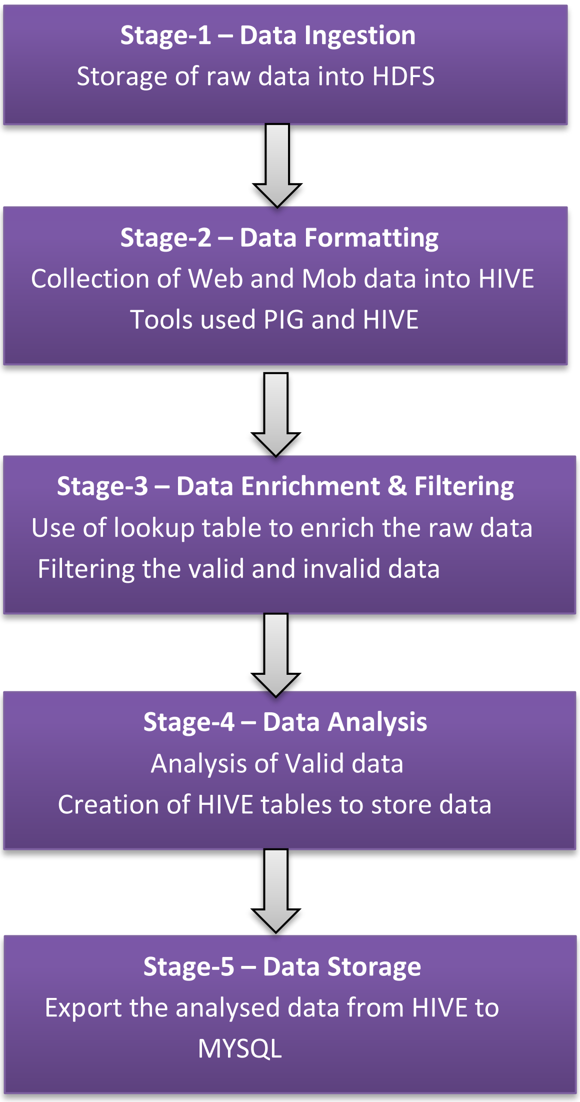
Stage – 2
Data Formatting

Stage – 3
Data Enrichment & Filtering

Stage – 4
Data Analysis

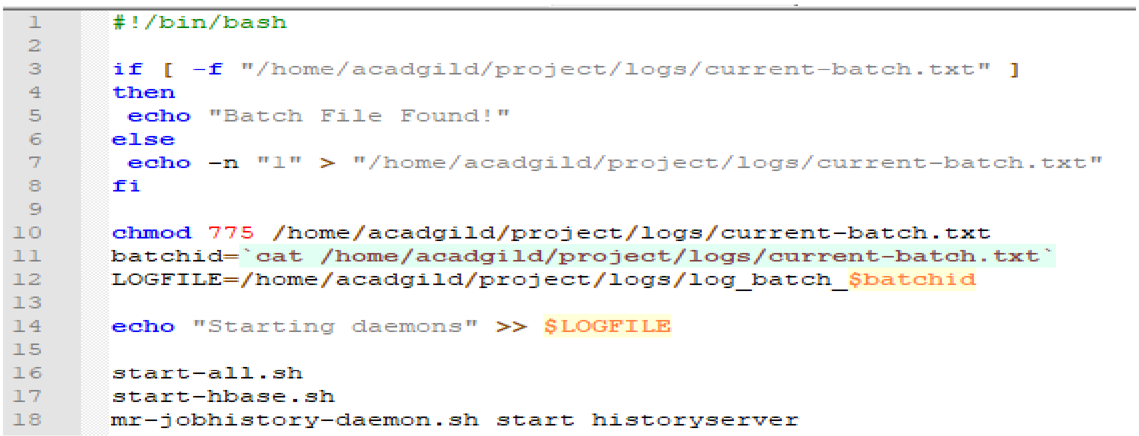
Stage – 5
Data Storage

2.2 High level Design


Stage-1 – Data Ingestion
Storage of raw data into HDFS
Stage-2 – Data Formatting
Collection of Web and Mob data into HIVE
Tools used PIG and HIVE
Stage-3 – Data Enrichment & Filtering
Use of lookup table to enrich the raw data
Filtering the valid and invalid data
Stage-4 – Data Analysis
Analysis of Valid data
Creation of HIVE tables to store data
Stage-5 – Data Storage
Export the analysed data from HIVE to
MYSQL

SECTION – 3 – Hadoop Eco-System Implementation

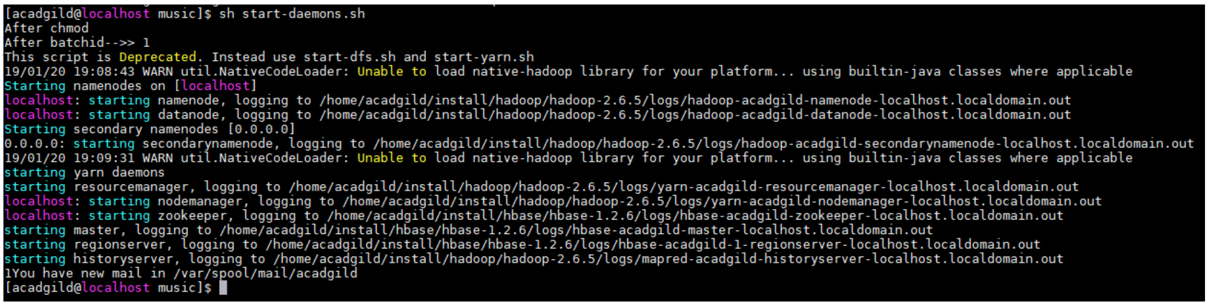
1. We have created a batch file **"start-daemon.sh"** which starts the daemons such as **hive**, **hbase**, **Mysql** and rest of the all **hadoop** daemons.



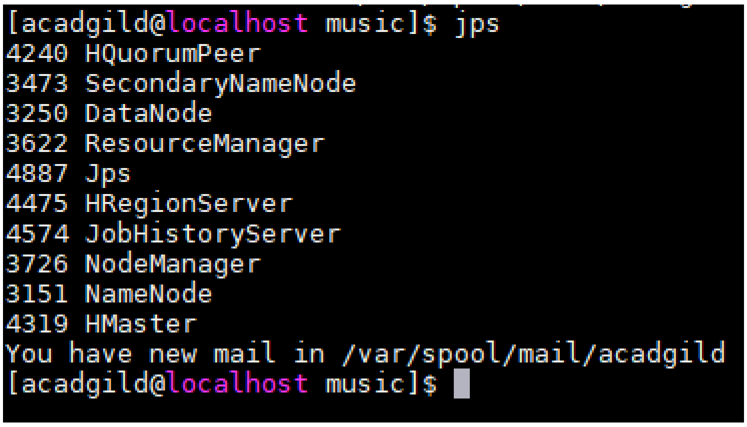
```
#!/bin/bash

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
```

2. Starting all daemons.

The batch file script **start-daemons.sh** will start all the hadoop daemons and Hbase daemons as shown in the below screen shot.

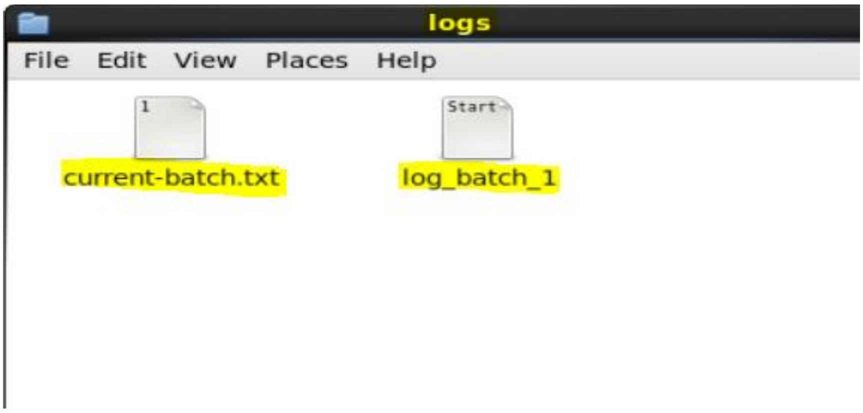


```
[acadgild@localhost music]$ sh start-daemons.sh
After chmod
After batchid-->> 1
This script is Deprecated. Instead use start-dfs.sh and start-yarn.sh
19/01/20 19:08:43 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Starting namenodes on [localhost]
localhost: starting namenode, logging to /home/acadgild/install/hadoop/hadoop-2.6.5/logs/hadoop-acadgild-namenode-localhost.localdomain.out
localhost: starting datanode, logging to /home/acadgild/install/hadoop/hadoop-2.6.5/logs/hadoop-acadgild-datanode-localhost.localdomain.out
Starting secondary namenodes [0.0.0.0]
0.0.0.0: starting secondarynamenode, logging to /home/acadgild/install/hadoop/hadoop-2.6.5/logs/hadoop-acadgild-secondarynamenode-localhost.localdomain.out
19/01/20 19:09:31 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
starting yarn daemons
starting resourcemanager, logging to /home/acadgild/install/hadoop/hadoop-2.6.5/logs/yarn-acadgild-resourcemanager-localhost.localdomain.out
localhost: starting nodemanager, logging to /home/acadgild/install/hadoop/hadoop-2.6.5/logs/yarn-acadgild-nodemanager-localhost.localdomain.out
localhost: starting zookeeper, logging to /home/acadgild/install/hbase/hbase-1.2.6/logs/hbase-acadgild-zookeeper-localhost.localdomain.out
starting master, logging to /home/acadgild/install/hbase/hbase-1.2.6/logs/hbase-acadgild-master-localhost.localdomain.out
starting regionserver, logging to /home/acadgild/install/hbase/hbase-1.2.6/logs/hbase-acadgild-1-regionserver-localhost.localdomain.out
starting historyserver, logging to /home/acadgild/install/hadoop/hadoop-2.6.5/logs/mapred-acadgild-historyserver-localhost.localdomain.out
1You have new mail in /var/spool/mail/acadgild
[acadgild@localhost music]$
```



```
[acadgild@localhost music]$ jps
4240 HQuorumPeer
3473 SecondaryNameNode
3250 DataNode
3622 ResourceManager
4887 Jps
4475 HRegionServer
4574 JobHistoryServer
3726 NodeManager
3151 NameNode
4319 HMaster
You have new mail in /var/spool/mail/acadgild
[acadgild@localhost music]$
```

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



SECTION – 4 – Data Ingestion, Formatting, Enrichment and Filtering

4.1 Stage – 1 – Data Ingestion

By using the “***populate-lookup.sh”*** script we will create lookup tables in **Hbase**. These tables have to be used in,

- Data Formatting
- Data Enrichment
- Analysis Stage

Lookup Tables –

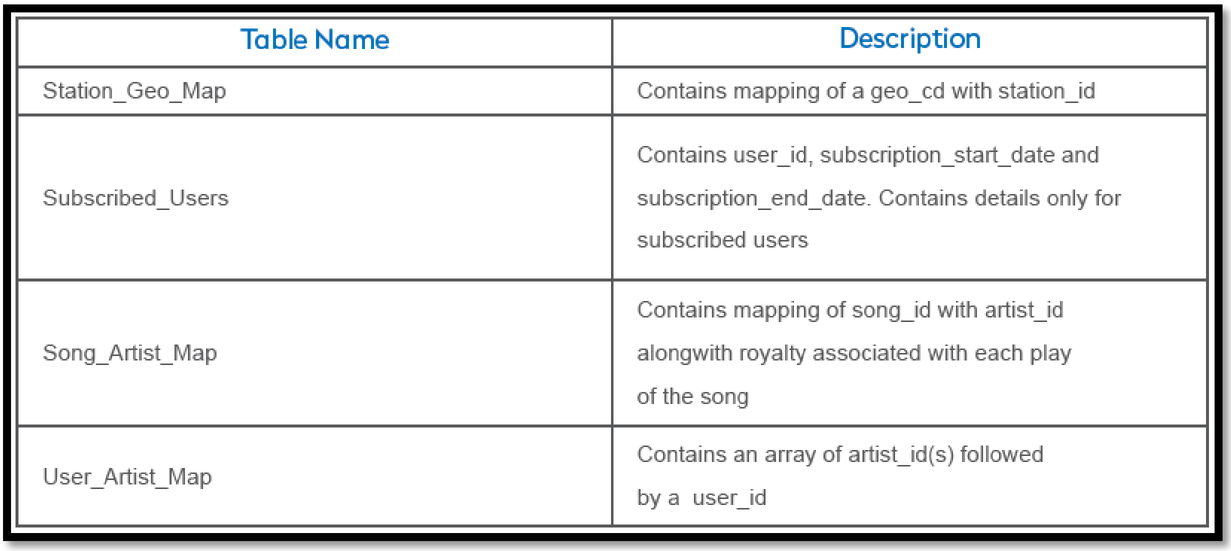
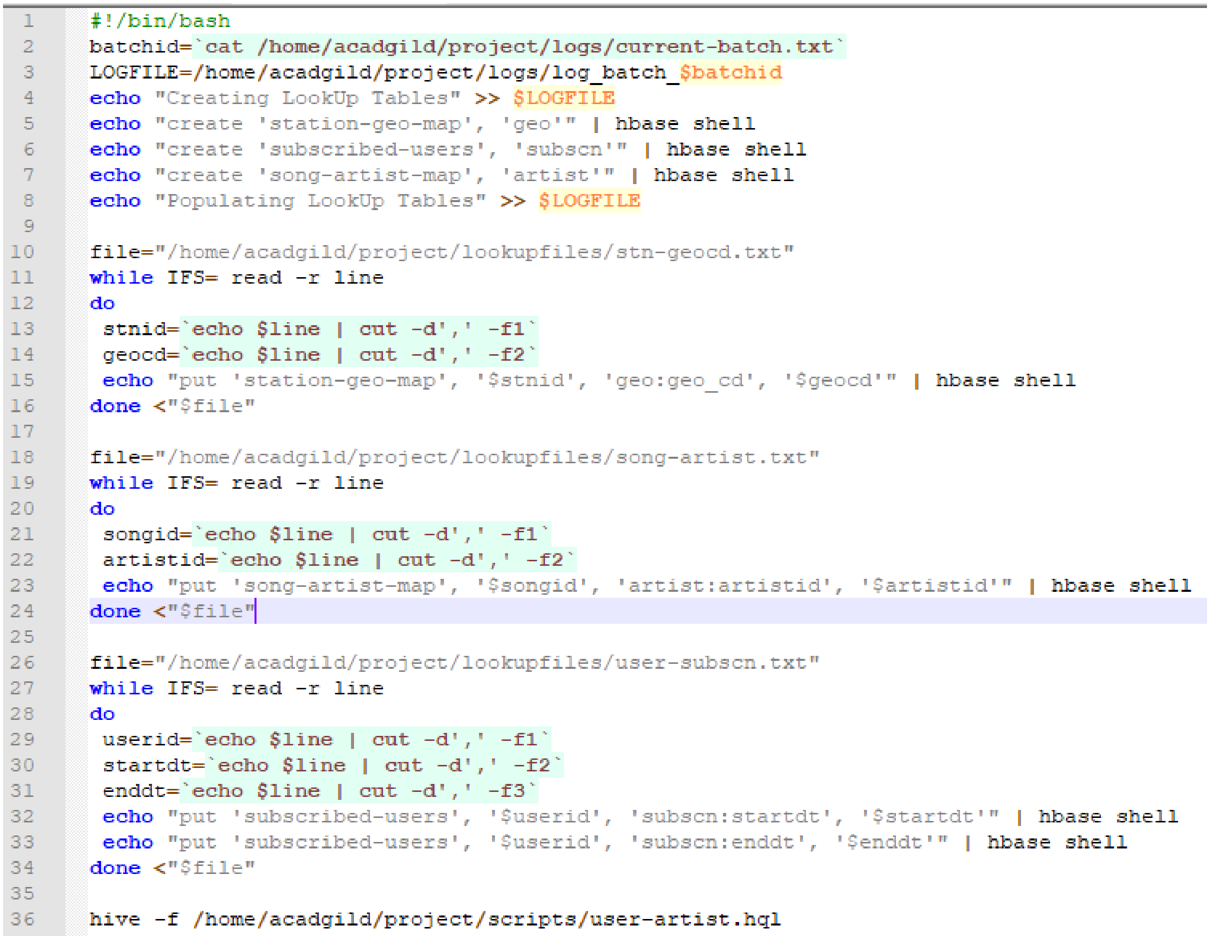


Table Name	Description
Station_Geo_Map	Contains mapping of a geo_cd with station_id
Subscribed_Users	Contains user_id, subscription_start_date and subscription_end_date. Contains details only for subscribed users
Song_Artist_Map	Contains mapping of song_id with artist_id alongwith royalty associated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

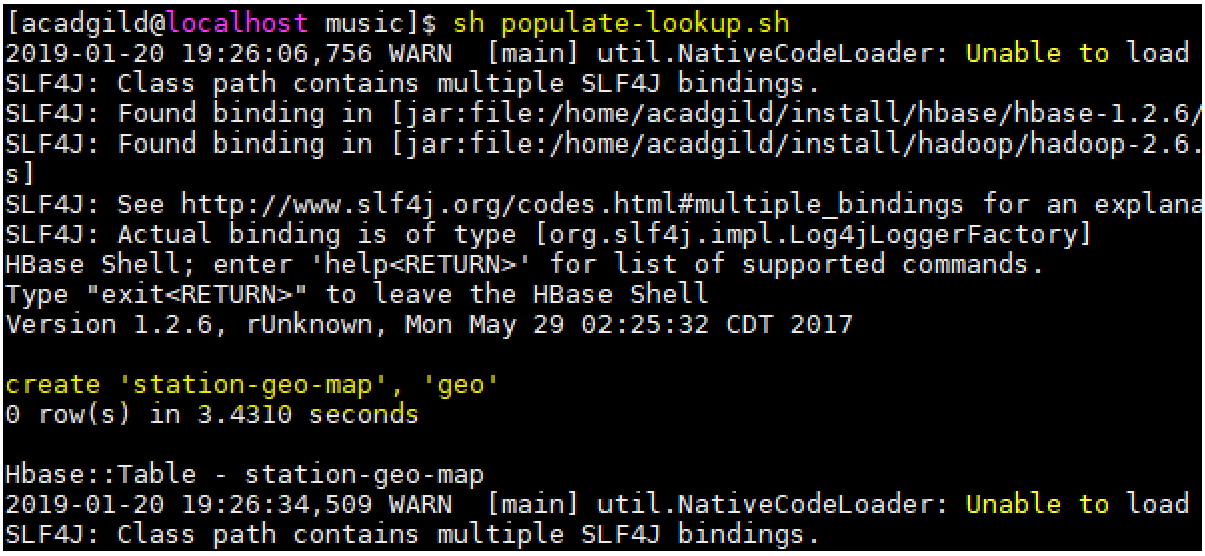
The “***populate-lookup.sh”*** shell script creates the above 4 lookup tables in the Hbase and populate the data into the lookup tables from the dataset files.

In the below screen shots, we can see the create-lookup.sh script and the following screen shots shows the table creation and population of the data in the Hbase. Also, the values loaded into the Hbase Tables are also shown, please see the below screen shots.

