Project – Music Data Analysis

Section 1: Project Overview

A leading music-catering company is planning to analyze large amount of data received from varieties of sources, namely mobile app and website to track the behavior of users, classify users, calculate royalties associated with the song and make appropriate business strategies. The file server receives data files periodically after every 3 hours.

1.1 Fields present in the data files

Data files contain below fields.

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

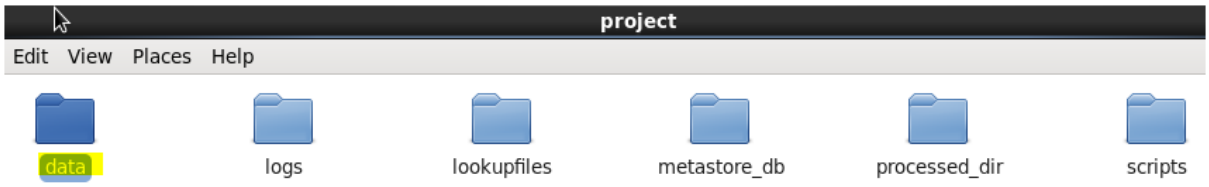
1.2 LookUp Tables

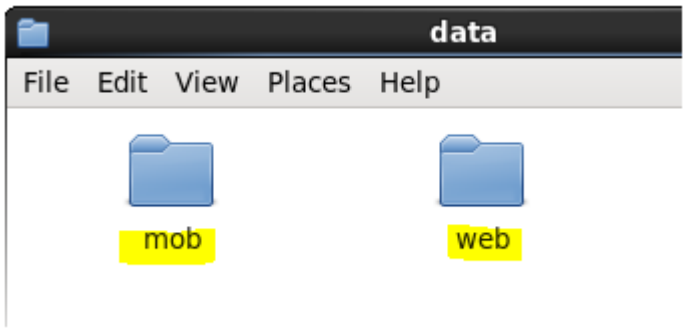
There are some existing look up tables present in **NoSQL** databases. They play an important role in data enrichment and analysis.

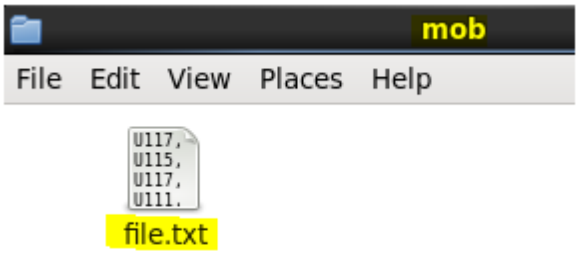
Table Name	Description
Station_Geo_Map	Contains mapping of a geo_cd with station_id
Subscribed_Users	Contains user_id, subscription_start_date and subscription_end_date. Contains details only for subscribed users
Song_Artist_Map	Contains mapping of song_id with artist_id alongwith royalty associated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

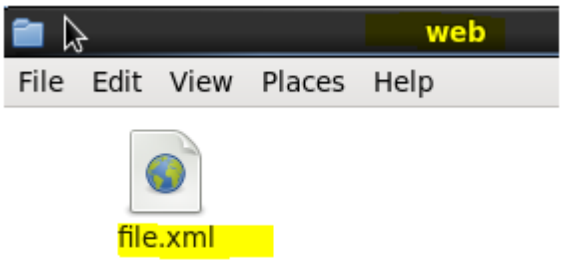
1.3 DATASET

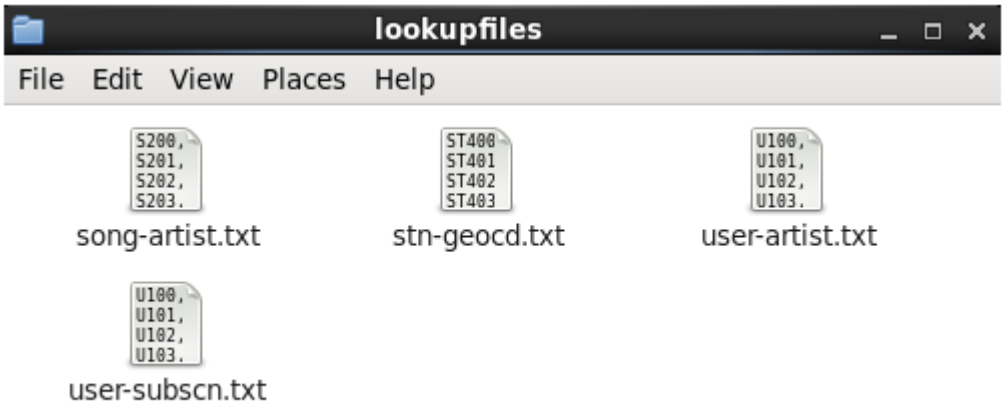
1. Data coming from web applications reside in /data/web and has xml format.
2. Data coming from mobile applications reside in /data/mob and has csv format.
3. Data present in lookup directory should be used in HBase.











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

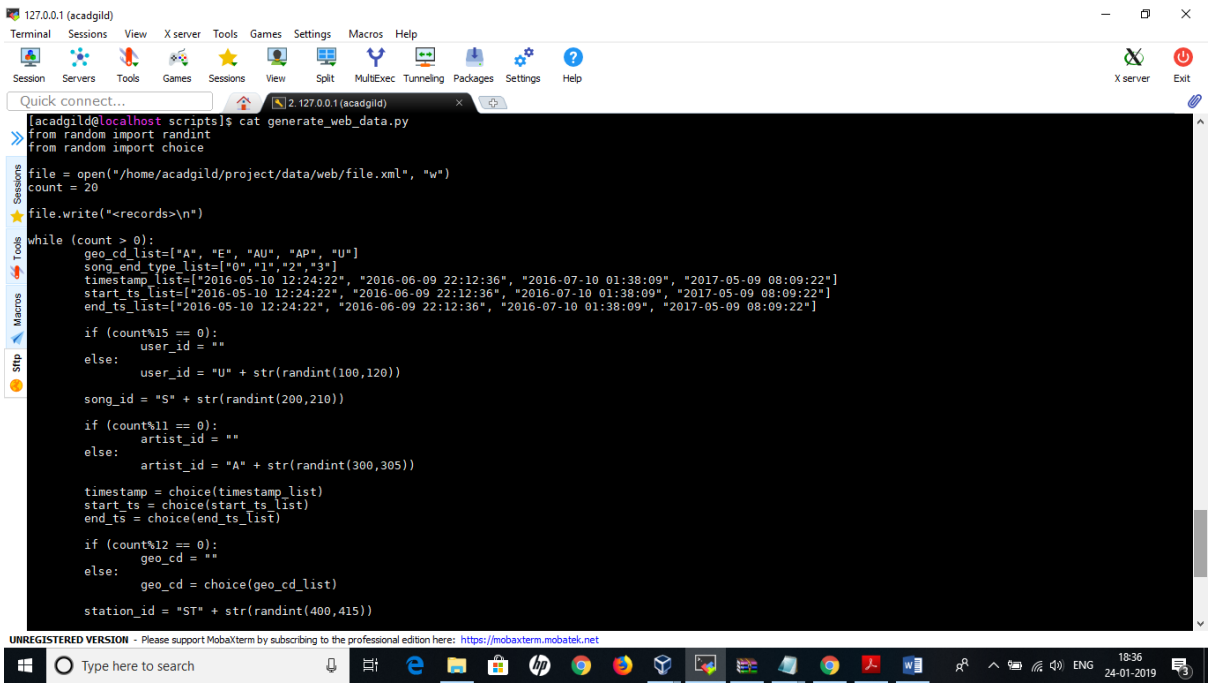
It is not only the data which is important, rather it is the insight it can be used to generate important. Once we have made the data ready for analysis, we have to perform below analysis on a daily basis.

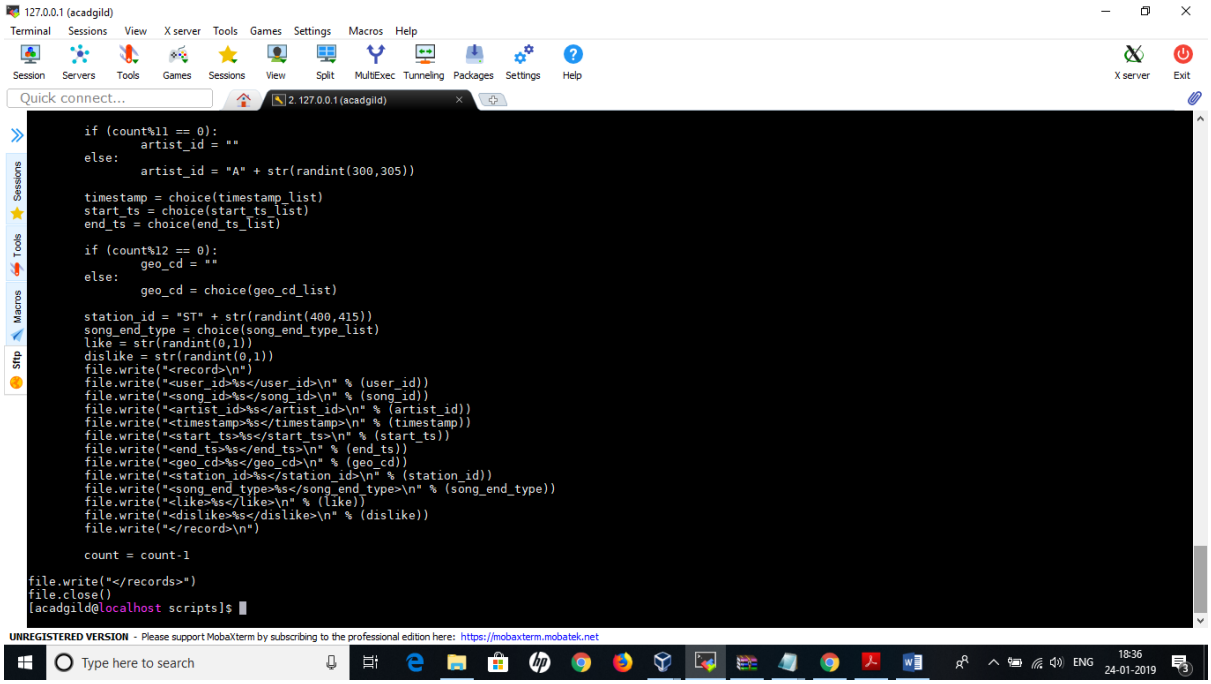
1. Determine top 10 station_id(s) where maximum number of songs were played, which were liked by unique users.
2. Determine total duration of songs played by each type of user, where type of user can be 'subscribed' or 'unsubscribed'. An unsubscribed user is the one whose record is either not present 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.

Section 2: Data Creation:

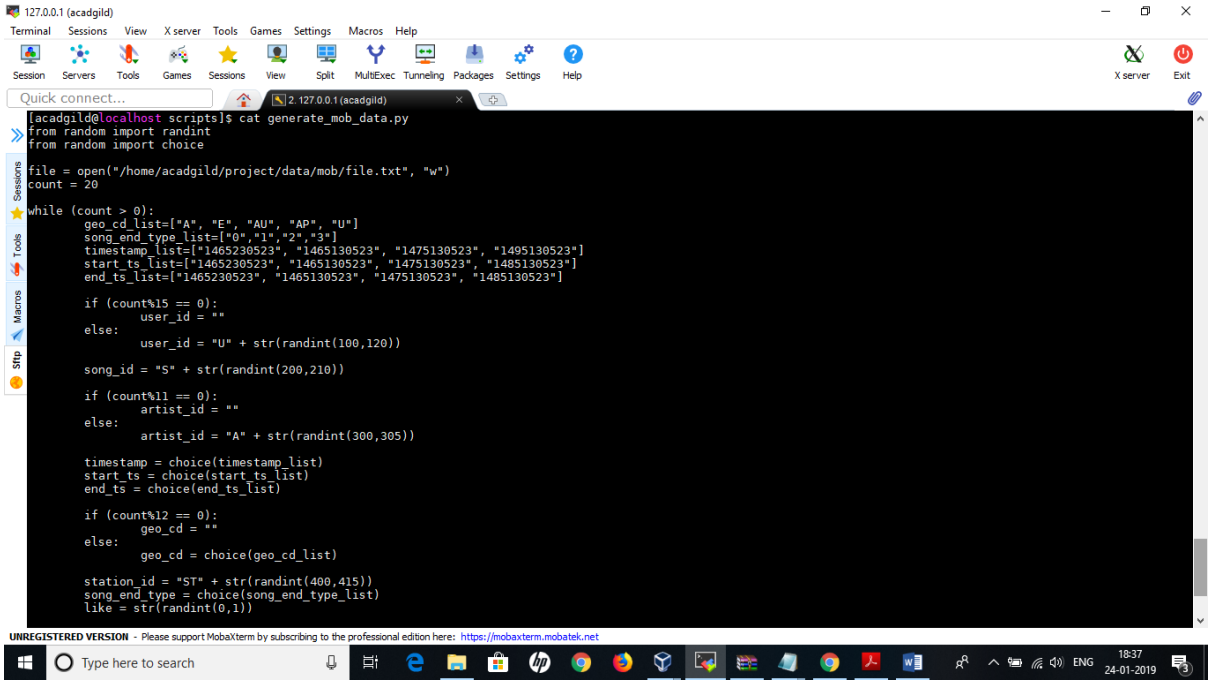
We have used the python script to generate the data. There are two script files one is for web data and another is for Mob data.

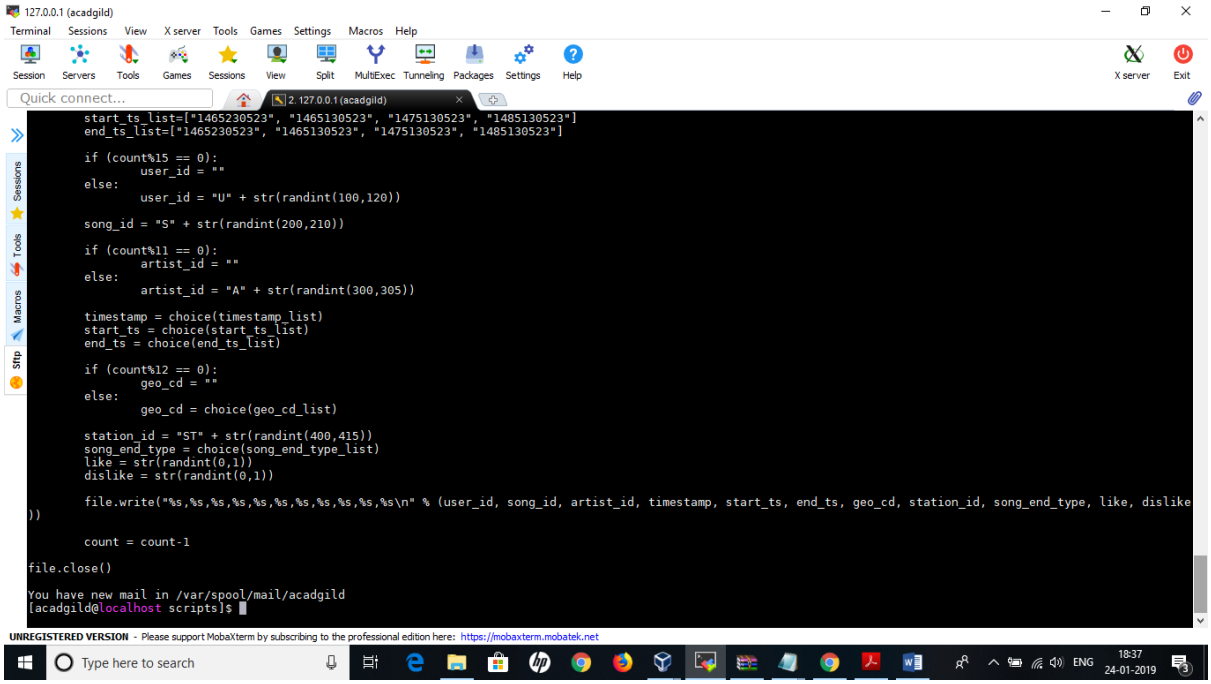
generate_web_data.py -- Generates some random data coming from web application
python /home/acadgild/project/scripts/generate_web_data.py





generate_mob_data.py -- Generates some random data coming from mobile application
python /home/acadgild/project/scripts/generate_mob_data.py



