Project:-Music Data Analysis

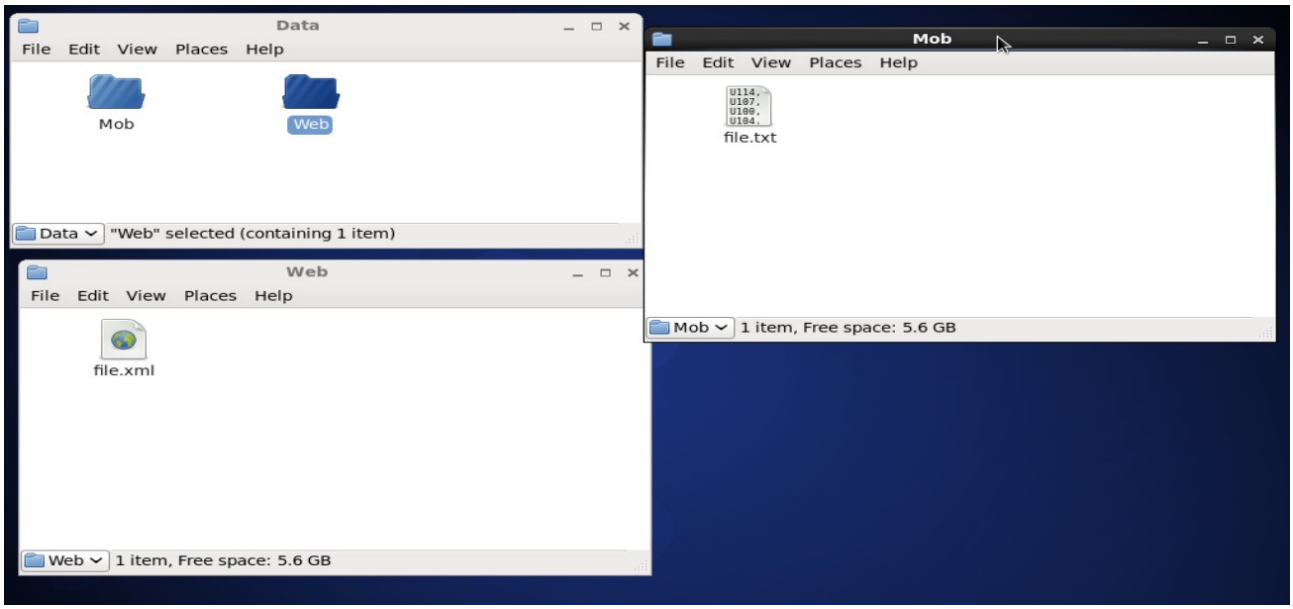
Problem Statement :-

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. we need to calculate the queries and return the results to business/customer.

Solution:-

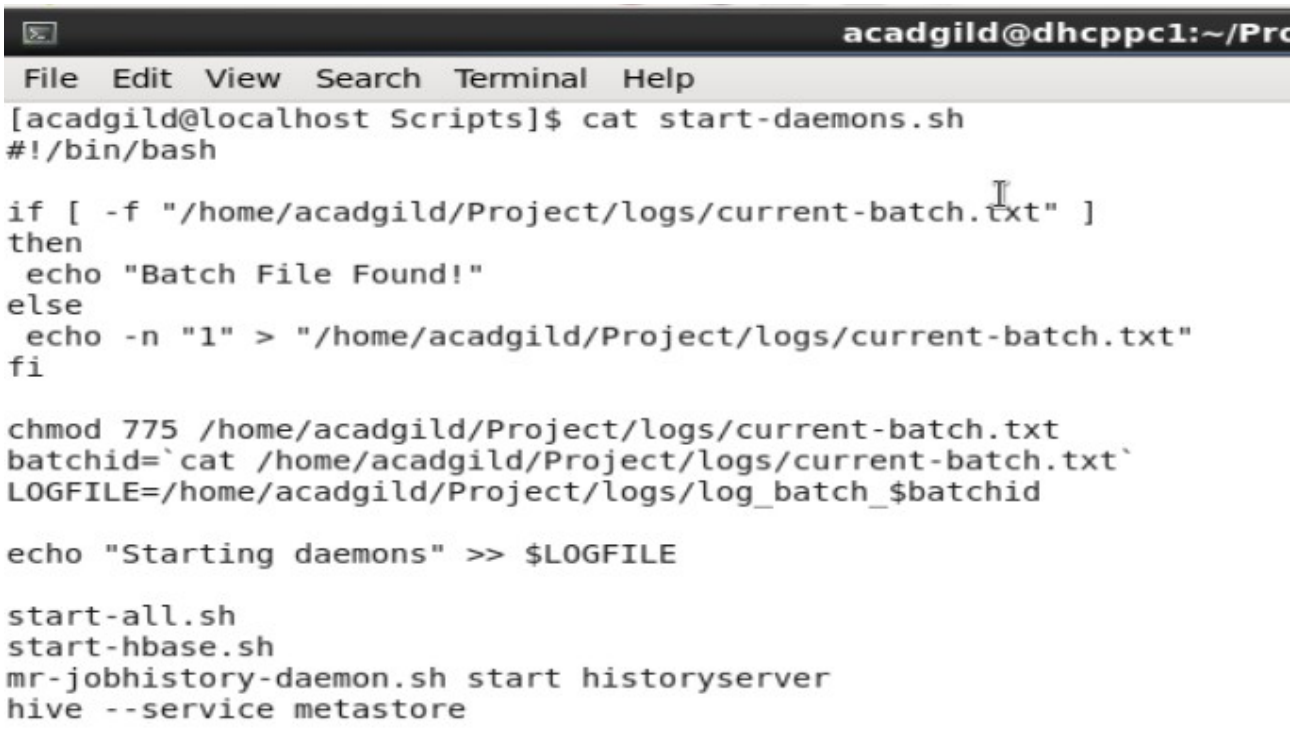
DATASET:

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.



Step 1: To launch required daemons.

- Launch all necessary daemons
- Launch the Mysql Service (needed for Hive)
- Run the shell script start-daemons.sh

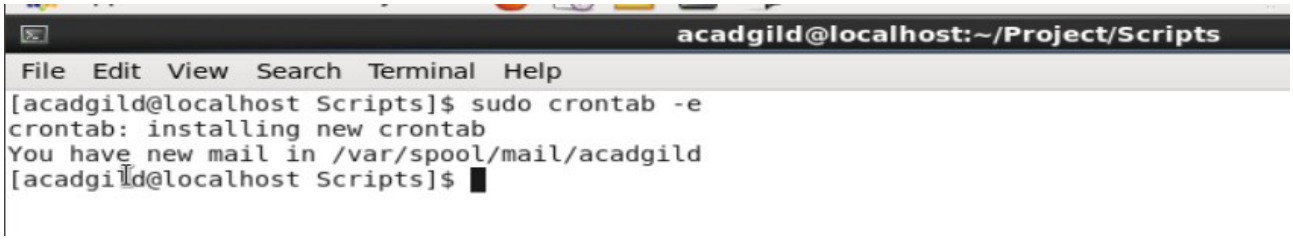


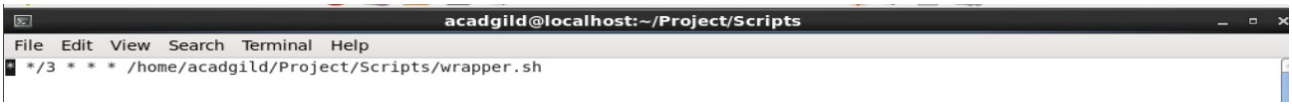
The script performs the operations:-

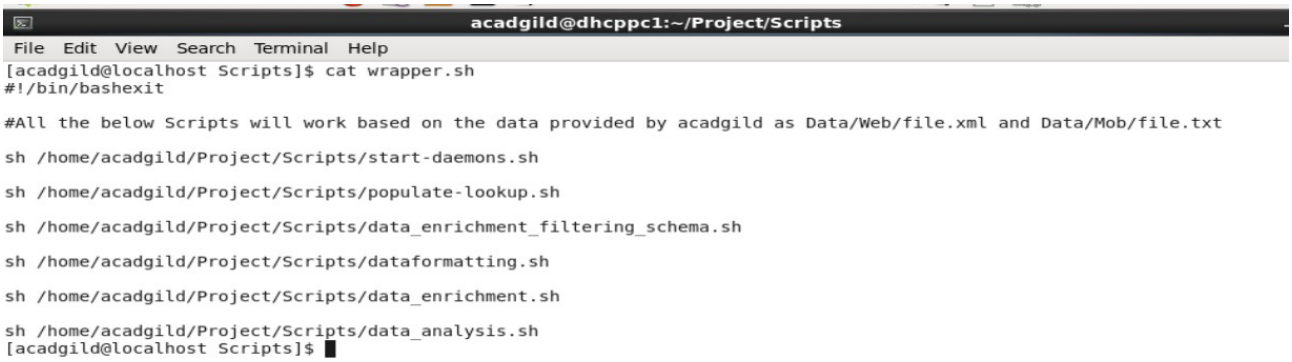
1. Check if a file current-batch.txt has been created or not,
2. If already created, print Batch File Found! else create the file and add 1 to it to signify batch 1.
3. Give permissions to the file, so that we are able to modify it on the run.
4. Get the batch id number from the batch file created above and create a Log File for the batch using the batch id. This will be log_batch_1.

Step 2 : Perform Job Scheduling

- The shell script **wrapper.sh** has all the scripts for scheduling all the jobs
- Using crontab to schedule a Job in the -e mode. We have scheduled this job for every 3 Hours.







In the shell script wrapper.sh used above, all the processes needed to perform analysis on the Music Data is called once every 3 hours thereby creating a new batch. This is the job scheduling.

Step 3: Populating LookUp tables

- Displaying the script **populate-lookup.sh** which creates tables in hbase and hive.

