

**ACADGILD**

---

**Big Data Engineering with Hadoop & Spark**

---

**PROJECT REPORT**

**MUSIC DATA ANALYSIS**

**PREPARED BY**

**AMAN TURATE**

# TABLE OF CONTENTS

Title Page .....	1
Table of Contents .....	2
Introduction & Data Sets .....	3
Flow of operations.....	7
Steps to perform Music Data Analysis.....	8
✓ Scheduling Crontab job.....	8
✓ Launch all necessary daemons .....	9
✓ Populate look up tables into HBase .....	11
✓ Perform Data Formatting.....	16
✓ Perform Data Enrichment and Cleaning .....	19
✓ Perform Data Analysis.....	26
Problem Statement Solution .....	30
View of Log File after Data Analysis.....	35

# INTRODUCTION

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

## **DATASET:**

Below is the link for same.

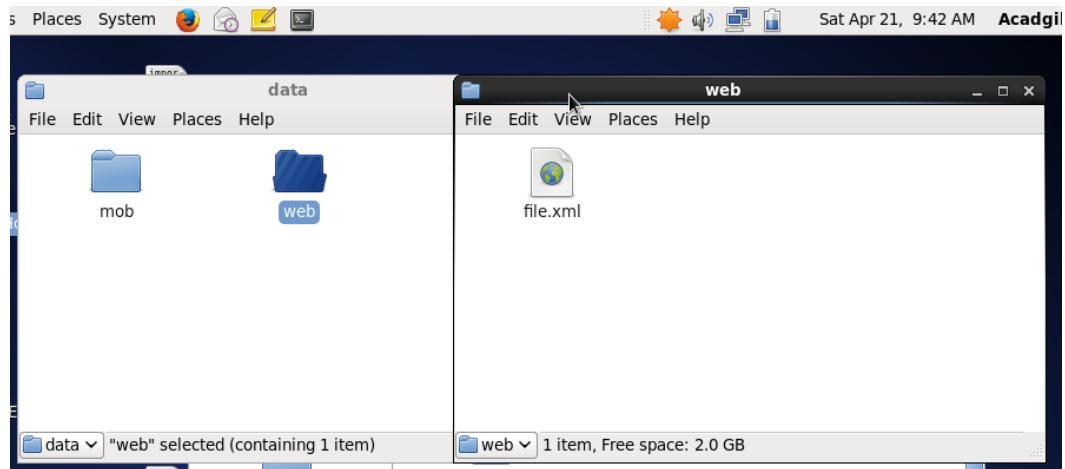
[https://drive.google.com/drive/folders/0B\\_P3pWagdIrrMjJGVINsSUEtbG8?usp=sharing](https://drive.google.com/drive/folders/0B_P3pWagdIrrMjJGVINsSUEtbG8?usp=sharing)

## **Data Description**

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

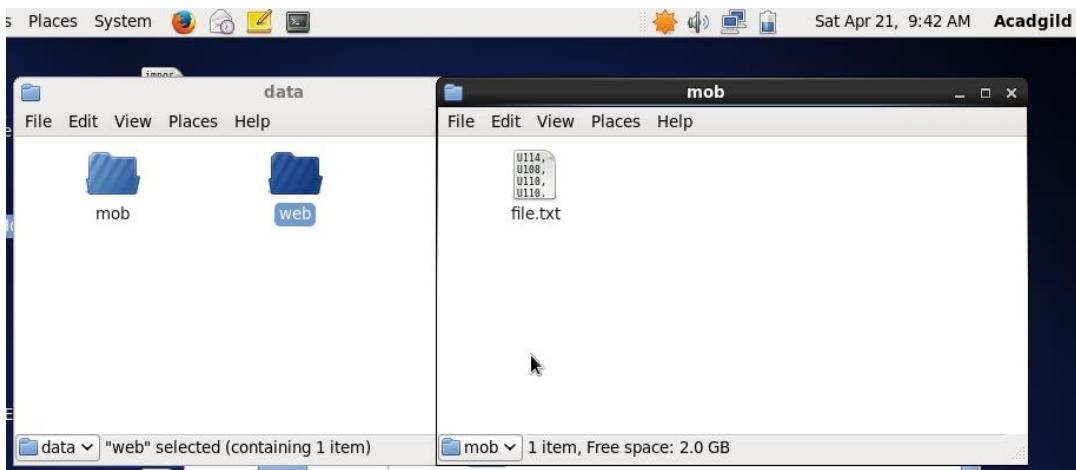
## Data Files

- ✓ Sample Data coming from web applications reside in /data/web and has xml format.



```
<dislike>0</dislike>
</record>
<record>
<user_id>U118</user_id>
<song_id>S202</song_id>
<artist_id>A304</artist_id>
<timestamp>2017-05-09 08:09:22</timestamp>
<start_ts>2017-05-09 08:09:22</start_ts>
<end_ts>2016-06-09 22:12:36</end_ts>
<geo_cd>U</geo_cd>
<station_id>ST405</station_id>
<song_end_type>0</song_end_type>
<like>1</like>
<dislike>1</dislike>
</record>
<record>
<user_id>U102</user_id>
<song_id>S204</song_id>
<artist_id>A305</artist_id>
<timestamp>2016-07-10 01:38:09</timestamp>
<start_ts>2016-05-10 12:24:22</start_ts>
<end_ts>2016-05-10 12:24:22</end_ts>
<geo_cd>E</geo_cd>
<station_id>ST413</station_id>
<song_end_type>0</song_end_type>
<like>0</like>
<dislike>1</dislike>
</record>
<record>
<user_id>U116</user_id>
<song_id>S209</song_id>
<artist_id>A301</artist_id>
<timestamp>2017-05-09 08:09:22</timestamp>
```