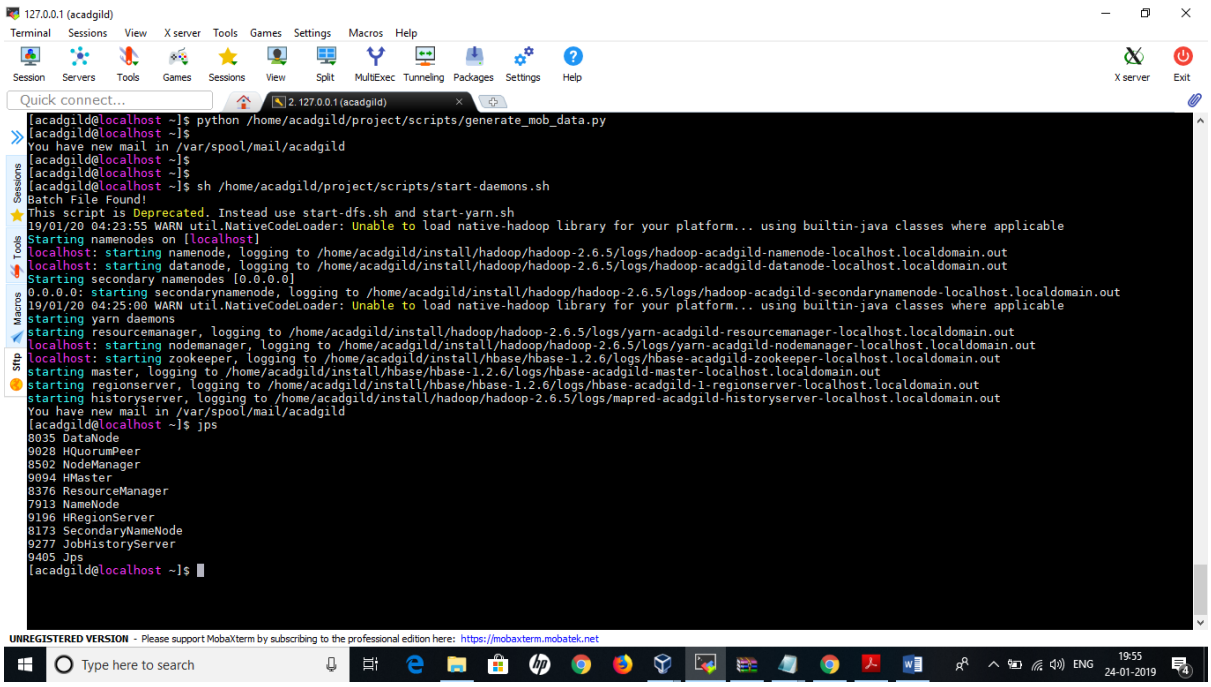


Section 3: Lookup table and Intial validation

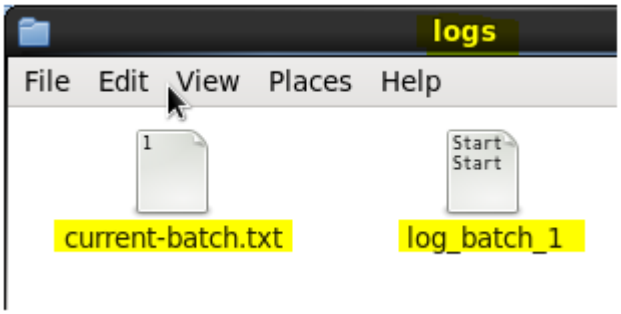
Start the Daemons

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

```
#!/bin/bash

if [ -f "/home/acadgild/Project_2_Music_Data_Analysis/logs/current-batch.txt" ]

then

echo "Batch File Found!"

else

echo -n "1" > "/home/acadgild/Project_2_Music_Data_Analysis/logs/current-batch.txt"

fi

chmod 775 /home/acadgild/Project_2_Music_Data_Analysis/logs/current-batch.txt
batchid=`cat /home/acadgild/Project_2_Music_Data_Analysis/logs/current-batch.txt`
LOGFILE=/home/acadgild/Project_2_Music_Data_Analysis/logs/log_batch_$batchid

echo "Starting daemons" >> $LOGFILE

# To Start Hadoop Daemons:
start-all.sh

# To start the HMASTER service:
start-hbase.sh

# To Start the JobHistory server Services:
mr-jobhistory-daemon.sh start historyserver

# To Start the mysql service
sudo service mysqld start

# To Start HIVE metastore:
hive --service metastore
```



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**.



Use the "***populate-lookup.sh"*** script to create lookup tables in **Hbase**. These tables have to be used in,

- Data formatting,
- Data enrichment and
- Analysis stage

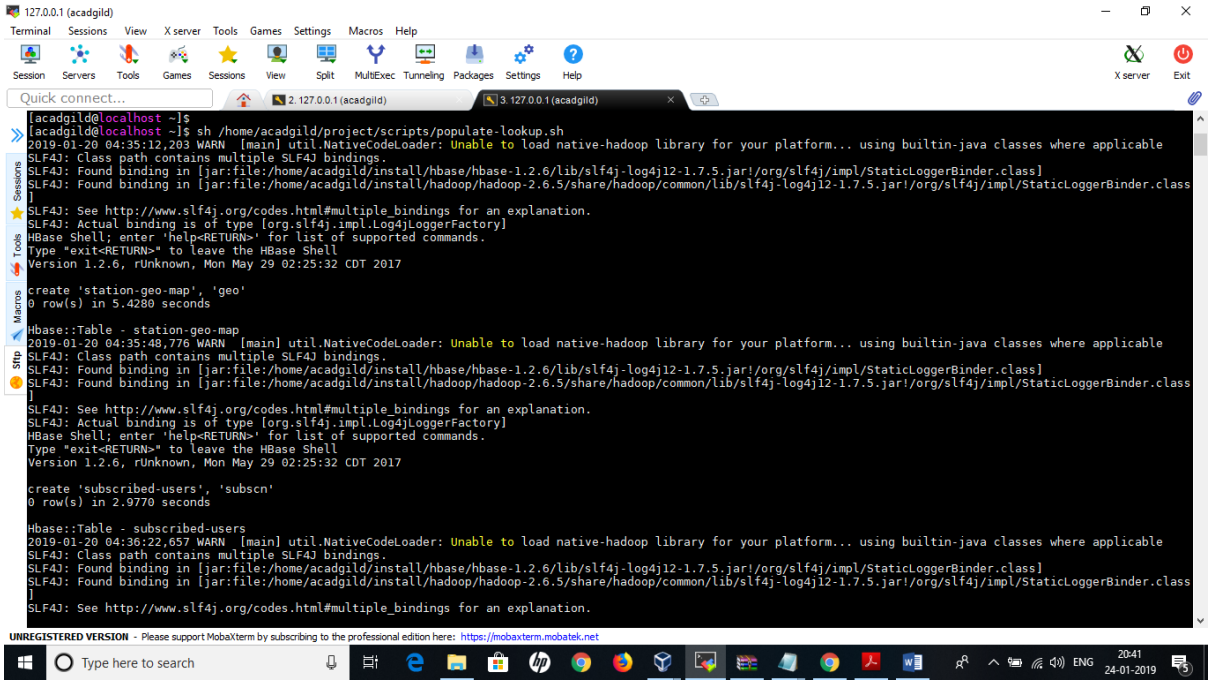
Lookup Tables

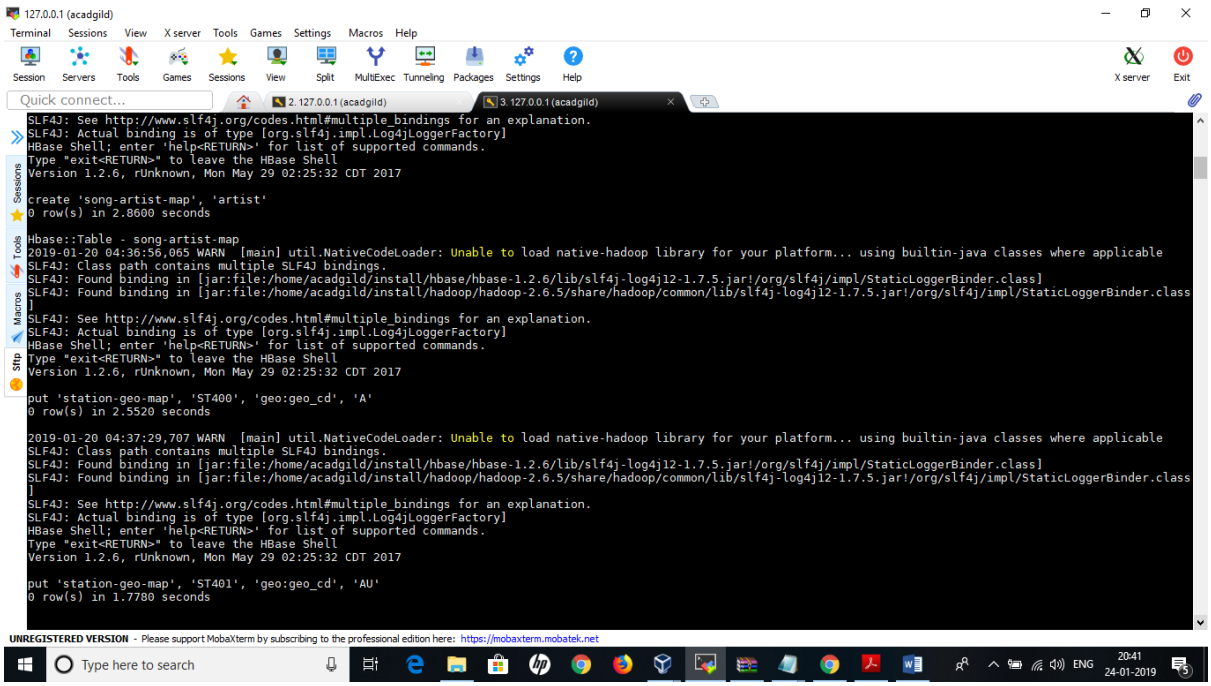
Table Name	Description
Station_Geo_Map	Contains mapping of a geo_cd with station_id
Subscribed_Users	Contains user_id, subscription_start_date and subscription_end_date. Contains details only for subscribed users
Song_Artist_Map	Contains mapping of song_id with artist_id alongwith royalty associated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