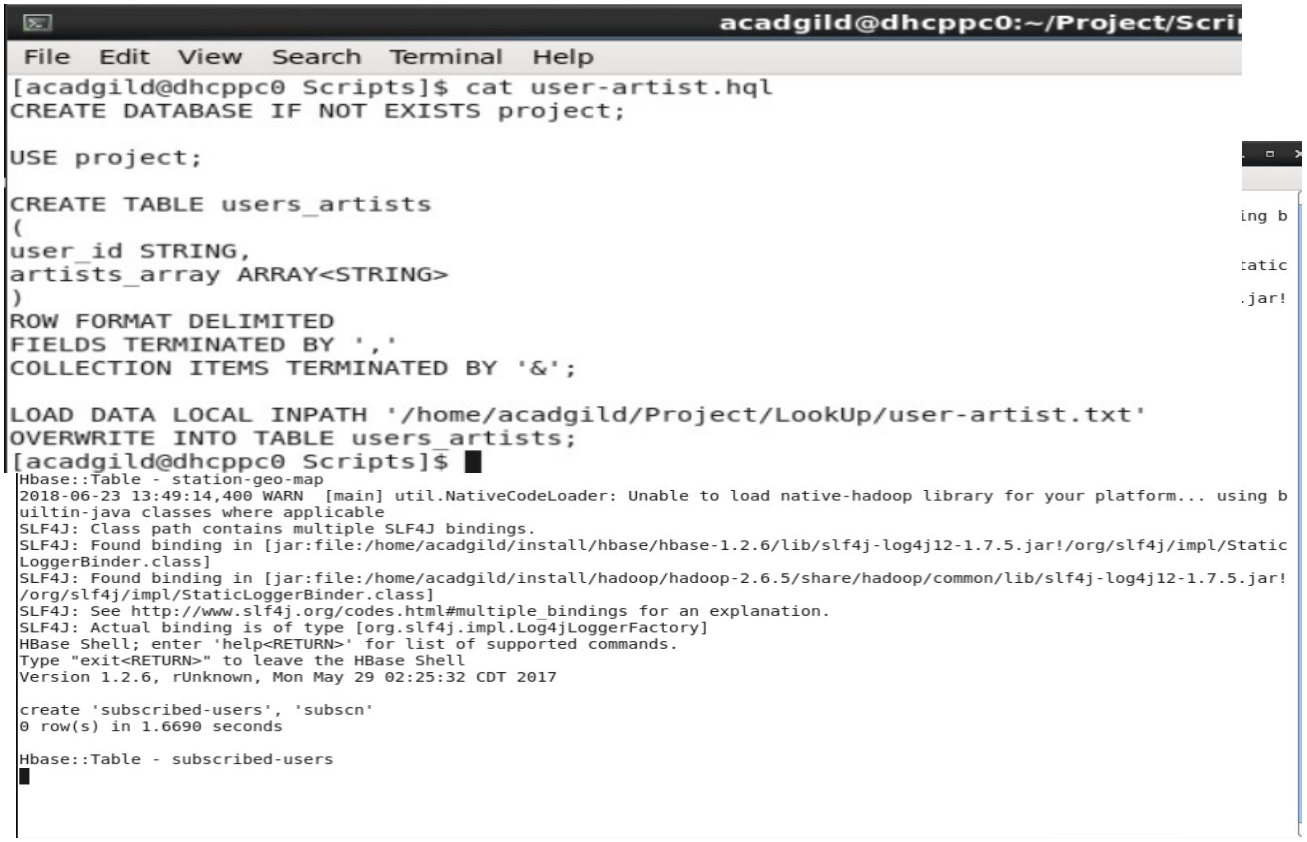
acadgild@dhcppc0:~/Project/Scripts

File Edit View Search Terminal Help

```
[acadgild@dhcppc0 Scripts]$ cat populate-lookup.sh
#!/bin/bash

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

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

echo "Creating LookUp Tables" >> $LOGFILE

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


echo "Populating LookUp Tables" >> $LOGFILE

file="/home/acadgild/Project/LookUp/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/LookUp/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/LookUp/user-subscn.txt"
while IFS= read -r line
do
 userid=`echo $line | cut -d',' -f1`
 startdt=`echo $line | cut -d',' -f2`
```

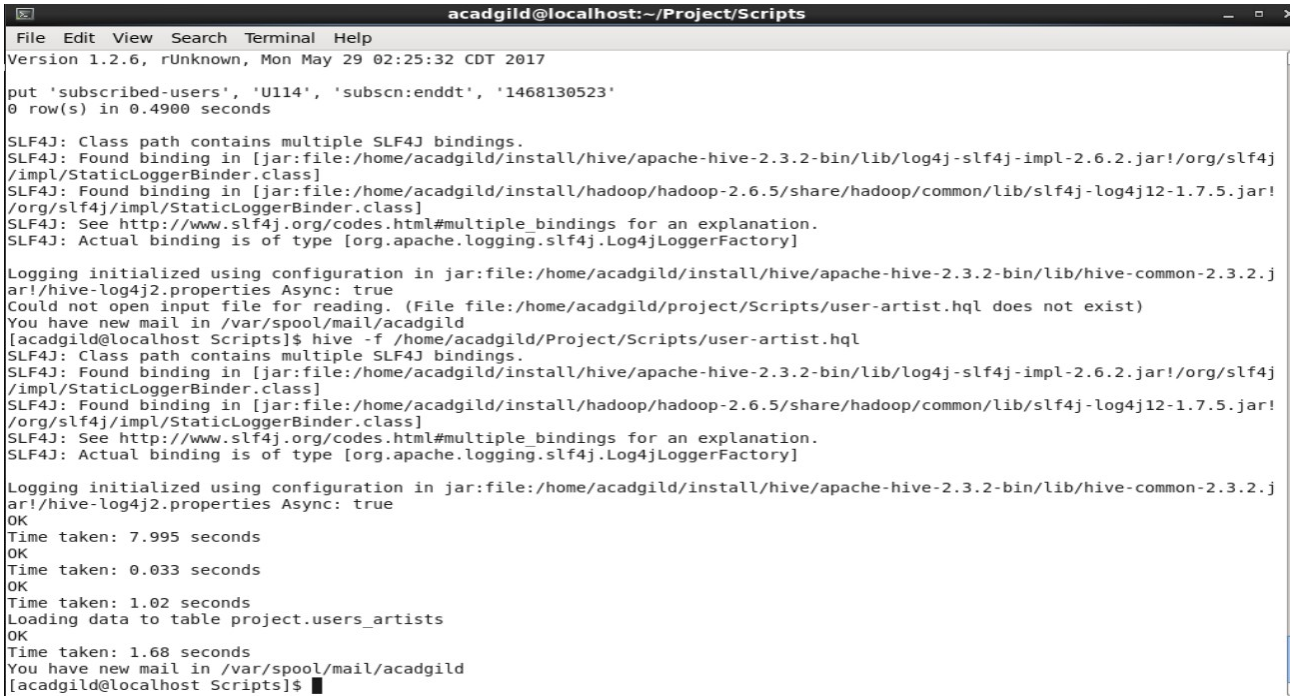
```
 enddt=`echo $line | cut -d',' -f3`
 echo "put 'subscribed-users', '$userid', 'subscn:startdt', '$startdt'" | hbase shell
 echo "put 'subscribed-users', '$userid', 'subscn:enddt', '$enddt'" | hbase shell
done <"$file"

hive -f /home/acadgild/Project/Scripts/user-artist.hql
```

- Displaying the **script user_artist.hql** which creates tables in hive



- Running populate-lookup.sh The following operations are performed:

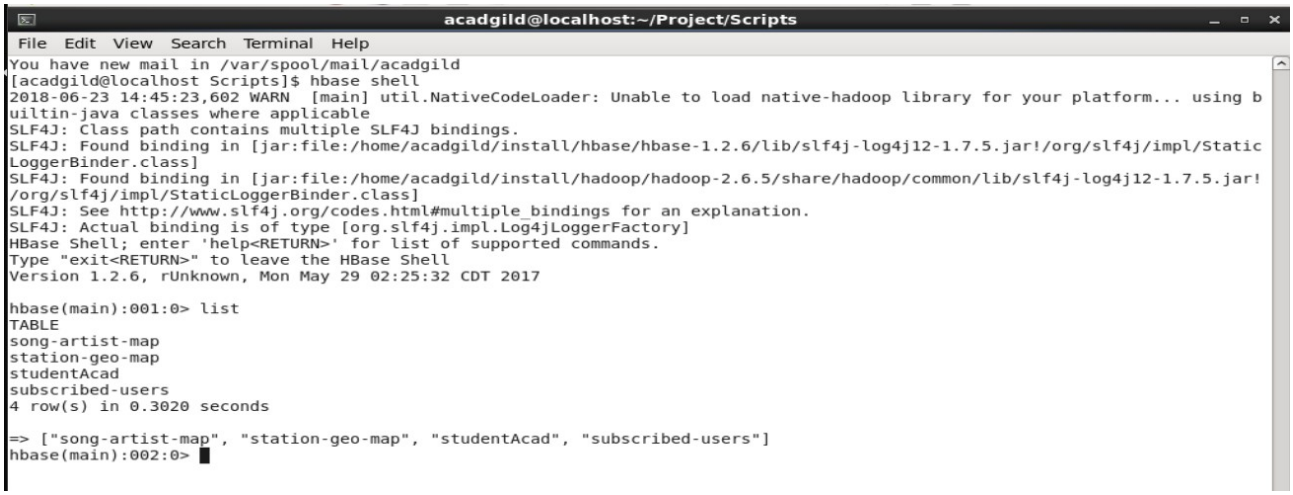
1. Get the batch id number from the batch file and get the Log File for the batch using the batch id. This will be batch1
2. Add logs to the Log File signifying that the lookup tables are being created and populated
3. Create the HBase tables for the lookup data files: song-artist, stn-geocd and user-subscn with their column families
4. For every lookup data file, read each line, extract the columns (comma separated) and add the data as rows to the corresponding HBase tables created above
5. Run the hive script user-artist.hql. This will populate a hive table with the data in the lookup data file user-artist. This is because this file has an array column that is difficult to populate in HBase.



By using the shell scripting , in the above screenshot we can see 4 lookup tables is created in Hbase NoSQL Database.