- ✓ Sample Data coming from mobile applications reside in /data/mob and has csv format.



```

U111,S204,A300,1465130523,1465130523,1475130523,E,ST405,2,1,0
U103,S202,A302,1465230523,1465130523,1485130523,AP,ST405,2,0,1
U113,S207,A302,1495130523,1465230523,1465230523,AU,ST410,3,0,1
U113,S205,A301,1465130523,1485130523,1465230523,A,ST404,1,0,1
U117,S203,A305,1475130523,1485130523,1465130523,AP,ST414,0,0,0
,S202,A304,1465130523,1485130523,1485130523,,ST410,3,1,0
U103,S208,A305,1475130523,1475130523,1465130523,U,ST408,2,0,1
U109,S207,A304,1495130523,1465130523,1485130523,U,ST413,1,0,1
U113,S203,A302,1465230523,1465130523,1485130523,,ST403,3,0,0
U115,S200,,1475130523,1485130523,1465130523,A,ST400,3,0,1
U103,S210,A301,1475130523,1465130523,1485130523,E,ST408,0,0,1
U114,S203,A305,1495130523,1465230523,1485130523,AU,ST413,2,1,0
U102,S206,A300,1465230523,1485130523,1475130523,U,ST401,2,0,0
U120,S210,A300,1495130523,1485130523,1475130523,E,ST412,1,0,1
U119,S201,A300,1465130523,1465230523,1475130523,AU,ST406,1,0,1
U115,S200,A302,1495130523,1475130523,1475130523,U,ST404,2,0,0
U102,S210,A303,1495130523,1465130523,1485130523,E,ST400,3,1,1
U101,S205,A305,1475130523,1465130523,1465130523,A,ST414,0,0,0
U105,S201,A302,1465230523,1475130523,1485130523,E,ST412,3,1,0
U103,S201,A300,1465130523,1465130523,1465230523,U,ST412,2,1,0

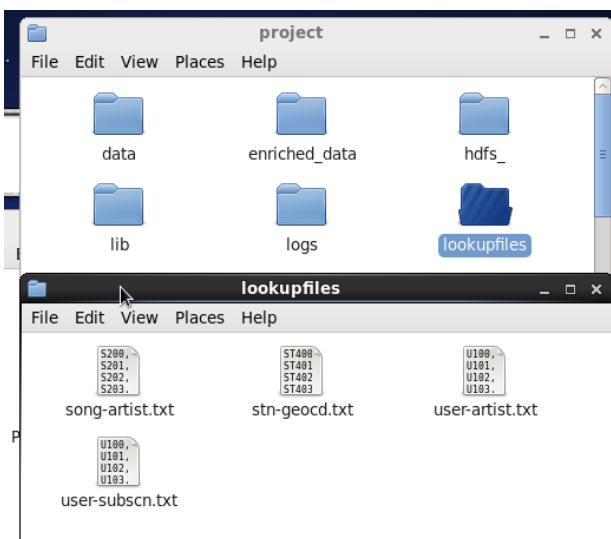
```

## 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

- ✓ Sample Data present in lookup directory is being used in HBase.



Song-artist.txt

Columns: song\_id, artist\_id

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

user-subscn.txt

user\_id,subscn\_start\_dt, subscn\_end\_dt

```
U100,1465230523,1465130523
U101,1465230523,1475130523
U102,1465230523,1475130523
U103,1465230523,1475130523
U104,1465230523,1475130523
U105,1465230523,1475130523
U106,1465230523,1485130523
U107,1465230523,1455130523
U108,1465230523,1465230623
U109,1465230523,1475130523
U110,1465230523,1475130523
U111,1465230523,1475130523
U112,1465230523,1475130523
U113,1465230523,1485130523
U114,1465230523,1468130523
```

stn-geocd.txt

Columns: station\_id, geo\_cd

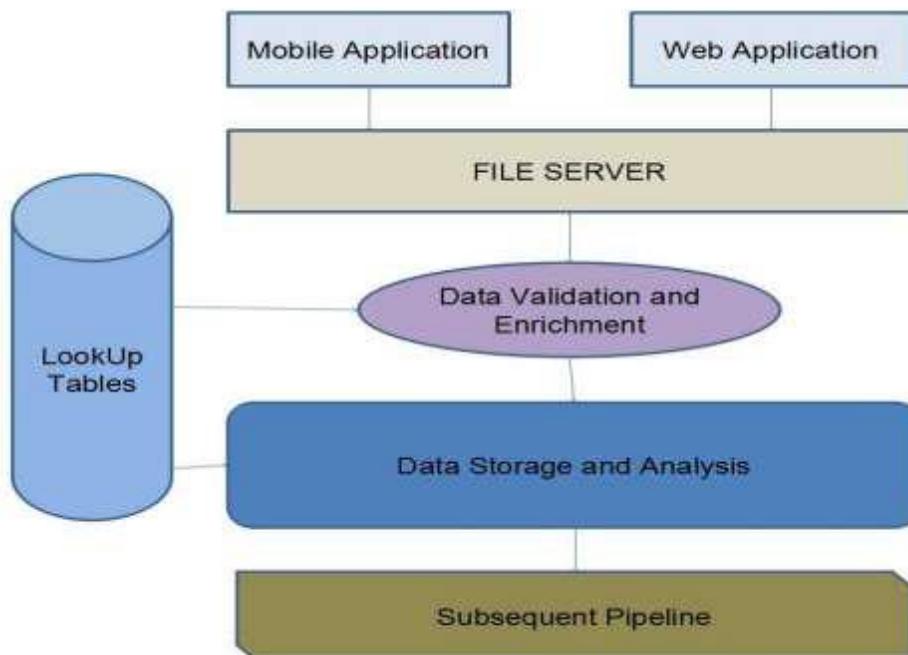
```
ST400,A  
ST401,AU  
ST402,AP  
ST403,J  
ST404,E  
ST405,A  
ST406,AU  
ST407,AP  
ST408,E  
ST409,E  
ST410,A  
ST411,A  
ST412,AP  
ST413,J  
ST414,E
```

user-artist.txt

user\_id, artists\_array

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

## FLOW OF OPERATIONS :



## **Steps to perform Music Data Analysis :**

- 1) Scheduling Crontab job
- 2) Launch all necessary daemons.
- 3) Populate look up tables into HBase.
- 4) Perform Data Formatting.
- 5) Perform Data Enrichment and Cleaning.
- 6) Perform Data Analysis.