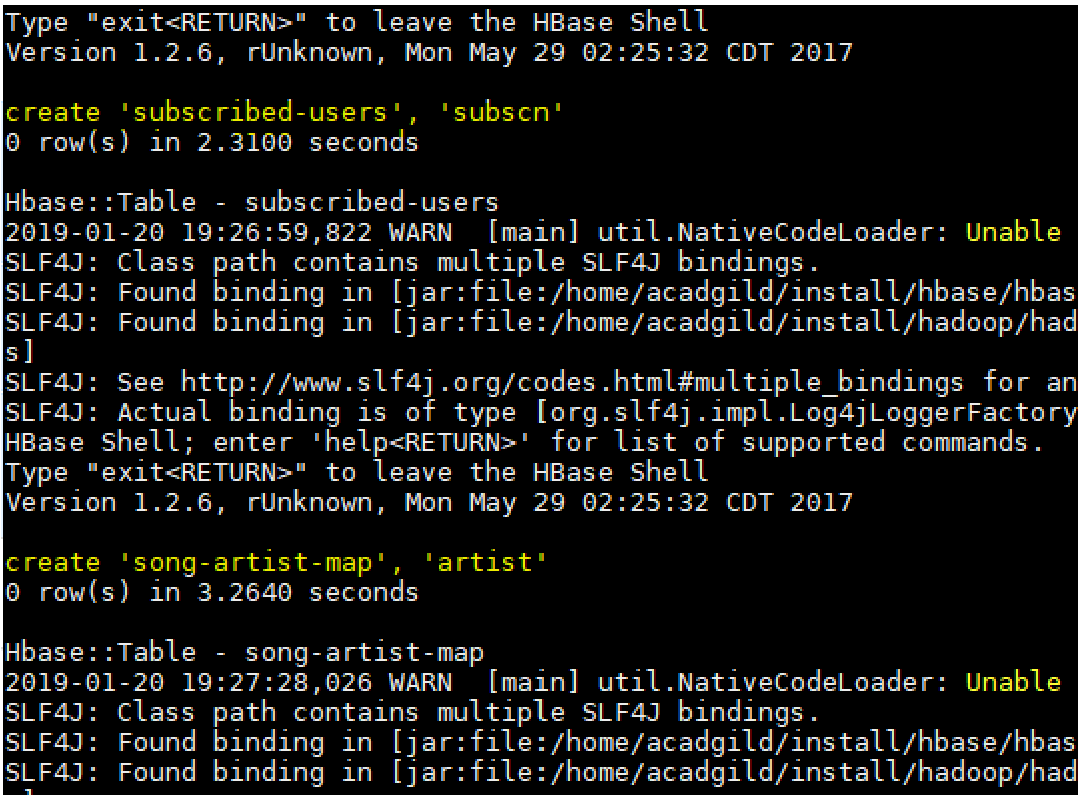


```
#!/bin/bash
batchid=`cat /home/acadgild/project/logs/current-batch.txt`
LOGFILE=/home/acadgild/project/logs/log_batch_$batchid
echo "Creating LookUp Tables" >> $LOGFILE
echo "create 'station-geo-map', 'geo'" | hbase shell
echo "create 'subscribed-users', 'subscn'" | hbase shell
echo "create 'song-artist-map', '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
```

Run Script: *./populate-lookup.sh*

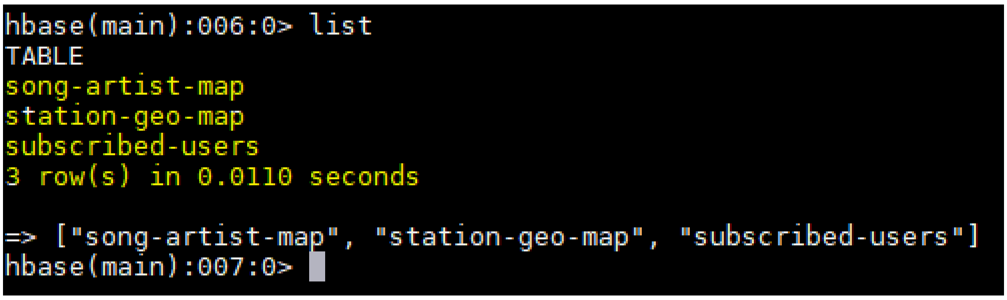


```
[acadgild@localhost music]$ sh populate-lookup.sh
2019-01-20 19:26:06,756 WARN  [main] util.NativeCodeLoader: Unable to load
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/acadgild/install/hbase/hbase-1.2.6/
SLF4J: Found binding in [jar:file:/home/acadgild/install/hadoop/hadoop-2.6.
s]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explana
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 3.4310 seconds

Hbase::Table - station-geo-map
2019-01-20 19:26:34,509 WARN  [main] util.NativeCodeLoader: Unable to load
SLF4J: Class path contains multiple SLF4J bindings.
```



```
Type "exit<RETURN>" to leave the HBase Shell
Version 1.2.6, rUnknown, Mon May 29 02:25:32 CDT 2017

create 'subscribed-users', 'subscn'
0 row(s) in 2.3100 seconds

Hbase::Table - subscribed-users
2019-01-20 19:26:59,822 WARN  [main] util.NativeCodeLoader: Unable
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/acadgild/install/hbase/hbas
SLF4J: Found binding in [jar:file:/home/acadgild/install/hadoop/had
s]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory
HBase Shell; enter 'help<RETURN>' for list of supported commands.
Type "exit<RETURN>" to leave the HBase Shell
Version 1.2.6, rUnknown, Mon May 29 02:25:32 CDT 2017

create 'song-artist-map', 'artist'
0 row(s) in 3.2640 seconds

Hbase::Table - song-artist-map
2019-01-20 19:27:28,026 WARN  [main] util.NativeCodeLoader: Unable
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/acadgild/install/hbase/hbas
SLF4J: Found binding in [jar:file:/home/acadgild/install/hadoop/had
```

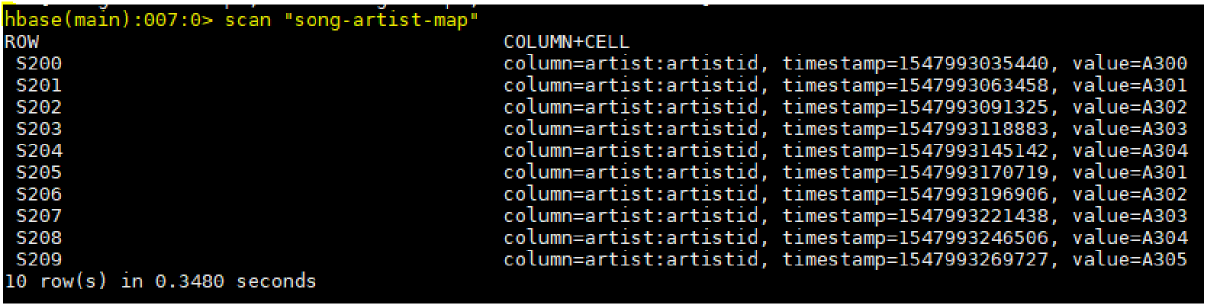
We can see the lookup tables created using the ***"populate-lookup.sh"*** in the below screen shot, Lookup Tables in the hbase shell,



```
hbase(main):006:0> list
TABLE
song-artist-map
station-geo-map
subscribed-users
3 row(s) in 0.0110 seconds

=> ["song-artist-map", "station-geo-map", "subscribed-users"]
hbase(main):007:0>
```

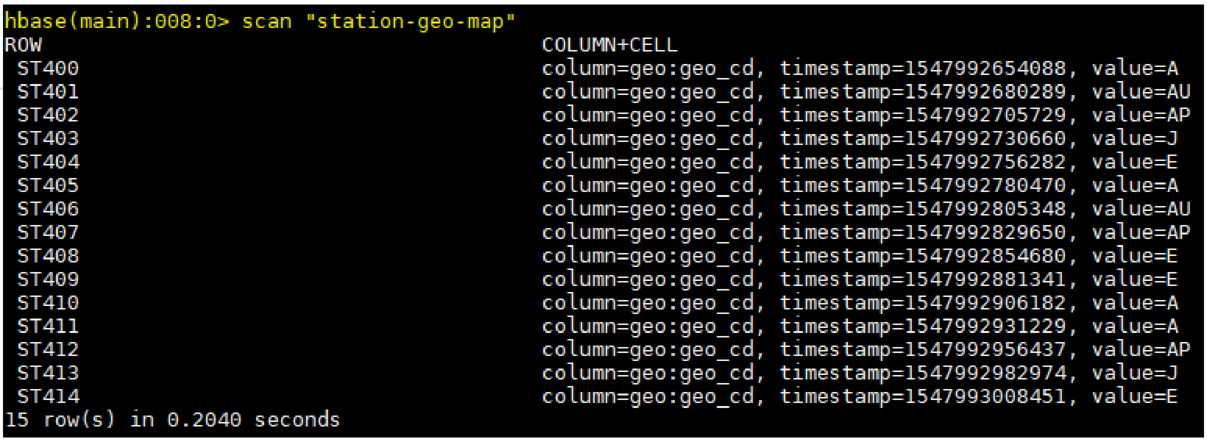
Values loaded in lookup tables –

song-artist-map



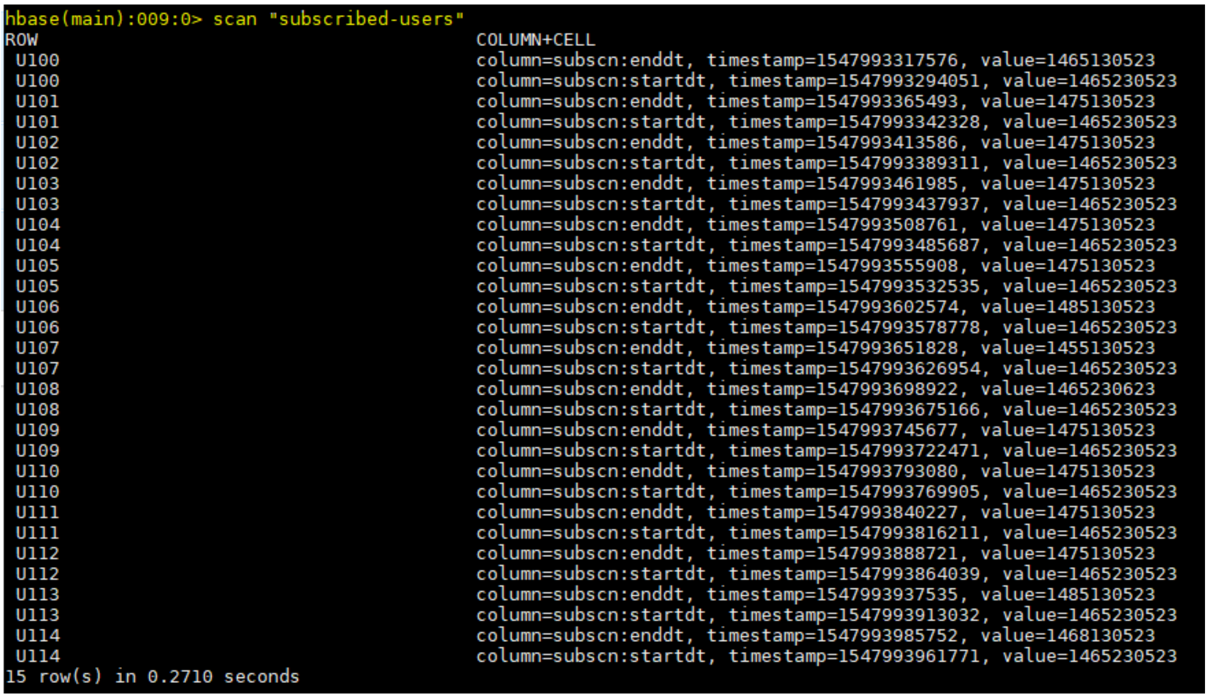
```
hbase(main):007:0> scan "song-artist-map"
ROW                                       COLUMN+CELL
 S200                                     column=artist:artistid, timestamp=1547993035440, value=A300
 S201                                     column=artist:artistid, timestamp=1547993063458, value=A301
 S202                                     column=artist:artistid, timestamp=1547993091325, value=A302
 S203                                     column=artist:artistid, timestamp=1547993118883, value=A303
 S204                                     column=artist:artistid, timestamp=1547993145142, value=A304
 S205                                     column=artist:artistid, timestamp=1547993170719, value=A301
 S206                                     column=artist:artistid, timestamp=1547993196906, value=A302
 S207                                     column=artist:artistid, timestamp=1547993221438, value=A303
 S208                                     column=artist:artistid, timestamp=1547993246506, value=A304
 S209                                     column=artist:artistid, timestamp=1547993269727, value=A305
10 row(s) in 0.3480 seconds
```

station-geo-map



```
hbase(main):008:0> scan "station-geo-map"
ROW                                       COLUMN+CELL
 ST400                                    column=geo:geo_cd, timestamp=1547992654088, value=A
 ST401                                    column=geo:geo_cd, timestamp=1547992680289, value=AU
 ST402                                    column=geo:geo_cd, timestamp=1547992705729, value=AP
 ST403                                    column=geo:geo_cd, timestamp=1547992730660, value=J
 ST404                                    column=geo:geo_cd, timestamp=1547992756282, value=E
 ST405                                    column=geo:geo_cd, timestamp=1547992780470, value=A
 ST406                                    column=geo:geo_cd, timestamp=1547992805348, value=AU
 ST407                                    column=geo:geo_cd, timestamp=1547992829650, value=AP
 ST408                                    column=geo:geo_cd, timestamp=1547992854680, value=E
 ST409                                    column=geo:geo_cd, timestamp=1547992881341, value=E
 ST410                                    column=geo:geo_cd, timestamp=1547992906182, value=A
 ST411                                    column=geo:geo_cd, timestamp=1547992931229, value=A
 ST412                                    column=geo:geo_cd, timestamp=1547992956437, value=AP
 ST413                                    column=geo:geo_cd, timestamp=1547992982974, value=J
 ST414                                    column=geo:geo_cd, timestamp=1547993008451, value=E
15 row(s) in 0.2040 seconds
```

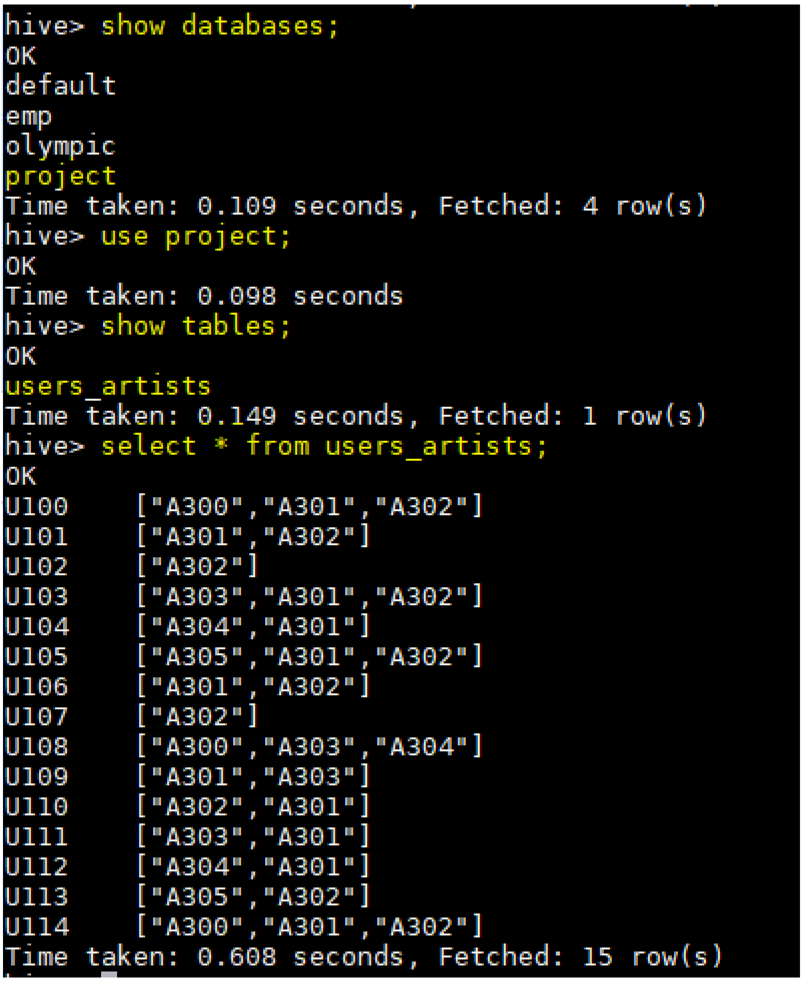
subscribed-users



```
hbase(main):009:0> scan "subscribed-users"
ROW                                         COLUMN+CELL
 U100                                       column=subscn:enddt, timestamp=1547993317576, value=1465130523
 U100                                       column=subscn:startdt, timestamp=1547993294051, value=1465230523
 U101                                       column=subscn:enddt, timestamp=1547993365493, value=1475130523
 U101                                       column=subscn:startdt, timestamp=1547993342328, value=1465230523
 U102                                       column=subscn:enddt, timestamp=1547993413586, value=1475130523
 U102                                       column=subscn:startdt, timestamp=1547993389311, value=1465230523
 U103                                       column=subscn:enddt, timestamp=1547993461985, value=1475130523
 U103                                       column=subscn:startdt, timestamp=1547993437937, value=1465230523
 U104                                       column=subscn:enddt, timestamp=1547993508761, value=1475130523
 U104                                       column=subscn:startdt, timestamp=1547993485687, value=1465230523
 U105                                       column=subscn:enddt, timestamp=1547993555908, value=1475130523
 U105                                       column=subscn:startdt, timestamp=1547993532535, value=1465230523
 U106                                       column=subscn:enddt, timestamp=1547993602574, value=1485130523
 U106                                       column=subscn:startdt, timestamp=1547993578778, value=1465230523
 U107                                       column=subscn:enddt, timestamp=1547993651828, value=1455130523
 U107                                       column=subscn:startdt, timestamp=1547993626954, value=1465230523
 U108                                       column=subscn:enddt, timestamp=1547993698922, value=1465230623
 U108                                       column=subscn:startdt, timestamp=1547993675166, value=1465230523
 U109                                       column=subscn:enddt, timestamp=1547993745677, value=1475130523
 U109                                       column=subscn:startdt, timestamp=1547993722471, value=1465230523
 U110                                       column=subscn:enddt, timestamp=1547993793080, value=1475130523
 U110                                       column=subscn:startdt, timestamp=1547993769905, value=1465230523
 U111                                       column=subscn:enddt, timestamp=1547993840227, value=1475130523
 U111                                       column=subscn:startdt, timestamp=1547993816211, value=1465230523
 U112                                       column=subscn:enddt, timestamp=1547993888721, value=1475130523
 U112                                       column=subscn:startdt, timestamp=1547993864039, value=1465230523
 U113                                       column=subscn:enddt, timestamp=1547993937535, value=1485130523
 U113                                       column=subscn:startdt, timestamp=1547993913032, value=1465230523
 U114                                       column=subscn:enddt, timestamp=1547993985752, value=1468130523
 U114                                       column=subscn:startdt, timestamp=1547993961771, value=1465230523
15 row(s) in 0.2710 seconds
```

We have successfully created the lookup tables in the Hbase.

The populate-lookup.sh also creates a lookup table **"users_artists"** in the HIVE, loading the data from the **user-artist.txt,** the below screen shot shows that the table has been created in the HIVE.



```
hive> show databases;
OK
default
emp
olympic
project
Time taken: 0.109 seconds, Fetched: 4 row(s)
hive> use project;
OK
Time taken: 0.098 seconds
hive> show tables;
OK
users_artists
Time taken: 0.149 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: 0.608 seconds, Fetched: 15 row(s)
```

Now we need to link theses lookup tables in hive using the Hbase Storage Handler.