- Verifying the tables created in HBase

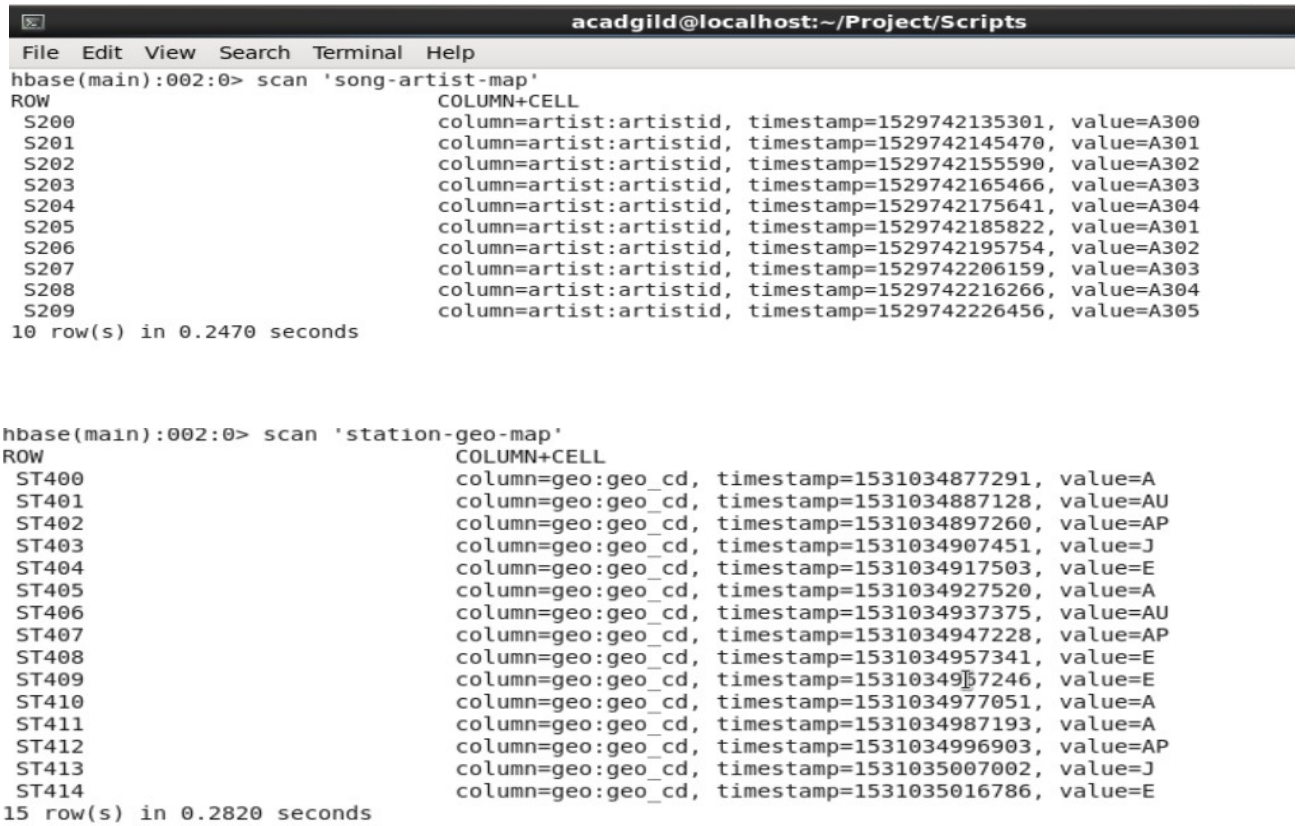
  Command used - **list**



In the above screenshot it is evident that in the hbase lookup tables are created. Output of the above (Hbase).

- Verifying the tables created in Hive

- Checking the content of look up tables in Hbase.

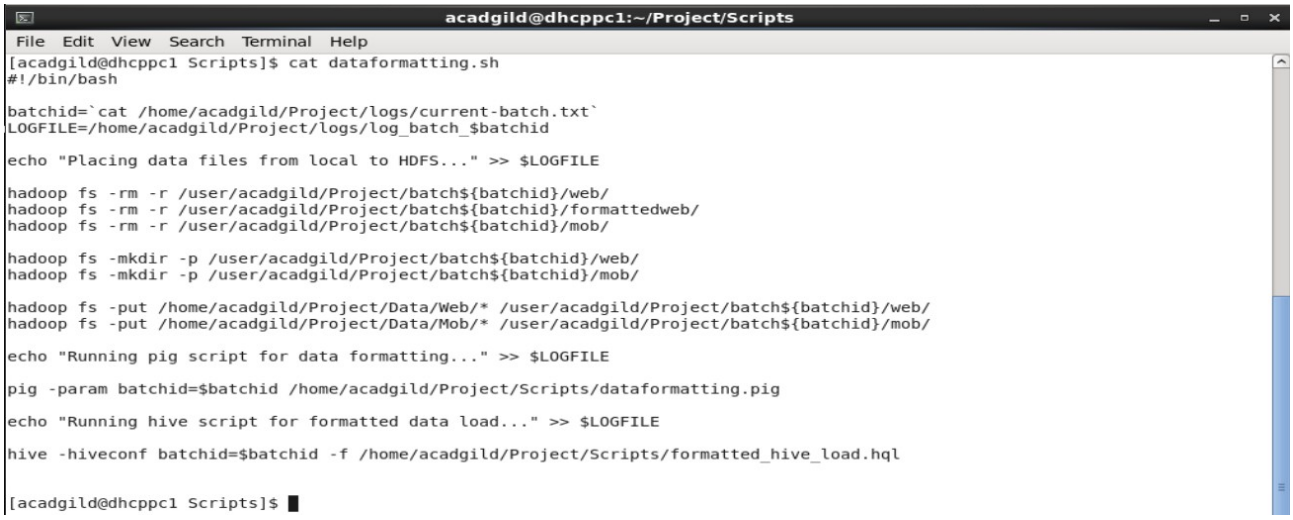


```
hbase(main):003:0> scan 'subscribed-users'
ROW                                COLUMN+CELL
 U100                              column=subscn:enddt, timestamp=1531035136421, value=1465130523
 U100                              column=subscn:startdt, timestamp=1531035126361, value=1465230523
 U101                              column=subscn:enddt, timestamp=1531035157128, value=1475130523
 U101                              column=subscn:startdt, timestamp=1531035147452, value=1465230523
 U102                              column=subscn:enddt, timestamp=1531035176769, value=1475130523
 U102                              column=subscn:startdt, timestamp=1531035167132, value=1465230523
 U103                              column=subscn:enddt, timestamp=1531035196456, value=1475130523
 U103                              column=subscn:startdt, timestamp=1531035186699, value=1465230523
 U104                              column=subscn:enddt, timestamp=1531035216070, value=1475130523
 U104                              column=subscn:startdt, timestamp=1531035206199, value=1465230523
 U105                              column=subscn:enddt, timestamp=1531035235561, value=1475130523
 U105                              column=subscn:startdt, timestamp=1531035225860, value=1465230523
 U106                              column=subscn:enddt, timestamp=1531035255309, value=1485130523
 U106                              column=subscn:startdt, timestamp=1531035245143, value=1465230523
 U107                              column=subscn:enddt, timestamp=1531035275057, value=1455130523
 U107                              column=subscn:startdt, timestamp=1531035265229, value=1465230523
 U108                              column=subscn:enddt, timestamp=1531035294922, value=1465230623
 U108                              column=subscn:startdt, timestamp=1531035285150, value=1465230523
 U109                              column=subscn:enddt, timestamp=1531035314672, value=1475130523
 U109                              column=subscn:startdt, timestamp=1531035305023, value=1465230523
 U110                              column=subscn:enddt, timestamp=1531035334181, value=1475130523
 U110                              column=subscn:startdt, timestamp=1531035324573, value=1465230523
 U111                              column=subscn:enddt, timestamp=1531035353901, value=1475130523
 U111                              column=subscn:startdt, timestamp=1531035343911, value=1465230523
 U112                              column=subscn:enddt, timestamp=1531035373772, value=1475130523
 U112                              column=subscn:startdt, timestamp=1531035363985, value=1465230523
 U113                              column=subscn:enddt, timestamp=1531035393295, value=1485130523
 U113                              column=subscn:startdt, timestamp=1531035383595, value=1465230523
 U114                              column=subscn:enddt, timestamp=1531035413135, value=1468130523
 U114                              column=subscn:startdt, timestamp=1531035403171, value=1465230523
15 row(s) in 0.1890 seconds
```

Step 4: Performing Data Formatting

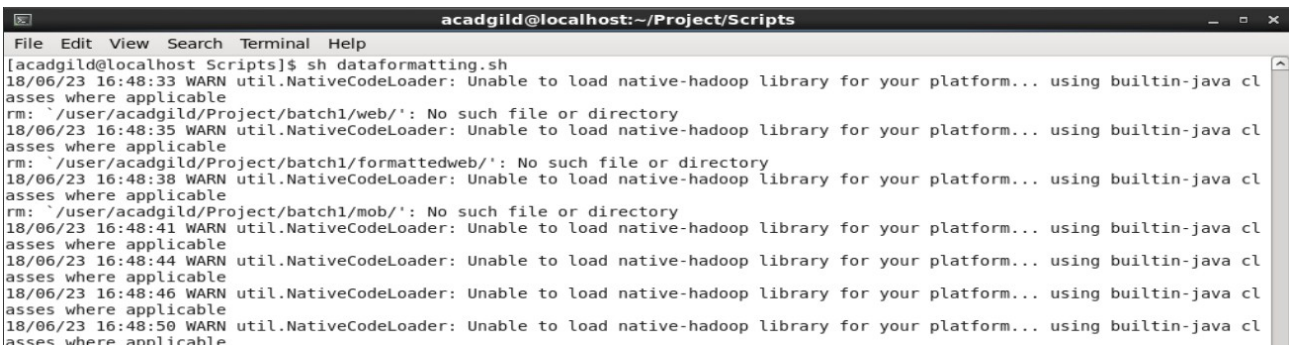
- Loading and Formatting data using pig and hive using the dataformatting script(i.e **dataformatting.pig**)

```
acadgild@dhcppc1:~/Project/Scripts
File  Edit  View  Search  Terminal  Help
[acadgild@dhcppc1 Scripts]$ cat 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:char
array);

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(',');

[acadgild@dhcppc1 Scripts]$
```



- By running the **dataformatting.sh** file The following operations are performed:

1. Get the batch id number from the batch file and get the Log File for the batch using the batch id. This will be log_batch_1
2. Add logs to the Log File signifying that the data is placed in the HDFS and the running of the Pig and Hive scripts for data formatting and loading respectively.
3. Delete, if they exist, folders for the mob, web and formattedweb. This is done in-case any old data remains because of execution failure.
4. Create the above folders web and mob that were deleted above and move the data from the Local FS to the HDFS. The formattedweb folder is created in the Pig Script.
5. Run the pig script dataformatting.pig. This will format the web data (stored in the web folder in the HDFS) in xml format to csv format and store it in the HDFS in the folder
6. formattedweb.
7. Run the hive script formatted_hive_load.hql. This will load the data in the mob folder and formattedweb folder in the HDFS to a table formatted_input in Hive which will
8. be used for data enrichment later.