"populate-lookup.sh" script

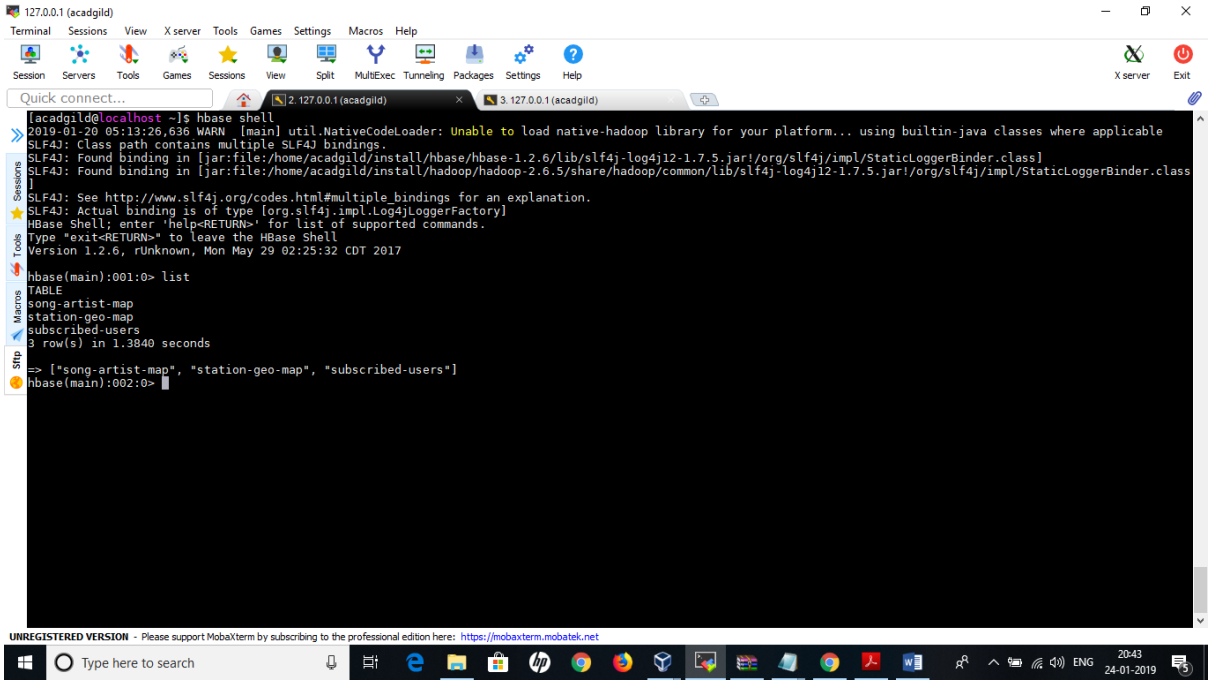
The "***populate-lookup.sh"*** shell script creates the above 4 lookup tables in the Hbase and populate the data into the lookup tables from the dataset files.
In the below screen shots, we can see the create-lookup.sh scripts and the following screen shots shows the tables creation and population of 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

```

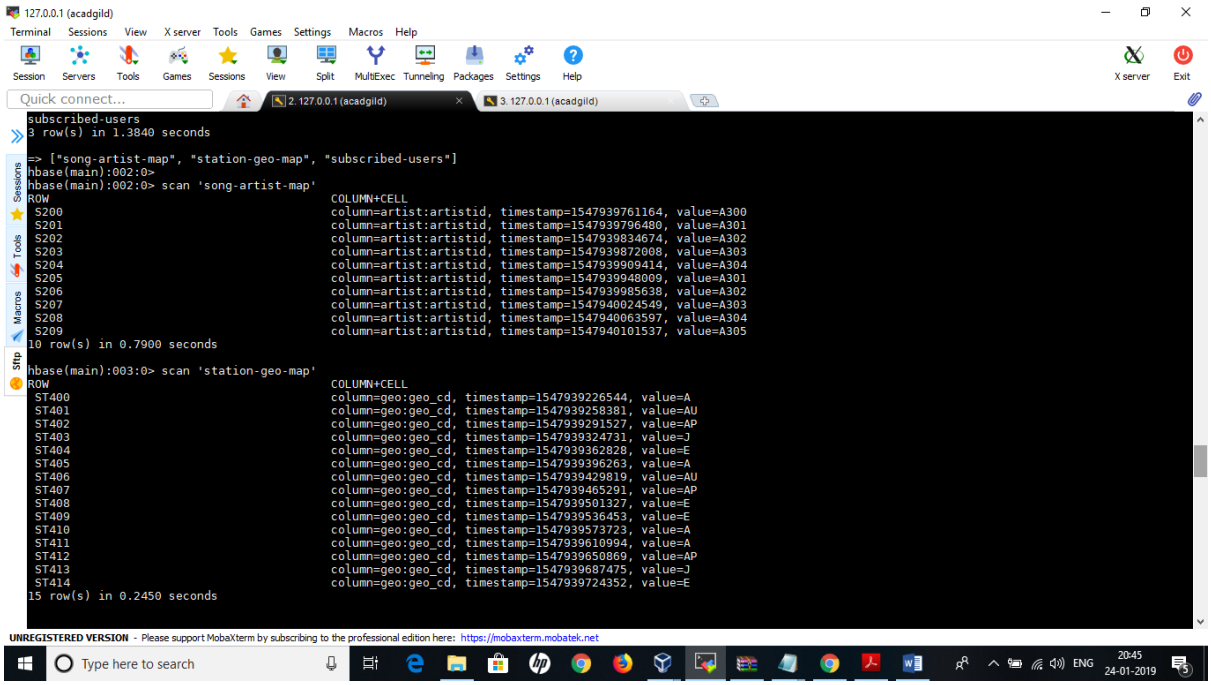


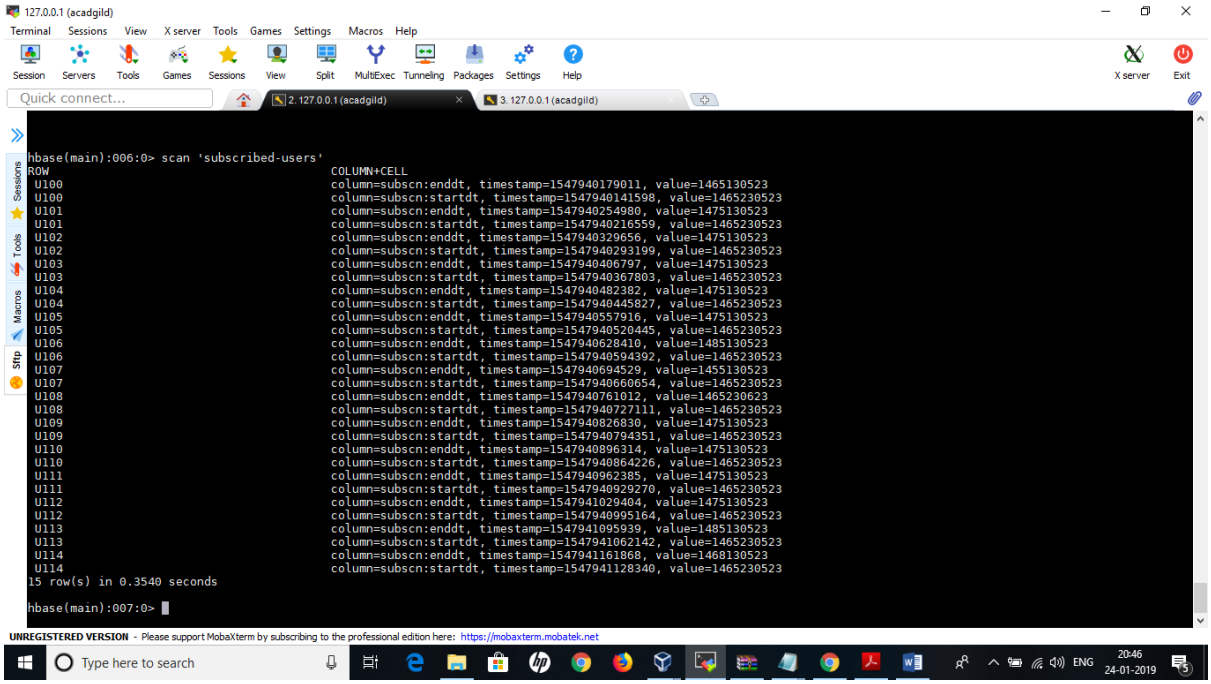


We can see the lookup tables created using the ***"populate-lookup.sh"*** in the below screen shot, Lookup Tables in the hbase shell,



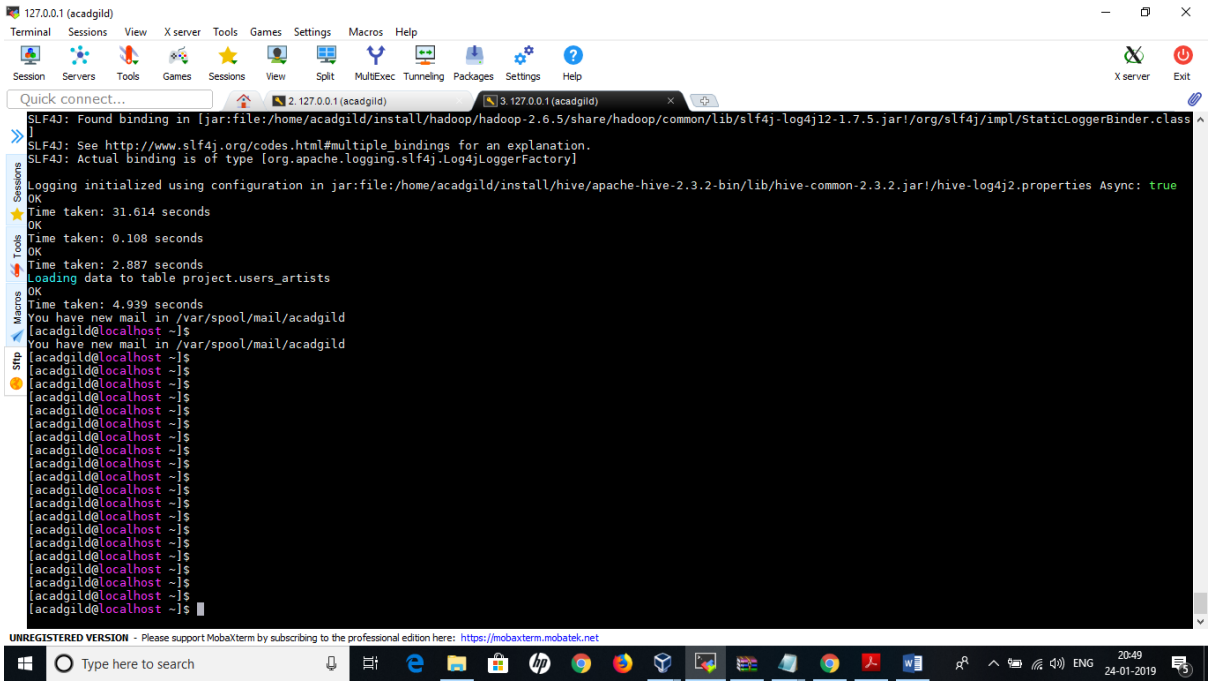
The values loaded in the Lookup tables are shown below,



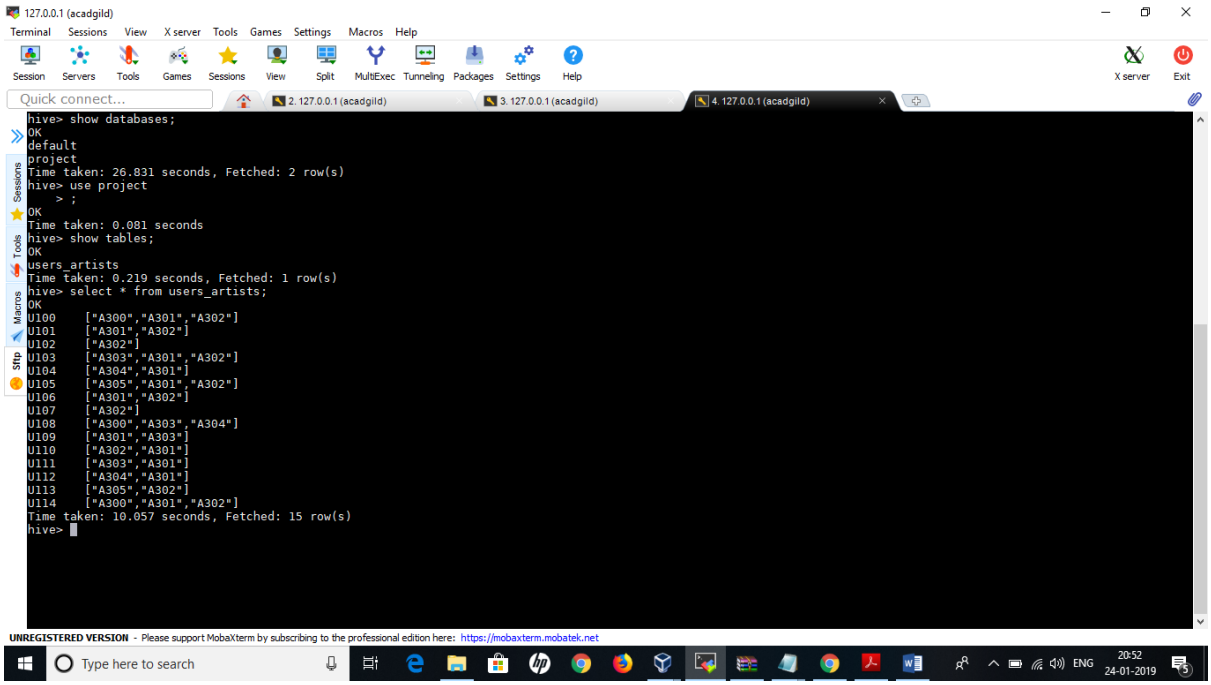


We have successfully created the lookup tables in the Hbase.

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> Select * From users_artists;



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 the 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: **sh /home/acadgild/project/scripts /data_enrichment_filtering_schema.sh**

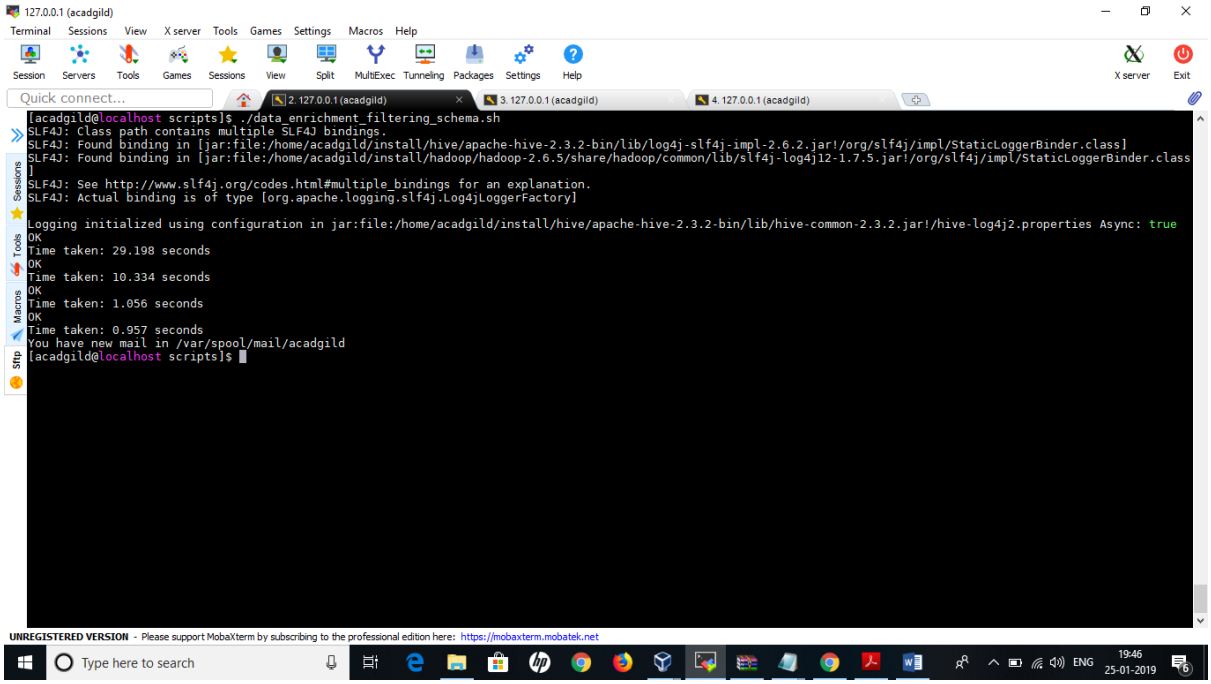
The script will run the **"create_hive_hbase_lookup.hql"** which will create the HIVE external tables with the help of **Hbase storage handler & SerDe properties**. The hive external tables will match the columns of **Hbase** tables to **HIVE** tables.

```
#!/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”**

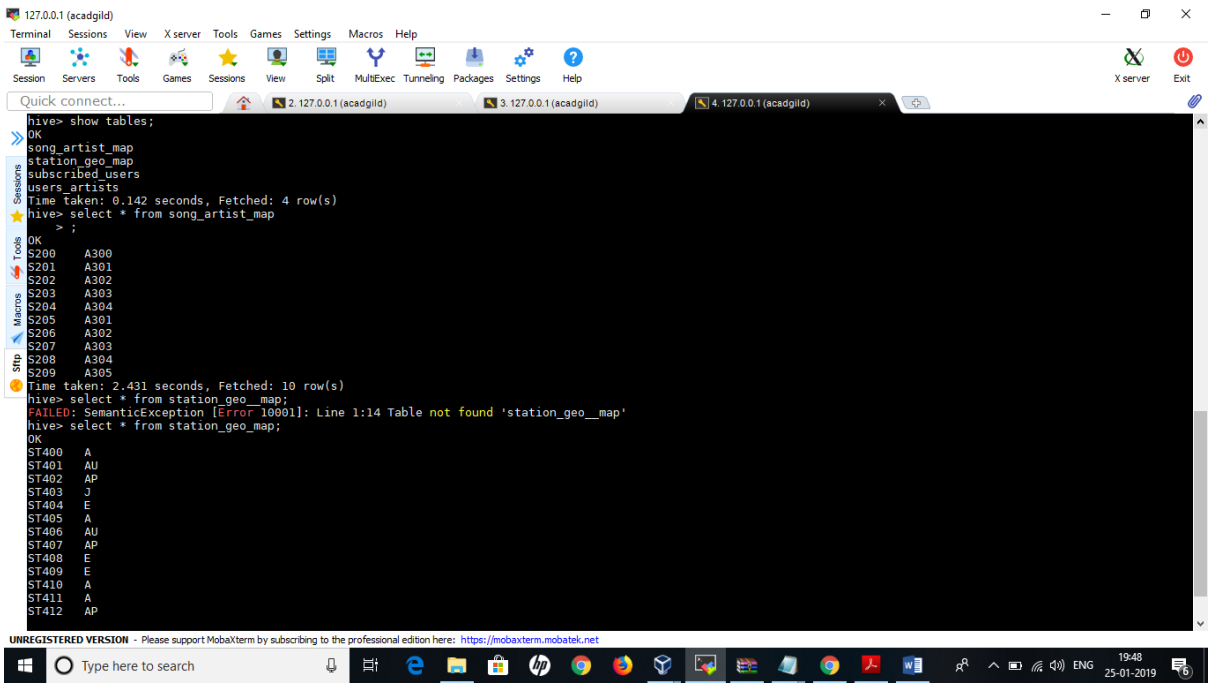


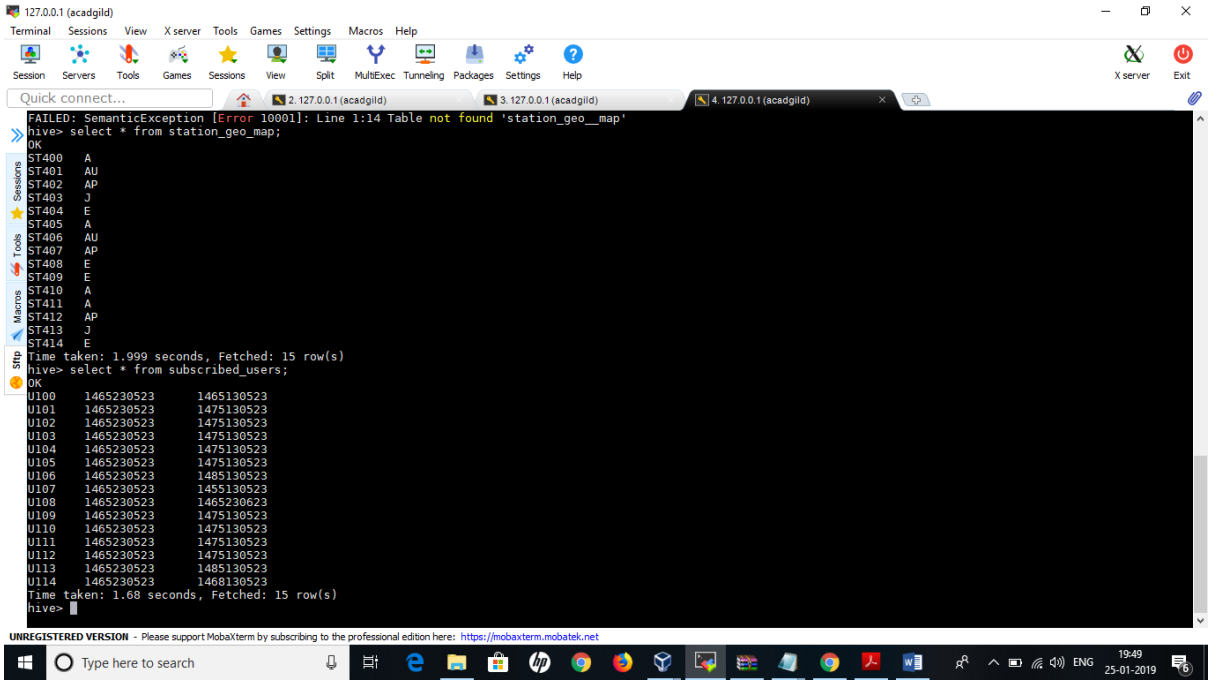
hive>Show Tables;

hive>Select * From song_artist_map

hive>Select * From station_geo_map

hive>Select * From Subscribed_users



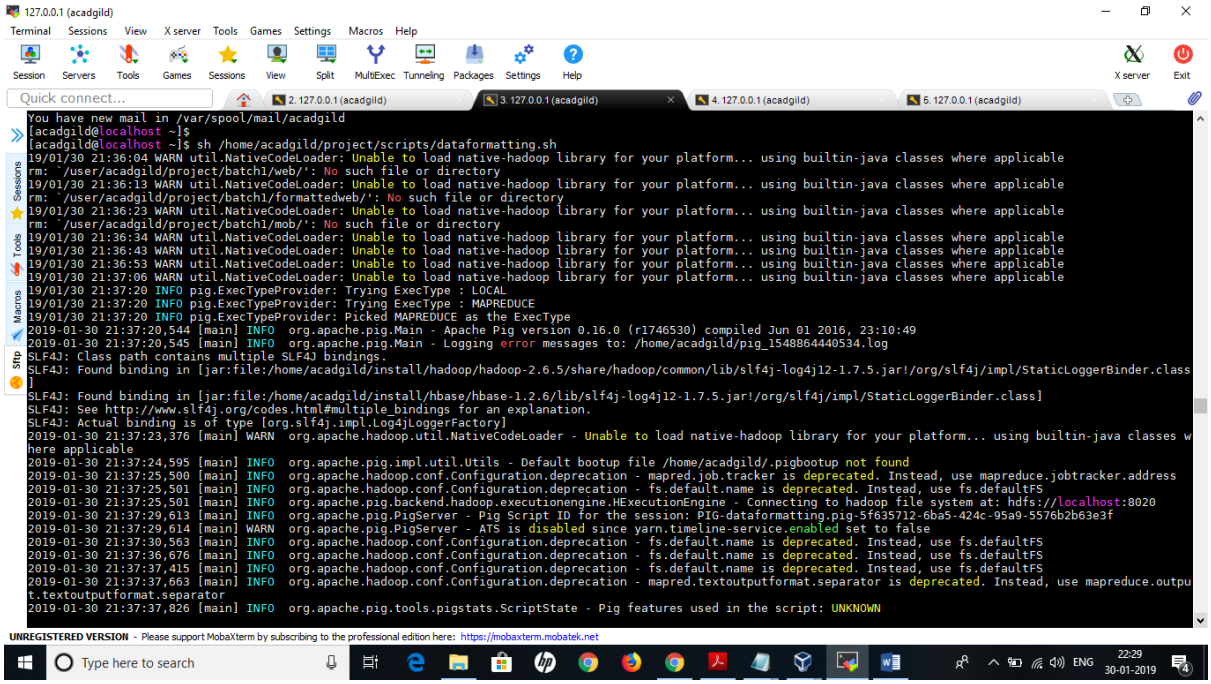


Section-4 Data Formatting:

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 scripts for every 3 hours.