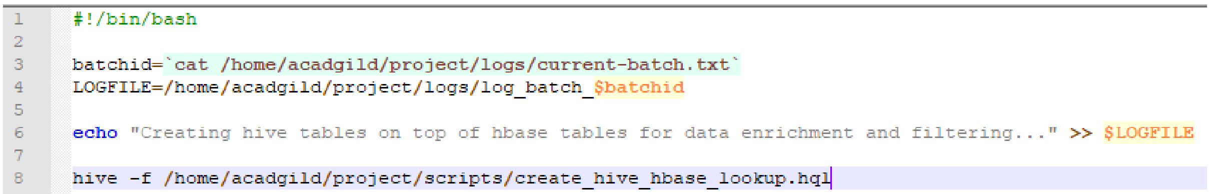
With the help of **“data_enrichment_filtering_schema.sh”** file we will create hive tables on the top of Hbase tables using **“create_hive_hbase_lookup.hql”.**

Creating Hive tables on top of HBASE-

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

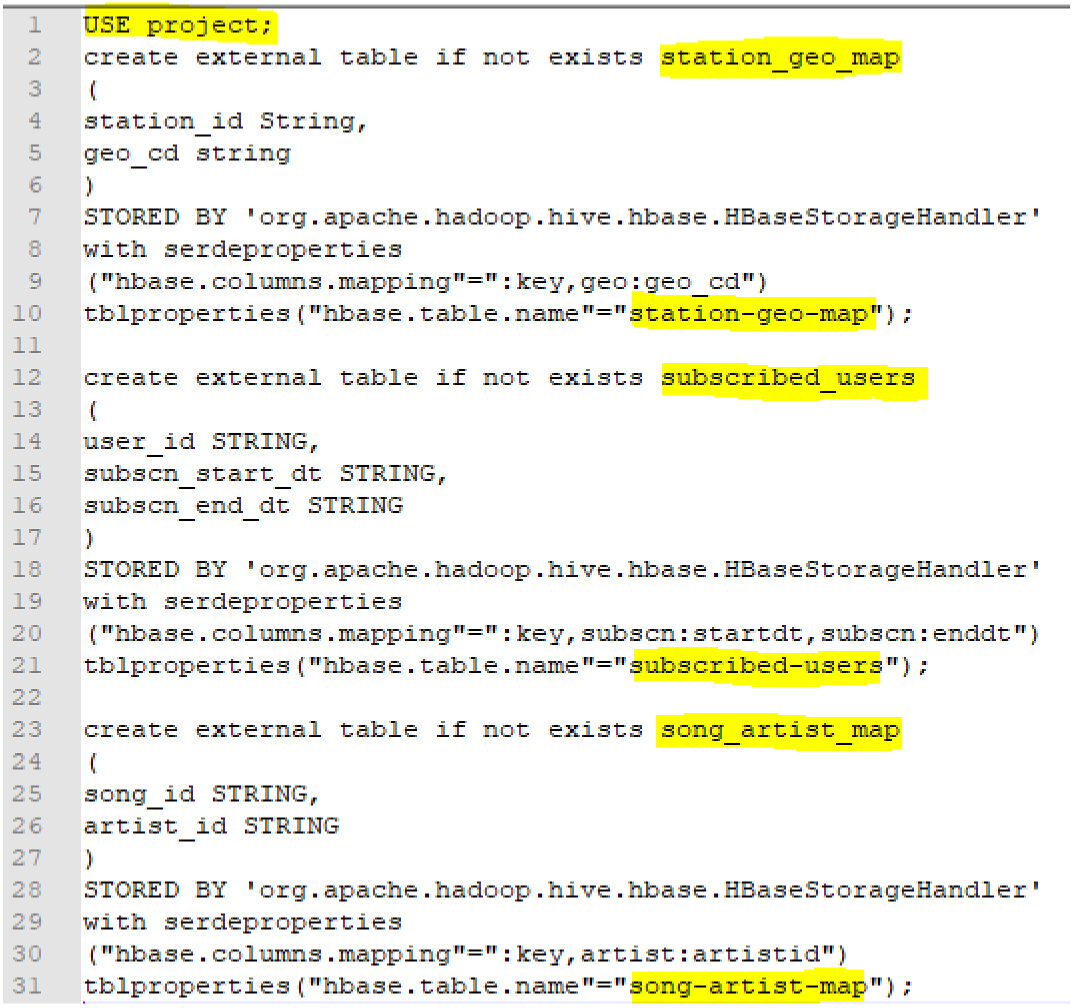
Run the script: ./data_enrichment_filtering_schema.sh

The script will run the **“create_hive_hbase_lookup.hql”** which will create the HIVE external tables with the help of **Hbase storage handler & SerDe properties**. The hive external tables will match the columns of **Hbase** tables to **HIVE** tables.



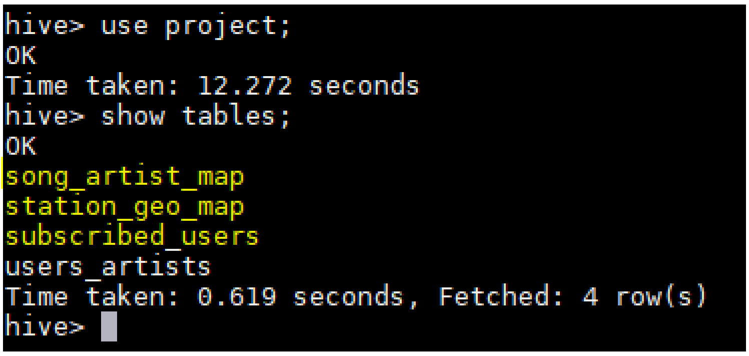
```
#!/bin/bash

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



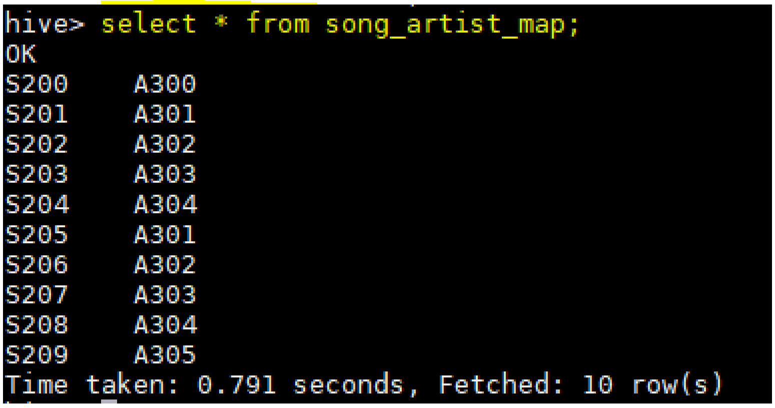
```
USE project;
create external table if not exists station_geo_map
(
station_id String,
geo_cd string
)
STORED BY 'org.apache.hadoop.hive.hbase.HBaseStorageHandler'
with serdeproperties
("hbase.columns.mapping"=":key,geo:geo_cd")
tblproperties("hbase.table.name"="station-geo-map");

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");
```

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



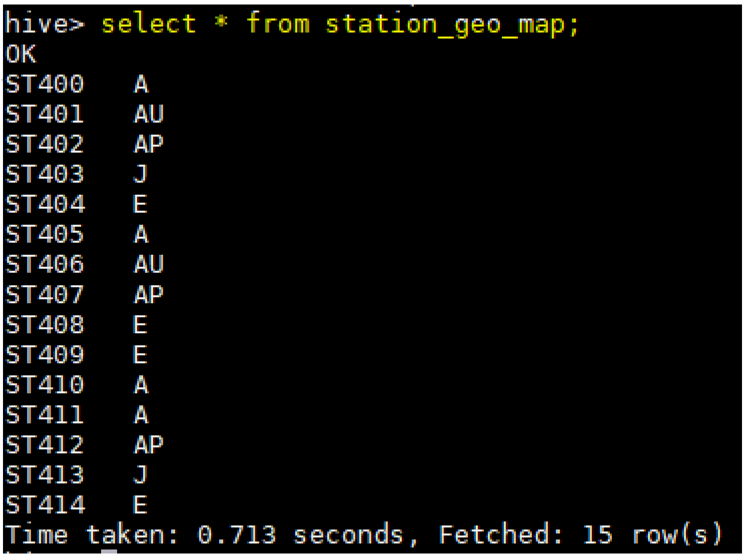
```
hive> use project;
OK
Time taken: 12.272 seconds
hive> show tables;
OK
song_artist_map
station_geo_map
subscribed_users
users_artists
Time taken: 0.619 seconds, Fetched: 4 row(s)
hive> 
```

*Select * from song_artist_map*



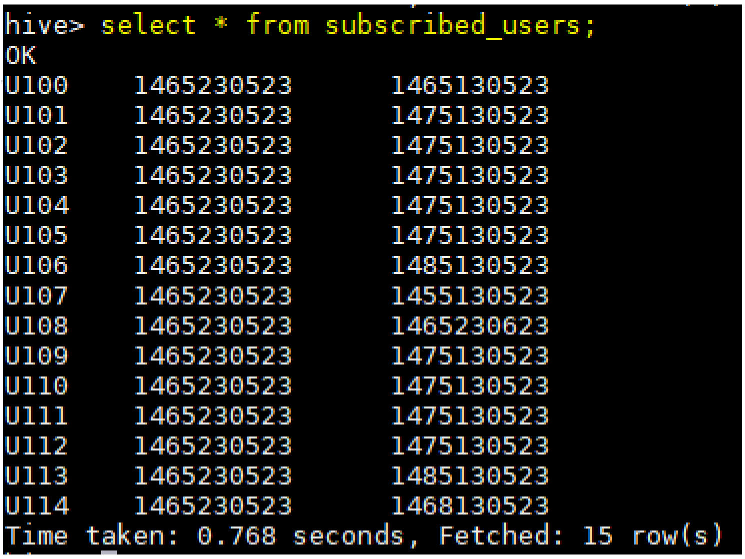
```
hive> select * from song_artist_map;
OK
S200    A300
S201    A301
S202    A302
S203    A303
S204    A304
S205    A301
S206    A302
S207    A303
S208    A304
S209    A305
Time taken: 0.791 seconds, Fetched: 10 row(s)
```

*Select * from station_geo_map;*



```
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: 0.713 seconds, Fetched: 15 row(s)
```

*Select * from subscribed_users;*

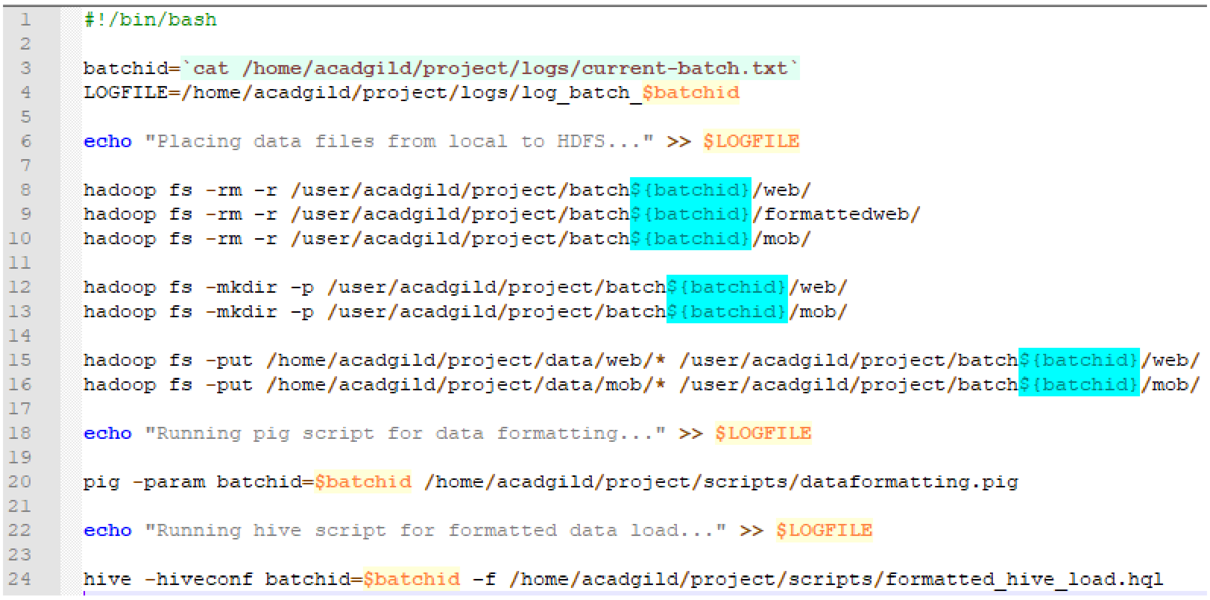


```
hive> select * from subscribed_users;
OK
U100    1465230523      1465130523
U101    1465230523      1475130523
U102    1465230523      1475130523
U103    1465230523      1475130523
U104    1465230523      1475130523
U105    1465230523      1475130523
U106    1465230523      1485130523
U107    1465230523      1455130523
U108    1465230523      1465230623
U109    1465230523      1475130523
U110    1465230523      1475130523
U111    1465230523      1475130523
U112    1465230523      1475130523
U113    1465230523      1485130523
U114    1465230523      1468130523
Time taken: 0.768 seconds, Fetched: 15 row(s)
```

4.2 Stage-2 – Data Formatting

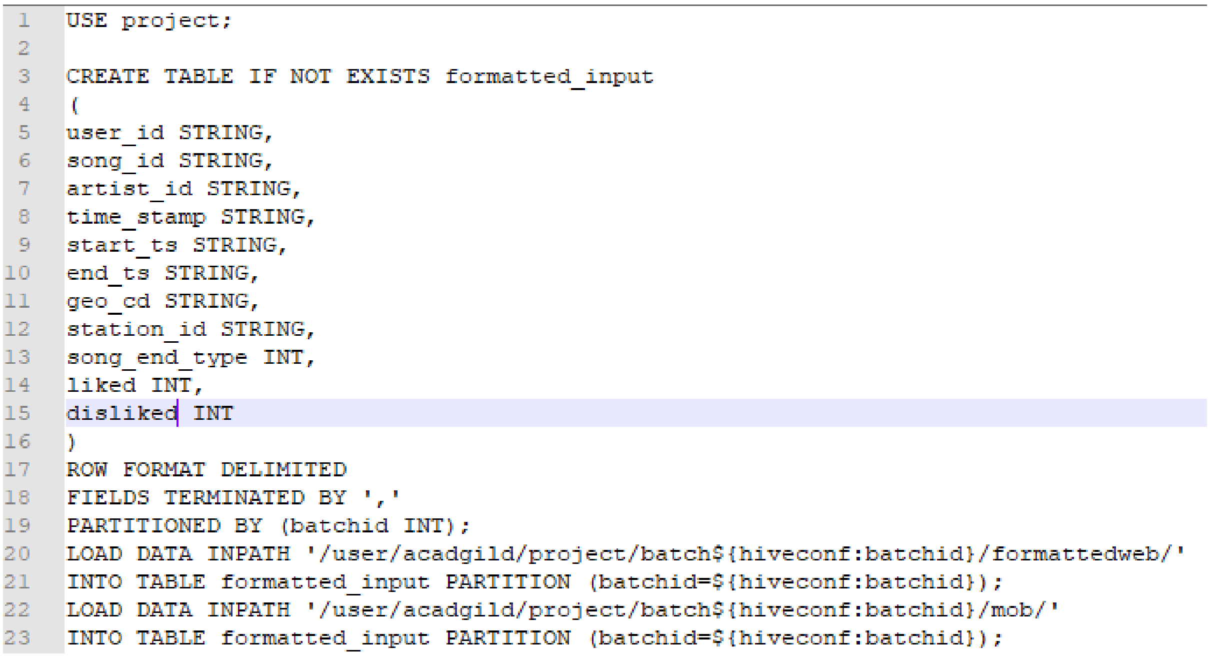
In this stage we are merging the data coming from both **web** applications and **mobile** applications and create a common table for analyzing purpose and create partitioned data based on **batchid**, since we are running this script for every 3 hours.

Run the script: *./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

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
```

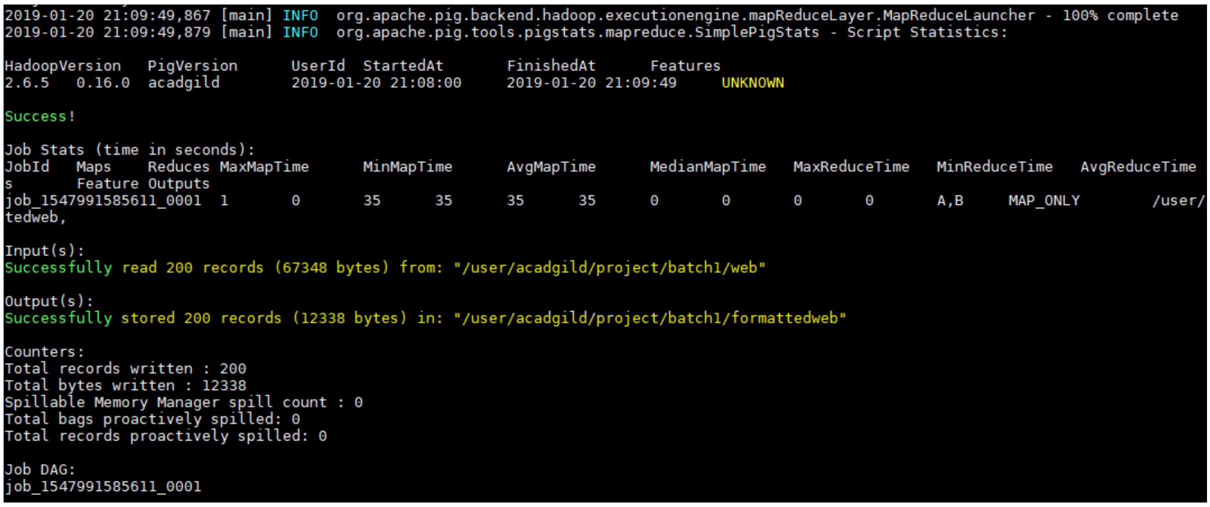
formatted_hive_load.hql



```
USE project;

CREATE TABLE IF NOT EXISTS formatted_input
(
user_id STRING,
song_id STRING,
artist_id STRING,
time_stamp STRING,
start_ts STRING,
end_ts STRING,
geo_cd STRING,
station_id STRING,
song_end_type INT,
liked INT,
disliked INT
)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
PARTITIONED BY (batchid INT);
LOAD DATA INPATH '/user/acadgild/project/batch${hiveconf:batchid}/formattedweb/'
INTO TABLE formatted_input PARTITION (batchid=${hiveconf:batchid});
LOAD DATA INPATH '/user/acadgild/project/batch${hiveconf:batchid}/mob/'
INTO TABLE formatted_input PARTITION (batchid=${hiveconf:batchid});
```

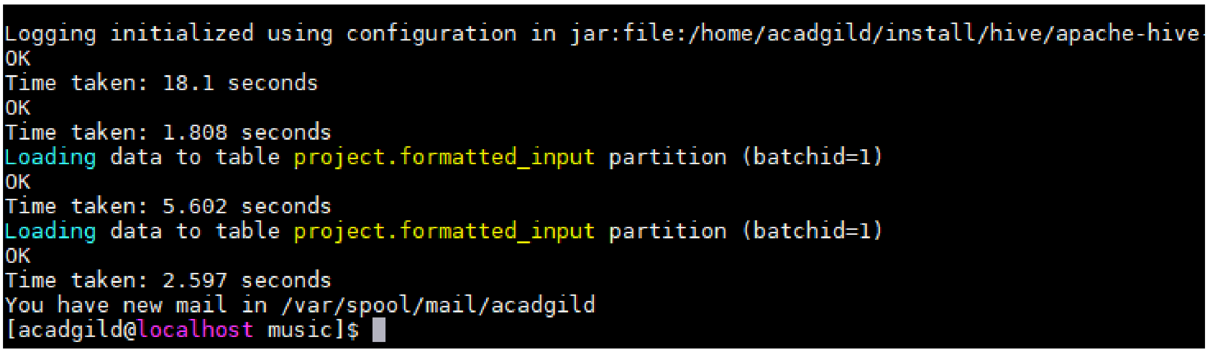
In the below screenshot we can see the data both the scripts in action, first pig script will parse the data and then hive script will load the data into hive terminal successfully.