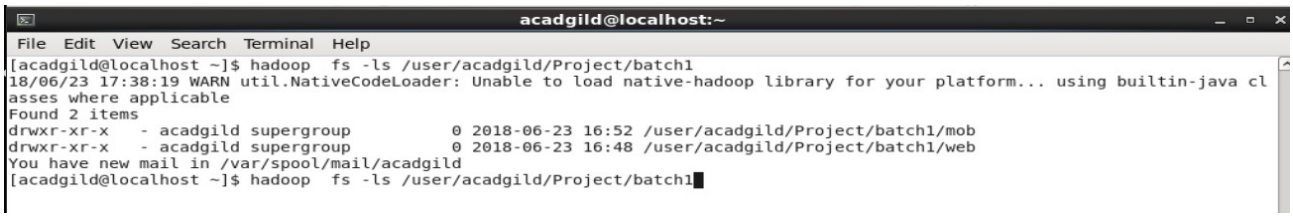


```
acadgild@localhost:~/Project/Scripts
File  Edit  View  Search  Terminal  Help
y tried 4 time(s); retry policy is RetryUpToMaximumCountWithFixedSleep(maxRetries=10, sleepTime=1000 MILLISECONDS)
2018-06-23 16:52:07,832 [main] INFO  org.apache.hadoop.ipc.Client - Retrying connect to server: 0.0.0.0/0.0.0.0:10020. Alread
y tried 5 time(s); retry policy is RetryUpToMaximumCountWithFixedSleep(maxRetries=10, sleepTime=1000 MILLISECONDS)
2018-06-23 16:52:08,833 [main] INFO  org.apache.hadoop.ipc.Client - Retrying connect to server: 0.0.0.0/0.0.0.0:10020. Alread
y tried 6 time(s); retry policy is RetryUpToMaximumCountWithFixedSleep(maxRetries=10, sleepTime=1000 MILLISECONDS)
2018-06-23 16:52:09,835 [main] INFO  org.apache.hadoop.ipc.Client - Retrying connect to server: 0.0.0.0/0.0.0.0:10020. Alread
y tried 7 time(s); retry policy is RetryUpToMaximumCountWithFixedSleep(maxRetries=10, sleepTime=1000 MILLISECONDS)
2018-06-23 16:52:10,836 [main] INFO  org.apache.hadoop.ipc.Client - Retrying connect to server: 0.0.0.0/0.0.0.0:10020. Alread
y tried 8 time(s); retry policy is RetryUpToMaximumCountWithFixedSleep(maxRetries=10, sleepTime=1000 MILLISECONDS)
2018-06-23 16:52:11,837 [main] INFO  org.apache.hadoop.ipc.Client - Retrying connect to server: 0.0.0.0/0.0.0.0:10020. Alread
y tried 9 time(s); retry policy is RetryUpToMaximumCountWithFixedSleep(maxRetries=10, sleepTime=1000 MILLISECONDS)
2018-06-23 16:52:11,938 [main] WARN  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - Unable
to retrieve job to compute warning aggregation.
2018-06-23 16:52:11,938 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - Success
!
2018-06-23 16:52:11,962 [main] INFO  org.apache.pig.Main - Pig script completed in 3 minutes, 18 seconds and 440 milliseconds
 (198440 ms)
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.j
ar!/hive-log4j2.properties Async: true
OK
Time taken: 7.936 seconds
OK
Time taken: 0.767 seconds
Loading data to table project.formatted_input partition (batchid=1)
OK
Time taken: 2.319 seconds
Loading data to table project.formatted_input partition (batchid=1)
OK
Time taken: 1.647 seconds
You have new mail in /var/spool/mail/acadgild
[acadgild@localhost Scripts]$
```

- Checking the formatted data stored in hive

```
acadgild@localhost:~
File  Edit  View  Search  Terminal  Help
hive> show tables;
OK
formatted_input
users_artists
Time taken: 0.067 seconds, Fetched: 2 row(s)
hive> select * from formatted_input;
OK
U114    S207    A303    1465130523      1465230523      1475130523      A       ST415   3       1       0       1
U107    S202    A303    1495130523      1465230523      1465230523      U       ST415   0       1       1       1
U100    S204    A302    1495130523      1475130523      1465130523      AU      ST408   2       1       1       1
U104    S202    A303    1465230523      1475130523      1465130523      A       ST409   2       0       1       1
U102    S207    A301    1465230523      1485130523      1465230523      AU      ST403   3       1       1       1
        S203    A302    1495130523      1475130523      1465230523      E       ST400   0       0       1       1
U106    S202    A302    1465230523      1465130523      1465130523      AU      ST408   0       1       1       1
U105    S207    A300    1465230523      1485130523      1465130523      U       ST400   2       0       1       1
U108    S205    A304    1465130523      1465130523      1475130523              ST410   2       1       0       1
U105    S203            1475130523      1465230523      1465130523      AU      ST408   2       0       1       1
U110    S203    A300    1465230523      1465130523      1485130523      A       ST415   0       1       1       1
U113    S200    A303    1465230523      1475130523      1465130523      E       ST413   3       1       1       1
U119    S208    A302    1495130523      1465230523      1465230523      U       ST415   3       0       0       1
U118    S208    A303    1475130523      1465130523      1465230523      E       ST415   3       0       0       1
U107    S210    A302    1475130523      1485130523      1485130523      AP      ST404   2       1       0       1
U118    S202    A300    1495130523      1465230523      1465230523      AP      ST410   1       0       0       1
U111    S206    A305    1465130523      1465130523      1485130523      AU      ST415   0       1       1       1
U116    S208    A303    1465230523      1485130523      1475130523      A       ST413   1       0       1       1
U101    S202    A300    1465230523      1465130523      1475130523      U       ST401   0       0       1       1
U120    S206    A303    1495130523      1485130523      1465130523      AU      ST414   0       0       0       1
U106    S205    A300    1462863262      1462863262      1494297562      AP      ST407   2       1       1       1
U114    S209    A303    1465490556      1462863262      1494297562      U       ST411   2       1       0       1
U113    S203    A304    1465490556      1465490556      1462863262      U       ST405   0       0       1       1
U108    S200    A302    1468094889      1462863262      1468094889      U       ST414   0       0       1       1
U102    S203    A305    1465490556      1465490556      1494297562      U       ST404   2       0       0       1
        S208    A300    1465490556      1494297562      1465490556      U       ST411   1       0       1       1
U115    S200    A300    1465490556      1494297562      1465490556      AU      ST404   3       0       0       1
U111    S204    A300    1465490556      1465490556      1468094889      U       ST410   3       1       1       1
U120    S201    A300    1494297562      1465490556      1468094889              ST410   3       0       1       1
U113    S203            1465490556      1465490556      1465490556      A       ST402   1       1       0       1
U109    S203    A304    1462863262      1494297562      1468094889      E       ST405   1       1       1       1
U110    S202    A303    1494297562      1494297562      1468094889      AU      ST402   2       1       0       1
```

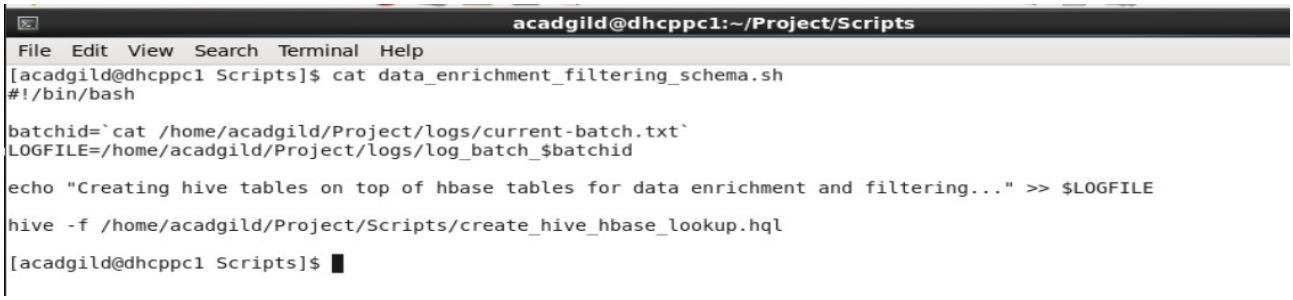
- Checking the data files stored in HDFS