Run the script: ***sh /home/acadgild/project/scripts/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
```



We are running two scripts to format the data. They are:

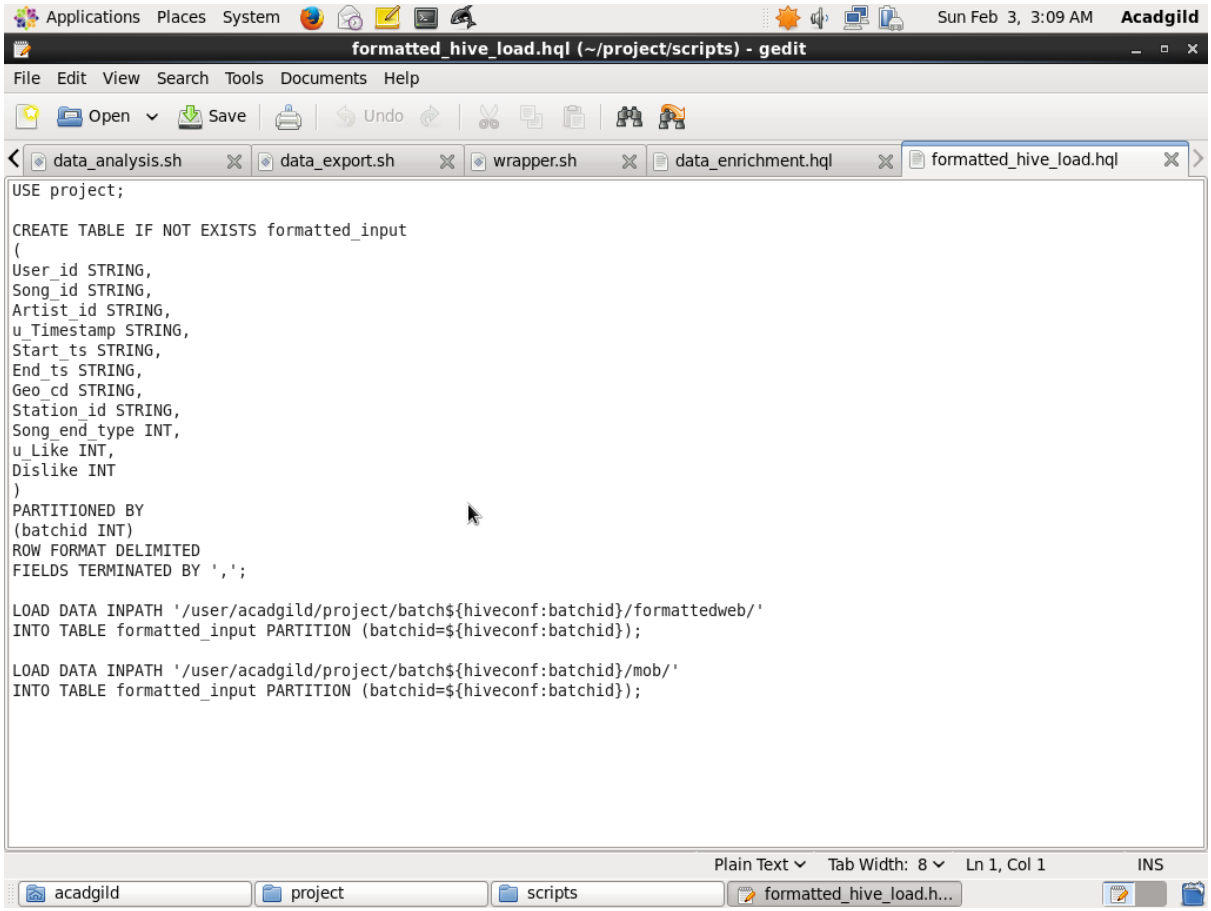
- Dataformatting.pig
- Formatted_hive_load.hql

Pig script to parse the data from coming from **web_data.xml** to **csv** format and partition both web and mob data based on based on batch ID's

Dataformatting.pig

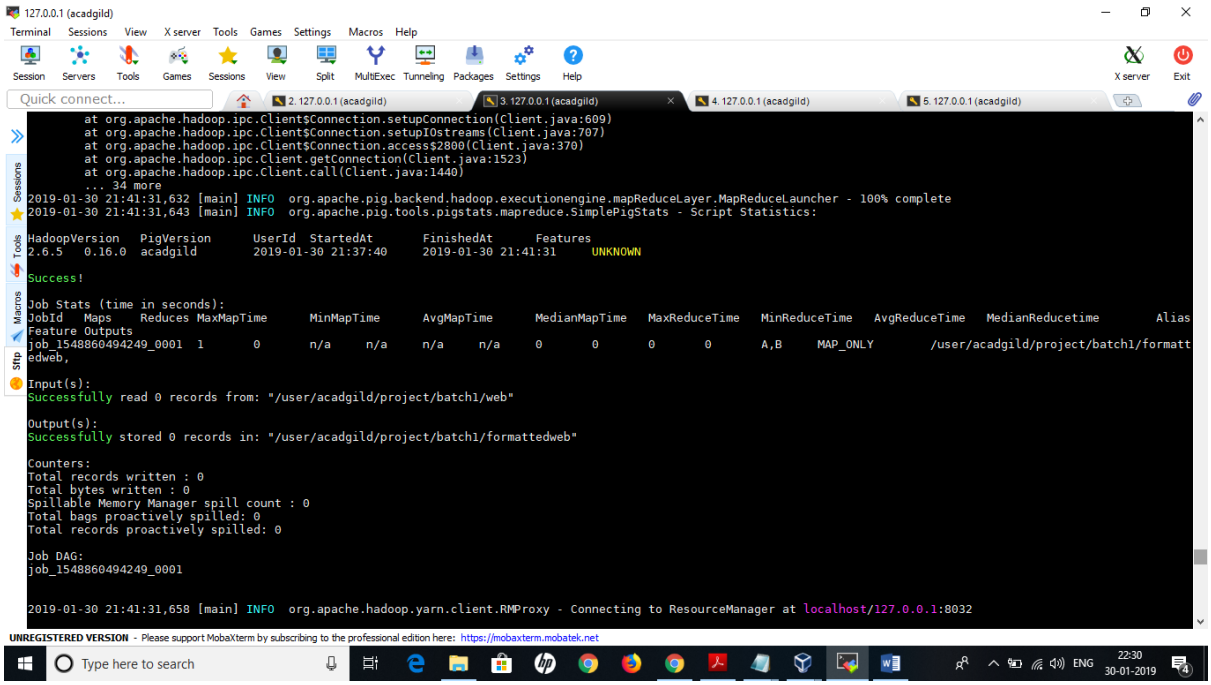
```
REGISTER /home/acadgild/project/lib/piggybank.jar;

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

A =  LOAD '/user/acadgild/project/batch${batchid}/web/' using org.apache.pig.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

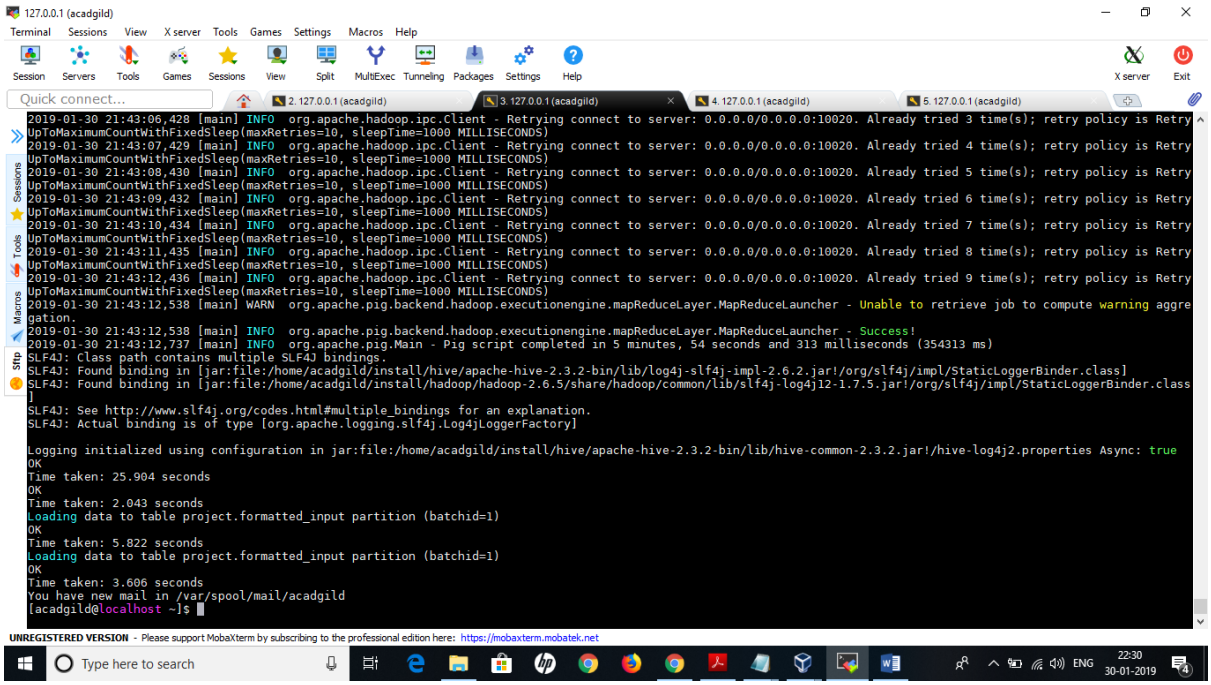


In the below screenshot we can see the data both the scripts in action, first pig script will parse the data and then hive script will load the data into hive terminal successfully.

Pig script successful completion,



Hive script successfully load the data into hive terminal,



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:

```
drwxr-xr-x   - acadgild supergroup          0 2018-01-20 16:29 project
[acadgild@localhost ~]$ hadoop fs -ls /user/acadgild/project
Java HotSpot(TM) Client VM warning: You have loaded library /home/acadgild/hadoop-2.7.2/lib/native/libhadoop.so.1.0.0 whi
 stack guard. The VM will try to fix the stack guard now.
It's highly recommended that you fix the library with 'execstack -c <libfile>', or link it with '-z noexecstack'.
18/01/20 19:05:34 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-jav
able
Found 1 items
drwxr-xr-x   - acadgild supergroup          0 2018-01-20 18:12 /user/acadgild/project/batch1
[acadgild@localhost ~]$ hadoop fs -ls /user/acadgild/project/batch1
Java HotSpot(TM) Client VM warning: You have loaded library /home/acadgild/hadoop-2.7.2/lib/native/libhadoop.so.1.0.0 whi
 stack guard. The VM will try to fix the stack guard now.
It's highly recommended that you fix the library with 'execstack -c <libfile>', or link it with '-z noexecstack'.
18/01/20 19:05:47 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-jav
able
Found 3 items
drwxr-xr-x   - acadgild supergroup          0 2018-01-20 18:12 /user/acadgild/project/batch1/formattedweb
drwxr-xr-x   - acadgild supergroup          0 2018-01-20 18:12 /user/acadgild/project/batch1/mob
drwxr-xr-x   - acadgild supergroup          0 2018-01-20 18:11 /user/acadgild/project/batch1/web
[acadgild@localhost ~]$
[acadgild@localhost ~]$
[acadgild@localhost ~]$
[acadgild@localhost ~]$
[acadgild@localhost ~]$ hadoop fs -ls /user/acadgild/project/batch1/formattedweb
Java HotSpot(TM) Client VM warning: You have loaded library /home/acadgild/hadoop-2.7.2/lib/native/libhadoop.so.1.0.0 whi
 stack guard. The VM will try to fix the stack guard now.
It's highly recommended that you fix the library with 'execstack -c <libfile>', or link it with '-z noexecstack'.
18/01/20 19:07:25 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-jav
able
Found 2 items
-rw-r--r--   1 acadgild supergroup          0 2018-01-20 18:12 /user/acadgild/project/batch1/formattedweb/_SUCCESS
-rw-r--r--   1 acadgild supergroup       1241 2018-01-20 18:12 /user/acadgild/project/batch1/formattedweb/part-m-00000
[acadgild@localhost ~]$
```

The output of the **formattedweb** data obtained from the **Dataformatting.pig** is shown in the below screen shot,

Command,

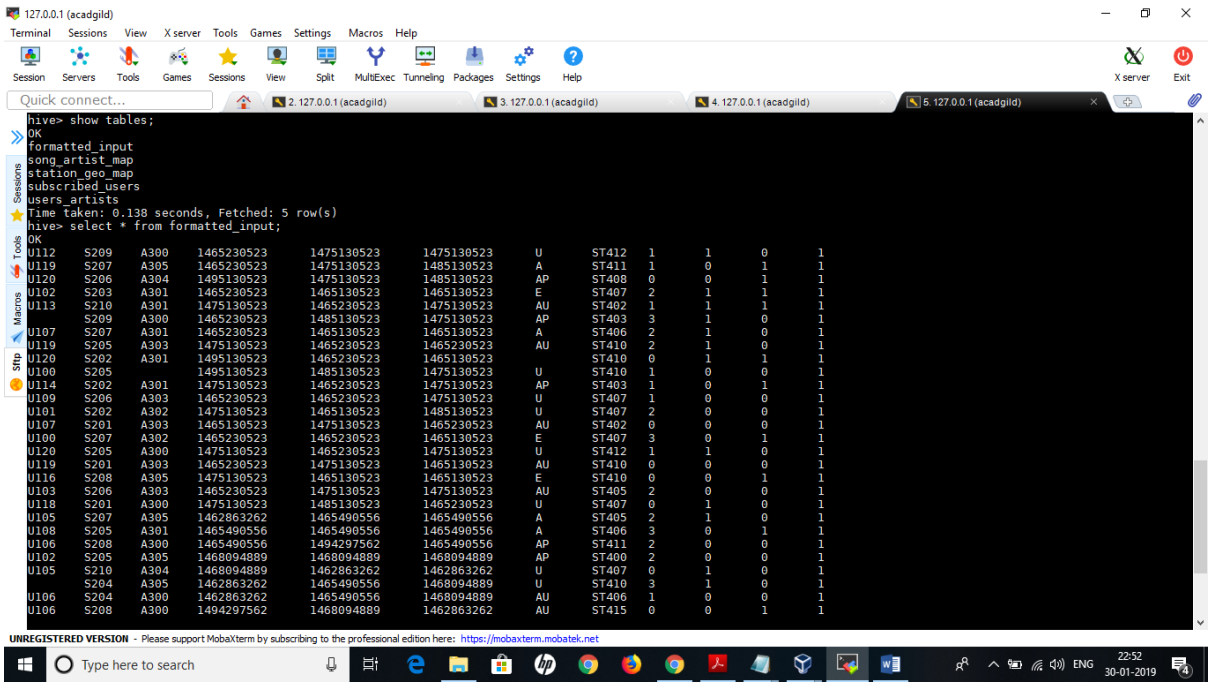
hadoop fs -cat /user/acadgild/project/batch1/formattedweb/*

```
[acadgild@localhost ~]$
[acadgild@localhost ~]$ hadoop fs -cat /user/acadgild/project/batch1/formattedweb/*
Java HotSpot(TM) Client VM warning: You have loaded library /home/acadgild/hadoop-2.7
 stack guard. The VM will try to fix the stack guard now.
It's highly recommended that you fix the library with 'execstack -c <libfile>', or li
18/01/20 19:09:24 WARN util.NativeCodeLoader: Unable to load native-hadoop library fo
able
U113,S205,A305,1462863262,1465490556,1462863262,AP,ST407,3,0,1
U102,S200,A301,1494297562,1465490556,1465490556,A,ST400,1,0,1
U115,S207,A301,1494297562,1468094889,1465490556,AU,ST406,2,1,1
U110,S201,A300,1468094889,1462863262,1468094889,AU,ST413,2,0,1
U102,S203,A305,1465490556,1494297562,1465490556,A,ST414,2,0,0
,S209,A304,1465490556,1462863262,1465490556,E,ST412,0,0,1
U105,S203,A300,1462863262,1468094889,1468094889,U,ST407,2,1,1
U113,S205,A303,1462863262,1468094889,1468094889,E,ST415,2,0,1
U120,S205,A302,1494297562,1494297562,1494297562,,ST400,0,1,0
U105,S210,,1468094889,1462863262,1494297562,E,ST410,1,0,1
U117,S206,A300,1468094889,1468094889,1465490556,A,ST414,2,0,0
U114,S200,A301,1462863262,1468094889,1462863262,AP,ST408,1,1,1
U110,S208,A303,1494297562,1468094889,1468094889,E,ST405,1,0,1
U115,S201,A303,1465490556,1465490556,1494297562,AU,ST407,2,1,1
U103,S209,A305,1465490556,1468094889,1468094889,AU,ST408,3,0,1
U112,S210,A303,1494297562,1494297562,1462863262,AU,ST408,2,1,0
U118,S202,A301,1468094889,1465490556,1468094889,AP,ST414,0,0,1
U100,S200,A301,1462863262,1494297562,1494297562,AU,ST408,2,0,0
U113,S210,A304,1468094889,1465490556,1494297562,E,ST403,2,0,1
U104,S203,A300,1468094889,1468094889,1494297562,AU,ST406,1,0,1
[acadgild@localhost ~]$
[acadgild@localhost ~]$
```

The new Tables has been created and show below,

DataFormatting.sh output in hive terminal,

hive> select * from formatted_input;



- In the above screenshot we can see the formatted input data with some null values in **user_id, aritist_id** and **geo_cd** columns which we will fill the enrichment script based on

rules of enrichment for **artist_id** and **geo_cd** only. We will get neglect **user_id** because they didn't mentioned anything about **user_id** for enrichment purpose.