Pig script completion -



```
2019-01-20 21:09:49,867 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - 100% complete
2019-01-20 21:09:49,879 [main] INFO  org.apache.pig.tools.pigstats.mapreduce.SimplePigStats - Script Statistics:

HadoopVersion   PigVersion      UserId  StartedAt       FinishedAt      Features
2.6.5   0.16.0  acadgild        2019-01-20 21:08:00     2019-01-20 21:09:49     UNKNOWN

Success!

Job Stats (time in seconds):
JobId   Maps    Reduces MaxMapTime      MinMapTime      AvgMapTime      MedianMapTime   MaxReduceTime   MinReduceTime   AvgReduceTime
s       Feature Outputs
job_1547991585611_0001  1       0       35      35      35      35      0       0       0       0       A,B     MAP_ONLY        /user/
tedweb,

Input(s):
Successfully read 200 records (67348 bytes) from: "/user/acadgild/project/batch1/web"

Output(s):
Successfully stored 200 records (12338 bytes) in: "/user/acadgild/project/batch1/formattedweb"

Counters:
Total records written : 200
Total bytes written : 12338
Spillable Memory Manager spill count : 0
Total bags proactively spilled: 0
Total records proactively spilled: 0

Job DAG:
job_1547991585611_0001
```

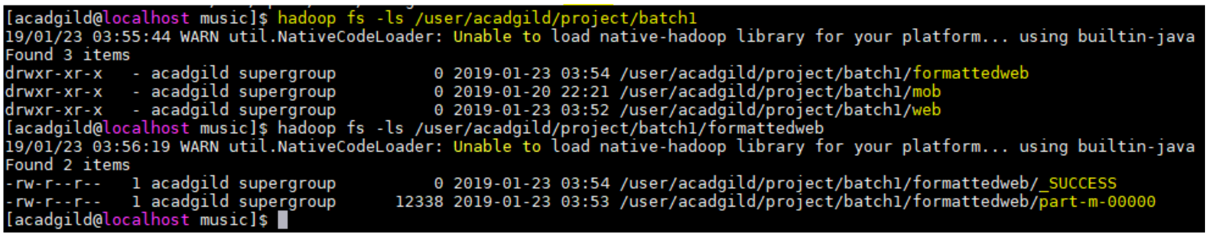
Hive script successfully load the data into hive terminal –



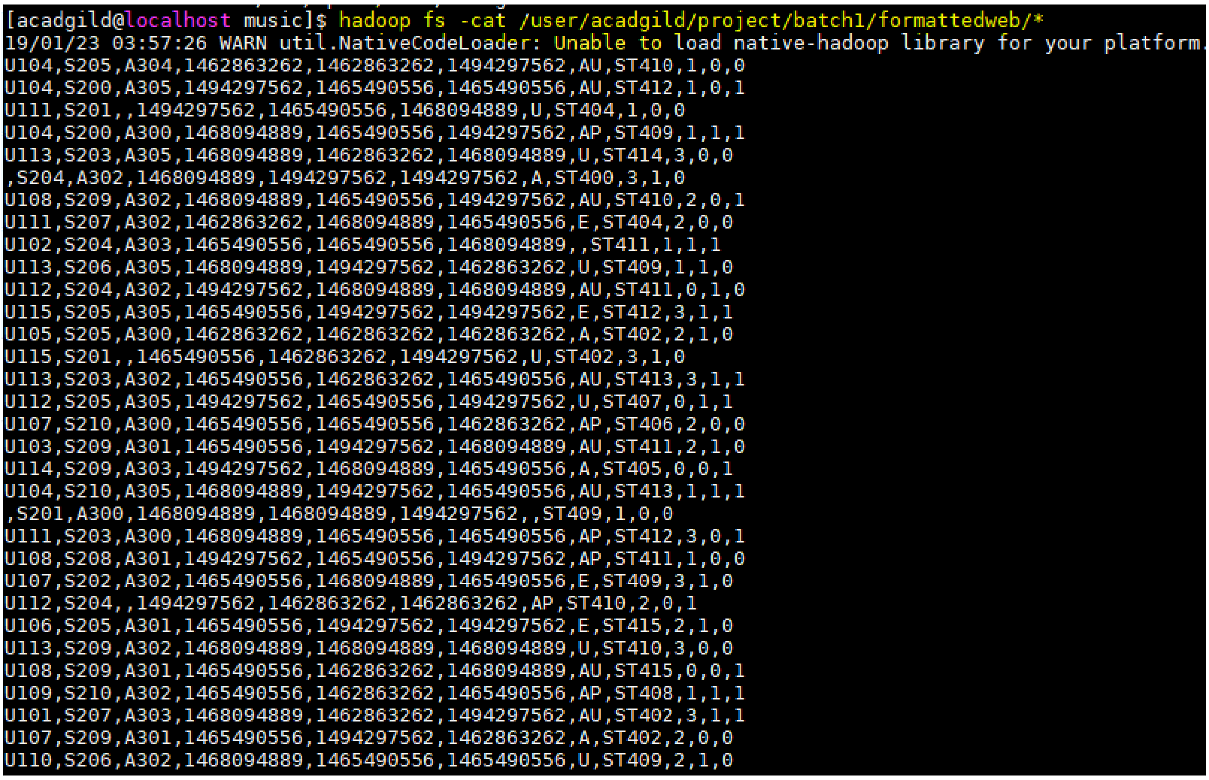
```
Logging initialized using configuration in jar:file:/home/acadgild/install/hive/apache-hive-
OK
Time taken: 18.1 seconds
OK
Time taken: 1.808 seconds
Loading data to table project.formatted_input partition (batchid=1)
OK
Time taken: 5.602 seconds
Loading data to table project.formatted_input partition (batchid=1)
OK
Time taken: 2.597 seconds
You have new mail in /var/spool/mail/acadgild
[acadgild@localhost music]$ 
```

In the above screenshot we can see the **dataformatting.pig** along with the **formatted_hive_load.hql** executed successfully.

The output of **dataformatting.sh** script in HDFS folders:

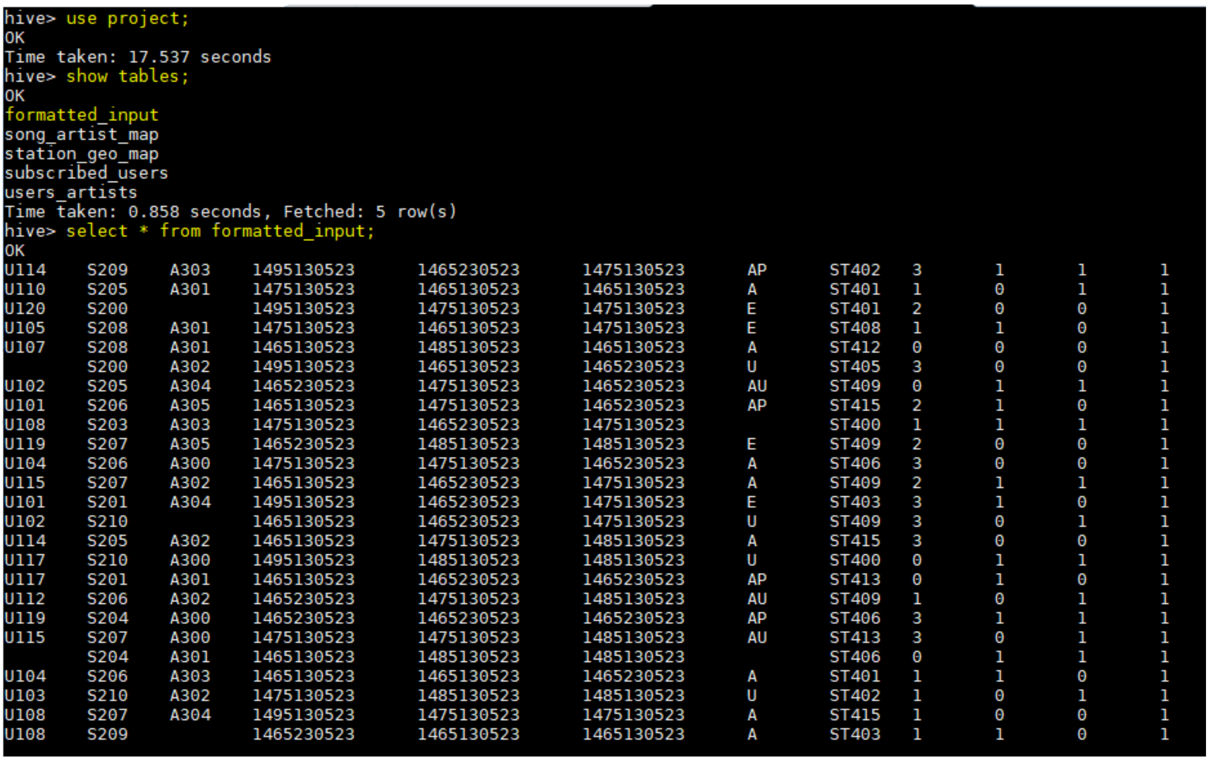


```
[acadgild@localhost music]$ hadoop fs -ls /user/acadgild/project/batch1
19/01/23 03:55:44 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java
Found 3 items
drwxr-xr-x   - acadgild supergroup          0 2019-01-23 03:54 /user/acadgild/project/batch1/formattedweb
drwxr-xr-x   - acadgild supergroup          0 2019-01-20 22:21 /user/acadgild/project/batch1/mob
drwxr-xr-x   - acadgild supergroup          0 2019-01-23 03:52 /user/acadgild/project/batch1/web
[acadgild@localhost music]$ hadoop fs -ls /user/acadgild/project/batch1/formattedweb
19/01/23 03:56:19 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java
Found 2 items
-rw-r--r--   1 acadgild supergroup          0 2019-01-23 03:54 /user/acadgild/project/batch1/formattedweb/_SUCCESS
-rw-r--r--   1 acadgild supergroup      12338 2019-01-23 03:53 /user/acadgild/project/batch1/formattedweb/part-m-00000
[acadgild@localhost music]$ 
```



```
[acadgild@localhost music]$ hadoop fs -cat /user/acadgild/project/batch1/formattedweb/*
19/01/23 03:57:26 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform.
U104,S205,A304,1462863262,1462863262,1494297562,AU,ST410,1,0,0
U104,S200,A305,1494297562,1465490556,1465490556,AU,ST412,1,0,1
U111,S201,,1494297562,1465490556,1468094889,U,ST404,1,0,0
U104,S200,A300,1468094889,1465490556,1494297562,AP,ST409,1,1,1
U113,S203,A305,1468094889,1462863262,1468094889,U,ST414,3,0,0
,S204,A302,1468094889,1494297562,1494297562,A,ST400,3,1,0
U108,S209,A302,1468094889,1465490556,1494297562,AU,ST410,2,0,1
U111,S207,A302,1462863262,1468094889,1465490556,E,ST404,2,0,0
U102,S204,A303,1465490556,1465490556,1468094889,,ST411,1,1,1
U113,S206,A305,1468094889,1494297562,1462863262,U,ST409,1,1,0
U112,S204,A302,1494297562,1468094889,1468094889,AU,ST411,0,1,0
U115,S205,A305,1465490556,1494297562,1494297562,E,ST412,3,1,1
U105,S205,A300,1462863262,1462863262,1462863262,A,ST402,2,1,0
U115,S201,,1465490556,1462863262,1494297562,U,ST402,3,1,0
U113,S203,A302,1465490556,1462863262,1465490556,AU,ST413,3,1,1
U112,S205,A305,1494297562,1465490556,1494297562,U,ST407,0,1,1
U107,S210,A300,1465490556,1465490556,1462863262,AP,ST406,2,0,0
U103,S209,A301,1465490556,1494297562,1468094889,AU,ST411,2,1,0
U114,S209,A303,1494297562,1468094889,1465490556,A,ST405,0,0,1
U104,S210,A305,1468094889,1494297562,1465490556,AU,ST413,1,1,1
,S201,A300,1468094889,1468094889,1494297562,,ST409,1,0,0
U111,S203,A300,1468094889,1465490556,1465490556,AP,ST412,3,0,1
U108,S208,A301,1468094889,1494297562,1465490556,AP,ST411,1,0,0
U107,S202,A302,1465490556,1468094889,1465490556,E,ST409,3,1,0
U112,S204,,1494297562,1462863262,1462863262,AP,ST410,2,0,1
U106,S205,A301,1465490556,1494297562,1494297562,E,ST415,2,1,0
U113,S209,A302,1468094889,1468094889,1468094889,U,ST410,3,0,0
U108,S209,A301,1465490556,1462863262,1468094889,AU,ST415,0,0,1
U109,S210,A302,1465490556,1462863262,1465490556,AP,ST408,1,1,1
U101,S207,A303,1468094889,1462863262,1494297562,AU,ST402,3,1,1
U107,S209,A301,1465490556,1494297562,1462863262,A,ST402,2,0,0
U110,S206,A302,1468094889,1465490556,1465490556,U,ST409,2,1,0
```

The ***formatted_input*** table is created in HIVE after running **dataformatting.sh** script,



```
hive> use project;
OK
Time taken: 17.537 seconds
hive> show tables;
OK
formatted_input
song_artist_map
station_geo_map
subscribed_users
users_artists
Time taken: 0.858 seconds, Fetched: 5 row(s)
hive> select * from formatted_input;
OK
U114    S209    A303    1495130523      1465230523      1475130523      AP      ST402   3       1       1       1
U110    S205    A301    1475130523      1465130523      1465130523      A       ST401   1       0       1       1
U120    S200            1495130523      1475130523      1475130523      E       ST401   2       0       0       1
U105    S208    A301    1475130523      1465130523      1475130523      E       ST408   1       1       0       1
U107    S208    A301    1465130523      1485130523      1465130523      A       ST412   0       0       0       1
        S200    A302    1495130523      1465130523      1465230523      U       ST405   3       0       0       1
U102    S205    A304    1465230523      1475130523      1465230523      AU      ST409   0       1       1       1
U101    S206    A305    1465130523      1475130523      1465230523      AP      ST415   2       1       0       1
U108    S203    A303    1475130523      1465230523      1475130523              ST400   1       1       1       1
U119    S207    A305    1465230523      1485130523      1485130523      E       ST409   2       0       0       1
U104    S206    A300    1475130523      1475130523      1465230523      A       ST406   3       0       0       1
U115    S207    A302    1465130523      1465230523      1475130523      A       ST409   2       1       1       1
U101    S201    A304    1495130523      1465230523      1475130523      E       ST403   3       1       0       1
U102    S210            1465130523      1465230523      1475130523      U       ST409   3       0       1       1
U114    S205    A302    1465130523      1475130523      1485130523      A       ST415   3       0       0       1
U117    S210    A300    1495130523      1485130523      1485130523      U       ST400   0       1       1       1
U117    S201    A301    1465130523      1465230523      1465230523      AP      ST413   0       1       0       1
U112    S206    A302    1465230523      1475130523      1485130523      AU      ST409   1       0       1       1
U119    S204    A300    1465230523      1465230523      1465230523      AP      ST406   3       1       1       1
U115    S207    A300    1475130523      1475130523      1485130523      AU      ST413   3       0       1       1
        S204    A301    1465130523      1485130523      1485130523              ST406   0       1       1       1
U104    S206    A303    1465130523      1465130523      1465230523      A       ST401   1       1       0       1
U103    S210    A302    1475130523      1485130523      1485130523      U       ST402   1       0       1       1
U108    S207    A304    1495130523      1475130523      1475130523      A       ST415   1       0       0       1
U108    S209            1465230523      1465130523      1465130523      A       ST403   1       1       0       1
```

- In the above screenshot we can see the formatted input data with some null values in **user_id, aritist_id** and **geo_cd** columns which we will fill in stage-3 (Data Enrichment) based on rules of enrichment for **artist_id** and **geo_cd** only. We will get neglect **user_id** because they didn't mention anything about **user_id** for enrichment purpose.

- Data formatting phase is executed successfully by loading both **mobile** and **web** data and partitioned based on **batchid**.

4.3 Stage – 3 – Data Enrichment & Filtering

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

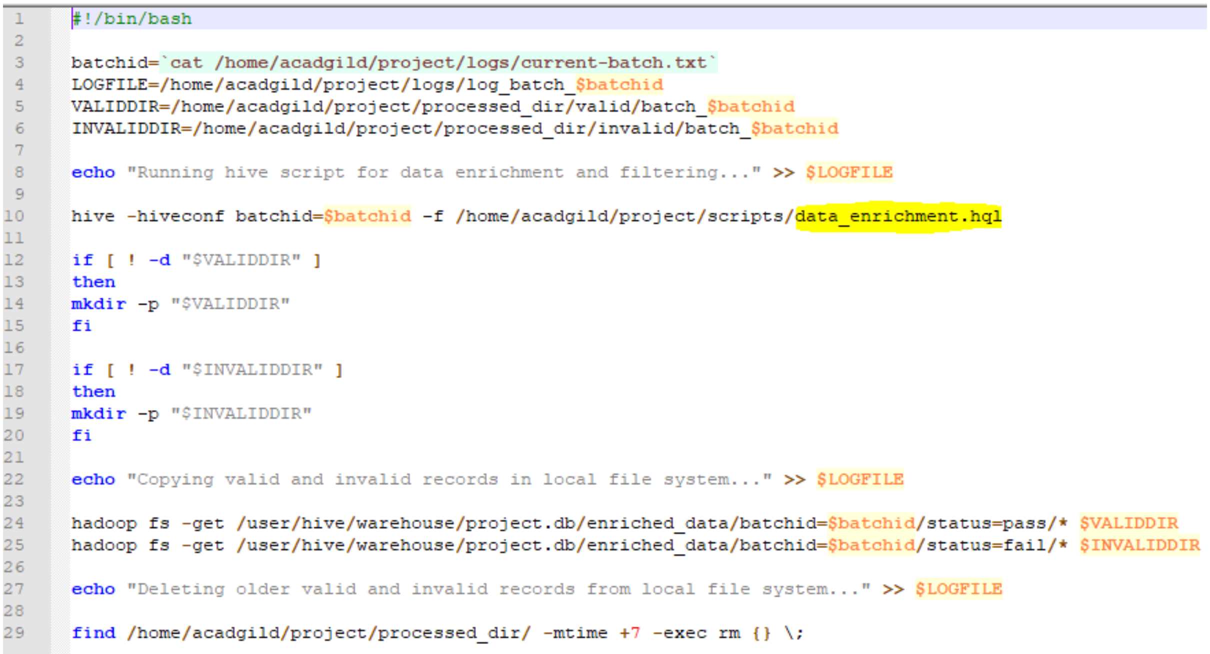
Rules for data enrichment -

- If any of like or dislike is **NULL** or **absent**, consider it as **0**.
- If fields like **Geo_cd** and **Artist_id** are NULL or absent, consult the lookup tables for fields

  Station_id and **Song_id** respectively to get the values of **Geo_cd** and **Artist_id.**
- If corresponding lookup entry is not found, consider that **record** to be **invalid**

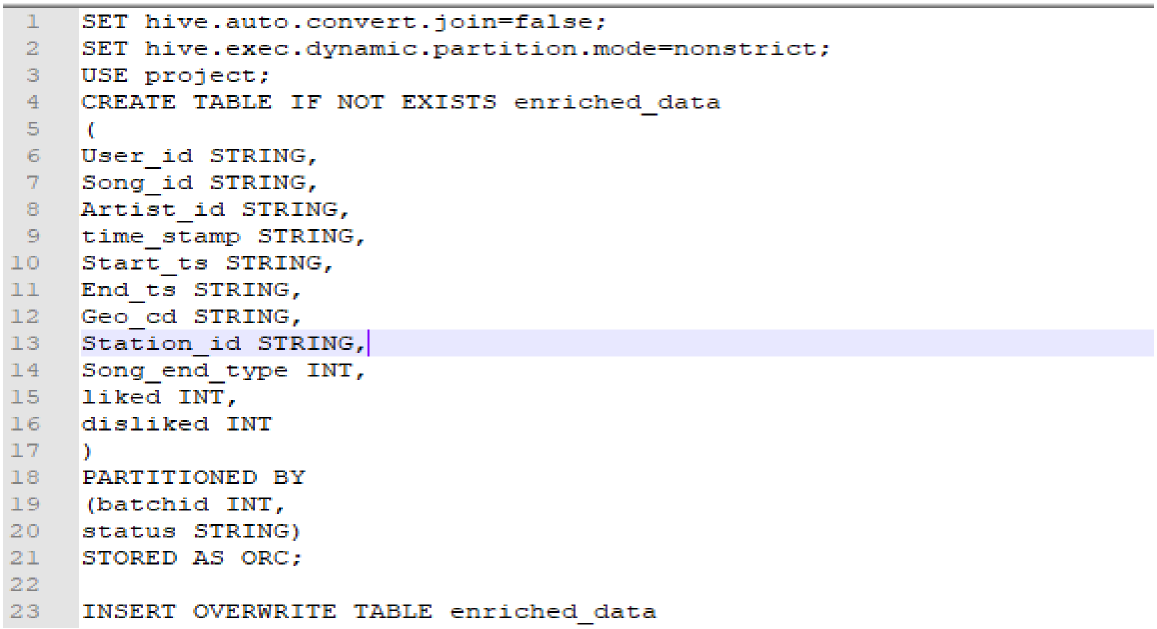
So based on the enrichment rules we will fill the null **geo_cd** and **artist_id** values with the help of corresponding lookup values in **song-artist-map** and **station-geo-map** tables in **Hive-Hbase** tables.