### **1) Scheduling Crontab job :-**

- ✓ Open Crontab File using following command -

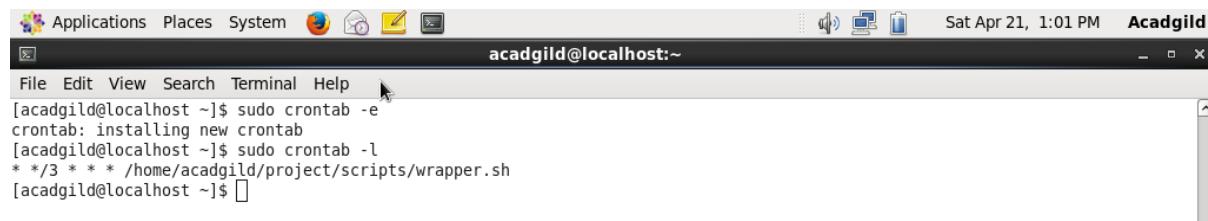
```
sudo crontab -e
```

- ✓ Press I to enter into insert mode.
- ✓ Insert the following command -

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

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

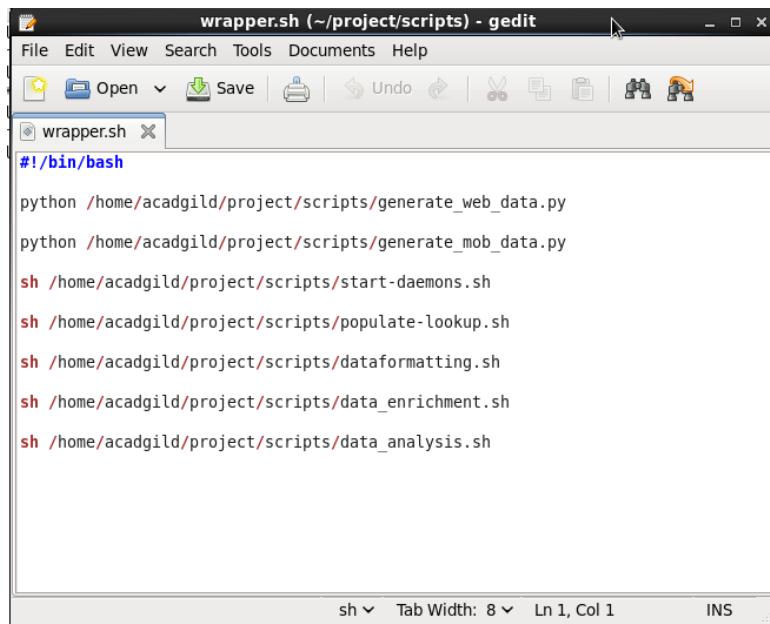
The statement above runs the **wrapper.sh** shell script every 3 hours.



A screenshot of a terminal window titled "acadgild@localhost:~". The window shows the following command history:

```
File Edit View Search Terminal Help
[acadgild@localhost ~]$ sudo crontab -e
crontab: installing new crontab
[acadgild@localhost ~]$ sudo crontab -l
* */3 * * * /home/acadgild/project/scripts/wrapper.sh
[acadgild@localhost ~]$
```

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.



```
wrapper.sh (~/project/scripts) - gedit
File Edit View Search Tools Documents Help
Open Save Undo | | | | | | |
wrapper.sh x
#!/bin/bash

python /home/acadgild/project/scripts/generate_web_data.py
python /home/acadgild/project/scripts/generate_mob_data.py
sh /home/acadgild/project/scripts/start-daemons.sh
sh /home/acadgild/project/scripts/populate-lookup.sh
sh /home/acadgild/project/scripts/dataformatting.sh
sh /home/acadgild/project/scripts/data_enrichment.sh
sh /home/acadgild/project/scripts/data_analysis.sh

sh v Tab Width: 8 v Ln 1, Col 1 INS
```

## 2) Launch all necessary daemons :-

### Execution Script - [start-daemons.sh](#)

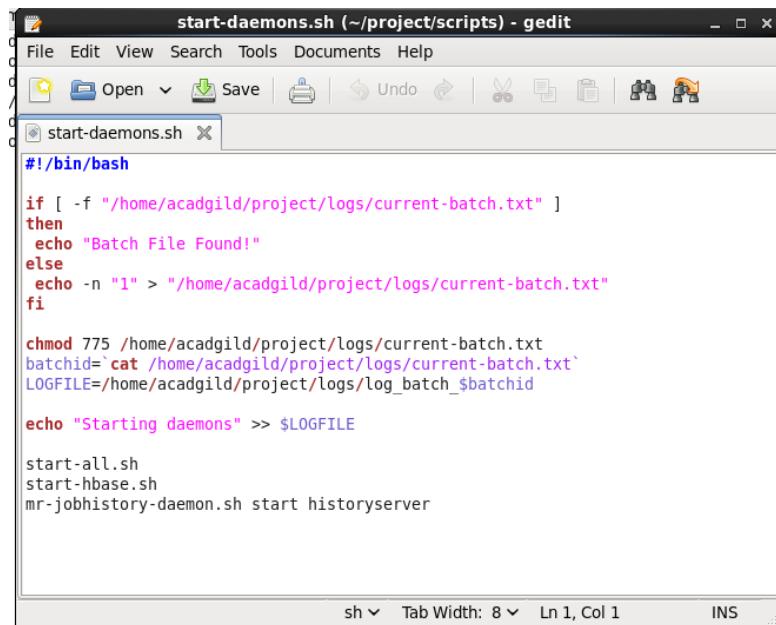
Launch Mysql service



```
sudo service mysql start
[acadgild@localhost ~]$ sudo service mysql start
[sudo] password for acadgild:
Starting mysqld: [ OK ]
[acadgild@localhost ~]$
```

[Start-daemons.sh](#) is the file that contains command to start all the daemons required for this project.

Daemons include - dfs, yarn, hbase and jobhistory daemons.



```

#!/bin/bash

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

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

echo "Starting daemons" >> $LOGFILE

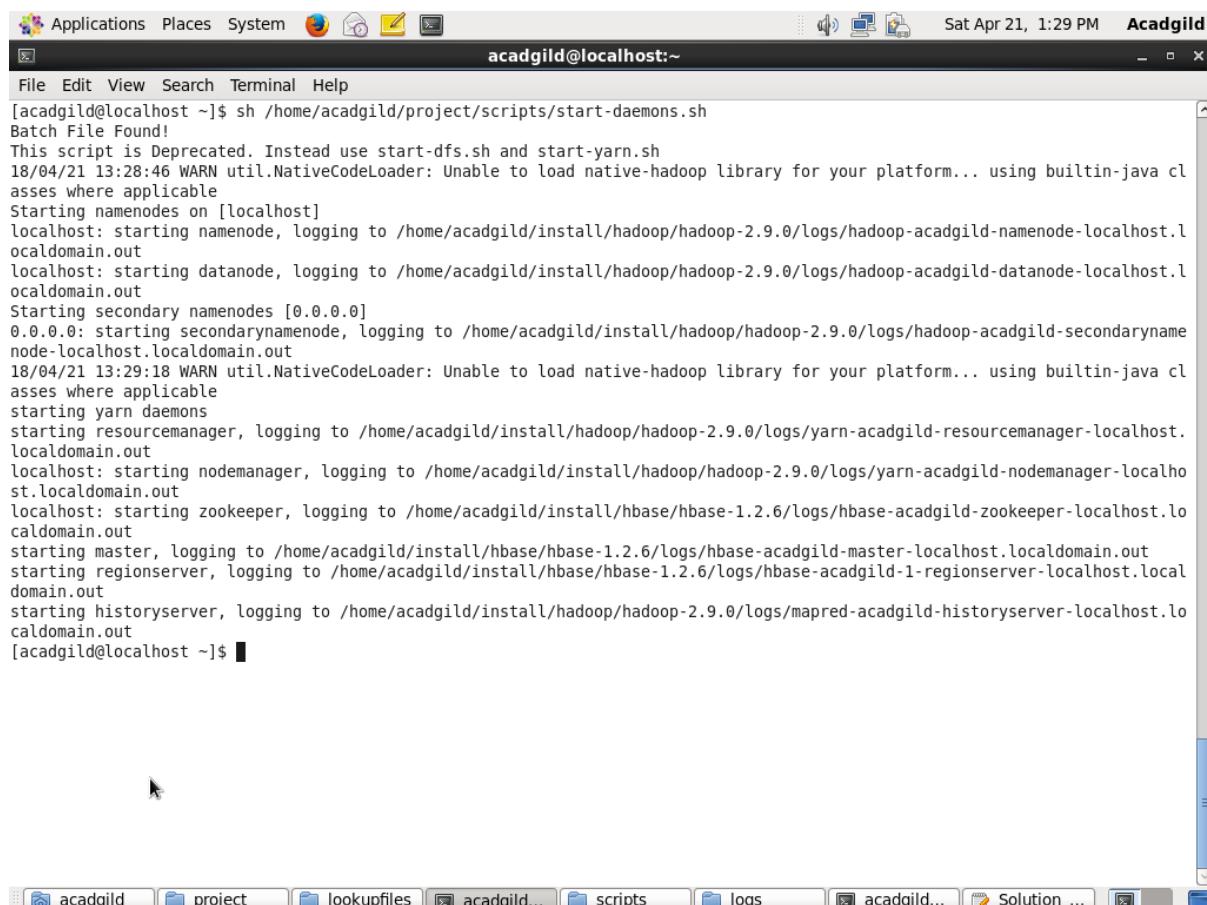
start-all.sh
start-hbase.sh
mr-jobhistory-daemon.sh start historyserver