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

Section 5 - 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,

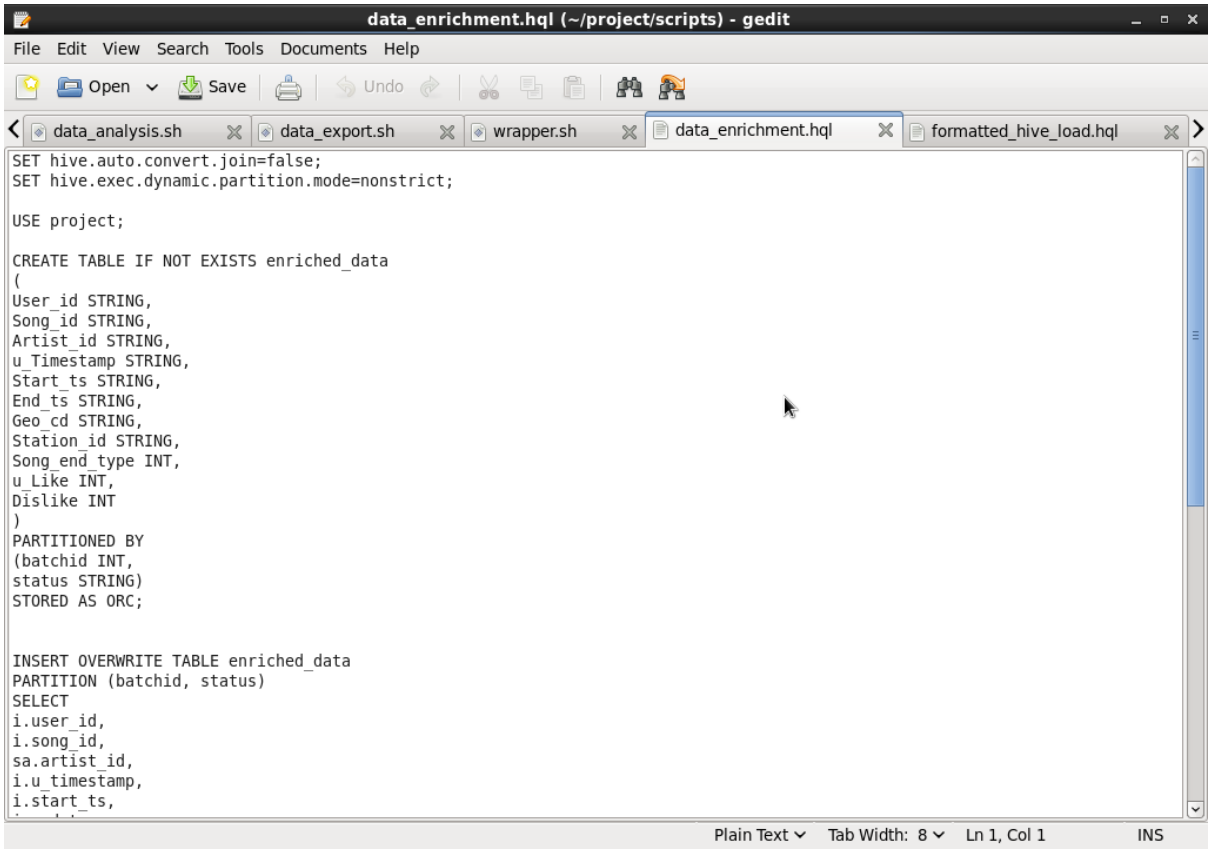
1. If any of like or dislike is **NULL** or **absent**, consider it as **0**.
2. If fields like **Geo_cd** and **Artist_id** are NULL or absent, consult the lookup tables for fields **Station_id** and **Song_id** respectively to get the values of **Geo_cd** and **Artist_id.**
3. If corresponding lookup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

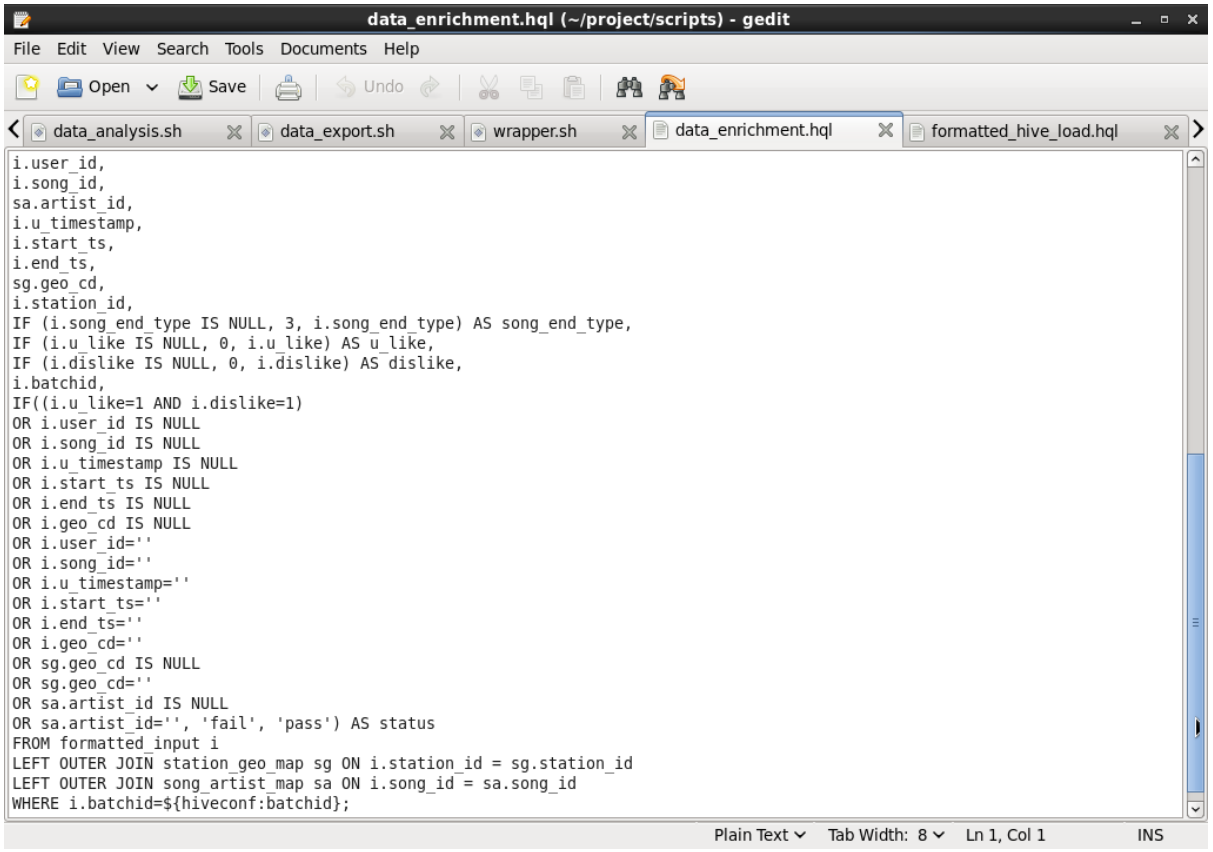
So based on the enrichment rules we will fill the null **geo_cd** and **artist_id** values with the help of corresponding lookup values in **song-artist-map** and **station-geo-map** tables in **Hive-Hbase** tables.