data_enrichment.sh

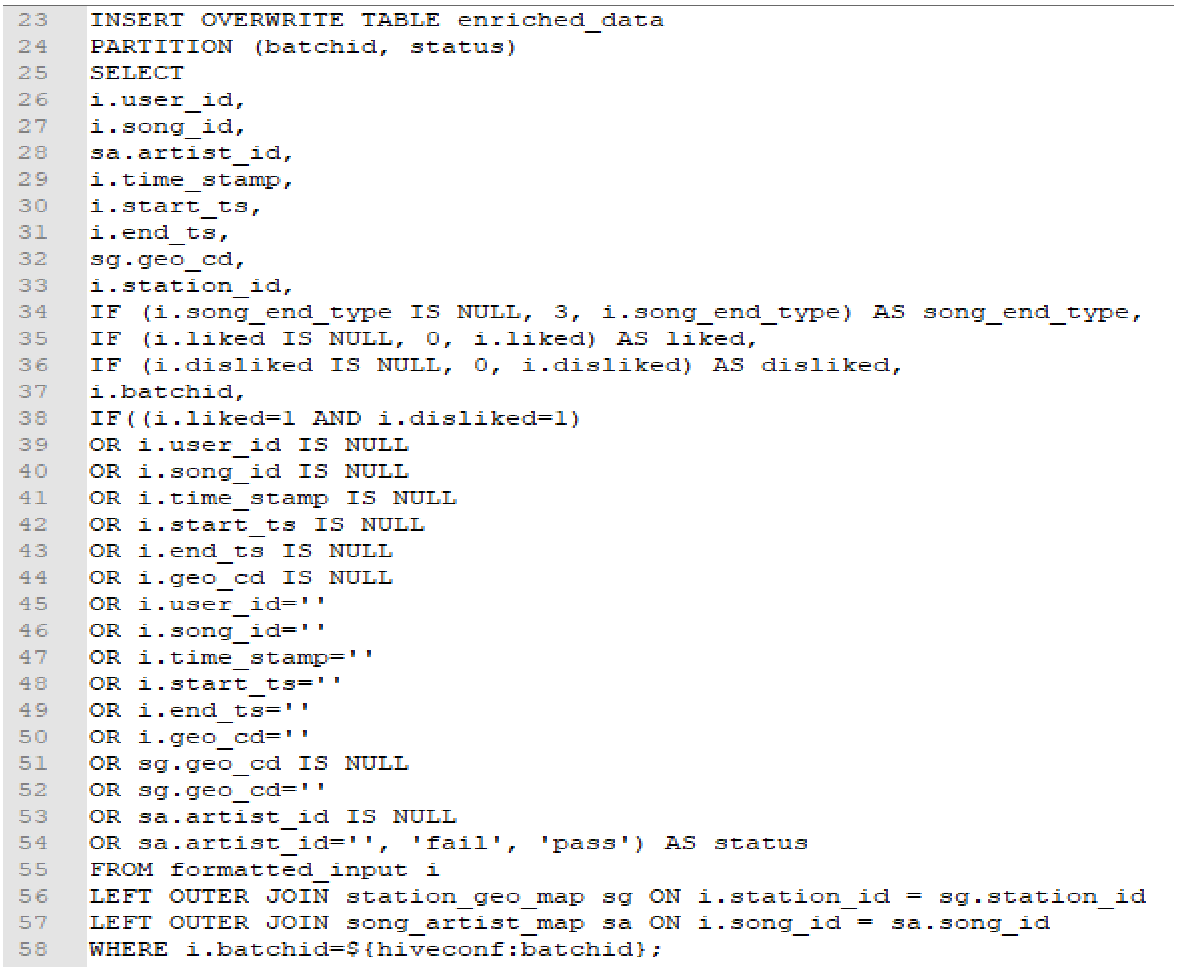


```bash
#!/bin/bash

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 {} \;
```

data_enrichment.hql

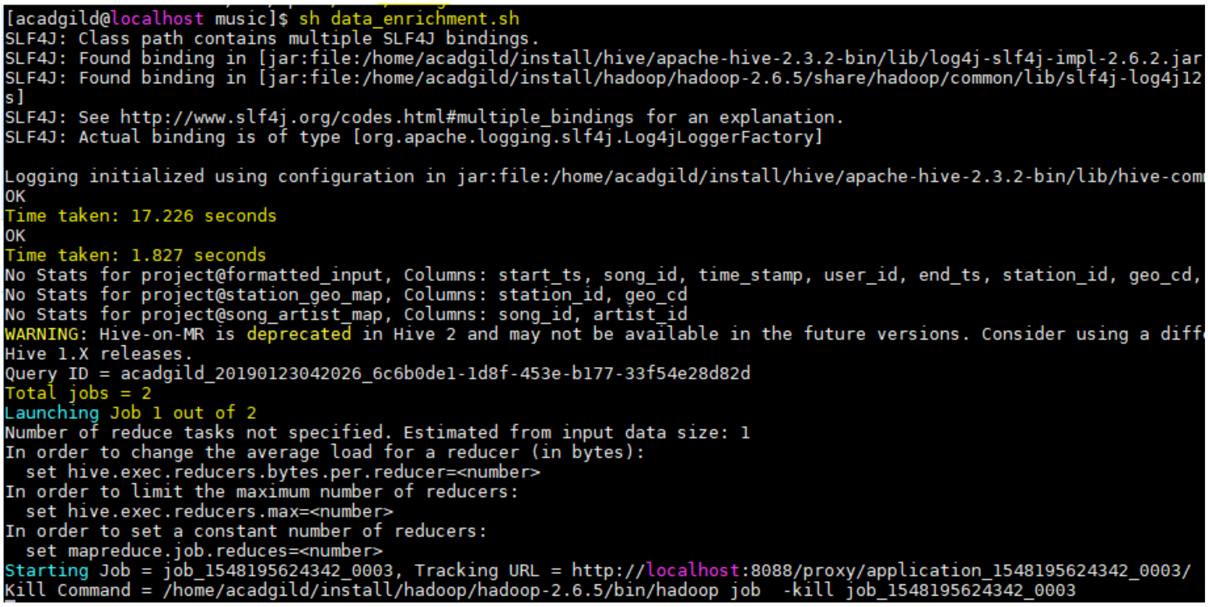


```
SET hive.auto.convert.join=false;
SET hive.exec.dynamic.partition.mode=nonstrict;
USE project;
CREATE TABLE IF NOT EXISTS enriched_data
(
User_id STRING,
Song_id STRING,
Artist_id STRING,
time_stamp STRING,
Start_ts STRING,
End_ts STRING,
Geo_cd STRING,
Station_id STRING,
Song_end_type INT,
liked INT,
disliked INT
)
PARTITIONED BY
(batchid INT,
status STRING)
STORED AS ORC;

INSERT OVERWRITE TABLE enriched_data
```

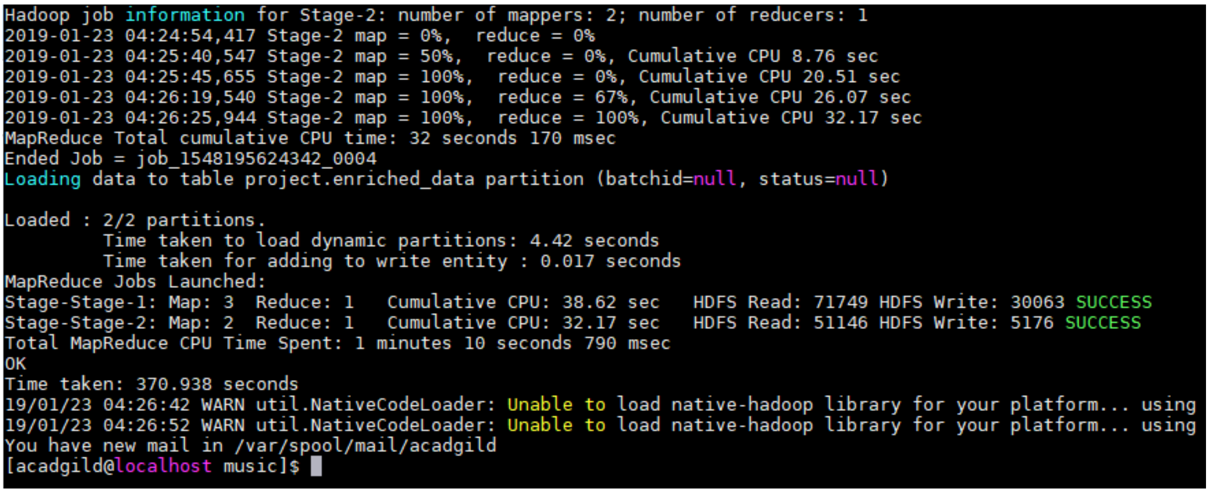


```
INSERT OVERWRITE TABLE enriched_data
PARTITION (batchid, status)
SELECT
i.user_id,
i.song_id,
sa.artist_id,
i.time_stamp,
i.start_ts,
i.end_ts,
sg.geo_cd,
i.station_id,
IF (i.song_end_type IS NULL, 3, i.song_end_type) AS song_end_type,
IF (i.liked IS NULL, 0, i.liked) AS liked,
IF (i.disliked IS NULL, 0, i.disliked) AS disliked,
i.batchid,
IF((i.liked=1 AND i.disliked=1)
OR i.user_id IS NULL
OR i.song_id IS NULL
OR i.time_stamp 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.time_stamp=''
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};
```

Run Script: *./data_enrichment.sh*

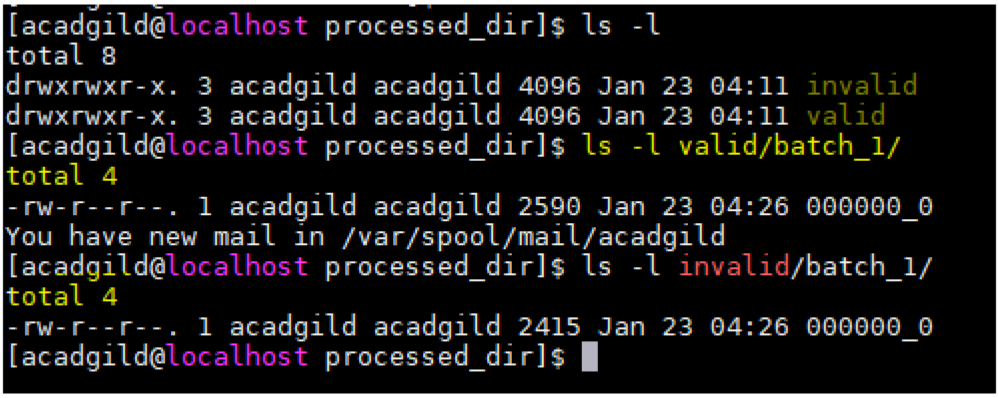


```
[acadgild@localhost music]$ sh data_enrichment.sh
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/acadgild/install/hive/apache-hive-2.3.2-bin/lib/log4j-slf4j-impl-2.6.2.jar
SLF4J: Found binding in [jar:file:/home/acadgild/install/hadoop/hadoop-2.6.5/share/hadoop/common/lib/slf4j-log4j12
s]
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-com
OK
Time taken: 17.226 seconds
OK
Time taken: 1.827 seconds
No Stats for project@formatted_input, Columns: start_ts, song_id, time_stamp, user_id, end_ts, station_id, geo_cd,
No Stats for project@station_geo_map, Columns: station_id, geo_cd
No Stats for project@song_artist_map, Columns: song_id, artist_id
WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a diff
Hive 1.X releases.
Query ID = acadgild_20190123042026_6c6b0de1-1d8f-453e-b177-33f54e28d82d
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_1548195624342_0003, Tracking URL = http://localhost:8088/proxy/application_1548195624342_0003/
Kill Command = /home/acadgild/install/hadoop/hadoop-2.6.5/bin/hadoop job  -kill job_1548195624342_0003
```



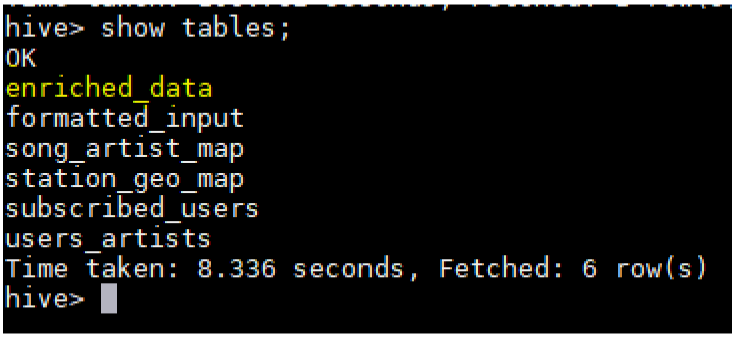
```
Hadoop job information for Stage-2: number of mappers: 2; number of reducers: 1
2019-01-23 04:24:54,417 Stage-2 map = 0%,  reduce = 0%
2019-01-23 04:25:40,547 Stage-2 map = 50%,  reduce = 0%, Cumulative CPU 8.76 sec
2019-01-23 04:25:45,655 Stage-2 map = 100%,  reduce = 0%, Cumulative CPU 20.51 sec
2019-01-23 04:26:19,540 Stage-2 map = 100%,  reduce = 67%, Cumulative CPU 26.07 sec
2019-01-23 04:26:25,944 Stage-2 map = 100%,  reduce = 100%, Cumulative CPU 32.17 sec
MapReduce Total cumulative CPU time: 32 seconds 170 msec
Ended Job = job_1548195624342_0004
Loading data to table project.enriched_data partition (batchid=null, status=null)

Loaded : 2/2 partitions.
         Time taken to load dynamic partitions: 4.42 seconds
         Time taken for adding to write entity : 0.017 seconds
MapReduce Jobs Launched:
Stage-Stage-1: Map: 3  Reduce: 1   Cumulative CPU: 38.62 sec   HDFS Read: 71749 HDFS Write: 30063 SUCCESS
Stage-Stage-2: Map: 2  Reduce: 1   Cumulative CPU: 32.17 sec   HDFS Read: 51146 HDFS Write: 5176 SUCCESS
Total MapReduce CPU Time Spent: 1 minutes 10 seconds 790 msec
OK
Time taken: 370.938 seconds
19/01/23 04:26:42 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using
19/01/23 04:26:52 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using
You have new mail in /var/spool/mail/acadgild
[acadgild@localhost music]$
```

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



```
[acadgild@localhost processed_dir]$ ls -l
total 8
drwxrwxr-x. 3 acadgild acadgild 4096 Jan 23 04:11 invalid
drwxrwxr-x. 3 acadgild acadgild 4096 Jan 23 04:11 valid
[acadgild@localhost processed_dir]$ ls -l valid/batch_1/
total 4
-rw-r--r--. 1 acadgild acadgild 2590 Jan 23 04:26 000000_0
You have new mail in /var/spool/mail/acadgild
[acadgild@localhost processed_dir]$ ls -l invalid/batch_1/
total 4
-rw-r--r--. 1 acadgild acadgild 2415 Jan 23 04:26 000000_0
[acadgild@localhost processed_dir]$
```

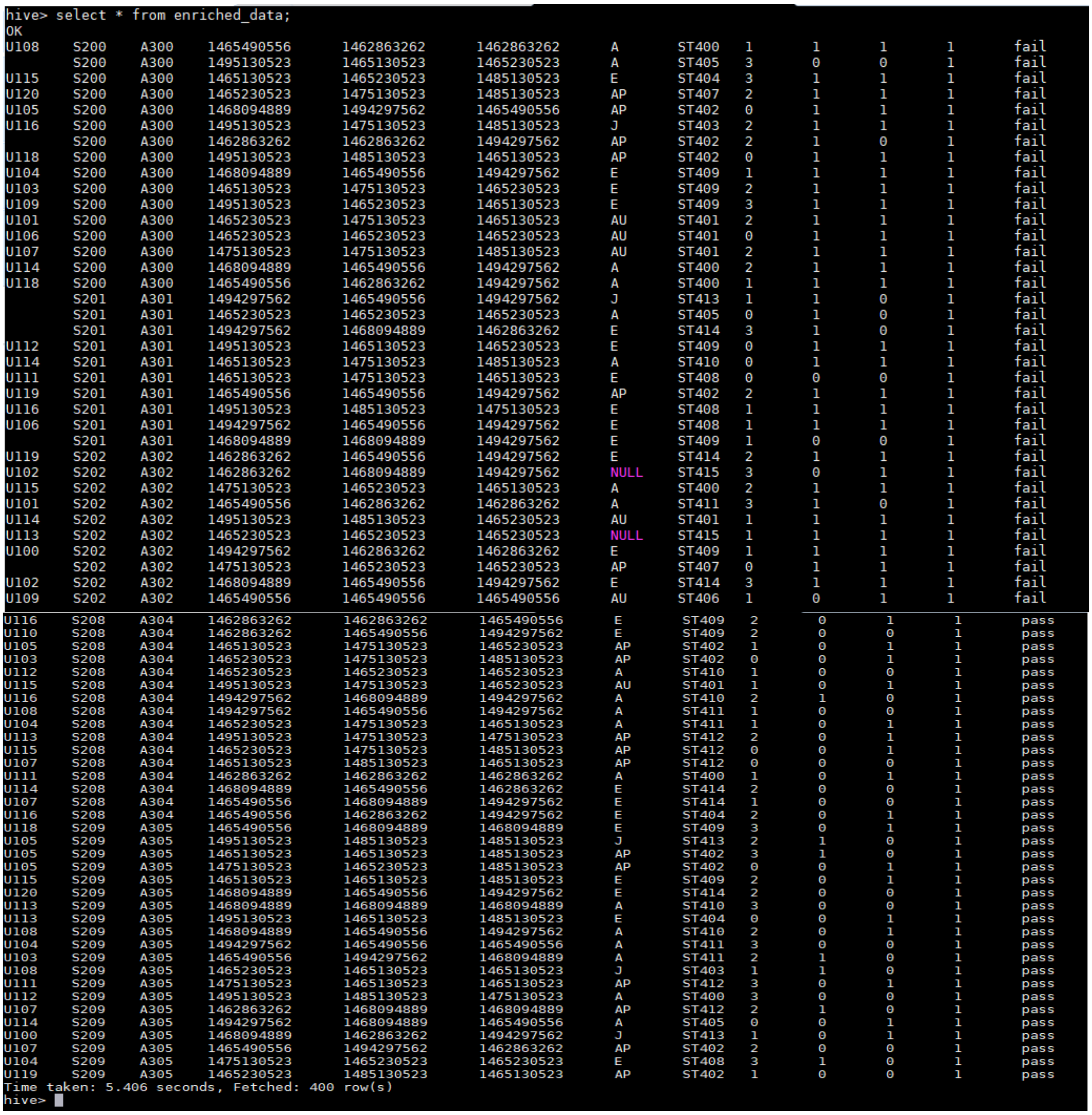
Now we can check whether the data properly loaded in the HIVE.



```
hive> show tables;
OK
enriched_data
formatted_input
song_artist_map
station_geo_map
subscribed_users
users_artists
Time taken: 8.336 seconds, Fetched: 6 row(s)
hive>
```