```

**Current-batch.sh** is the file which contains batch id.

**Chmod 775** provides all the necessary permissions to current-batch.sh file.

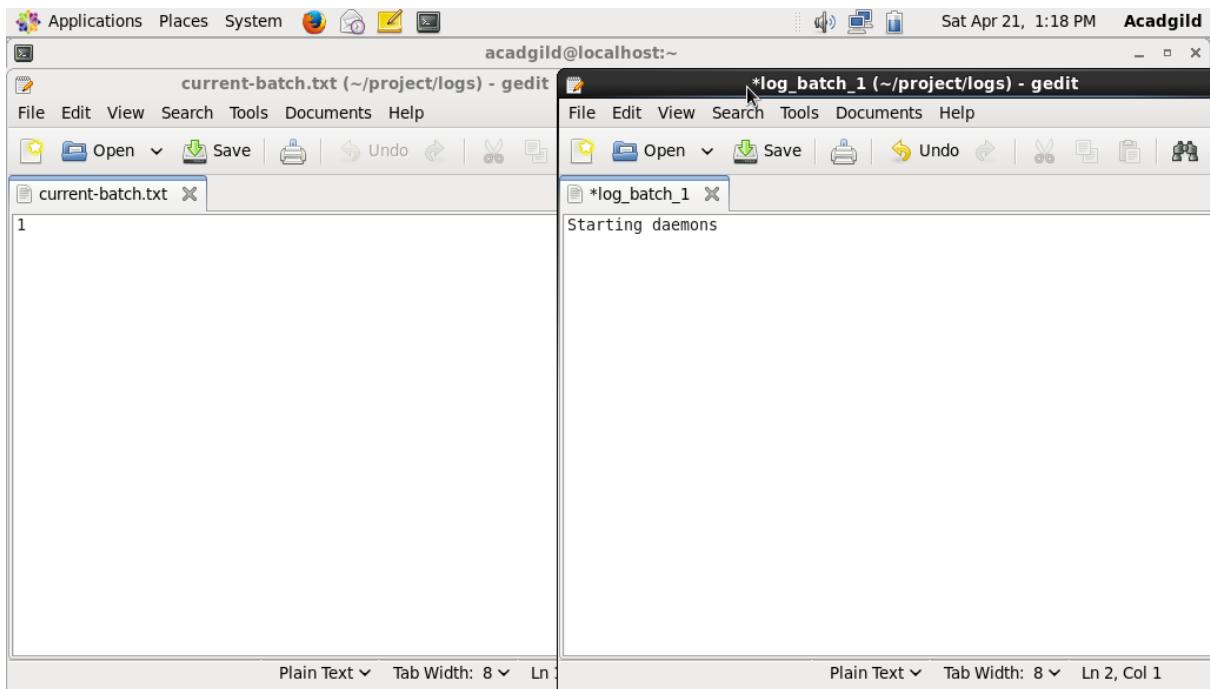
**Echo** command writes the string to the Logfile.



```

[acadgild@localhost ~]$ sh /home/acadgild/project/scripts/start-daemons.sh
Batch File Found!
This script is Deprecated. Instead use start-dfs.sh and start-yarn.sh
18/04/21 13:28:46 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Starting namenodes on [localhost]
localhost: starting namenode, logging to /home/acadgild/install/hadoop/hadoop-2.9.0/logs/hadoop-acadgild-namenode-localhost.localdomain.out
localhost: starting datanode, logging to /home/acadgild/install/hadoop/hadoop-2.9.0/logs/hadoop-acadgild-datanode-localhost.localdomain.out
Starting secondary namenodes [0.0.0.0]
0.0.0.0: starting secondarynamenode, logging to /home/acadgild/install/hadoop/hadoop-2.9.0/logs/hadoop-acadgild-secondaryname-node-localhost.localdomain.out
18/04/21 13:29:18 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
starting yarn daemons
starting resourcemanager, logging to /home/acadgild/install/hadoop/hadoop-2.9.0/logs/yarn-acadgild-resourcemanager-localhost.localdomain.out
localhost: starting nodemanager, logging to /home/acadgild/install/hadoop/hadoop-2.9.0/logs/yarn-acadgild-nodemanager-localhost.localdomain.out
localhost: starting zookeeper, logging to /home/acadgild/install/hbase/hbase-1.2.6/logs/hbase-acadgild-zookeeper-localhost.localdomain.out
starting master, logging to /home/acadgild/install/hbase/hbase-1.2.6/logs/hbase-acadgild-master-localhost.localdomain.out
starting regionserver, logging to /home/acadgild/install/hbase/hbase-1.2.6/logs/hbase-acadgild-1-regionserver-localhost.localdomain.out
starting historyserver, logging to /home/acadgild/install/hadoop/hadoop-2.9.0/logs/mapred-acadgild-historyserver-localhost.localdomain.out
[acadgild@localhost ~]$

```



### 3) Populate look up tables into HBase :-

#### Execution script - [populate-lookup.sh](#)

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

The following operations are performed:

- ✓ Get the batch id number from the batch file and get the Log File for the batch using the batch id.
- ✓ Create the HBase tables for the lookup data files: **song-artist**, **stn-geocd** and **user-subscn** with their column families.
- ✓ For every lookup data file, read each line, extract the columns (comma separated) and add the data as rows to the corresponding HBase tables.
- ✓ 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.