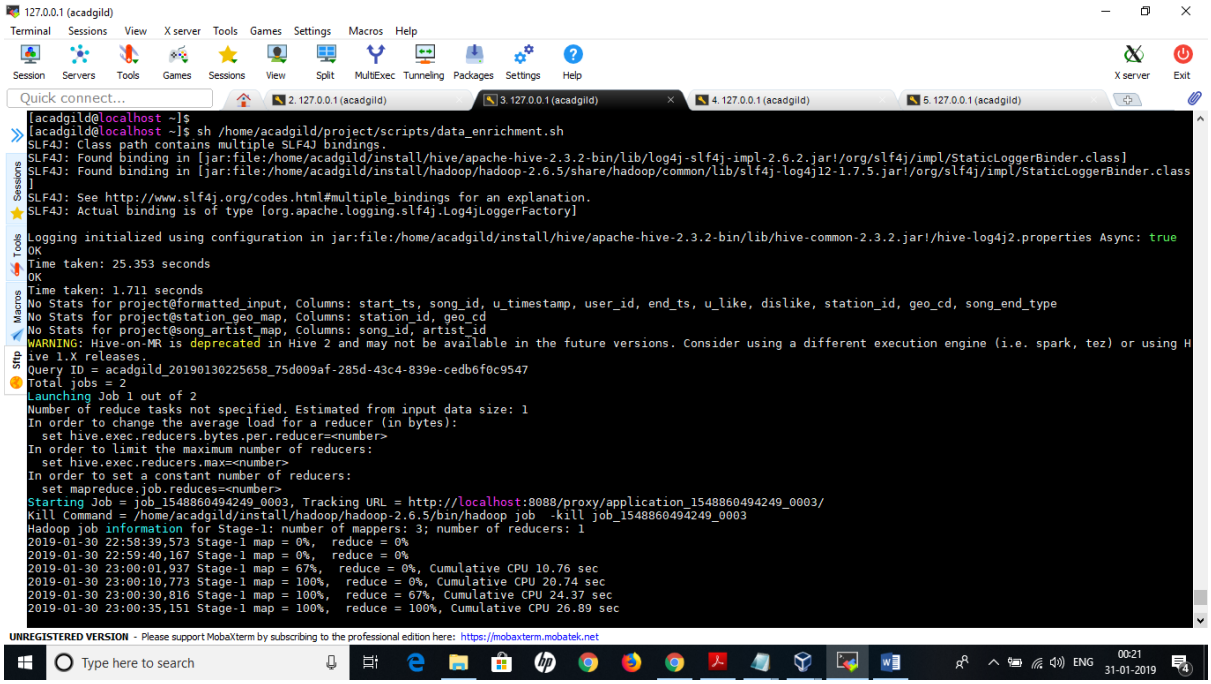
data_enrichment.sh

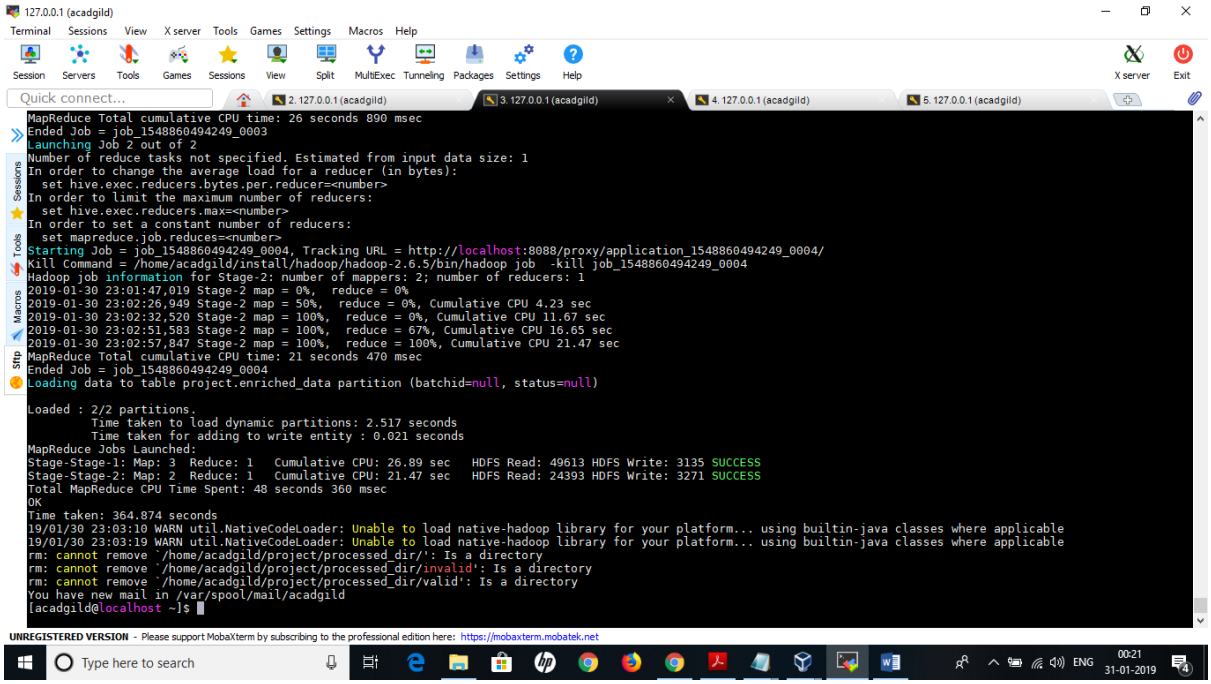
```
#!/bin/bash

batchid=`cat /home/acadgild/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

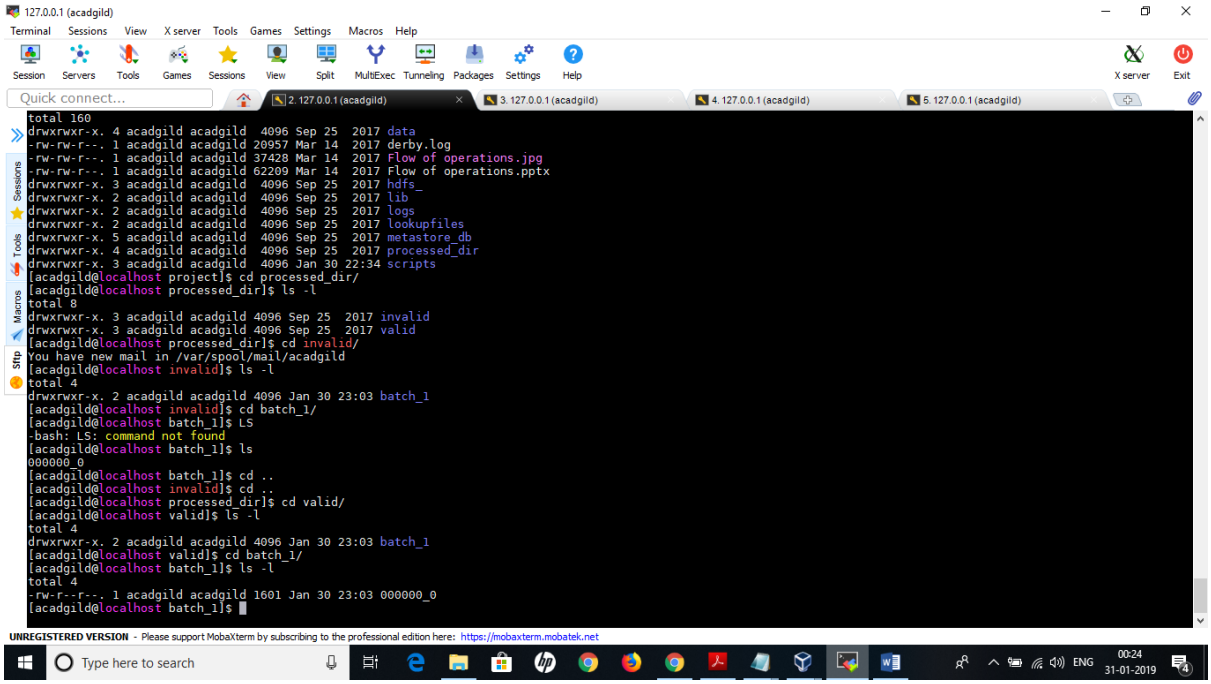

data_enrichment.hql (~/project/scripts) - gedit
File Edit View Search Tools Documents Help
Open Save Undo
data_analysis.sh data_export.sh wrapper.sh data_enrichment.hql formatted_hive_load.hql
SET hive.auto.convert.join=false;
SET hive.exec.dynamic.partition.mode=nonstrict;
USE project;
CREATE TABLE IF NOT EXISTS enriched_data
(
User_id STRING,
Song_id STRING,
Artist_id STRING,
u_Timestamp STRING,
Start_ts STRING,
End_ts STRING,
Geo_cd STRING,
Station_id STRING,
Song_end_type INT,
u_Like INT,
Dislike INT
)
PARTITIONED BY
(batchid INT,
status STRING)
STORED AS ORC;
INSERT OVERWRITE TABLE enriched_data
PARTITION (batchid, status)
SELECT
i.user_id,
i.song_id,
sa.artist_id,
i.u_timestamp,
i.start_ts,
Plain Text Tab Width: 8 Ln 1, Col 1 INS


data_enrichment.hql (~/project/scripts) - gedit
File Edit View Search Tools Documents Help
Open Save Undo
data_analysis.sh data_export.sh wrapper.sh data_enrichment.hql formatted_hive_load.hql
i.user_id,
i.song_id,
sa.artist_id,
i.u_timestamp,
i.start_ts,
i.end_ts,
sg.geo_cd,
i.station_id,
IF (i.song_end_type IS NULL, 3, i.song_end_type) AS song_end_type,
IF (i.u_like IS NULL, 0, i.u_like) AS u_like,
IF (i.dislike IS NULL, 0, i.dislike) AS dislike,
i.batchid,
IF((i.u_like=1 AND i.dislike=1)
OR i.user_id IS NULL
OR i.song_id IS NULL
OR i.u_timestamp IS NULL
OR i.start_ts IS NULL
OR i.end_ts IS NULL
OR i.geo_cd IS NULL
OR i.user_id=''
OR i.song_id=''
OR i.u_timestamp=''
OR i.start_ts=''
OR i.end_ts=''
OR i.geo_cd=''
OR sg.geo_cd IS NULL
OR sg.geo_cd=''
OR sa.artist_id IS NULL
OR sa.artist_id='', 'fail', 'pass') AS status
FROM formatted_input i
LEFT OUTER JOIN station_geo_map sg ON i.station_id = sg.station_id
LEFT OUTER JOIN song_artist_map sa ON i.song_id = sa.song_id
WHERE i.batchid=${hiveconf:batchid};
Plain Text Tab Width: 8 Ln 1, Col 1 INS

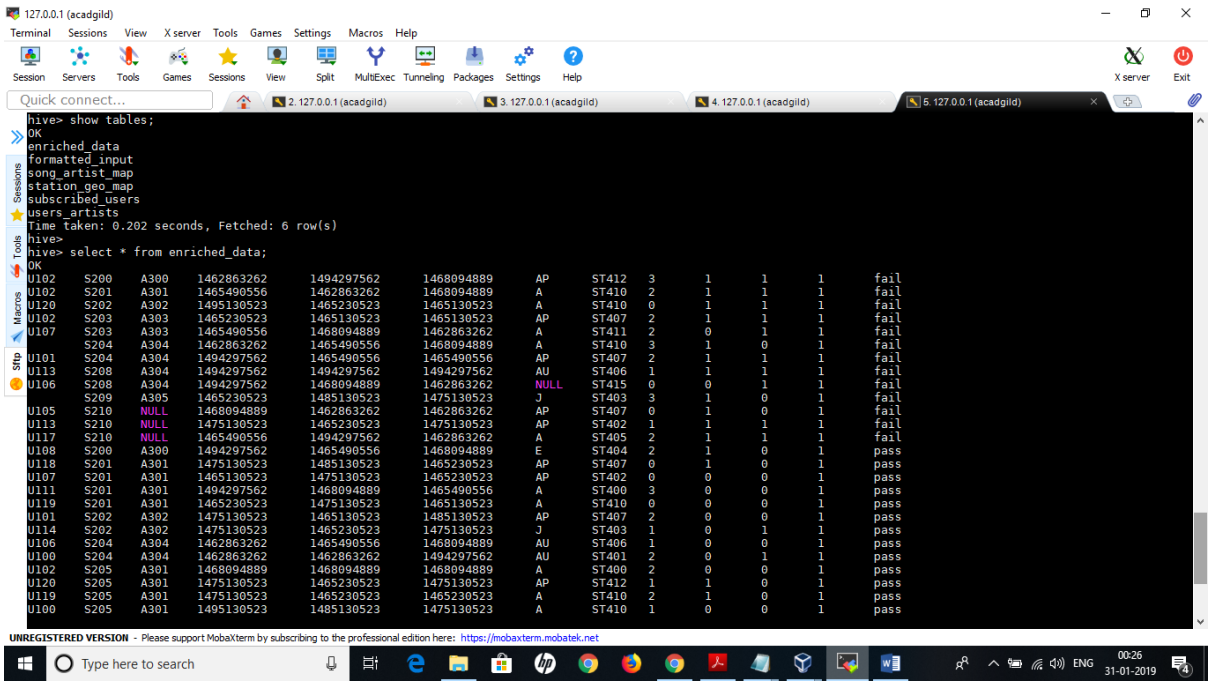




At the end script will automatically divide the records based on status **pass** & **fail** and dump the result into **processed_dir** folder with valid and invalid folders.



Now we can check whether the data properly loaded in the hive terminal or not.



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

Section 6: Data Analysis:

In this stage we will do analysis on enriched_data by the use of hive. We will perform the analysis with the help of script file DataAnalysis.sh

DataAnalysis.sh

data_analysis.sh (~/project/scripts) - gedit

File Edit View Search Tools Documents Help

Open Save Undo

data_analysis.sh | data_export.sh | wrapper.sh | data_enrichment.hql | formatted_hive_load.hql

```
#!/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
```

sh Tab Width: 8 Ln 10, Col 19 INS

data_analysis.hql (~/project/scripts) - gedit

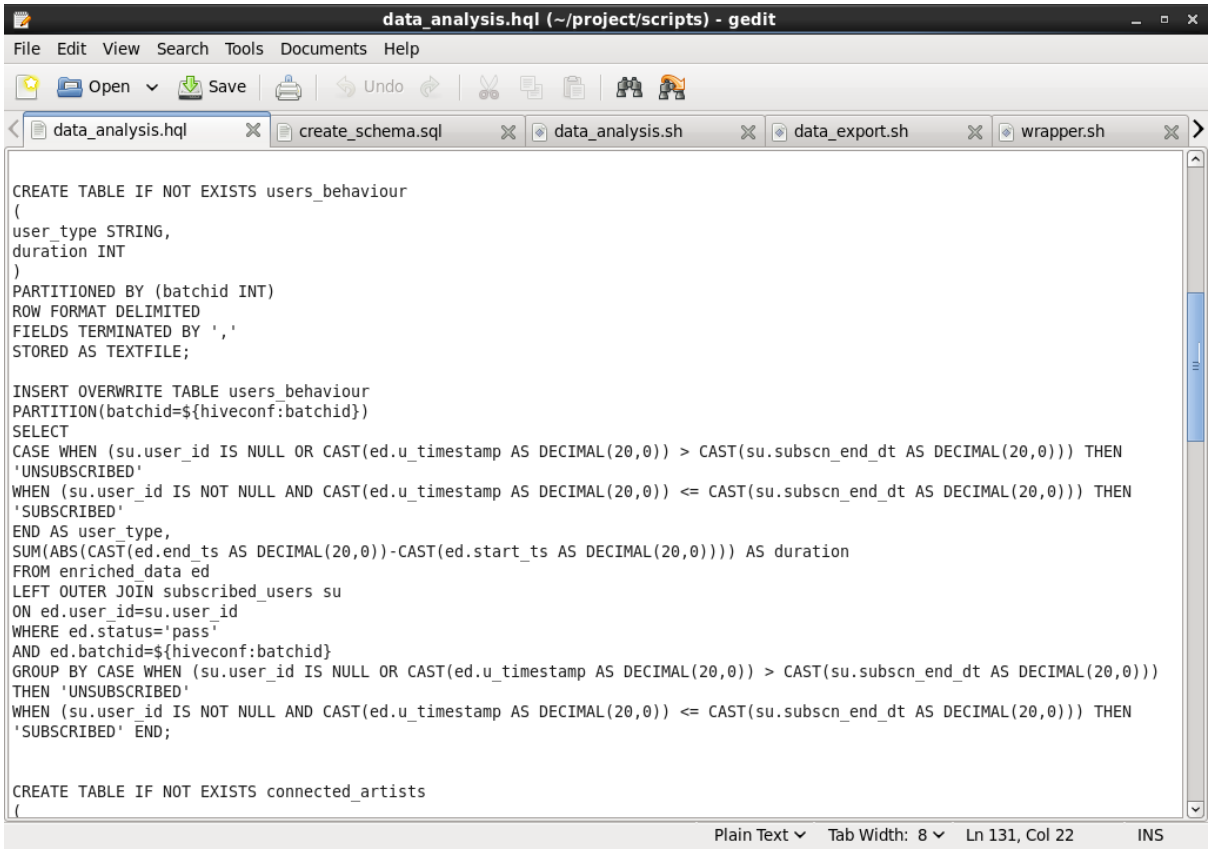
File Edit View Search Tools Documents Help

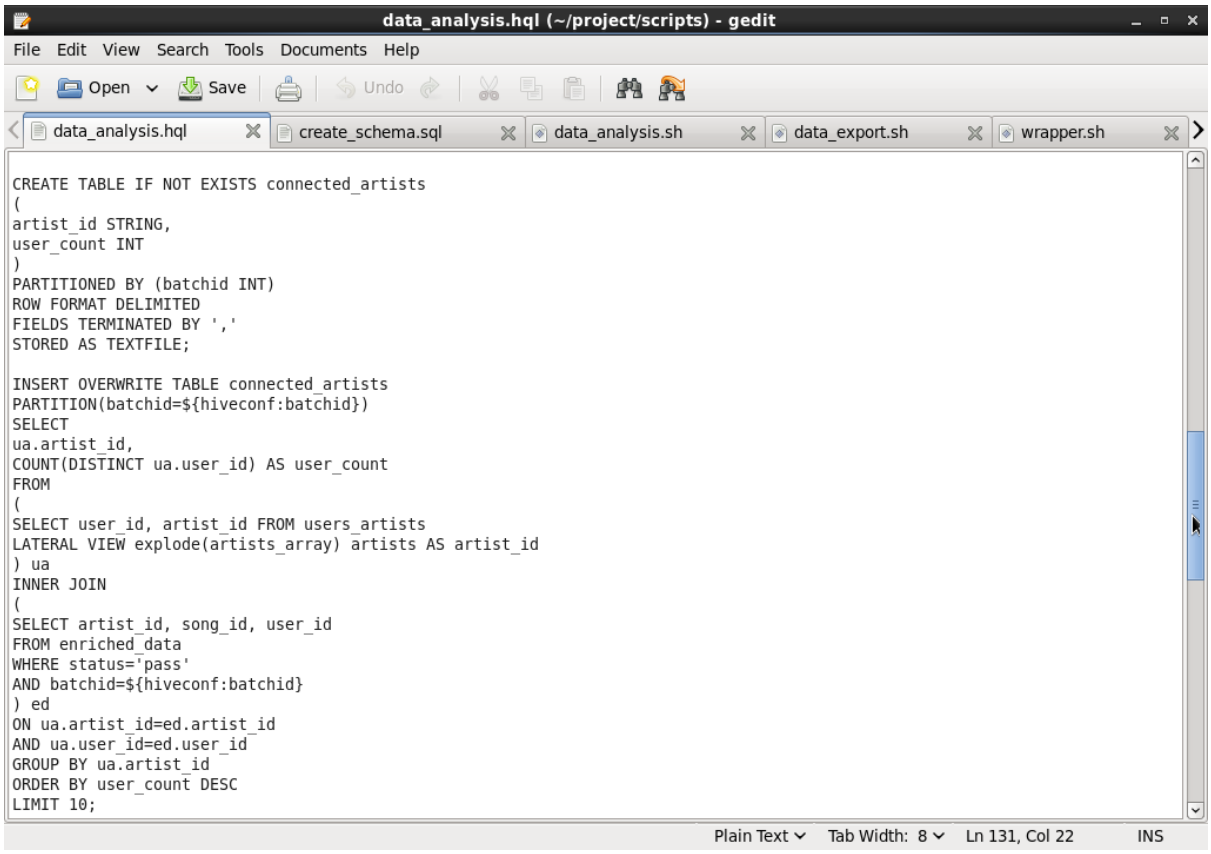
Open Save Undo

data_analysis.hql | create_schema.sql | data_analysis.sh | data_export.sh | wrapper.sh

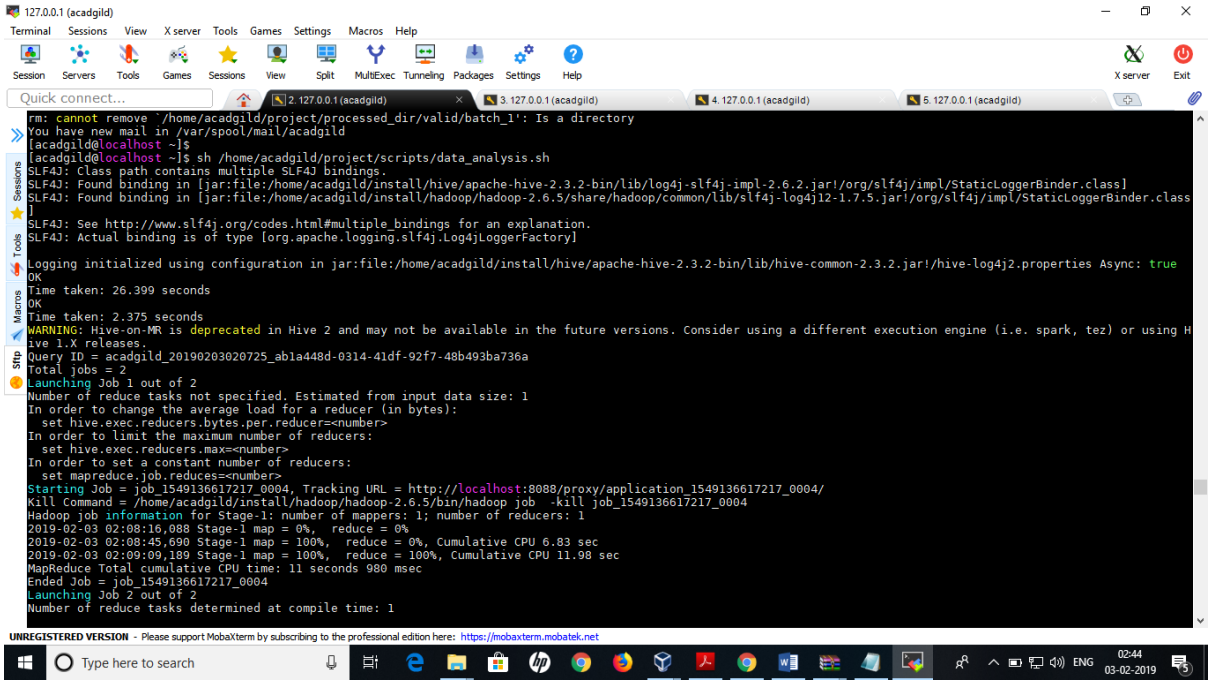
```
SET hive.auto.convert.join=false;
USE project;