In the below screenshot we have data for **enriched_data** table where we filled the null values of **artist_id** and **geo_cd** of formatted input with the help of lookup tables.

select * from enriched_data;



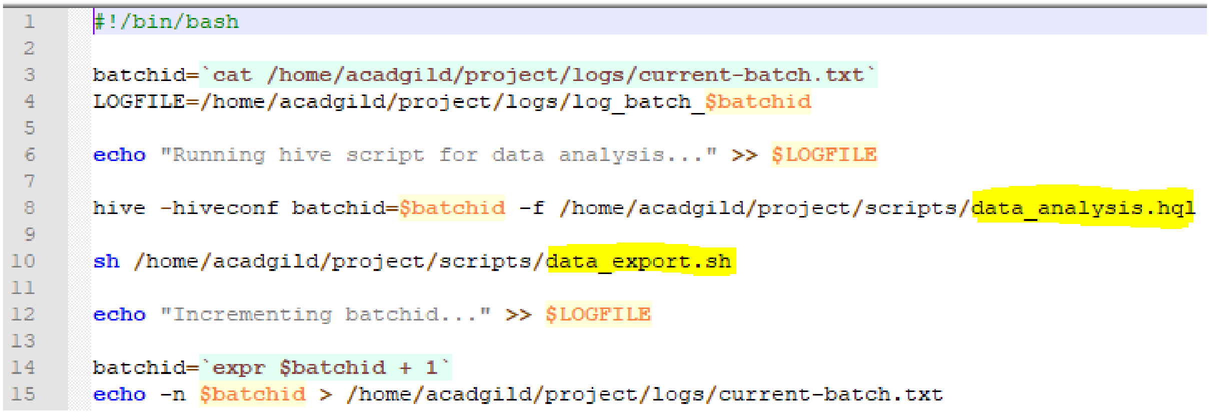
```
hive> select * from enriched_data;
OK
U108      S200      A300      1465490556          1462863262          1462863262          A         ST400 1         1         1         1         fail
          S200      A300      1495130523          1465130523          1465230523          A         ST405 3         0         0         1         fail
U115      S200      A300      1465130523          1465230523          1485130523          E         ST404 3         1         1         1         fail
U120      S200      A300      1465230523          1475130523          1485130523          AP        ST407 2         1         1         1         fail
U105      S200      A300      1468094889          1494297562          1465490556          AP        ST402 0         1         1         1         fail
U116      S200      A300      1495130523          1475130523          1485130523          J         ST403 2         1         1         1         fail
          S200      A300      1462863262          1462863262          1494297562          AP        ST402 2         1         0         1         fail
U118      S200      A300      1495130523          1485130523          1465130523          AP        ST402 0         1         1         1         fail
U104      S200      A300      1468094889          1465490556          1494297562          E         ST409 1         1         1         1         fail
U103      S200      A300      1465130523          1475130523          1465230523          E         ST409 2         1         1         1         fail
U109      S200      A300      1495130523          1465230523          1465130523          E         ST409 3         1         1         1         fail
U101      S200      A300      1465230523          1475130523          1465130523          AU        ST401 2         1         1         1         fail
U106      S200      A300      1465230523          1465230523          1465230523          AU        ST401 0         1         1         1         fail
U107      S200      A300      1475130523          1475130523          1485130523          AU        ST401 2         1         1         1         fail
U114      S200      A300      1468094889          1465490556          1494297562          A         ST400 2         1         1         1         fail
U118      S200      A300      1465490556          1462863262          1494297562          A         ST400 1         1         1         1         fail
          S201      A301      1494297562          1465490556          1494297562          J         ST413 1         1         0         1         fail
          S201      A301      1465230523          1465230523          1465230523          A         ST405 0         1         0         1         fail
          S201      A301      1494297562          1468094889          1462863262          E         ST414 3         1         0         1         fail
U112      S201      A301      1495130523          1465130523          1465230523          E         ST409 0         1         1         1         fail
U114      S201      A301      1465130523          1475130523          1485130523          A         ST410 0         1         1         1         fail
U111      S201      A301      1465130523          1475130523          1465130523          E         ST408 0         0         0         1         fail
U119      S201      A301      1465490556          1465490556          1494297562          AP        ST402 2         1         1         1         fail
U116      S201      A301      1495130523          1485130523          1475130523          E         ST408 1         1         1         1         fail
U106      S201      A301      1494297562          1465490556          1494297562          E         ST408 1         1         1         1         fail
          S201      A301      1468094889          1468094889          1494297562          E         ST409 1         0         0         1         fail
U119      S202      A302      1462863262          1465490556          1494297562          E         ST414 2         1         1         1         fail
U102      S202      A302      1462863262          1468094889          1494297562          NULL      ST415 3         0         1         1         fail
U115      S202      A302      1475130523          1465230523          1465130523          A         ST400 2         1         1         1         fail
U101      S202      A302      1465490556          1462863262          1462863262          A         ST411 3         1         0         1         fail
U114      S202      A302      1495130523          1485130523          1465230523          AU        ST401 1         1         1         1         fail
U113      S202      A302      1465230523          1465230523          1465230523          NULL      ST415 1         1         1         1         fail
U100      S202      A302      1494297562          1462863262          1462863262          E         ST409 1         1         1         1         fail
          S202      A302      1475130523          1465230523          1465230523          AP        ST407 0         1         1         1         fail
U102      S202      A302      1468094889          1465490556          1494297562          E         ST414 3         1         1         1         fail
U109      S202      A302      1465490556          1465490556          1465490556          AU        ST406 1         0         1         1         fail
U116      S208      A304      1462863262          1462863262          1465490556          E         ST409 2         0         1         1         pass
U110      S208      A304      1462863262          1465490556          1494297562          E         ST409 2         0         0         1         pass
U105      S208      A304      1465130523          1475130523          1465230523          AP        ST402 1         0         1         1         pass
U103      S208      A304      1465230523          1475130523          1485130523          AP        ST402 0         0         1         1         pass
U112      S208      A304      1465230523          1465230523          1465230523          A         ST410 1         0         0         1         pass
U115      S208      A304      1495130523          1475130523          1465230523          AU        ST401 1         0         1         1         pass
U116      S208      A304      1494297562          1468094889          1494297562          A         ST410 2         1         0         1         pass
U108      S208      A304      1494297562          1465490556          1494297562          A         ST411 1         0         0         1         pass
U104      S208      A304      1465230523          1475130523          1465130523          A         ST411 1         0         1         1         pass
U113      S208      A304      1495130523          1475130523          1475130523          AP        ST412 2         0         1         1         pass
U115      S208      A304      1465230523          1475130523          1485130523          AP        ST412 0         0         1         1         pass
U107      S208      A304      1465130523          1485130523          1465130523          AP        ST412 0         0         0         1         pass
U111      S208      A304      1462863262          1462863262          1462863262          A         ST400 1         0         1         1         pass
U114      S208      A304      1468094889          1465490556          1462863262          E         ST414 2         0         0         1         pass
U107      S208      A304      1465490556          1468094889          1494297562          E         ST414 1         0         0         1         pass
U116      S208      A304      1465490556          1462863262          1494297562          E         ST404 2         0         1         1         pass
U118      S209      A305      1465490556          1468094889          1468094889          E         ST409 3         0         1         1         pass
U105      S209      A305      1495130523          1485130523          1485130523          J         ST413 2         1         0         1         pass
U105      S209      A305      1465130523          1465130523          1485130523          AP        ST402 3         1         0         1         pass
U105      S209      A305      1475130523          1465230523          1485130523          AP        ST402 0         0         1         1         pass
U115      S209      A305      1465130523          1465130523          1485130523          E         ST409 2         0         1         1         pass
U120      S209      A305      1468094889          1465490556          1494297562          E         ST414 2         0         0         1         pass
U113      S209      A305      1468094889          1468094889          1468094889          A         ST410 3         0         0         1         pass
U113      S209      A305      1495130523          1465130523          1485130523          E         ST404 0         0         1         1         pass
U108      S209      A305      1468094889          1465490556          1494297562          A         ST410 2         0         1         1         pass
U104      S209      A305      1494297562          1465490556          1465490556          A         ST411 3         0         0         1         pass
U103      S209      A305      1465490556          1494297562          1468094889          A         ST411 2         1         0         1         pass
U108      S209      A305      1465230523          1465130523          1465130523          J         ST403 1         1         0         1         pass
U111      S209      A305      1475130523          1465130523          1465130523          AP        ST412 3         0         1         1         pass
U112      S209      A305      1495130523          1485130523          1475130523          A         ST400 3         0         0         1         pass
U107      S209      A305      1462863262          1468094889          1468094889          AP        ST412 2         1         0         1         pass
U114      S209      A305      1494297562          1468094889          1465490556          A         ST405 0         0         1         1         pass
U100      S209      A305      1468094889          1462863262          1494297562          J         ST413 1         0         1         1         pass
U107      S209      A305      1465490556          1494297562          1462863262          AP        ST402 2         0         0         1         pass
U104      S209      A305      1475130523          1465230523          1465230523          E         ST408 3         1         0         1         pass
U119      S209      A305      1465230523          1485130523          1465130523          AP        ST402 1         0         0         1         pass
Time taken: 5.406 seconds, Fetched: 400 row(s)
hive>
```

By applying the mentioned rules, we have successfully accomplished Data enrichment and Filtering stage.

4.4 Stage – 4 – Data Analysis

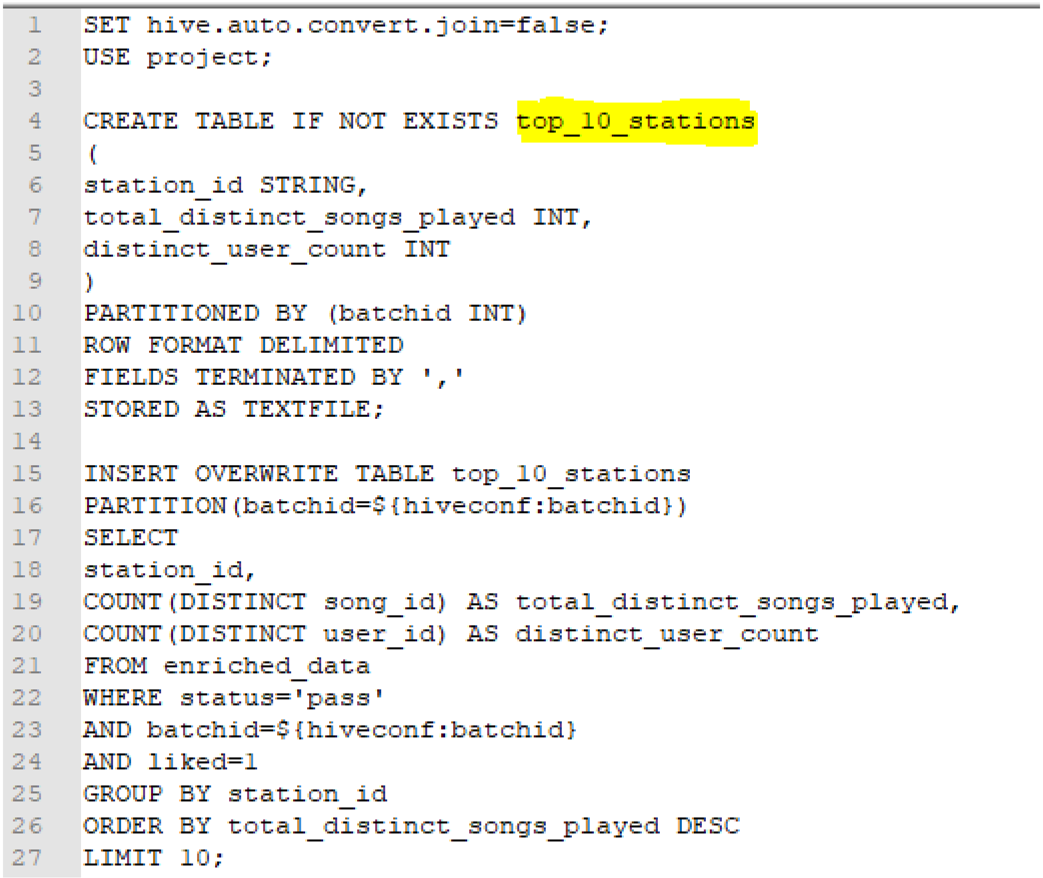
In this stage we will do analysis on enriched data.

data_analysis.sh

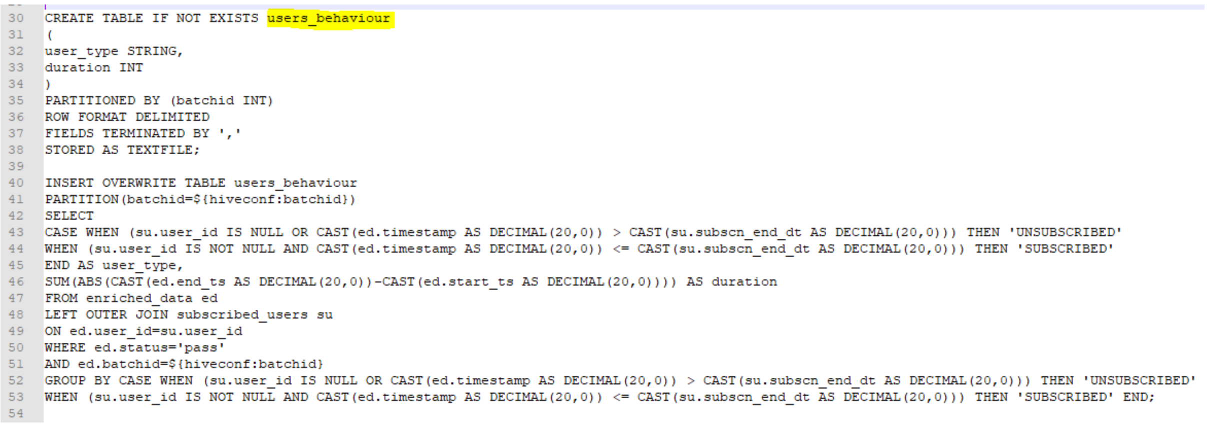


```
#!/bin/bash

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

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

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

sh /home/acadgild/project/scripts/data_export.sh

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

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

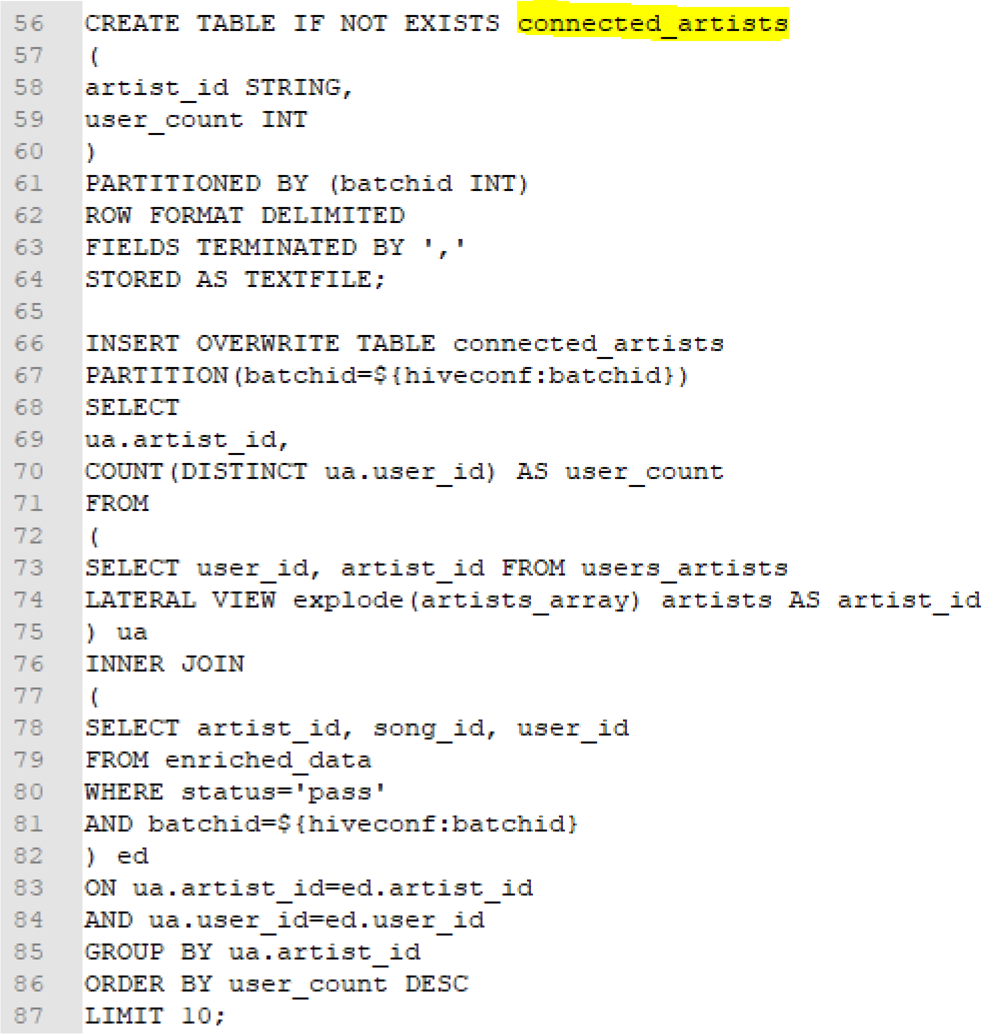
data_analysis.hql



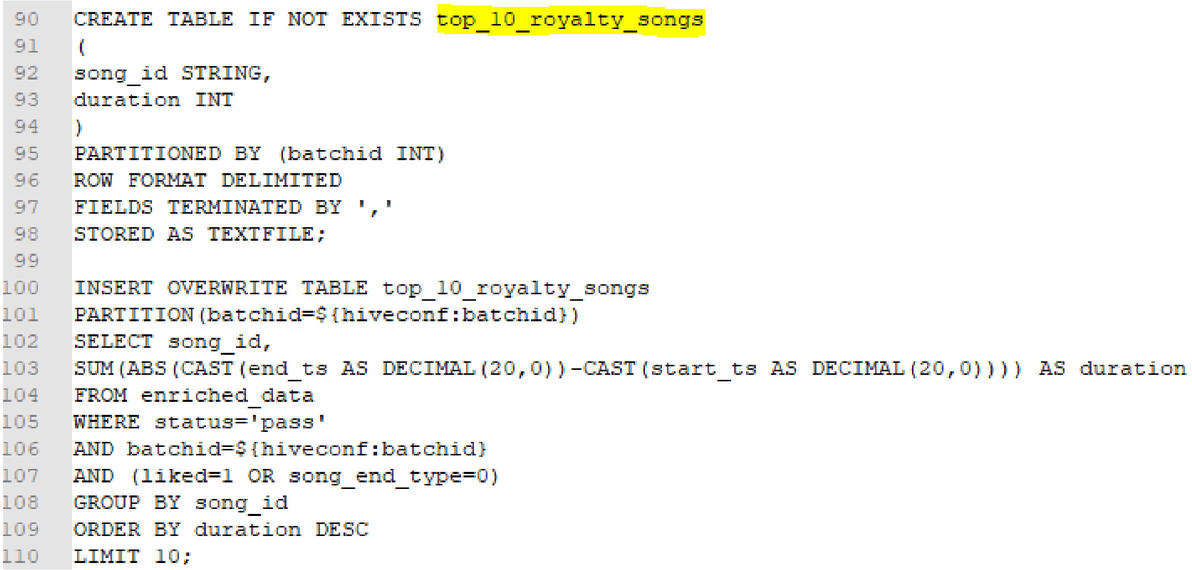
```
SET hive.auto.convert.join=false;
USE project;