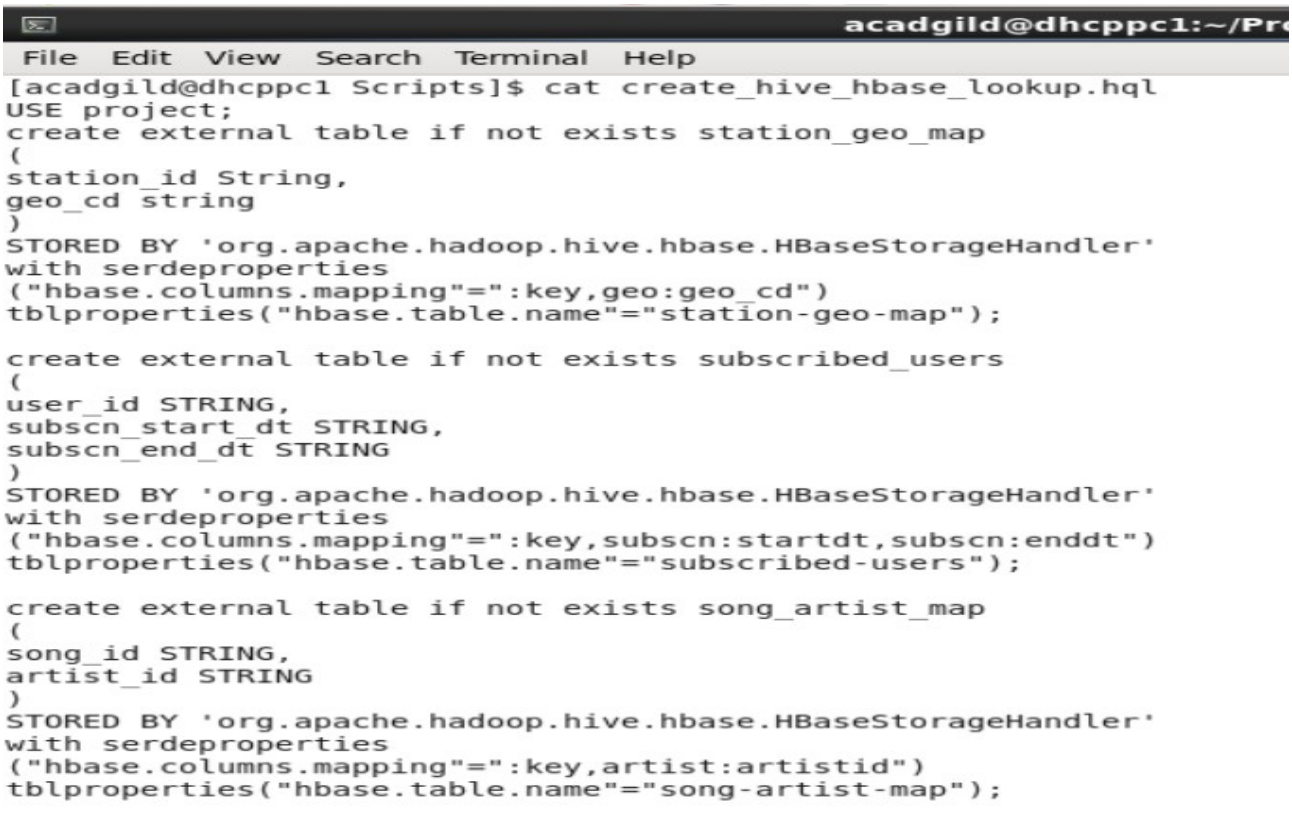


Step 5: Performing Data Enrichment and Cleaning

- Creating lookup tables in Hive by importing data from Hbase

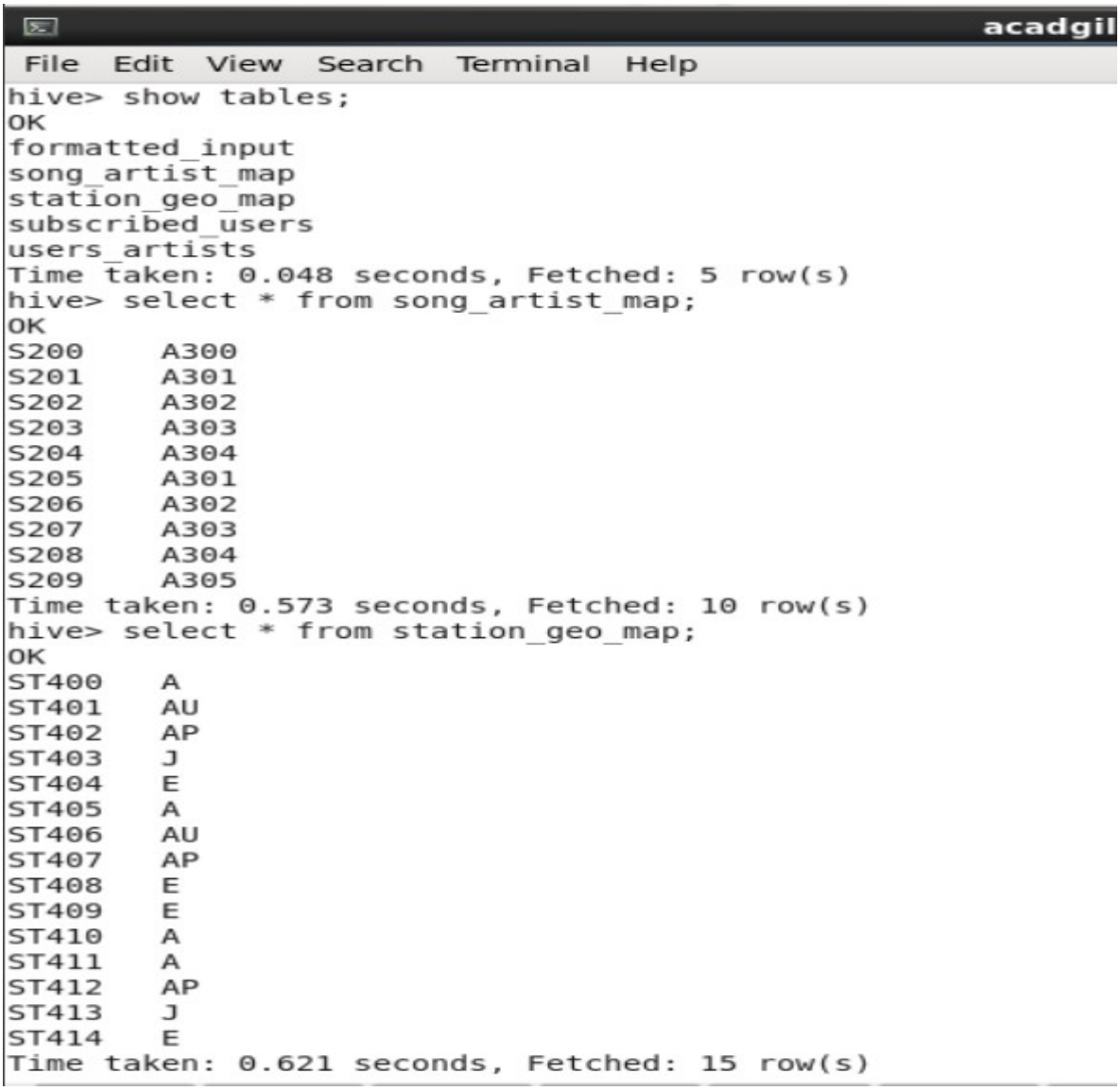
In this stage with the help of Hbase storage handler & SerDe properties we are creating the hive external tables by matching the columns of Hbase tables to hive tables.





In the above screenshot we can see tables getting created in hive by running the **data_enrichement_filtering_schema.sh** file.

- Verifying the tables created and data inserted



```
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: 0.568 seconds, Fetched: 15 row(s)
hive>
```

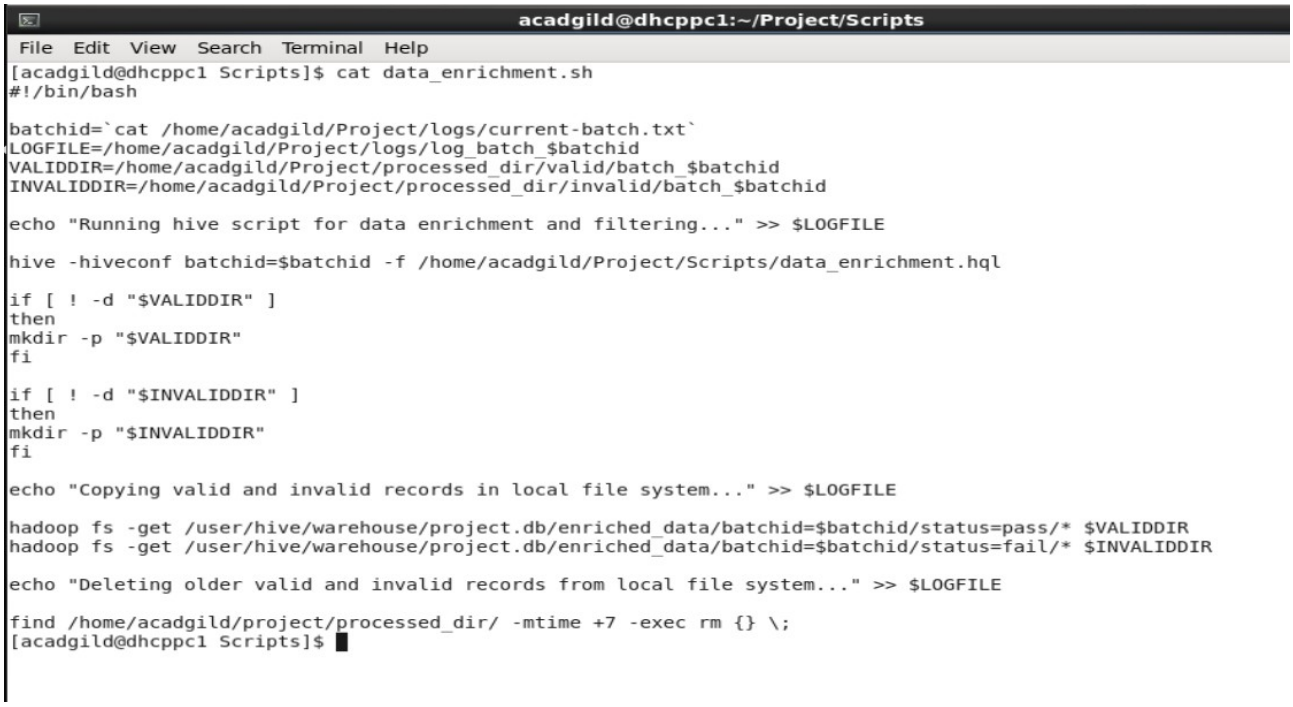
acadgild@dhcppc2:~

File Edit View Search Terminal Help

```
hive> select * from formatted_input;
OK
U114    S207    A303    1465130523      1465230523      1475130523      A       ST415   3       1       0       1
U107    S202    A303    1495130523      1465230523      1465230523      U       ST415   0       1       1       1
U100    S204    A302    1495130523      1475130523      1465130523      AU      ST408   2       1       1       1
U104    S202    A303    1465230523      1475130523      1465130523      A       ST409   2       0       1       1
U102    S207    A301    1465230523      1485130523      1465230523      AU      ST403   3       1       1       1
        S203    A302    1495130523      1475130523      1465230523      E       ST400   0       0       1       1
U106    S202    A302    1465230523      1465130523      1465130523      AU      ST408   0       1       1       1
U105    S207    A300    1465230523      1485130523      1465130523      U       ST400   2       0       1       1
U108    S205    A304    1465130523      1465130523      1475130523              ST410   2       1       0       1
U105    S203            1475130523      1465230523      1465130523      AU      ST408   2       0       1       1
U110    S203    A300    1465230523      1465130523      1485130523      A       ST415   0       1       1       1
U113    S200    A303    1465230523      1475130523      1465130523      E       ST413   3       1       1       1
U119    S208    A302    1495130523      1465230523      1465230523      U       ST415   3       0       0       1
U118    S208    A303    1475130523      1465130523      1465230523      E       ST415   3       0       0       1
U107    S210    A302    1475130523      1485130523      1485130523      AP      ST404   2       1       0       1
U118    S202    A300    1495130523      1465230523      1465230523      AP      ST410   1       0       0       1
U111    S206    A305    1465130523      1465130523      1485130523      AU      ST415   0       1       1       1
U116    S208    A303    1465230523      1485130523      1475130523      A       ST413   1       0       1       1
U101    S202    A300    1465230523      1465130523      1475130523      U       ST401   0       0       1       1
U120    S206    A303    1495130523      1485130523      1465130523      AU      ST414   0       0       0       1
U106    S205    A300    1462863262      1462863262      1494297562      AP      ST407   2       1       1       1
U114    S209    A303    1465490556      1462863262      1494297562      U       ST411   2       1       0       1
U113    S203    A304    1465490556      1465490556      1462863262      U       ST405   0       0       1       1
U108    S200    A302    1468094889      1462863262      1468094889      U       ST414   0       0       1       1
U102    S203    A305    1465490556      1465490556      1494297562      U       ST404   2       0       0       1
        S208    A300    1465490556      1494297562      1465490556      U       ST411   1       0       1       1
U115    S200    A300    1465490556      1494297562      1465490556      AU      ST404   3       0       0       1
U111    S204    A300    1465490556      1465490556      1468094889      U       ST410   3       1       1       1
U120    S201    A300    1494297562      1465490556      1468094889              ST410   3       0       1       1
U113    S203            1465490556      1465490556      1465490556      A       ST402   1       1       0       1
U109    S203    A304    1462863262      1494297562      1468094889      E       ST405   1       1       1       1
U110    S202    A303    1494297562      1494297562      1468094889      AU      ST402   2       1       0       1
U100    S200    A301    1494297562      1494297562      1494297562      AP      ST410   3       1       1       1
U101    S208    A300    1462863262      1468094889      1462863262      E       ST408   0       1       1       1
U106    S206    A300    1494297562      1465490556      1462863262      A       ST405   3       1       0       1
U107    S202    A304    1494297562      1468094889      1462863262      U       ST409   0       0       0       1
U103    S204    A300    1468094889      1494297562      1465490556      AU      ST411   2       1       0       1

U103    S202    A300    1465490556      1465490556      1465490556      A       ST415   2       1       1       1
U113    S203    A303    1462863262      1468094889      1494297562      U       ST408   2       0       0       1
U113    S204    A301    1494297562      1494297562      1465490556      E       ST415   3       0       1       1
Time taken: 3.482 seconds, Fetched: 40 row(s)
```

- Running **data_enrichment.sh** to create a table in Hive which has enriched data and based on the partition rules