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 u_like=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
```

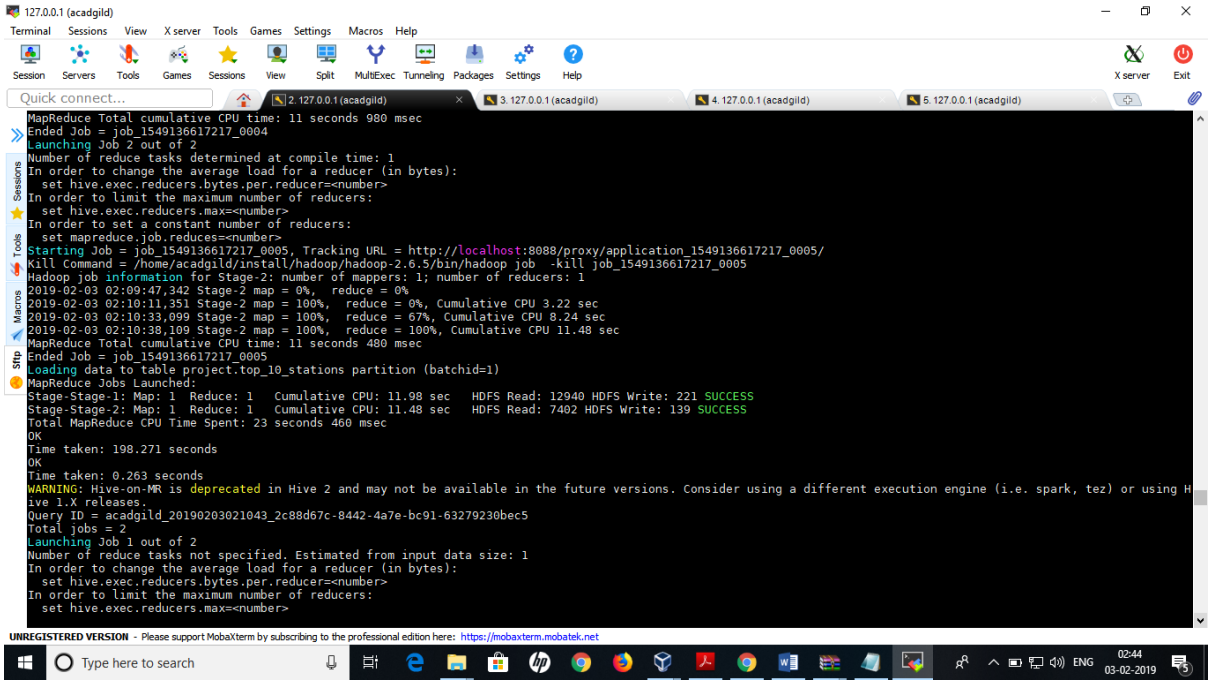
Plain Text Tab Width: 8 Ln 131, Col 22 INS

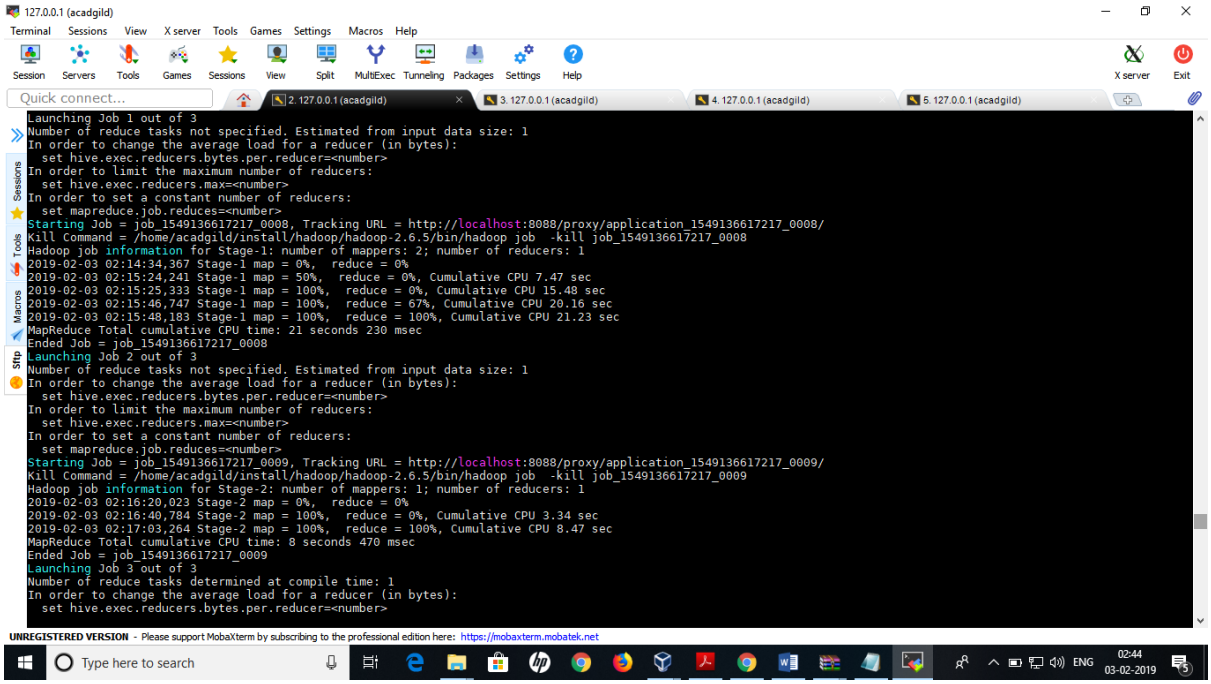




Command: sh /home/acadgild/project/scripts/data_analysis.sh


127.0.0.1 (acadgild)
Terminal Sessions View X server Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split MultiExec Tunneling Packages Settings Help X server Exit
Quick connect...
2. 127.0.0.1 (acadgild)
3. 127.0.0.1 (acadgild)
4. 127.0.0.1 (acadgild)
5. 127.0.0.1 (acadgild)
rm: cannot remove `/home/acadgild/project/processed_dir/valid/batch_1': Is a directory
You have new mail in /var/spool/mail/acadgild
[acadgild@localhost ~]$
[acadgild@localhost ~]$ sh /home/acadgild/project/scripts/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!/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: 26.399 seconds
OK
Time taken: 2.375 seconds
WARNING: 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.
Query ID = acadgild_20190203020725_ab1a448d-0314-41df-92f7-48b493ba736a
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_1549136617217_0004, Tracking URL = http://localhost:8088/proxy/application_1549136617217_0004/
Kill Command = /home/acadgild/install/hadoop/hadoop-2.6.5/bin/hadoop job -kill job_1549136617217_0004
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2019-02-03 02:08:16,088 Stage-1 map = 0%, reduce = 0%
2019-02-03 02:08:45,690 Stage-1 map = 100%, reduce = 0%, Cumulative CPU 6.83 sec
2019-02-03 02:09:09,189 Stage-1 map = 100%, reduce = 100%, Cumulative CPU 11.98 sec
MapReduce Total cumulative CPU time: 11 seconds 980 msec
Ended Job = job_1549136617217_0004
Launching Job 2 out of 2
Number of reduce tasks determined at compile time: 1
UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: https://mobaxterm.mobatek.net
Type here to search
ENG 02:44 03-02-2019


127.0.0.1 (acadgild)
Terminal Sessions View X server Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split MultiExec Tunneling Packages Settings Help X server Exit
Quick connect...
2. 127.0.0.1 (acadgild)
3. 127.0.0.1 (acadgild)
4. 127.0.0.1 (acadgild)
5. 127.0.0.1 (acadgild)
MapReduce Total cumulative CPU time: 11 seconds 980 msec
Ended Job = job_1549136617217_0004
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>
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_1549136617217_0005, Tracking URL = http://localhost:8088/proxy/application_1549136617217_0005/
Kill Command = /home/acadgild/install/hadoop/hadoop-2.6.5/bin/hadoop job -kill job_1549136617217_0005
Hadoop job information for Stage-2: number of mappers: 1; number of reducers: 1
2019-02-03 02:09:47,342 Stage-2 map = 0%, reduce = 0%
2019-02-03 02:10:11,351 Stage-2 map = 100%, reduce = 0%, Cumulative CPU 3.22 sec
2019-02-03 02:10:33,099 Stage-2 map = 100%, reduce = 67%, Cumulative CPU 8.24 sec
2019-02-03 02:10:38,109 Stage-2 map = 100%, reduce = 100%, Cumulative CPU 11.48 sec
MapReduce Total cumulative CPU time: 11 seconds 480 msec
Ended Job = job_1549136617217_0005
Loading data to table project.top_10_stations partition (batchid=1)
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1 Reduce: 1 Cumulative CPU: 11.98 sec HDFS Read: 12940 HDFS Write: 221 SUCCESS
Stage-Stage-2: Map: 1 Reduce: 1 Cumulative CPU: 11.48 sec HDFS Read: 7402 HDFS Write: 139 SUCCESS
Total MapReduce CPU Time Spent: 23 seconds 460 msec
OK
Time taken: 198.271 seconds
OK
Time taken: 0.263 seconds
WARNING: 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.
Query ID = acadgild_20190203021043_2c88d67c-8442-4a7e-bc91-63279230bec5
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>
UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: https://mobaxterm.mobatek.net
Type here to search
ENG 02:44 03-02-2019



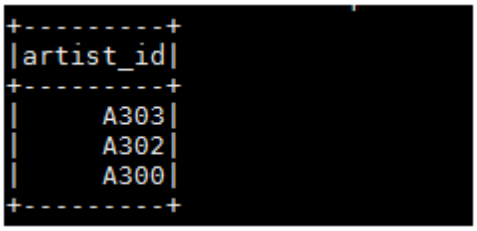
Query-1: Determine top 10 **station_id(s)** where maximum number of songs were played, which were liked by unique users.

```
+----------+
|station_id|
+----------+
|      ST407|
|      ST414|
|      ST411|
|      ST402|
|      ST406|
|      ST405|
+----------+
```

Query-2: Determine total duration of songs played by each type of user, where type of user can be **'subscribed'** or **'unsubscribed'**. An unsubscribed user is the one whose record is either not present in Subscribed_users lookup table or has subscription_end_date earlier than the timestamp of the song played by him.

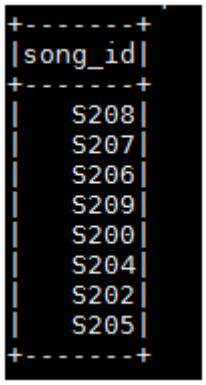
```
+------------+---------+
|   user_type| duration|
+------------+---------+
|  SUBSCRIBED| 93861594|
|UNSUBSCRIBED|105594881|
+------------+---------+
```

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



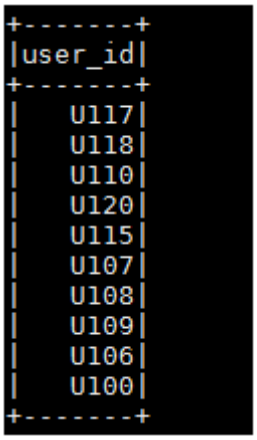
artist_id
A303
A302
A300

Query-4: Determine top 10 songs who have generated the maximum revenue. Royalty applies to a song only if it was liked or was completed successfully or both



song_id
S208
S207
S206
S209
S200
S204
S202
S205

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



user_id
U117
U118
U110
U120
U115
U107
U108
U109
U106
U100

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

```
Time taken: 0.097 seconds, Fetched: 11 row(s)
hive> Select * From connected_artists;
OK
connected_artists.artist_id     connected_artists.user_count    connected_artists.batchid
A303    2       1
A302    2       1
A300    1       1
Time taken: 0.225 seconds, Fetched: 3 row(s)
hive> Select * From top_10_royalty_songs;
OK
top_10_royalty_songs.song_id    top_10_royalty_songs.duration   top_10_royalty_songs.batchid
S208    22627294        1
S207    20000000        1
S206    19900000        1
S209    15254588        1
S200    9900000 1
S204    2604333 1
S202    100000  1
S205    0       1
Time taken: 0.237 seconds, Fetched: 8 row(s)
hive> Select * From top_10_stations;
OK
top_10_stations.station_id      top_10_stations.total_distinct_songs_played     top_10_stations.distinct_user_count     top_10_stations.batchid
ST407   2       3       1
ST414   1       1       1
ST411   1       1       1
ST402   1       2       1
ST406   1       1       1
ST405   1       1       1
Time taken: 0.336 seconds, Fetched: 6 row(s)
hive> Select * From top_10_unsubscribed_users;
OK
top_10_unsubscribed_users.user_id       top_10_unsubscribed_users.duration      top_10_unsubscribed_users.batchid
U117    20000000        1
U118    20000000        1
U110    20000000        1
U120    12627294        1
U115    12527294        1
```

```
Time taken: 0.237 seconds, Fetched: 8 row(s)
hive> Select * From top_10_stations;
OK
top_10_stations.station_id      top_10_stations.total_distinct_songs_played     top_10_stations.distinct_user_count     top_10_stations.batchid
ST407   2       3       1
ST414   1       1       1
ST411   1       1       1
ST402   1       2       1
ST406   1       1       1
ST405   1       1       1
Time taken: 0.336 seconds, Fetched: 6 row(s)
hive> Select * From top_10_unsubscribed_users;
OK
top_10_unsubscribed_users.user_id       top_10_unsubscribed_users.duration      top_10_unsubscribed_users.batchid
U117    20000000        1
U118    20000000        1
U110    20000000        1
U120    12627294        1
U115    12527294        1
U107    10000000        1
U108    5231627 1
U109    2604333 1
U106    2604333 1
U100    0       1
Time taken: 0.275 seconds, Fetched: 10 row(s)
hive> Select * From users_behaviour;
OK
users_behaviour.user_type       users_behaviour.duration        users_behaviour.batchid
SUBSCRIBED      93861594        1
UNSUBSCRIBED    105594881       1
Time taken: 0.274 seconds, Fetched: 2 row(s)
hive>
    >
    >
```

Section 7 – 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.

create_schema.sql – Make sure that you logged in to MySql. The below schema will create the database and tables in the MySQl.

create_schema.sql (~/project/scripts) - gedit

File Edit View Search Tools Documents Help

Open Save Undo

data_analysis.hql | create_schema.sql | data_analysis.sh | data_export.sh | wrapper.sh

```
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
```

SQL Tab Width: 8 Ln 1, Col 1 INS

data_export.sh

data_export.sh (~/project/scripts) - gedit

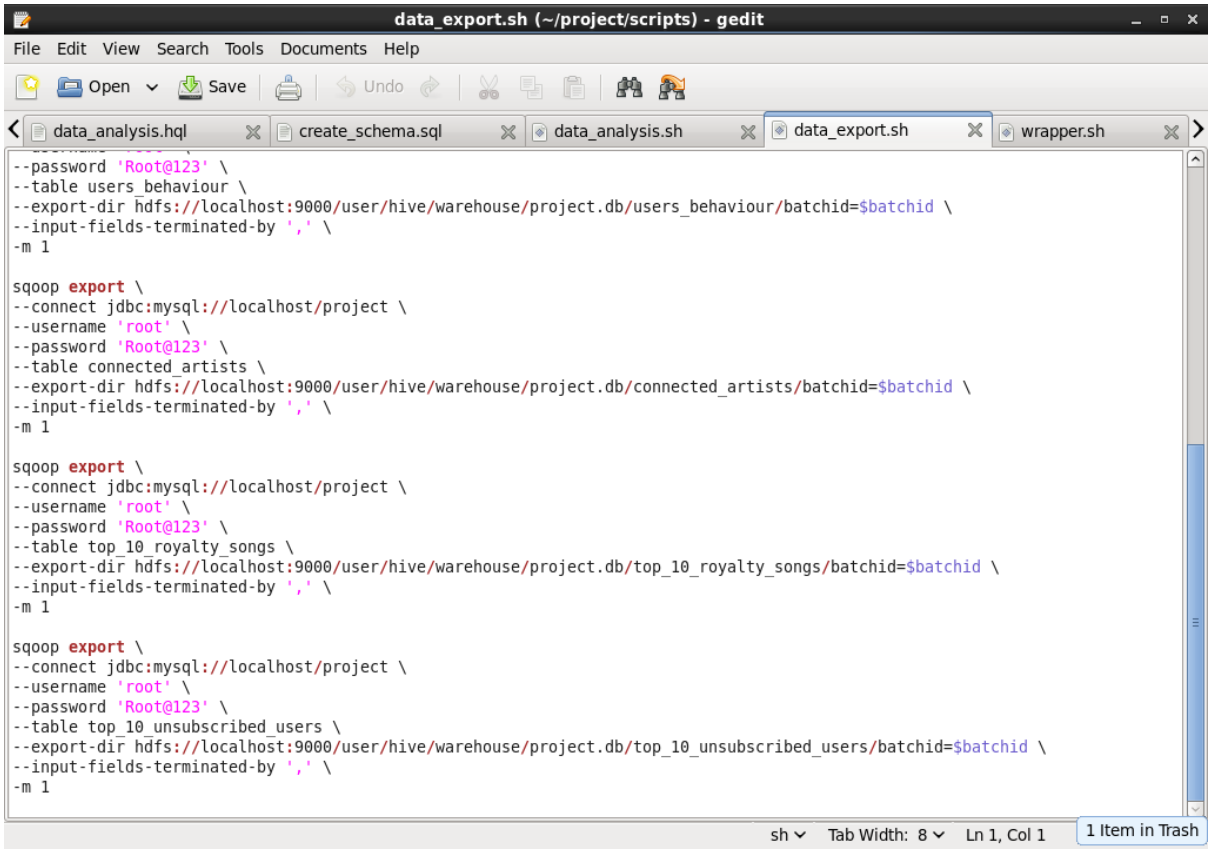
File Edit View Search Tools Documents Help

Open Save Undo

data_analysis.hql | create_schema.sql | data_analysis.sh | data_export.sh | wrapper.sh

```
#!/bin/bash

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

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

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

mysql -u root < /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 'Root@123' \
--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 'Root@123' \
--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 \
```

sh Tab Width: 8 Ln 1, Col 1 INS