CREATE TABLE IF NOT EXISTS top_10_stations
(
station_id STRING,
total_distinct_songs_played INT,
distinct_user_count INT
)
PARTITIONED BY (batchid INT)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
STORED AS TEXTFILE;

INSERT OVERWRITE TABLE top_10_stations
PARTITION(batchid=${hiveconf:batchid})
SELECT
station_id,
COUNT(DISTINCT song_id) AS total_distinct_songs_played,
COUNT(DISTINCT user_id) AS distinct_user_count
FROM enriched_data
WHERE status='pass'
AND batchid=${hiveconf:batchid}
AND liked=1
GROUP BY station_id
ORDER BY total_distinct_songs_played DESC
LIMIT 10;
```



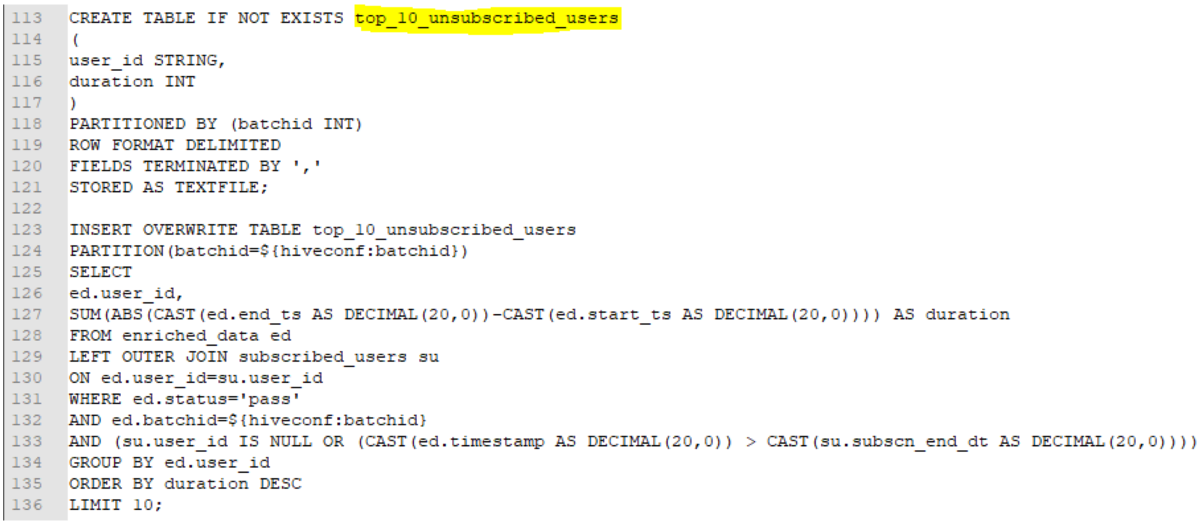
```
CREATE TABLE IF NOT EXISTS users_behaviour
(
user_type STRING,
duration INT
)
PARTITIONED BY (batchid INT)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
STORED AS TEXTFILE;

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



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

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

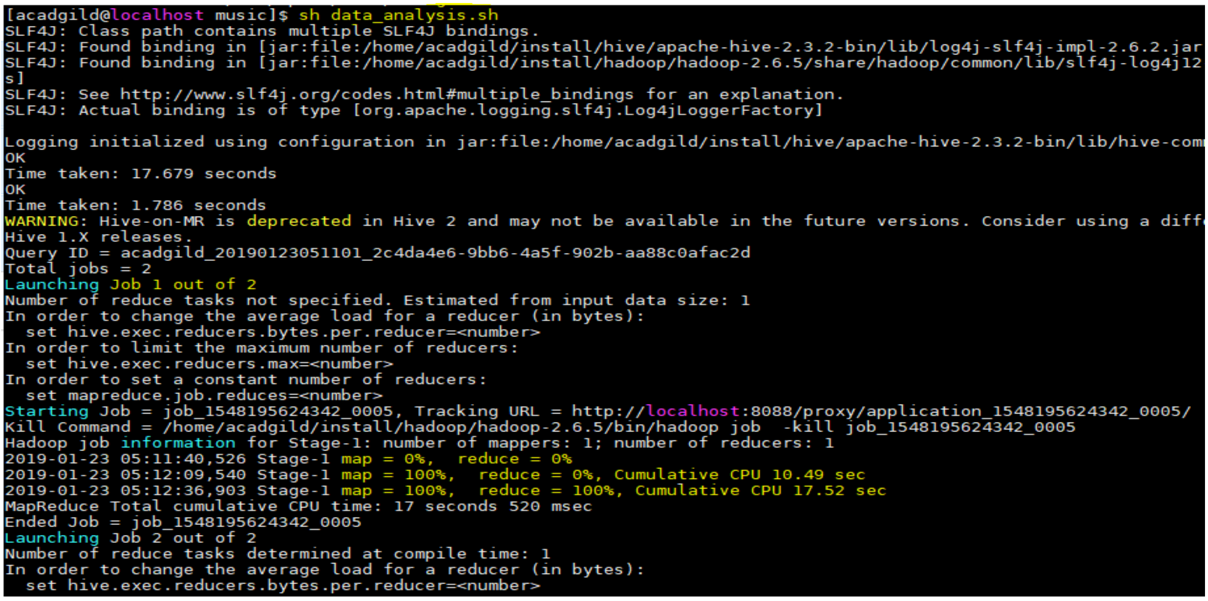


```
CREATE TABLE IF NOT EXISTS top_10_royalty_songs
(
song_id STRING,
duration INT
)
PARTITIONED BY (batchid INT)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
STORED AS TEXTFILE;

INSERT OVERWRITE TABLE top_10_royalty_songs
PARTITION(batchid=${hiveconf:batchid})
SELECT song_id,
SUM(ABS(CAST(end_ts AS DECIMAL(20,0))-CAST(start_ts AS DECIMAL(20,0)))) AS duration
FROM enriched_data
WHERE status='pass'
AND batchid=${hiveconf:batchid}
AND (liked=1 OR song_end_type=0)
GROUP BY song_id
ORDER BY duration DESC
LIMIT 10;
```



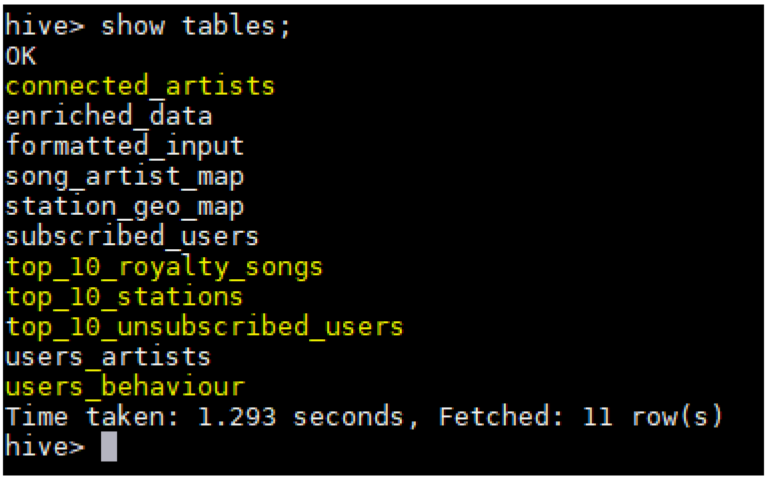
```
CREATE TABLE IF NOT EXISTS top_10_unsubscribed_users
(
user_id STRING,
duration INT
)
PARTITIONED BY (batchid INT)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
STORED AS TEXTFILE;

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

Run script*: ./data_analysis.sh*



```
[acadgild@localhost music]$ sh data_analysis.sh
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
SLF4J: Found binding in [jar:file:/home/acadgild/install/hadoop/hadoop-2.6.5/share/hadoop/common/lib/slf4j-log4j12
s]
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-com
OK
Time taken: 17.679 seconds
OK
Time taken: 1.786 seconds
WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a diff
Hive 1.X releases.
Query ID = acadgild_20190123051101_2c4da4e6-9bb6-4a5f-902b-aa88c0afac2d
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_1548195624342_0005, Tracking URL = http://localhost:8088/proxy/application_1548195624342_0005/
Kill Command = /home/acadgild/install/hadoop/hadoop-2.6.5/bin/hadoop job  -kill job_1548195624342_0005
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2019-01-23 05:11:40,526 Stage-1 map = 0%,  reduce = 0%
2019-01-23 05:12:09,540 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 10.49 sec
2019-01-23 05:12:36,903 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 17.52 sec
MapReduce Total cumulative CPU time: 17 seconds 520 msec
Ended Job = job_1548195624342_0005
Launching Job 2 out of 2
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
```

The tables have been created in the Hive,

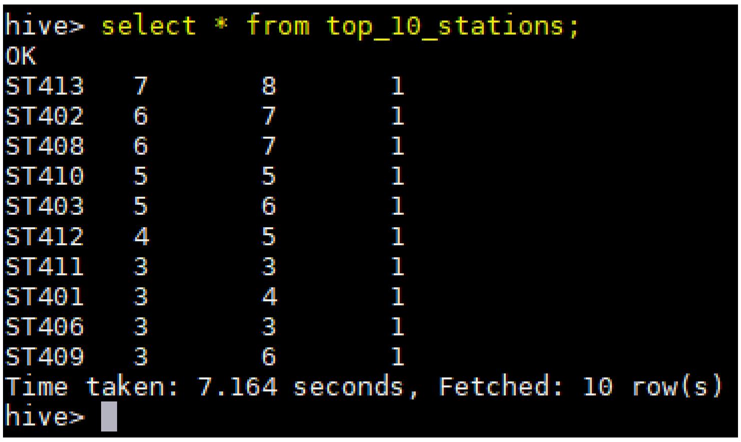


```
hive> show tables;
OK
connected_artists
enriched_data
formatted_input
song_artist_map
station_geo_map
subscribed_users
top_10_royalty_songs
top_10_stations
top_10_unsubscribed_users
users_artists
users_behaviour
Time taken: 1.293 seconds, Fetched: 11 row(s)
hive>
```

Problem statement 1:

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

select * from top_10_stations;

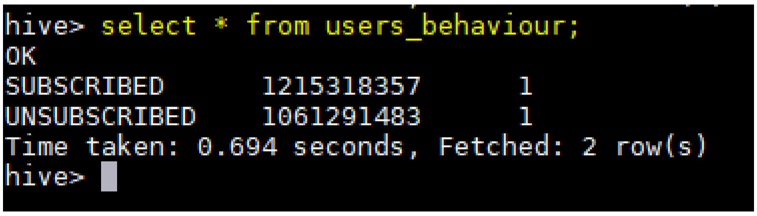


```
hive> select * from top_10_stations;
OK
ST413   7       8       1
ST402   6       7       1
ST408   6       7       1
ST410   5       5       1
ST403   5       6       1
ST412   4       5       1
ST411   3       3       1
ST401   3       4       1
ST406   3       3       1
ST409   3       6       1
Time taken: 7.164 seconds, Fetched: 10 row(s)
hive>
```

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.

select * from users_behaviour;

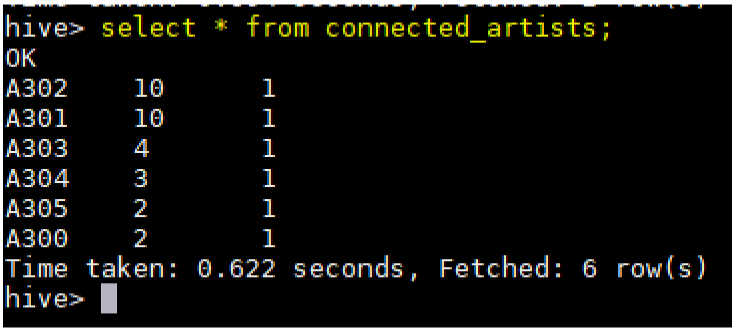


```
hive> select * from users_behaviour;
OK
SUBSCRIBED      1215318357      1
UNSUBSCRIBED    1061291483      1
Time taken: 0.694 seconds, Fetched: 2 row(s)
hive>
```

Problem statement 3:

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

select * from connected_artists;

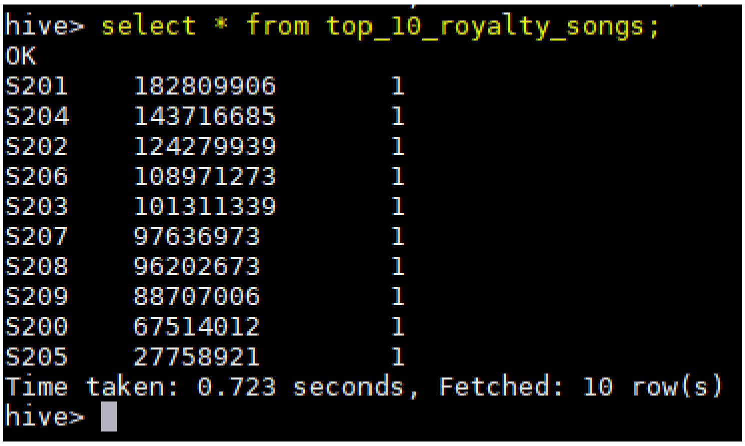


```
hive> select * from connected_artists;
OK
A302    10      1
A301    10      1
A303    4       1
A304    3       1
A305    2       1
A300    2       1
Time taken: 0.622 seconds, Fetched: 6 row(s)
hive>
```

Problem statement 4:

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

select * from top_10_royalty_songs;

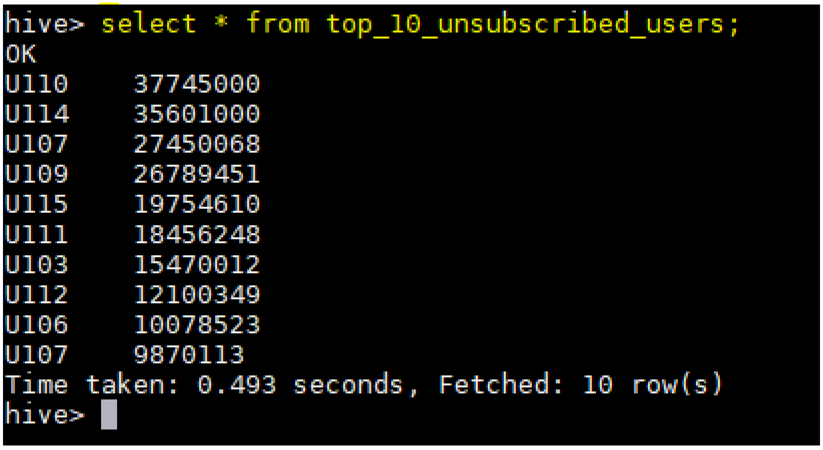


```
hive> select * from top_10_royalty_songs;
OK
S201    182809906       1
S204    143716685       1
S202    124279939       1
S206    108971273       1
S203    101311339       1
S207    97636973        1
S208    96202673        1
S209    88707006        1
S200    67514012        1
S205    27758921        1
Time taken: 0.723 seconds, Fetched: 10 row(s)
hive>
```

Problem statement 5:

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

select * from top_10_unsubscribed_users;

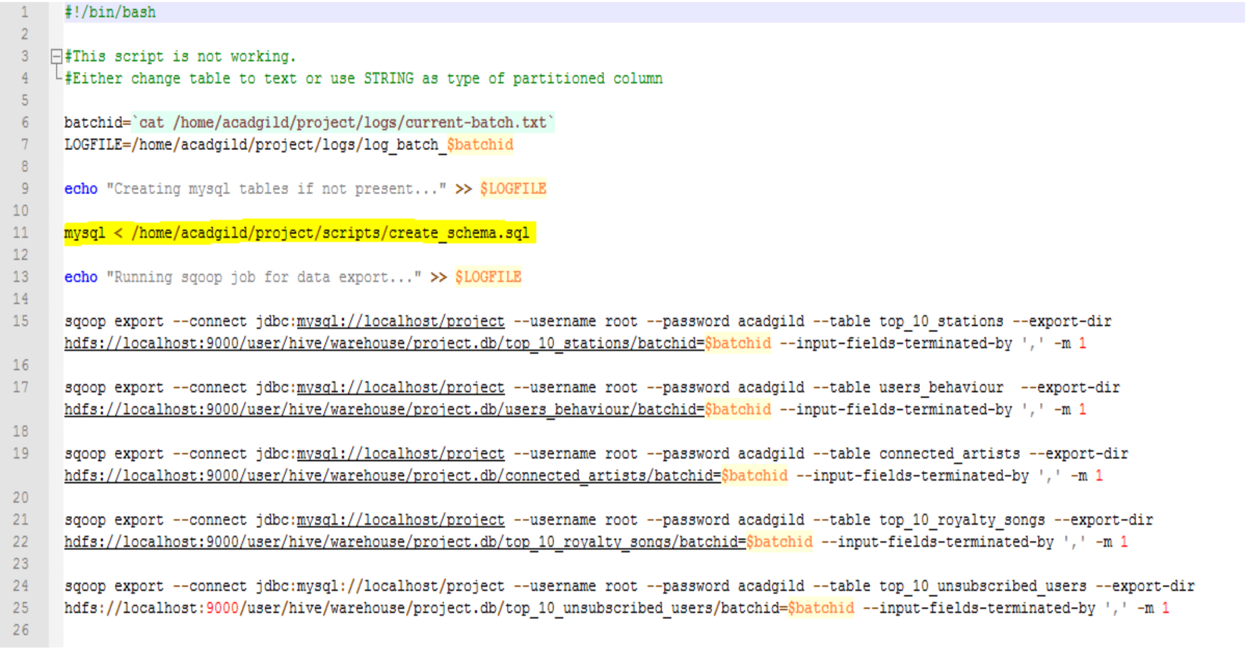


```
hive> select * from top_10_unsubscribed_users;
OK
U110    37745000
U114    35601000
U107    27450068
U109    26789451
U115    19754610
U111    18456248
U103    15470012
U112    12100349
U106    10078523
U107    9870113
Time taken: 0.493 seconds, Fetched: 10 row(s)
hive>
```

Now, we need to export all the data to the **MYSQL** using sqoop, run the script **data_export.sh**

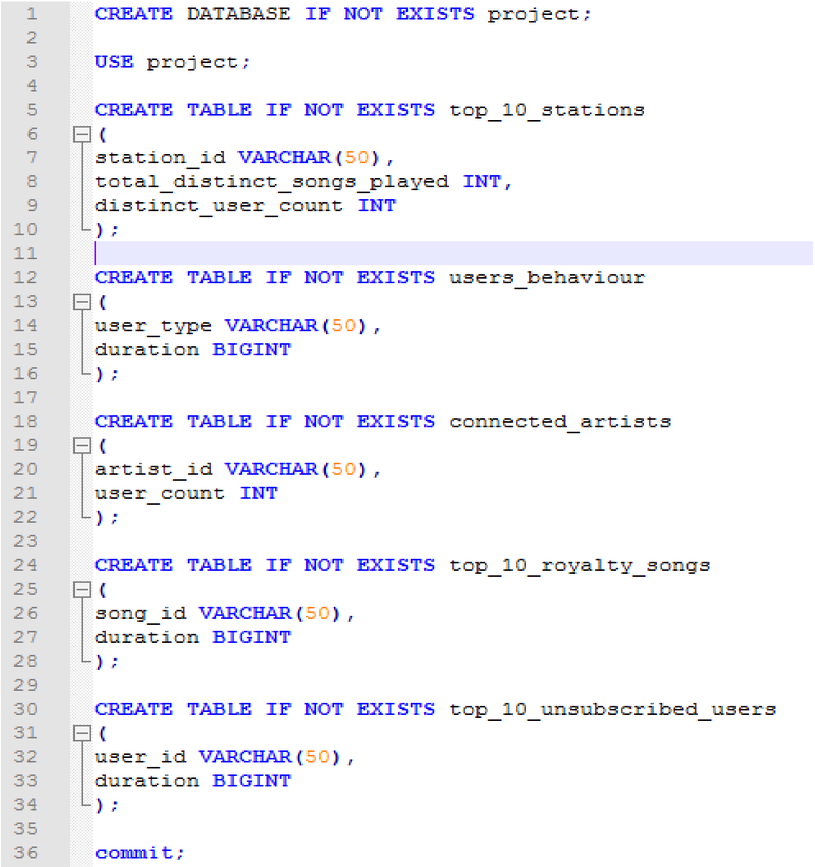
4.5 Stage – 5 – Data Storage in MYSQL

Using the bash file shown below, **data_export.sh** we are going to export the data from the HIVE tables into MYSQL using **SQOOP** export.



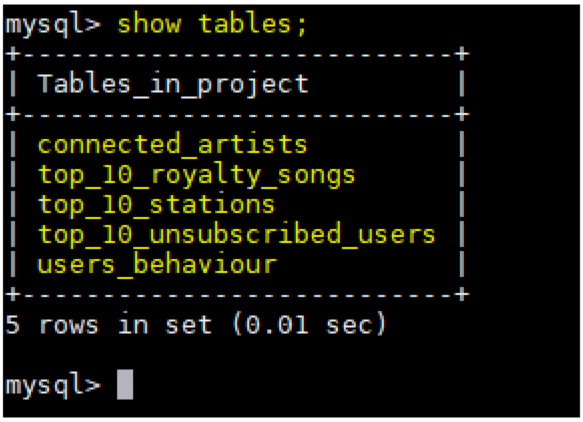
```
#!/bin/bash