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

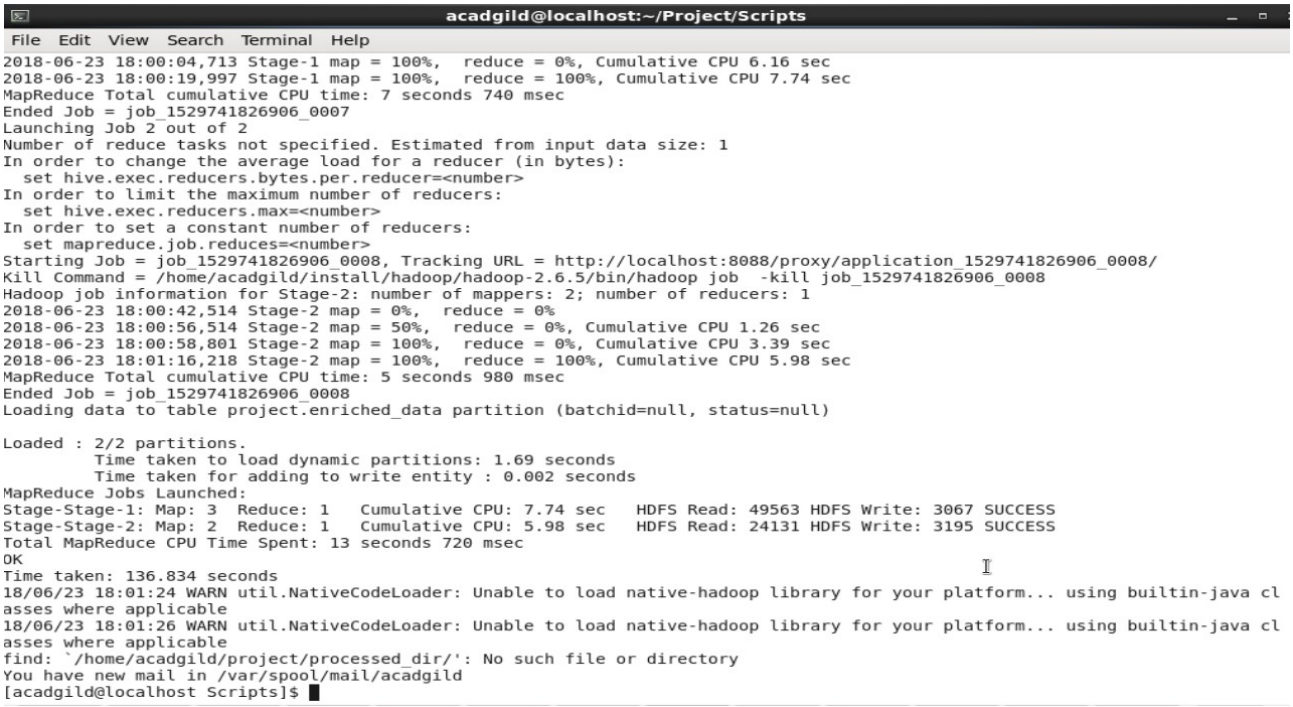



acadgild@dhcppc1:~/Project/Scr
File Edit View Search Terminal Help
[acadgild@dhcppc1 Scripts]$ cat data_enrichment.hql
SET hive.auto.convert.join=false;
SET hive.exec.dynamic.partition.mode=nonstrict;
USE project;
CREATE TABLE IF NOT EXISTS enriched_data
(
User_id STRING,
Song_id STRING,
Artist_id STRING,
`Timestamp` STRING,
Start_ts STRING,
End_ts STRING,
Geo_cd STRING,
Station_id STRING,
Song_end_type INT,
`Like` INT,
Dislike INT
)
PARTITIONED BY
(batchid INT,status STRING)
STORED AS ORC;
INSERT OVERWRITE TABLE enriched_data
PARTITION (batchid, status)
SELECT
i.user_id,
i.song_id,
sa.artist_id,
i.`timestamp`,
i.start_ts,
i.end_ts,
sg.geo_cd,
i.station_id,
IF (i.song_end_type IS NULL, 3, i.song_end_type) AS song_end_type,
IF (i.`like` IS NULL, 0, i.`like`) AS `like`,
IF (i.dislike IS NULL, 0, i.dislike) AS dislike,
i.batchid,
IF((i.`like`=1 AND i.dislike=1)
OR i.user_id IS NULL
OR i.song_id IS NULL
OR i.`timestamp` IS NULL
OR i.start_ts IS NULL
OR i.end_ts IS NULL
OR i.geo_cd IS NULL
OR i.user_id=''
OR i.song_id=''
OR i.`timestamp`=''
OR i.start_ts=''
OR i.end_ts=''
OR i.geo_cd=''
OR sg.geo_cd IS NULL
OR sg.geo_cd=''
OR sa.artist_id IS NULL
OR sa.artist_id='', 'fail', 'pass') AS status
FROM formatted_input i
LEFT OUTER JOIN station_geo_map sg ON i.station_id = sg.station_id
LEFT OUTER JOIN song_artist_map sa ON i.song_id = sa.song_id
WHERE i.batchid=${hiveconf:batchid};
[acadgild@dhcppc1 Scripts]$

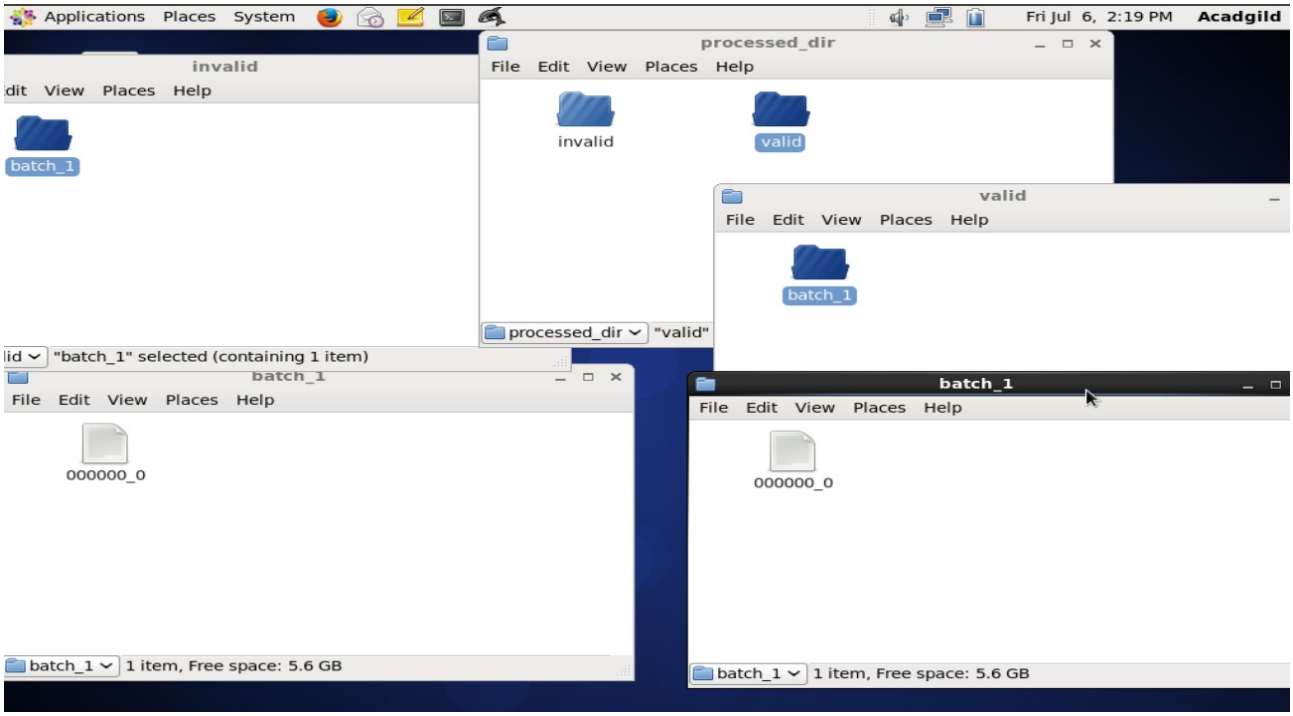
- **data_enrichment.sh** is executed

In this phase , a table enriched_data is created and the table is overwritten with the result of the below operations:

1. The data in the formatted_input table is joined with the lookup tables station_geo_map and song_artist_map to fill in the data gaps that can be obtained by said tables.
2. The same data is then filtered by the rules given above and



partitioned by status (pass or fail) & batchid.



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

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

```
hive> select * from enriched_data;
OK
U113    S200    A300    1465230523      1475130523      1465130523      J       ST413   3       1       1       1       fail
U100    S200    A300    1494297562      1494297562      1494297562      A       ST410   3       1       1       1       fail
U120    S201    A301    1494297562      1465490556      1468094889      A       ST410   3       0       1       1       fail
U107    S202    A302    1495130523      1465230523      1465230523      NULL    ST415   0       1       1       1       fail
U103    S202    A302    1465490556      1465490556      1465490556      NULL    ST415   2       1       1       1       fail
U106    S202    A302    1465230523      1465130523      1465130523      E       ST408   0       1       1       1       fail
U109    S203    A303    1462863262      1494297562      1468094889      A       ST405   1       1       1       1       fail
        S203    A303    1495130523      1475130523      1465230523      A       ST400   0       0       1       1       fail
U110    S203    A303    1465230523      1465130523      1485130523      NULL    ST415   0       1       1       1       fail
U111    S204    A304    1465490556      1465490556      1468094889      A       ST410   3       1       1       1       fail
U113    S204    A304    1494297562      1494297562      1465490556      NULL    ST415   3       0       1       1       fail
U100    S204    A304    1495130523      1475130523      1465130523      E       ST408   2       1       1       1       fail
U106    S205    A301    1462863262      1462863262      1494297562      AP      ST407   2       1       1       1       fail
U108    S205    A301    1465130523      1465130523      1475130523      A       ST410   2       1       0       1       fail
U111    S206    A302    1465130523      1465130523      1485130523      NULL    ST415   0       1       1       1       fail
U114    S207    A303    1465130523      1465230523      1475130523      NULL    ST415   3       1       0       1       fail
U102    S207    A303    1465230523      1485130523      1465230523      J       ST403   3       1       1       1       fail
        S208    A304    1465490556      1494297562      1465490556      A       ST411   1       0       1       1       fail
U118    S208    A304    1475130523      1465130523      1465230523      NULL    ST415   3       0       0       1       fail
U119    S208    A304    1495130523      1465230523      1465230523      NULL    ST415   3       0       0       1       fail
U101    S208    A304    1462863262      1468094889      1462863262      E       ST408   0       1       1       1       fail
U107    S210    NULL    1475130523      1485130523      1485130523      E       ST404   2       1       0       1       fail
U115    S200    A300    1465490556      1494297562      1465490556      E       ST404   3       0       0       1       pass
U108    S200    A300    1468094889      1462863262      1468094889      E       ST414   0       0       1       1       pass
U107    S202    A302    1494297562      1468094889      1462863262      E       ST409   0       0       0       1       pass
U101    S202    A302    1465230523      1465130523      1475130523      AU      ST401   0       0       1       1       pass
U110    S202    A302    1494297562      1494297562      1468094889      AP      ST402   2       1       0       1       pass
U118    S202    A302    1495130523      1465230523      1465230523      A       ST410   1       0       0       1       pass
U104    S202    A302    1465230523      1475130523      1465130523      E       ST409   2       0       1       1       pass
U102    S203    A303    1465490556      1465490556      1494297562      E       ST404   2       0       0       1       pass
U113    S203    A303    1465490556      1465490556      1462863262      A       ST405   0       0       1       1       pass
U113    S203    A303    1462863262      1468094889      1494297562      E       ST408   2       0       0       1       pass
U105    S203    A303    1475130523      1465230523      1465130523      E       ST408   2       0       1       1       pass
U113    S203    A303    1465490556      1465490556      1465490556      AP      ST402   1       1       0       1       pass
U103    S204    A304    1468094889      1494297562      1465490556      A       ST411   2       1       0       1       pass
U106    S206    A302    1494297562      1465490556      1462863262      A       ST405   3       1       0       1       pass
U120    S206    A302    1495130523      1485130523      1465130523      E       ST414   0       0       0       1       pass

U105    S207    A303    1465230523      1485130523      1465130523      A       ST400   2       0       1       1       pass
U116    S208    A304    1465230523      1485130523      1475130523      J       ST413   1       0       1       1       pass
U114    S209    A305    1465490556      1462863262      1494297562      A       ST411   2       1       0       1       pass
Time taken: 0.382 seconds, Fetched: 40 row(s)
```