Sat Apr 21, 1:39 PM Acadgild

```

#!/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"

```

sh Tab Width: 8 Ln 1, Col 1 INS

Sun Apr 15, 1:37 PM Acadgild

```

[acadgild@localhost ~]$ sh /home/acadgild/project/scripts/populate-lookup.sh
2018-04-15 13:36:49,144 WARN [main] util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using built-in Java classes where applicable
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/acadgild/install/hbase/hbase-1.2.6/lib/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/acadgild/install/hadoop/hadoop-2.9.0/share/hadoop/common/lib/slf4j-log4j12-1.7.25.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
HBase Shell; enter 'help<RETURN>' for list of supported commands.
Type "exit<RETURN>" to leave the HBase Shell
Version 1.2.6, rUnknown, Mon May 29 02:25:32 CDT 2017

create 'station-geo-map', 'geo'
0 row(s) in 3.1800 seconds

Hbase::Table - station-geo-map
2018-04-15 13:37:04,954 WARN [main] util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using built-in Java classes where applicable
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/acadgild/install/hbase/hbase-1.2.6/lib/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/acadgild/install/hadoop/hadoop-2.9.0/share/hadoop/common/lib/slf4j-log4j12-1.7.25.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
HBase Shell; enter 'help<RETURN>' for list of supported commands.
Type "exit<RETURN>" to leave the HBase Shell
Version 1.2.6, rUnknown, Mon May 29 02:25:32 CDT 2017

create 'subscribed-users', 'subscn'
0 row(s) in 1.7760 seconds

Hbase::Table - subscribed-users

```

[acadgild@localhost ~] project - File Browser acadgild@localhost:~

Applications Places System

Sun Apr 15, 1:37 PM Acadgild

acadgild@localhost:~

File Edit View Search Terminal Help

```

create 'subscribed-users', 'subscn'
0 row(s) in 1.7760 seconds

Hbase::Table - subscribed-users
2018-04-15 13:37:20,021 WARN [main] util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using built-in java classes where applicable
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/acadgild/install/hbase/hbase-1.2.6/lib/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/acadgild/install/hadoop/hadoop-2.9.0/share/hadoop/common/lib/slf4j-log4j12-1.7.25.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
HBase Shell; enter 'help<RETURN>' for list of supported commands.
Type "exit<RETURN>" to leave the HBase Shell
Version 1.2.6, rUnknown, Mon May 29 02:25:32 CDT 2017

create 'song-artist-map', 'artist'
0 row(s) in 1.8760 seconds

Hbase::Table - song-artist-map
2018-04-15 13:37:35,233 WARN [main] util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using built-in java classes where applicable
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/acadgild/install/hbase/hbase-1.2.6/lib/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/acadgild/install/hadoop/hadoop-2.9.0/share/hadoop/common/lib/slf4j-log4j12-1.7.25.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
HBase Shell; enter 'help<RETURN>' for list of supported commands.
Type "exit<RETURN>" to leave the HBase Shell
Version 1.2.6, rUnknown, Mon May 29 02:25:32 CDT 2017

put 'station-geo-map', 'ST400', 'geo:geo_cd', 'A'
0 row(s) in 0.6400 seconds

```

[acadgild@localhost:~] project - File Browser acadgild@localhost:~

Applications Places System

Sun Apr 15, 1:56 PM Acadgild

acadgild@localhost:~

File Edit View Search Terminal Help

```

hbase(main):006:0> scan 'song-artist-map'
ROW                                COLUMN+CELL
S200                               column=artist:artistid, timestamp=1523779859156, value=A300
S201                               column=artist:artistid, timestamp=1523779872166, value=A301
S202                               column=artist:artistid, timestamp=1523779884773, value=A302
S203                               column=artist:artistid, timestamp=1523779897975, value=A303
S204                               column=artist:artistid, timestamp=1523779911190, value=A304
S205                               column=artist:artistid, timestamp=1523779924346, value=A301
S206                               column=artist:artistid, timestamp=1523779937362, value=A302
S207                               column=artist:artistid, timestamp=1523779950624, value=A303
S208                               column=artist:artistid, timestamp=1523779963671, value=A304
S209                               column=artist:artistid, timestamp=1523779976967, value=A305
10 row(s) in 0.3720 seconds

hbase(main):007:0> scan 'station-geo-map'
ROW                                COLUMN+CELL
ST400                              column=geo:geo_cd, timestamp=1523779658767, value=A
ST401                              column=geo:geo_cd, timestamp=1523779675270, value=AP
ST402                              column=geo:geo_cd, timestamp=1523779689003, value=AP
ST403                              column=geo:geo_cd, timestamp=1523779702356, value=J
ST404                              column=geo:geo_cd, timestamp=1523779715296, value=E
ST405                              column=geo:geo_cd, timestamp=1523779728259, value=A
ST406                              column=geo:geo_cd, timestamp=1523779741383, value=AU
ST407                              column=geo:geo_cd, timestamp=1523779754427, value=AP
ST408                              column=geo:geo_cd, timestamp=1523779767330, value=E
ST409                              column=geo:geo_cd, timestamp=1523779780718, value=E
ST410                              column=geo:geo_cd, timestamp=1523779793458, value=A
ST411                              column=geo:geo_cd, timestamp=1523779807206, value=A
ST412                              column=geo:geo_cd, timestamp=1523779819957, value=AP
ST413                              column=geo:geo_cd, timestamp=1523779832899, value=J
ST414                              column=geo:geo_cd, timestamp=1523779846044, value=E
15 row(s) in 0.1810 seconds

hbase(main):008:0> scan 'subscribed-users'
ROW                                COLUMN+CELL
U100                               column=subscn:enddt, timestamp=1523780002236, value=1465130523
U100                               column=subscn:startdt, timestamp=1523779989824, value=1465230523
U101                               column=subscn:enddt, timestamp=1523780028224, value=1475130523
U101                               column=subscn:startdt, timestamp=1523780015056, value=1465230523