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

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

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

mysql < /home/acadgild/project/scripts/create_schema.sql

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

sqoop export --connect jdbc:mysql://localhost/project --username root --password acadgild --table top_10_stations --export-dir
hdfs://localhost:9000/user/hive/warehouse/project.db/top_10_stations/batchid=$batchid --input-fields-terminated-by ',' -m 1

sqoop export --connect jdbc:mysql://localhost/project --username root --password acadgild --table users_behaviour  --export-dir
hdfs://localhost:9000/user/hive/warehouse/project.db/users_behaviour/batchid=$batchid --input-fields-terminated-by ',' -m 1

sqoop export --connect jdbc:mysql://localhost/project --username root --password acadgild --table connected_artists --export-dir
hdfs://localhost:9000/user/hive/warehouse/project.db/connected_artists/batchid=$batchid --input-fields-terminated-by ',' -m 1

sqoop export --connect jdbc:mysql://localhost/project --username root --password acadgild --table top_10_royalty_songs --export-dir
hdfs://localhost:9000/user/hive/warehouse/project.db/top_10_royalty_songs/batchid=$batchid --input-fields-terminated-by ',' -m 1

sqoop export --connect jdbc:mysql://localhost/project --username root --password acadgild --table top_10_unsubscribed_users --export-dir
hdfs://localhost:9000/user/hive/warehouse/project.db/top_10_unsubscribed_users/batchid=$batchid --input-fields-terminated-by ',' -m 1

```

***create_schema.sql* –** The below schema will create the database and tables in the MySQL.

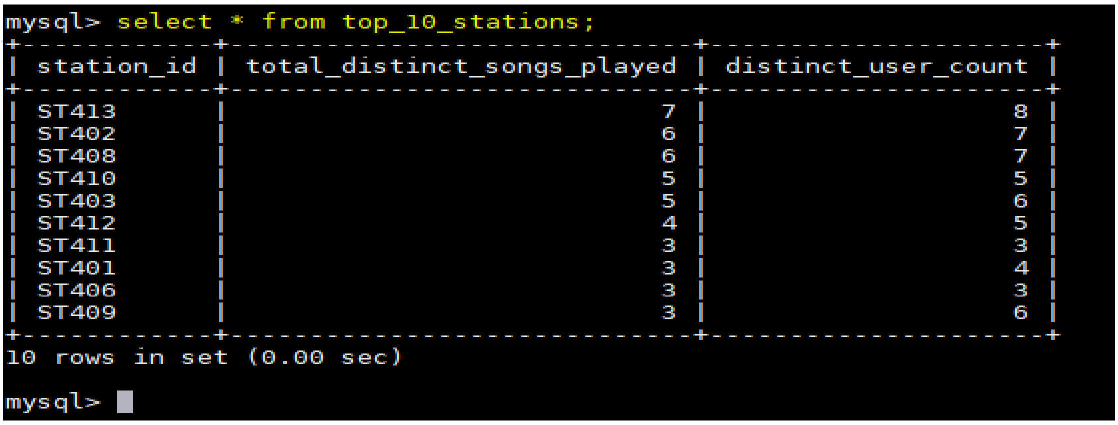


```
CREATE DATABASE IF NOT EXISTS project;

USE project;

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

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

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

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

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

commit;
```

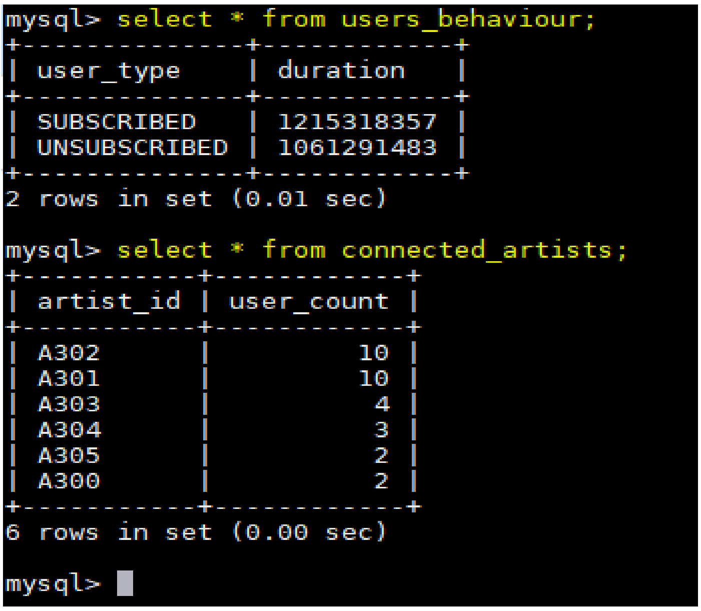
The database ***project*** has been exported from **HIVE** and the below screenshot shows the exported data to **MYSQL**.



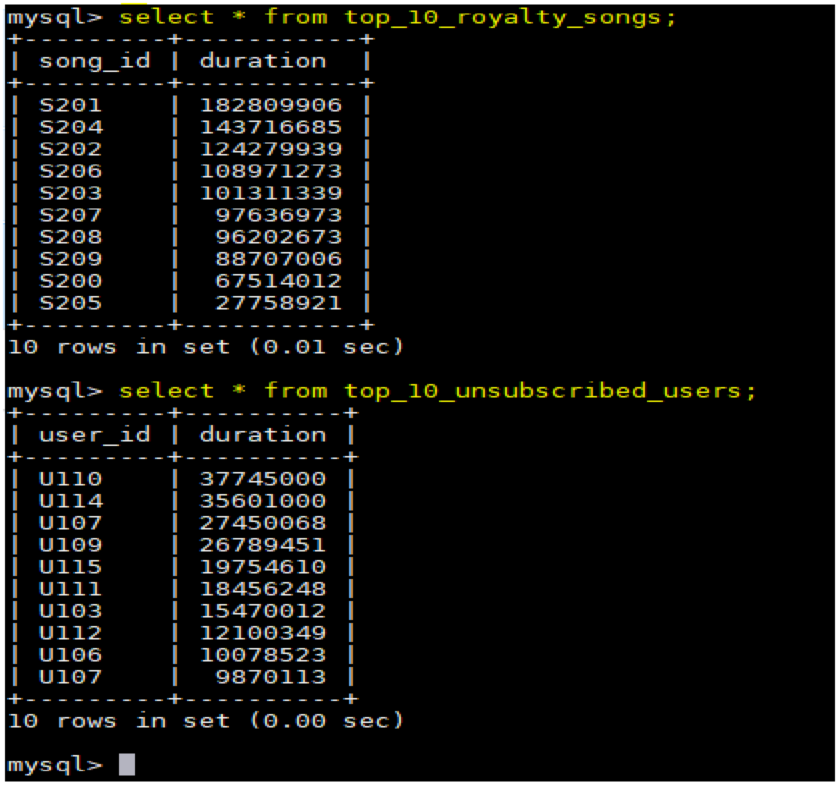
```
mysql> show tables;
+---------------------------+
| Tables_in_project         |
+---------------------------+
| connected_artists         |
| top_10_royalty_songs      |
| top_10_stations           |
| top_10_unsubscribed_users |
| users_behaviour           |
+---------------------------+
5 rows in set (0.01 sec)

mysql>
```



```
mysql> select * from top_10_stations;
+------------+-----------------------------+---------------------+
| station_id | total_distinct_songs_played | distinct_user_count |
+------------+-----------------------------+---------------------+
| ST413      |                           7 |                   8 |
| ST402      |                           6 |                   7 |
| ST408      |                           6 |                   7 |
| ST410      |                           5 |                   5 |
| ST403      |                           5 |                   6 |
| ST412      |                           4 |                   5 |
| ST411      |                           3 |                   3 |
| ST401      |                           3 |                   4 |
| ST406      |                           3 |                   3 |
| ST409      |                           3 |                   6 |
+------------+-----------------------------+---------------------+
10 rows in set (0.00 sec)

mysql> 
```



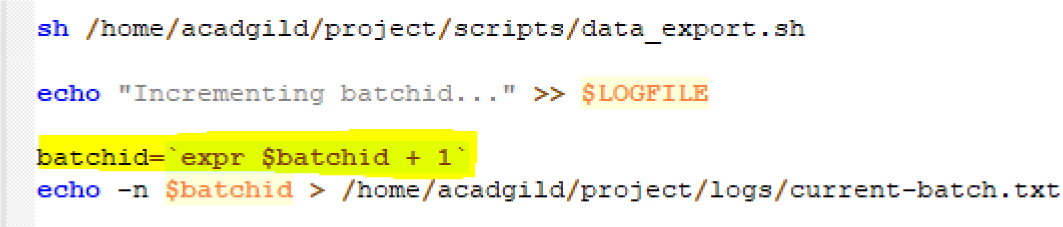
```
mysql> select * from users_behaviour;
+--------------+------------+
| user_type    | duration   |
+--------------+------------+
| SUBSCRIBED   | 1215318357 |
| UNSUBSCRIBED | 1061291483 |
+--------------+------------+
2 rows in set (0.01 sec)

mysql> select * from connected_artists;
+-----------+------------+
| artist_id | user_count |
+-----------+------------+
| A302      |         10 |
| A301      |         10 |
| A303      |          4 |
| A304      |          3 |
| A305      |          2 |
| A300      |          2 |
+-----------+------------+
6 rows in set (0.00 sec)

mysql> 
```



```
mysql> select * from top_10_royalty_songs;
+---------+-----------+
| song_id | duration  |
+---------+-----------+
| S201    | 182809906 |
| S204    | 143716685 |
| S202    | 124279939 |
| S206    | 108971273 |
| S203    | 101311339 |
| S207    |  97636973 |
| S208    |  96202673 |
| S209    |  88707006 |
| S200    |  67514012 |
| S205    |  27758921 |
+---------+-----------+
10 rows in set (0.01 sec)

mysql> select * from top_10_unsubscribed_users;
+---------+----------+
| user_id | duration |
+---------+----------+
| U110    | 37745000 |
| U114    | 35601000 |
| U107    | 27450068 |
| U109    | 26789451 |
| U115    | 19754610 |
| U111    | 18456248 |
| U103    | 15470012 |
| U112    | 12100349 |
| U106    | 10078523 |
| U107    |  9870113 |
+---------+----------+
10 rows in set (0.00 sec)

mysql> 
```

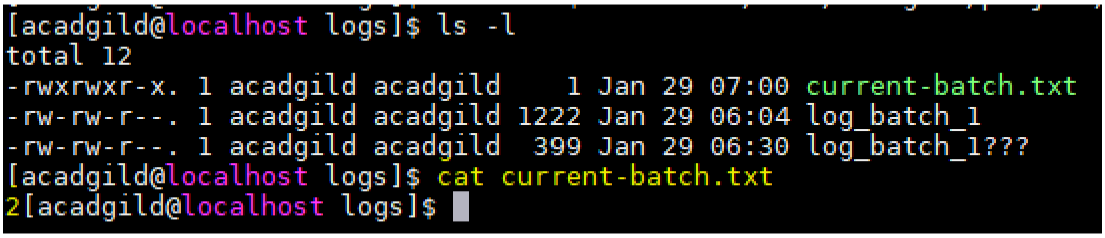
JOB SCHEDULING -

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



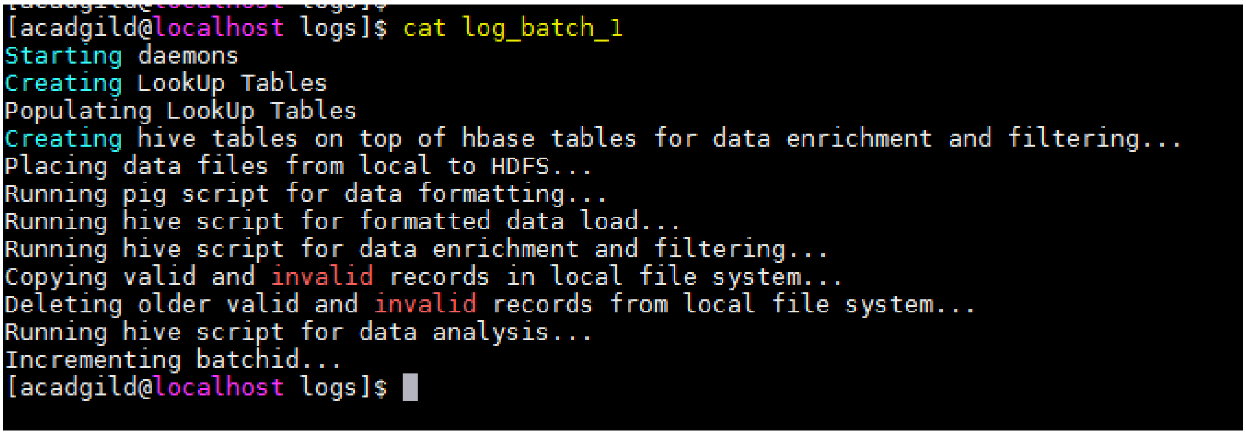
```
sh /home/acadgild/project/scripts/data_export.sh

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

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

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



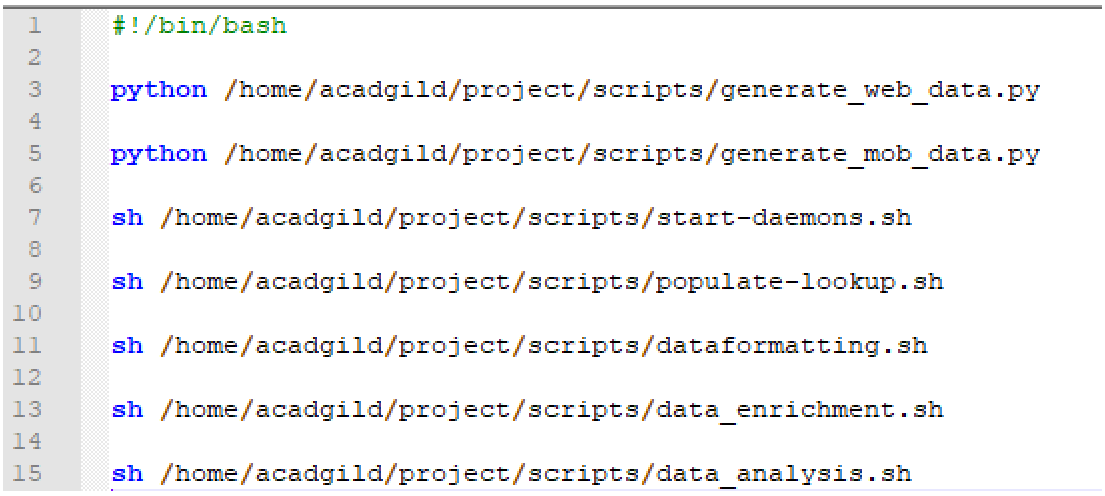
```
[acadgild@localhost logs]$ ls -l
total 12
-rwxrwxr-x. 1 acadgild acadgild    1 Jan 29 07:00 current-batch.txt
-rw-rw-r--. 1 acadgild acadgild 1222 Jan 29 06:04 log_batch_1
-rw-rw-r--. 1 acadgild acadgild  399 Jan 29 06:30 log_batch_1???
[acadgild@localhost logs]$ cat current-batch.txt
2[acadgild@localhost logs]$
```

The log file captured all the data and steps we performed so far,



```
[acadgild@localhost logs]$ cat log_batch_1
Starting daemons
Creating LookUp Tables
Populating LookUp Tables
Creating hive tables on top of hbase tables for data enrichment and filtering...
Placing data files from local to HDFS...
Running pig script for data formatting...
Running hive script for formatted data load...
Running hive script for data enrichment and filtering...
Copying valid and invalid records in local file system...
Deleting older valid and invalid records from local file system...
Running hive script for data analysis...
Incrementing batchid...
[acadgild@localhost logs]$
```

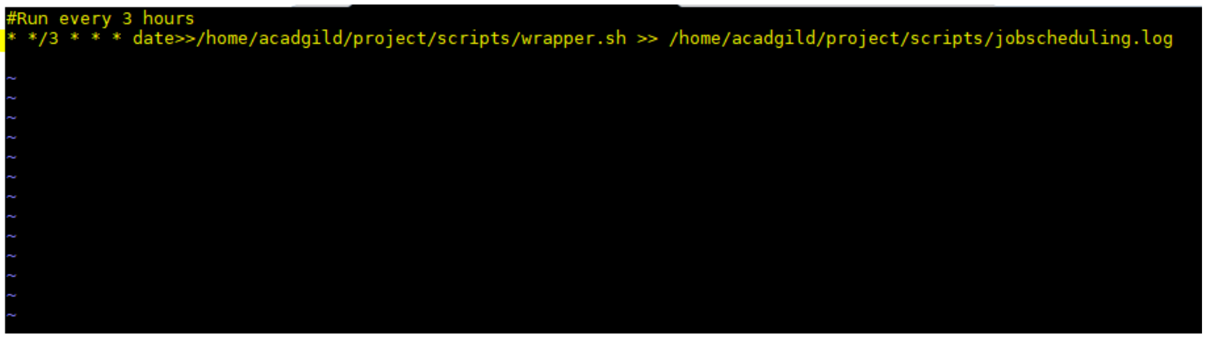
Wrapping all the scripts inside the single script file and scheduling this file to run at the periodic interval of every 3 hours.



```
#!/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
```

The wrapper.sh will be running for every 3 hours as per the job scheduling done below, as per the above order the wrapper.sh will run the scripts.

Creating **Crontab** to schedule the wrapper.sh script to run for every 3 hour interval -



```
#Run every 3 hours
* */3 * * * date>>/home/acadgild/project/scripts/wrapper.sh >> /home/acadgild/project/scripts/jobscheduling.log

~
~
~
~
~
~
~
~
~
~
~
~
~
```

The **crontab** job scheduler will run the **wrappr.sh** every 3 hours and for every 3 hours we will get incremental batch ID's. As per the requirement the job scheduling is done.

Highlights of the Project :

- LookUp tables in HBASE have been integrated with actual flow of data.
- Joins were optimized for analysis. Data was enriched with new fields and using broadcast maps on Lookup tables so as to avoid joins.
- Data cleaning, validation, enrichment and analysis have been automated using bash scripts and schedulers.
- Logs have been maintained to track the behavior and overcome failures in the pipeline.

Project End Conclusion:

All the data operation has been performed as per the sequence mentioned in the **wrapper.sh** file and obtained results successfully for the leading music company to make appropriate business strategies. The results can be used by data science or machine learning pipelines for further forecast and form visualization on the analyzed data.

--- END ---