Step 6: Performing Data Analysis

Below is the shell script will **data_analysis.sh** which will perform :

- Get the batch id number from the batch file and get the Log File for the batch using the batch id.
- Add logs to the Log File signifying that the data analysis is being performed using Spark
- and that the result is being exported to the Local FS.
- Run the spark script SparkAnalysis.scala. This will perform the data analysis required in the problem statement given and save the result to the HDFS in csv format.
- Add logs to the Log File signifying that the data analysis has completed and that the batch is being incremented. Here from 1 to 2
- Get batchid number from batch file and increment the batchid by 1.

```
Spark context Web UI available at http://192.168.1.3:4040
Spark context available as 'sc' (master = local[*], app id = local-1531057582300).
Spark session available as 'spark'.
Loading /home/acadgild/Project/Scripts/SparkAnalysis.scala...
import org.apache.spark.sql.hive.orc._
import org.apache.spark.sql._
import org.apache.hadoop.hbase.util.Bytes
warning: there was one deprecation warning; re-run with -deprecation for details
hiveContext: org.apache.spark.sql.hive.HiveContext = org.apache.spark.sql.hive.HiveContext@a120b9
import hiveContext.implicits._
args: Array[String] = Array(3)
batchid: String = 3
outputdirpath: String = /Test/batch3
res0: org.apache.spark.sql.DataFrame = []
df1: org.apache.spark.sql.DataFrame = [station_id: string, count(DISTINCT song_id): bigint ... 1 more field]
```

In the **SparkAnalysis.scala** operation performed:

1. Created an hive context.
2. Get the batchid from the batch file and store it in the variable batchid
3. Get the data that was exported and saved in the Local FS from the steps above i.e. enriched_data, subscribed_user and user_artists and perform the foll. on each of them:
4. Create the schema for the data
5. Create a DataFrame from the schema and data
6. Stored the the putput in csv format in HDFS.

SparkAnalysis.scala

```
import org.apache.spark.sql.hive.orc._
import org.apache.spark.sql._
import org.apache.hadoop.hbase.util.Bytes


//Create HiveContext
val hiveContext = new org.apache.spark.sql.hive.HiveContext(sc)

import hiveContext.implicits._

//Read the passed batchid as parameter
val args=sc.getConf.get("spark.driver.args").split("\\s+")
val batchid=args(0)


//Set the output path where all output files will be generated
val outputdirpath="/Project/batch"+batchid

//Set the database to be used in hive
hiveContext.sql("use project")

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

val df1=hiveContext.sql("select station_id,count(distinct song_id),count(distinct user_id) from enriched_data where
status='pass' and batchid='"+batchid+"'group by station_id")
df1.repartition(1).write.option("header","true").csv(outputdirpath+"/top10stations")

/************************** Problem 1 - End **********************/
```

SparkAnalysis.scala

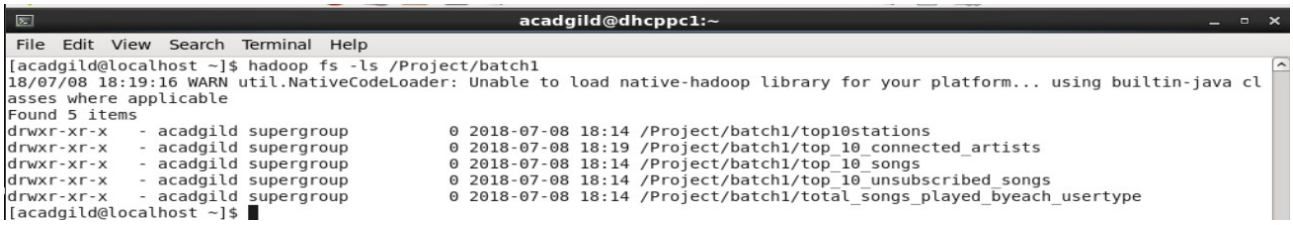
```
import org.apache.spark.sql.hive.orc._
import org.apache.spark.sql._
import org.apache.hadoop.hbase.util.Bytes


//Create HiveContext
val hiveContext = new org.apache.spark.sql.hive.HiveContext(sc)

import hiveContext.implicits._

//Read the passed batchid as parameter
val args=sc.getConf.get("spark.driver.args").split("\\s+")
val batchid=args(0)


//Set the output path where all output files will be generated
val outputdirpath="/Project/batch"+batchid

//Set the database to be used in hive
hiveContext.sql("use project")

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

val df1=hiveContext.sql("select station_id,count(distinct song_id),count(distinct user_id) from enriched_data where
status='pass' and batchid='"+batchid+"'group by station_id")
df1.repartition(1).write.option("header","true").csv(outputdirpath+"/top10stations")

/************************** Problem 1 - End **********************/
```

SparkAnalysis.scala

```
/************************** Problem 2 - Start **********************/
//Determine total duration of songs played by each type of user, where type of user can be 'subscribed' or 'unsubscribed'

val df2=hiveContext.sql("SELECT CASE WHEN (su.user_id IS NULL OR CAST(ed.timestamp AS DECIMAL(20,0)) > CAST
(su.subscn_end_dt AS DECIMAL(20,0))) THEN 'UNSUBSCRIBED' WHEN (su.user_id IS NOT NULL AND CAST(ed.timestamp AS DECIMAL
(20,0)) <= CAST(su.subscn_end_dt AS DECIMAL(20,0))) THEN 'SUBSCRIBED' END AS user_type,SUM(ABS(CAST(ed.end_ts AS DECIMAL
(20,0))-CAST(ed.start_ts AS DECIMAL(20,0)))) AS duration FROM enriched_data ed LEFT OUTER JOIN subscribed_users su ON
ed.user_id=su.user_id WHERE ed.status='pass' AND ed.batchid='"+batchid+"' GROUP BY CASE WHEN (su.user_id IS NULL OR CAST
(ed.timestamp AS DECIMAL(20,0)) > CAST(su.subscn_end_dt AS DECIMAL(20,0))) THEN 'UNSUBSCRIBED' WHEN (su.user_id IS NOT NULL
AND CAST(ed.timestamp AS DECIMAL(20,0)) <= CAST(su.subscn_end_dt AS DECIMAL(20,0))) THEN 'SUBSCRIBED' END")

//Write output of the above query to csv file
df2.repartition(1).write.option("header","true").csv(outputdirpath+"/total_songs_played_byeach_usertype")

/************************** Problem 2 - End **********************/

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

val df3=hiveContext.sql("SELECT ua.artist_id,COUNT(DISTINCT ua.user_id) AS user_count FROM(SELECT user_id, artist_id FROM
user_artist_map LATERAL VIEW explode(artists_array) artists AS artist_id ) ua INNER JOIN(SELECT artist_id, song_id, user_id
FROM enriched_data WHERE status='pass' AND batchid='"+batchid+"') ed ON ua.artist_id=ed.artist_id AND ua.user_id=ed.user_id
GROUP BY ua.artist_id ORDER BY user_count DESC LIMIT 10")

//Write output of the above query to csv file
df3.repartition(1).write.option("header","true").csv(outputdirpath+"/top_10_connected_artists")

/************************** Problem 3 - End **********************/
```

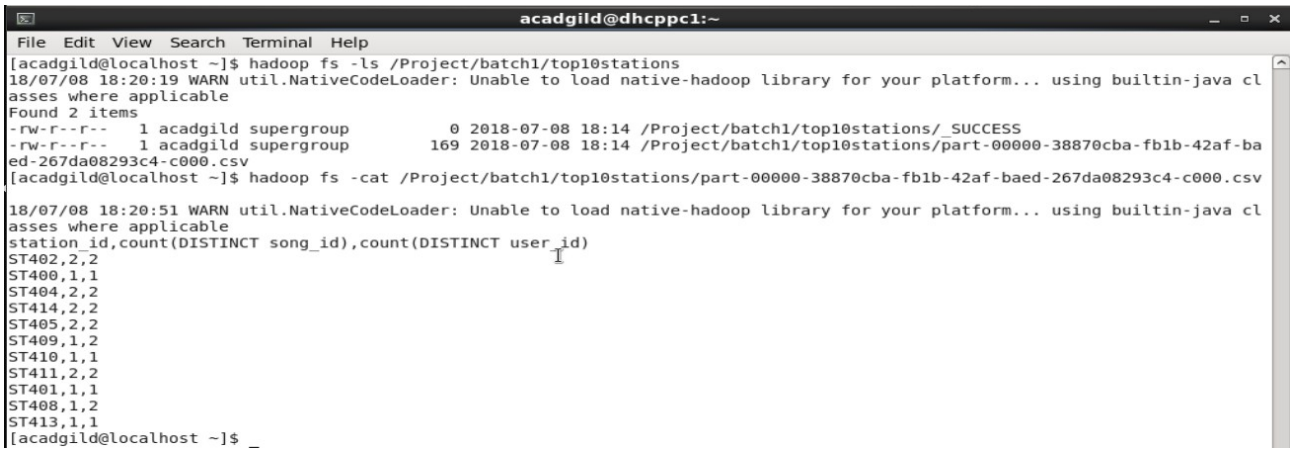
```
/************************** Problem 4 - Start **********************/
//Determine top 10 songs who have generated the maximum revenue

val df4 = hiveContext.sql("SELECT song_id,SUM(ABS(CAST(end_ts AS DECIMAL(20,0))-CAST(start_ts AS DECIMAL(20,0)))) AS
duration FROM enriched_data WHERE status='pass' AND batchid='"+batchid+"' AND (like=1 OR song_end_type=0)GROUP BY song_id
ORDER BY duration DESC LIMIT 10")

//Write output of the above query to csv file
df4.repartition(1).write.option("header","true").csv(outputdirpath+"/top_10_songs")

/************************** Problem 4 - End **********************/

/************************** Problem 5 - Start **********************/
//Determine top 10 unsubscribed users who listened to the songs for the longest duration


val df5=hiveContext.sql("SELECT ed.user_id,SUM(ABS(CAST(ed.end_ts AS DECIMAL(20,0))-CAST(ed.start_ts AS DECIMAL(20,0)))) AS
duration FROM enriched_data ed LEFT OUTER JOIN subscribed_users su ON ed.user_id=su.user_id WHERE ed.status='pass' AND
ed.batchid='"+batchid+"' AND (su.user_id IS NULL OR (CAST(ed.timestamp AS DECIMAL(20,0)) > CAST(su.subscn_end_dt AS DECIMAL
(20,0))))GROUP BY ed.user_id ORDER BY duration DESC LIMIT 10")

//Write output of the above query to csv file
df5.repartition(1).write.option("header","true").csv(outputdirpath+"/top_10_unsubscribed_songs")

/************************** Problem 5 - End **********************/

System.exit(0)
```

- After the execution , if we check the hdfs . In the below screenshot, we can see the analysis is done and the output are stored in folders.