```

[acadgild@localhost:~] project - File Browser acadgild@localhost:~

## Content of user-artist.hql :

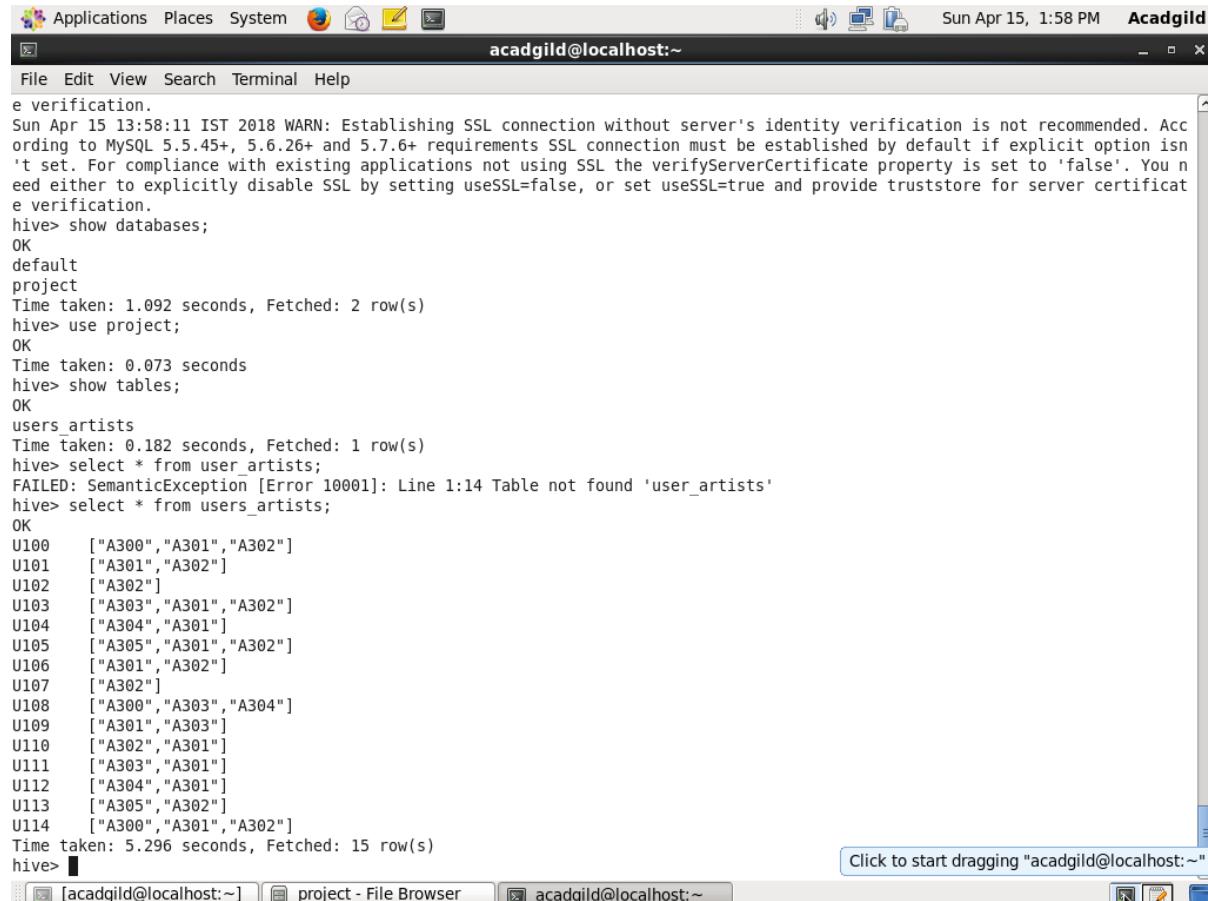
```
CREATE DATABASE IF NOT EXISTS project;
USE project;

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

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

//A lateral view with explode() can be used to convert artists_array into separate rows using the query

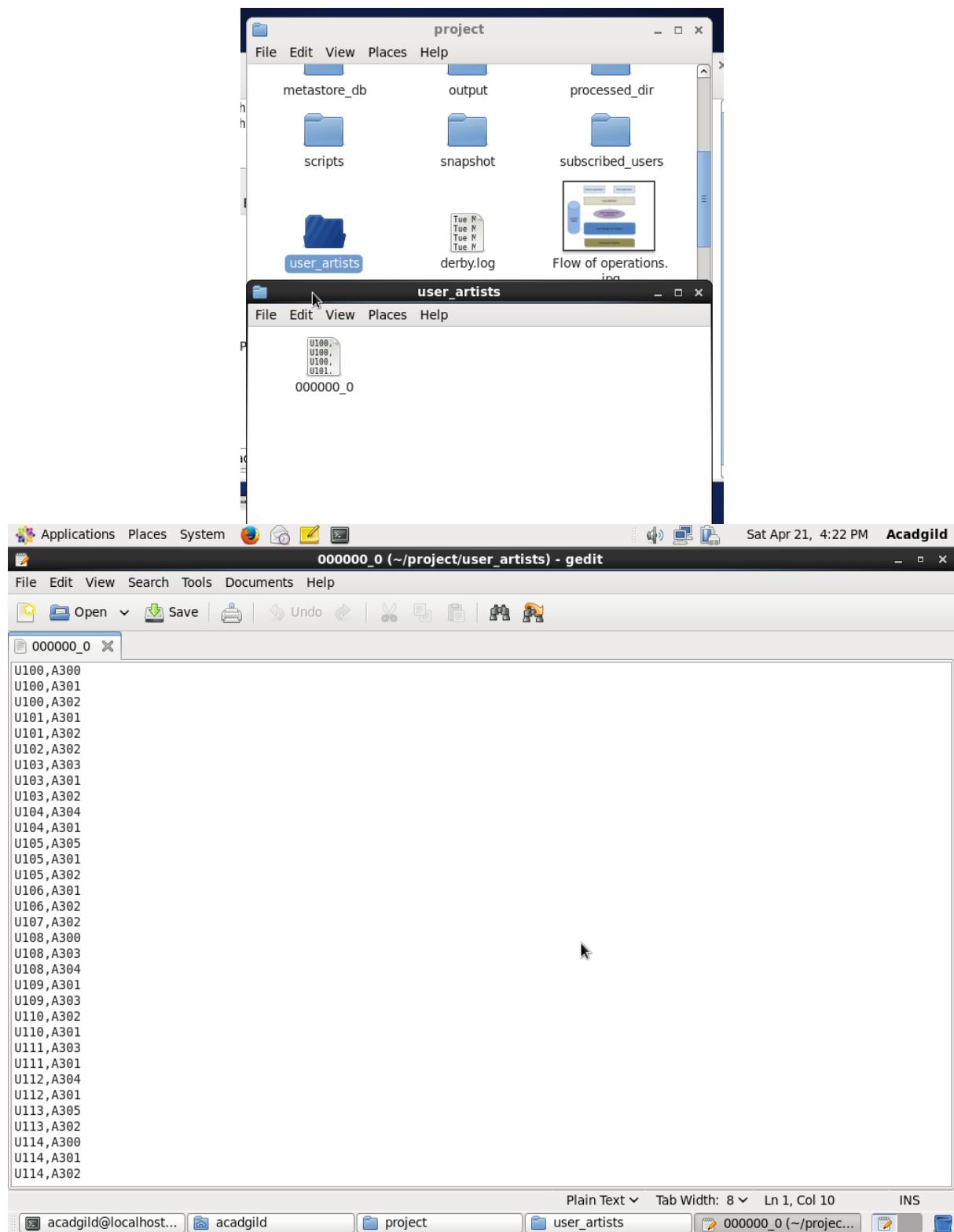
INSERT OVERWRITE LOCAL DIRECTORY '/home/acadgild/project/user_artists/'
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
STORED AS TEXTFILE
SELECT user_id, artists FROM users_artists LATERAL VIEW explode(artists_array) a AS artists;
```