```
 stack guard. The VM will try to fix the stack guard now.
It's highly recommended that you fix the library with 'execstack -c <libfile>', or link it with '-z noexecstack'.
18/01/24 09:57:24 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applic
able
Found 1 items
drwxrwxr-x   - acadgild supergroup          0 2018-01-24 09:34 hdfs://localhost:9000/user/hive/warehouse/project.db/top_10_stations/batchid=1
[acadgild@localhost ~]$ sqoop export --connect jdbc:mysql://localhost/project --username root --password acadgild --table top_10_stations --exp
ort-dir hdfs://localhost:9000/user/hive/warehouse/project.db/top_10_stations/batchid=1 --input-fields-terminated-by ',' -m 1
Warning: /home/acadgild/sqoop-1.4.6.bin__hadoop-2.0.4-alpha/../hcatalog does not exist! HCatalog jobs will fail.
Please set $HCAT_HOME to the root of your HCatalog installation.
Warning: /home/acadgild/sqoop-1.4.6.bin__hadoop-2.0.4-alpha/../accumulo does not exist! Accumulo imports will fail.
Please set $ACCUMULO_HOME to the root of your Accumulo installation.
Warning: /home/acadgild/sqoop-1.4.6.bin__hadoop-2.0.4-alpha/../zookeeper does not exist! Accumulo imports will fail.
Please set $ZOOKEEPER_HOME to the root of your Zookeeper installation.
2018-01-24 09:58:23,657 INFO  [main] sqoop.Sqoop: Running Sqoop version: 1.4.6
2018-01-24 09:58:23,694 WARN  [main] tool.BaseSqoopTool: Setting your password on the command-line is insecure. Consider using -P instead.
2018-01-24 09:58:24,084 INFO  [main] manager.MySQLManager: Preparing to use a MySQL streaming resultset.
2018-01-24 09:58:24,085 INFO  [main] tool.CodeGenTool: Beginning code generation
2018-01-24 09:58:24,696 INFO  [main] manager.SqlManager: Executing SQL statement: SELECT t.* FROM `top_10_stations` AS t LIMIT 1
2018-01-24 09:58:24,767 INFO  [main] manager.SqlManager: Executing SQL statement: SELECT t.* FROM `top_10_stations` AS t LIMIT 1
2018-01-24 09:58:24,810 INFO  [main] orm.CompilationManager: HADOOP_MAPRED_HOME is /home/acadgild/hadoop-2.7.2
Note: /tmp/sqoop-acadgild/compile/f1ae2653abc7121116a39d1b97e8735b/top_10_stations.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
2018-01-24 09:58:29,735 INFO  [main] orm.CompilationManager: Writing jar file: /tmp/sqoop-acadgild/compile/f1ae2653abc7121116a39d1b97e8735b/top
_10_stations.jar
2018-01-24 09:58:29,763 INFO  [main] mapreduce.ExportJobBase: Beginning export of top_10_stations
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/acadgild/hbase-1.0.3/lib/slf4j-log4j12-1.7.7.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/acadgild/hadoop-2.7.2/share/hadoop/common/lib/slf4j-log4j12-1.7.10.jar!/org/slf4j/impl/StaticLoggerBind
er.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
Java HotSpot(TM) Client VM warning: You have loaded library /home/acadgild/hadoop-2.7.2/lib/native/libhadoop.so.1.0.0 which might have disabled
 stack guard. The VM will try to fix the stack guard now.
It's highly recommended that you fix the library with 'execstack -c <libfile>', or link it with '-z noexecstack'.
2018-01-24 09:58:30,180 WARN  [main] util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classe
s where applicable
2018-01-24 09:58:30,199 INFO  [main] Configuration.deprecation: mapred.jar is deprecated. Instead, use mapreduce.job.jar
2018-01-24 09:58:32,079 INFO  [main] Configuration.deprecation: mapred.reduce.tasks.speculative.execution is deprecated. Instead, use mapreduce
```

```
2018-01-24 10:06:14,238 INFO  [main] input.FileInputFormat: Total input paths to process : 1
2018-01-24 10:06:14,435 INFO  [main] mapreduce.JobSubmitter: number of splits:1
2018-01-24 10:06:14,464 INFO  [main] Configuration.deprecation: mapred.map.tasks.speculative.execution is deprecated. Instead, use mapreduce.ma
p.speculative
2018-01-24 10:06:14,791 INFO  [main] mapreduce.JobSubmitter: Submitting tokens for job: job_1516764714140_0018
2018-01-24 10:06:16,203 INFO  [main] impl.YarnClientImpl: Submitted application application_1516764714140_0018
2018-01-24 10:06:16,420 INFO  [main] mapreduce.Job: The url to track the job: http://localhost:8088/proxy/application_1516764714140_0018/
2018-01-24 10:06:16,424 INFO  [main] mapreduce.Job: Running job: job_1516764714140_0018
2018-01-24 10:06:45,186 INFO  [main] mapreduce.Job: Job job_1516764714140_0018 running in uber mode : false
2018-01-24 10:06:45,191 INFO  [main] mapreduce.Job:  map 0% reduce 0%
2018-01-24 10:07:01,634 INFO  [main] mapreduce.Job:  map 100% reduce 0%
2018-01-24 10:07:02,707 INFO  [main] mapreduce.Job: Job job_1516764714140_0018 completed successfully
2018-01-24 10:07:03,246 INFO  [main] mapreduce.Job: Counters: 30
        File System Counters
                FILE: Number of bytes read=0
                FILE: Number of bytes written=136426
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=311
                HDFS: Number of bytes written=0
                HDFS: Number of read operations=4
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=0
        Job Counters
                Launched map tasks=1
                Data-local map tasks=1
                Total time spent by all maps in occupied slots (ms)=12452
                Total time spent by all reduces in occupied slots (ms)=0
                Total time spent by all map tasks (ms)=12452
                Total vcore-seconds taken by all map tasks=12452
                Total megabyte-seconds taken by all map tasks=12750848
        Map-Reduce Framework
                Map input records=10
                Map output records=10
                Input split bytes=178
                Spilled Records=0
                Failed Shuffles=0
                Merged Map outputs=0
```

Output:

```
mysql>
mysql> use project;
Database changed
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 (0.00 sec)

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

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

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

```
mysql> Select * From top_10_unsubscribed_users;
+---------+----------+
| user_id | duration |
+---------+----------+
| U117    | 20000000 |
| U118    | 20000000 |
| U110    | 20000000 |
| U120    | 12627294 |
| U115    | 12527294 |
| U107    | 10000000 |
| U108    |  5231627 |
| U109    |  2604333 |
| U106    |  2604333 |
| U100    |        0 |
+---------+----------+
10 rows in set (0.01 sec)

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

Job Scheduling:

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

```
21  --driver-class-path /home/acadgild/apache-hive-2.1.0-bin/lib/hive-hbase-handler-
22  /home/acadgild/project/lib/sparkanalysis.jar $batchid
23
24  echo "Exporting data to MYSQL using sqoop export..." >> $LOGFILE
25  sh /home/acadgild/project/scripts/data_export.sh
26
27  echo "Incrementing batchid..." >> $LOGFILE
28  batchid=`expr $batchid + 1`
29  echo -n $batchid > /home/acadgild/project/logs/current-batch.txt
30
```

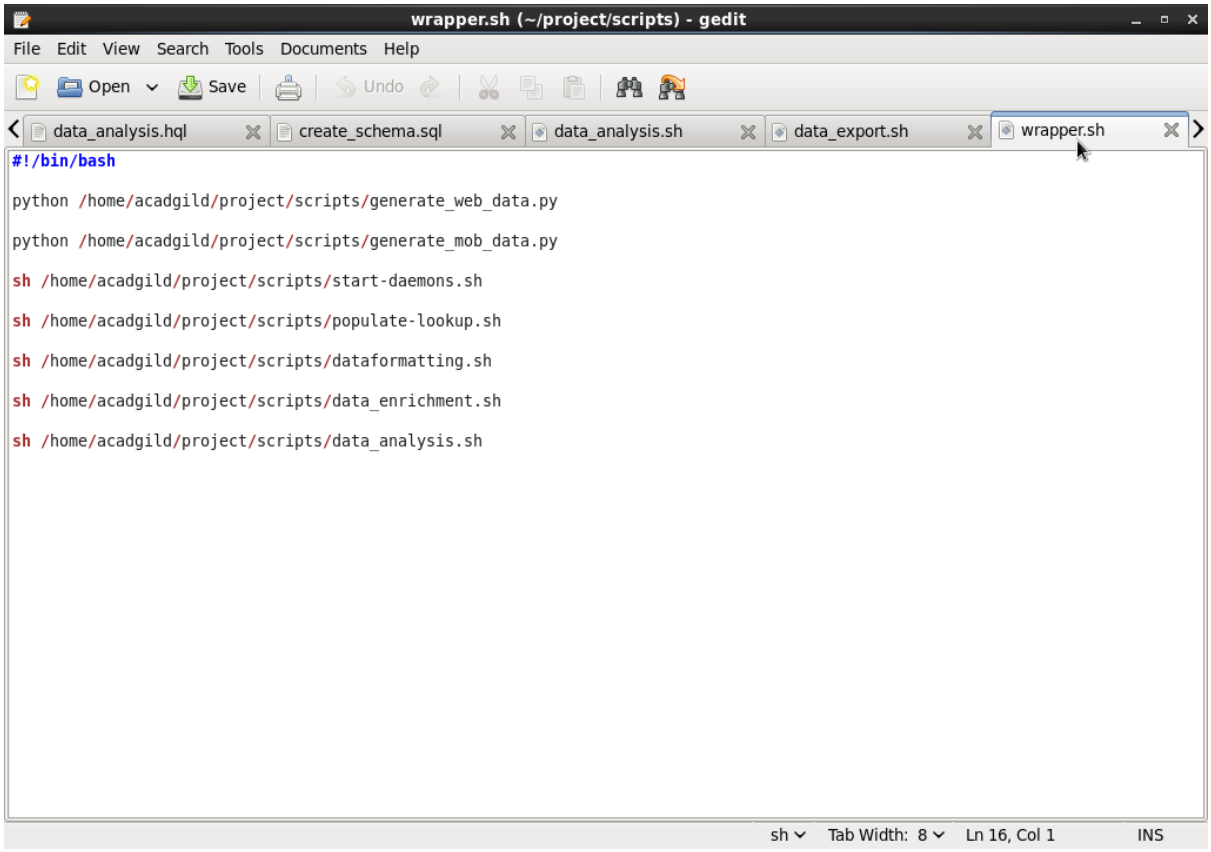
We can check logs to track the behavior of the operations we have done on the data and overcome failures in the pipeline and we can see the **batchid** incremented value in **current-batch.txt**

```
[acadgild@localhost project]$ cd logs
[acadgild@localhost logs]$ ls -l
total 24
-rwxrwxr-x. 1 acadgild acadgild    1 Jan 24 09:44 current-batch.txt
-rw-rw-r--. 1 acadgild acadgild  679 Jan 24 09:03 derby.log
drwxrwxr-x. 3 acadgild acadgild 4096 Jan 24 09:02 hdfs:
-rw-rw-r--. 1 acadgild acadgild  523 Jan 24 09:44 log_batch_1
-rw-rw-r--. 1 acadgild acadgild   77 Jan 24 09:44 log_batch_1???
drwxrwxr-x. 5 acadgild acadgild 4096 Jan 24 09:03 metastore_db
[acadgild@localhost logs]$ cat current-batch.txt
2[acadgild@localhost logs]$
[acadgild@localhost logs]$
[acadgild@localhost logs]$
```

```
[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]$
```

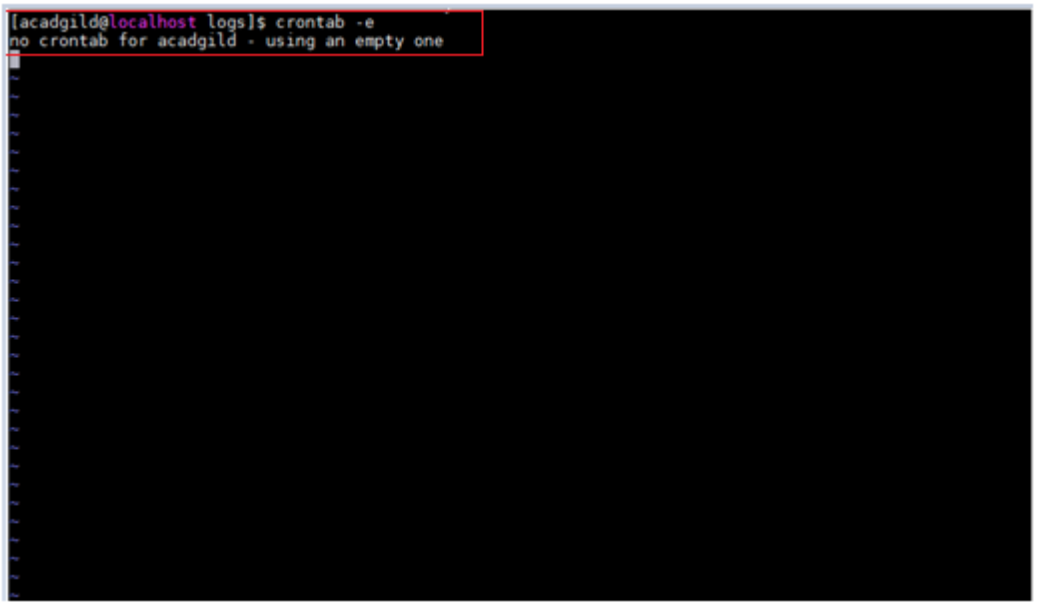
Wrapping all the scripts inside the single script file and scheduling this file to run at the periodic interval of every 3 hours.

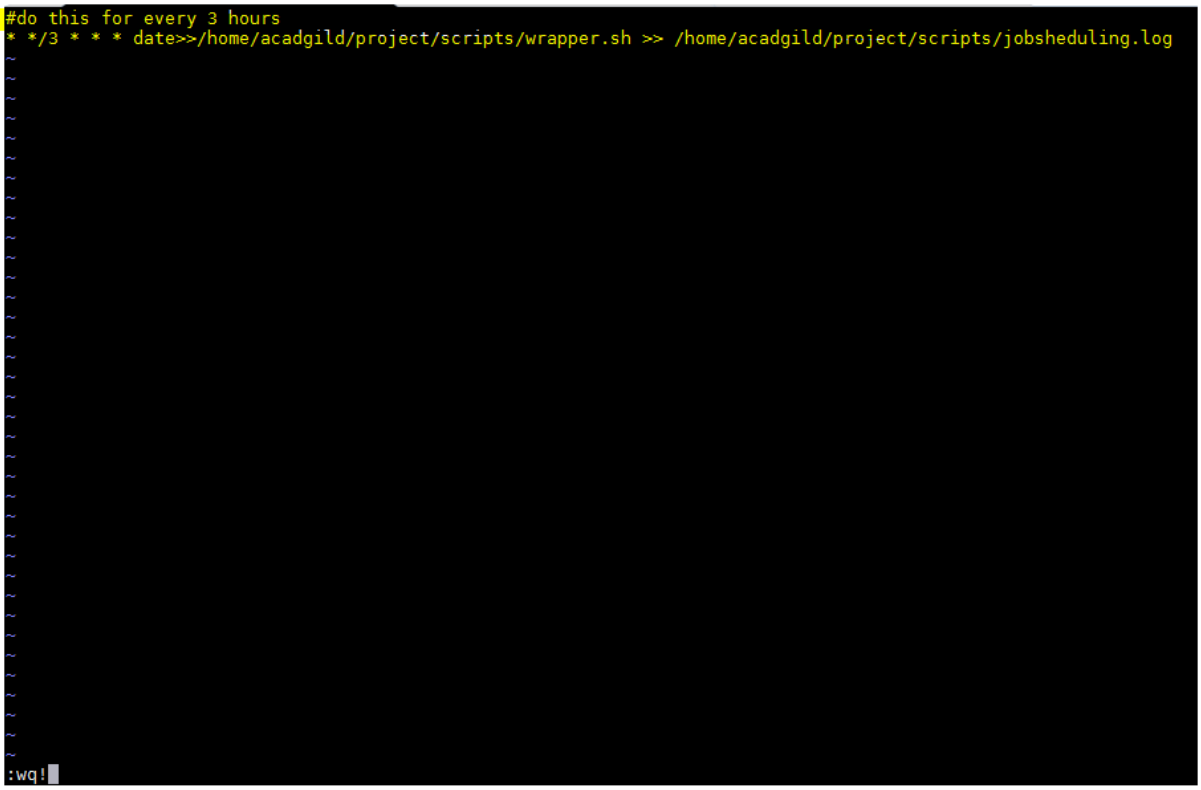
wrapper.sh

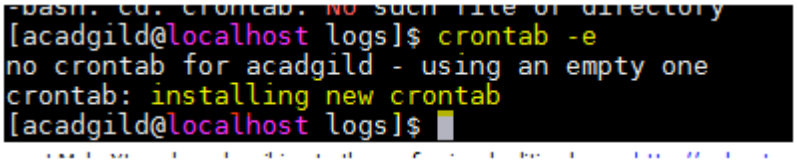


The wrapper.sh will be running for every 3 hours as per the job scheduling done below, as per the above order the wrapper.sh will run the scripts.

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







*Installing the **crontab** in the vm,*

The **crontab** job scheduler will run the **wrappr.sh** every 3 hours and for every 3 hours we will get incremental batch ID's. **Hence, as per the request this job scheduling has been done.**