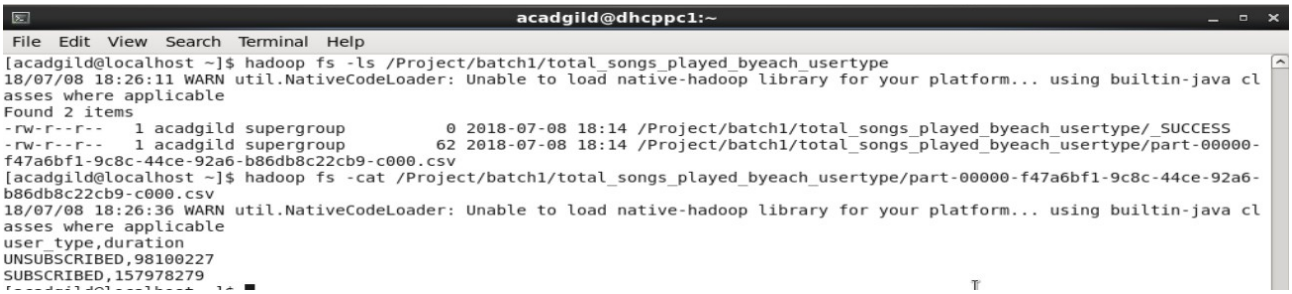
Problem Statement 1:
Determine top 10 station_id(s) where maximum number of songs were played, which were liked by unique users.

Output :



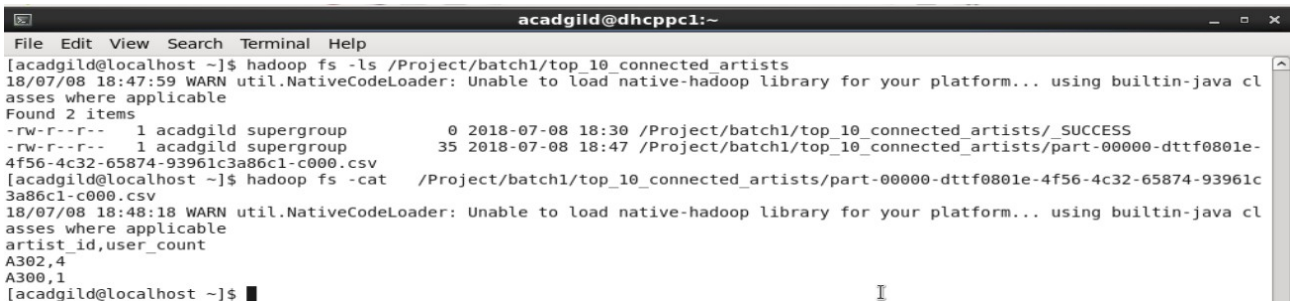
Problem Statement 2:
Determine total duration of songs played by each type of user, where type of user can be 'subscribed' or 'unsubscribed'. An unsubscribed user is the one whose record is either not present in Subscribed_users lookup table or has subscription_end_date earlier than the timestamp of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

Output -



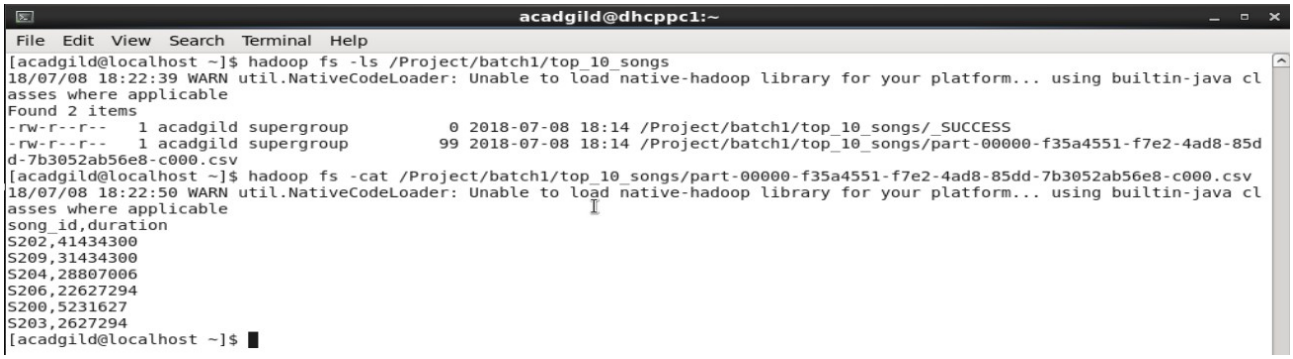
Problem Statement 3:
Determine top 10 connected artists. Connected artists are those whose songs are most listened by the unique users who follow them.

Outout -



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

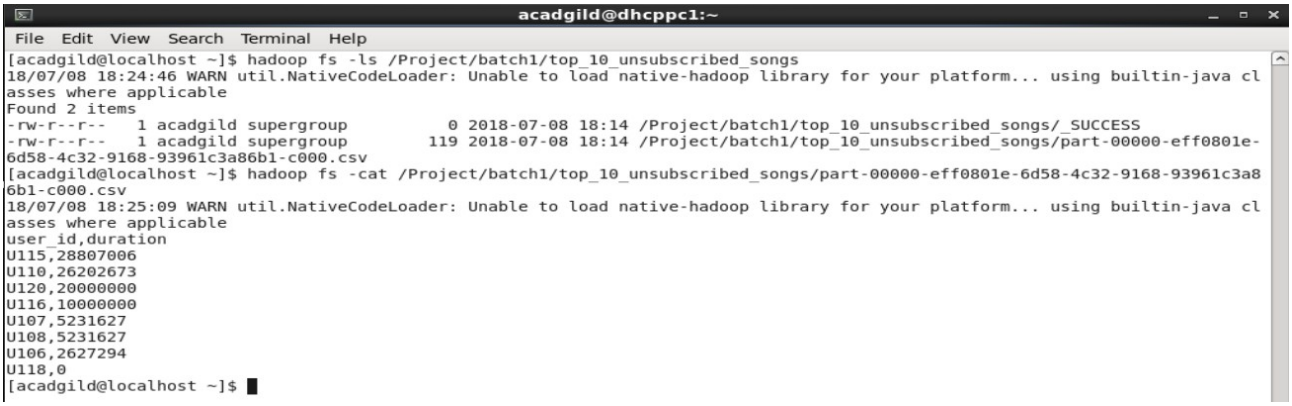
Output -



Problem Statement 5:

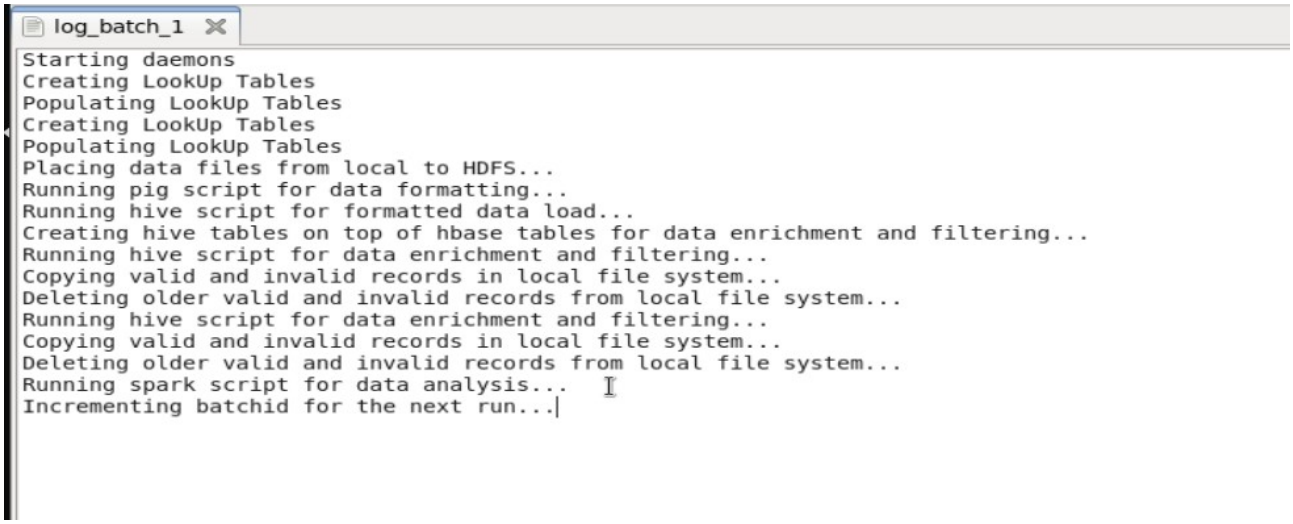
Determine top 10 unsubscribed users who listened to the songs for the longest duration.

Output -

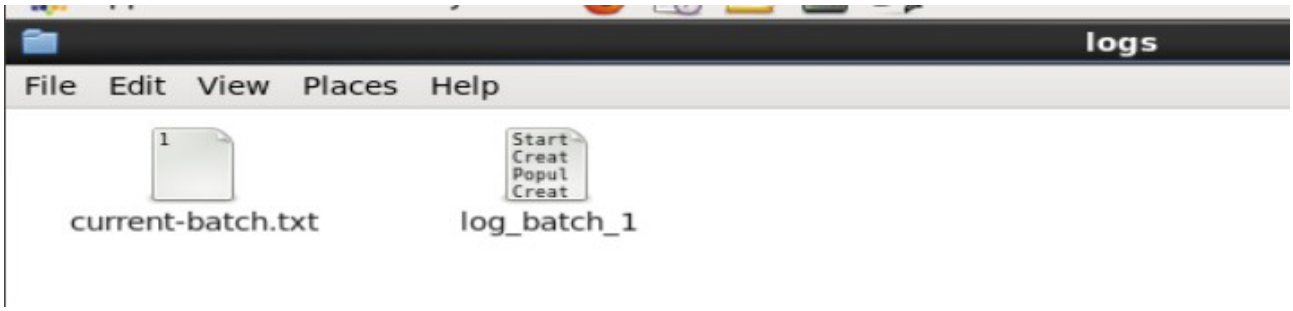


Post Analysis:

- A view of the log file post analysis.



- A view of the log folder:



- The batchid is incremented from 1 to 2 after execution of 1 batch