The screenshot shows a terminal window titled "acadgild@localhost:~". The window contains the following Hive commands and their output:

```
File Edit View Search Terminal Help
e verification.
Sun Apr 15 13:58:11 IST 2018 WARN: Establishing SSL connection without server's identity verification is not recommended. Acc
ording to MySQL 5.5.45+, 5.6.26+ and 5.7.6+ requirements SSL connection must be established by default if explicit option isn
't set. For compliance with existing applications not using SSL the verifyServerCertificate property is set to 'false'. You n
eed either to explicitly disable SSL by setting useSSL=false, or set useSSL=true and provide truststore for server certificat
e verification.
hive> show databases;
OK
default
project
Time taken: 1.092 seconds, Fetched: 2 row(s)
hive> use project;
OK
Time taken: 0.073 seconds
hive> show tables;
OK
users_artists
Time taken: 0.182 seconds, Fetched: 1 row(s)
hive> select * from user_artists;
FAILED: SemanticException [Error 10001]: Line 1:14 Table not found 'user_artists'
hive> select * from users_artists;
OK
U100  ["A300","A301","A302"]
U101  ["A301","A302"]
U102  ["A302"]
U103  ["A303","A301","A302"]
U104  ["A304","A301"]
U105  ["A305","A301","A302"]
U106  ["A301","A302"]
U107  ["A302"]
U108  ["A300","A303","A304"]
U109  ["A301","A303"]
U110  ["A302","A301"]
U111  ["A303","A301"]
U112  ["A304","A301"]
U113  ["A305","A302"]
U114  ["A300","A301","A302"]
Time taken: 5.296 seconds, Fetched: 15 row(s)
hive> 
```

## Output of Hive explode :



#### 4) Perform Data Formatting :-

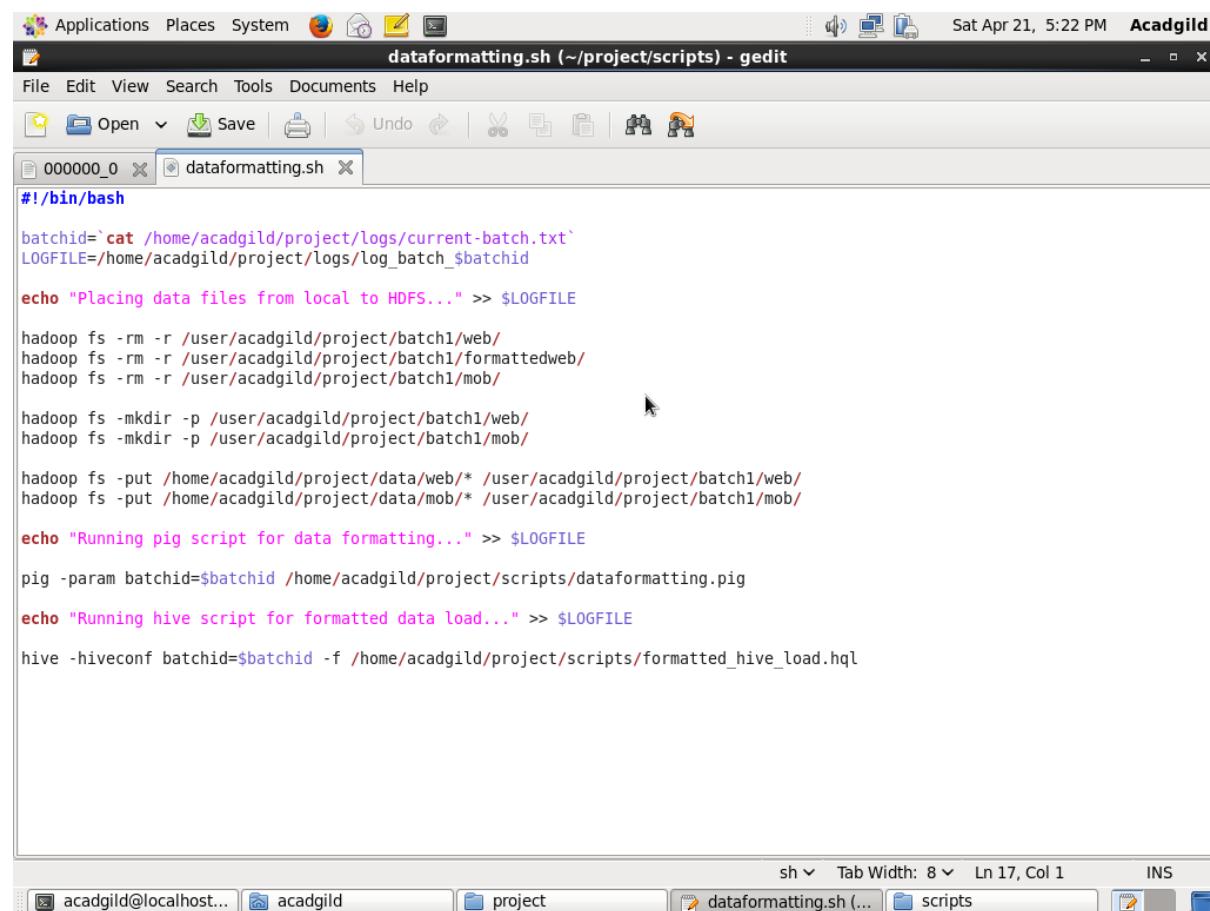
##### Execution Script - **dataformatting.sh**

- ✓ Firstly we will be executing **dataformatting.pig** which will format web xml data using pig to CSV.
- ✓ Then Load mob and formatted web data to hive using **formatted\_hive\_load.hql**

Following operations are performed in **dataformatting.sh** :

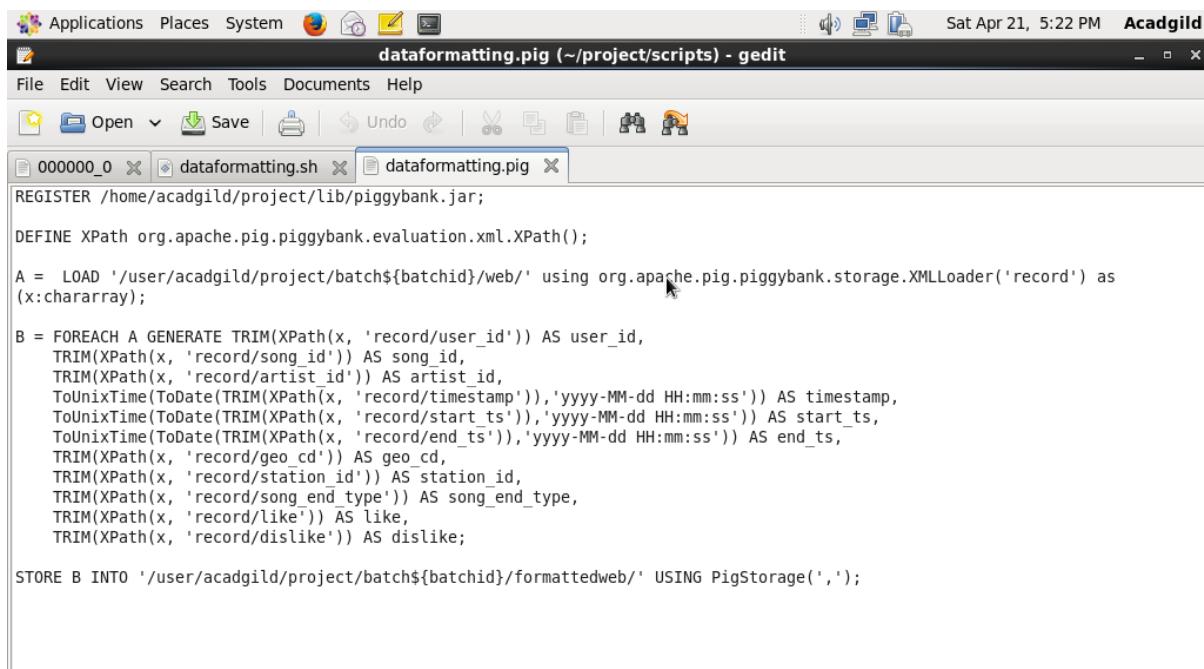
- ✓ Fetching batch id from the current\_batch file.
- ✓ Writing to the log file about process start.
- ✓ Checking if the directory is present in HDFS if yes, then deleting it and creating new blank directory.
- ✓ Copying mob and formatted web data to HDFS from local.
- ✓ Run pig script **dataformatting.pig**, to format web data stored in HDFS as xml to csv format and store formatted file to formattedweb directory.
- ✓ Run hive script **formatted\_hive\_load.hql** to load data from mob and formattedweb directory to hive tables.

##### **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/batch1/web/
hadoop fs -rm -r /user/acadgild/project/batch1/formattedweb/
hadoop fs -rm -r /user/acadgild/project/batch1/mob/
hadoop fs -mkdir -p /user/acadgild/project/batch1/web/
hadoop fs -mkdir -p /user/acadgild/project/batch1/mob/
hadoop fs -put /home/acadgild/project/data/web/* /user/acadgild/project/batch1/web/
hadoop fs -put /home/acadgild/project/data/mob/* /user/acadgild/project/batch1/mob/
echo "Running pig script for data formatting..." >> $LOGFILE
pig -param batchid=$batchid /home/acadgild/project/scripts/dataformatting.pig
echo "Running hive script for formatted data load..." >> $LOGFILE
hive -hiveconf batchid=$batchid -f /home/acadgild/project/scripts/formatted_hive_load.hql
```

## dataformatting.pig



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

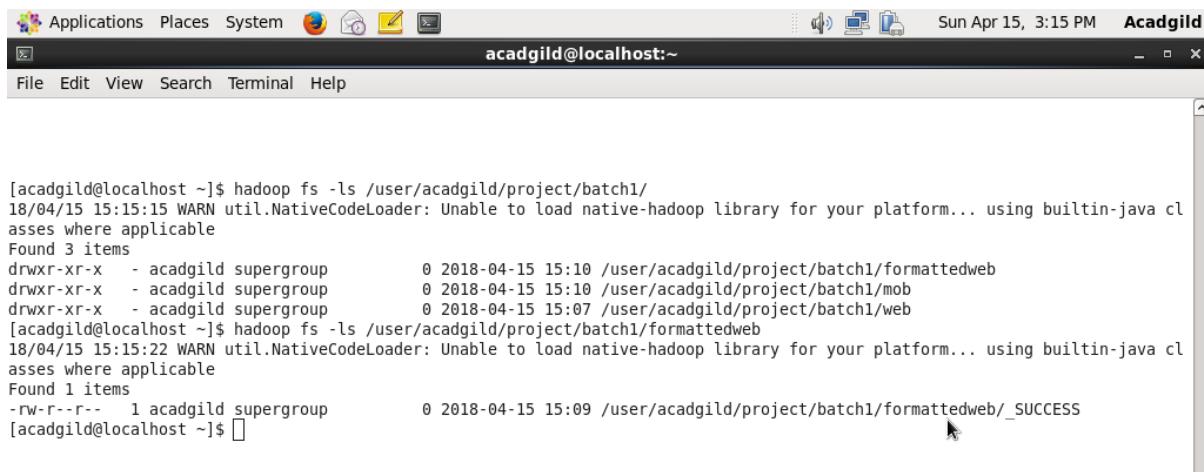
DEFINE XPath org.apache.pig.piggybank.evaluation.xml.XPath();

A = LOAD '/user/acadgild/project/batch${batchid}/web/' using org.apache.pig.piggybank.storageXMLLoader('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,
    ToUnixTimeToDate(TRIM(XPath(x, 'record/timestamp')), 'yyyy-MM-dd HH:mm:ss') AS timestamp,
    ToUnixTimeToDate(TRIM(XPath(x, 'record/start_ts')), 'yyyy-MM-dd HH:mm:ss') AS start_ts,
    ToUnixTimeToDate(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(',');
```

## Output :



```
[acadgild@localhost ~]$ hadoop fs -ls /user/acadgild/project/batch1/
18/04/15 15:15:15 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Found 3 items
drwxr-xr-x  - acadgild supergroup          0 2018-04-15 15:10 /user/acadgild/project/batch1/formattedweb
drwxr-xr-x  - acadgild supergroup          0 2018-04-15 15:10 /user/acadgild/project/batch1/mob
drwxr-xr-x  - acadgild supergroup          0 2018-04-15 15:07 /user/acadgild/project/batch1/web
[acadgild@localhost ~]$ hadoop fs -ls /user/acadgild/project/batch1/formattedweb
18/04/15 15:15:22 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Found 1 items
-rw-r--r--  1 acadgild supergroup          0 2018-04-15 15:09 /user/acadgild/project/batch1/formattedweb/_SUCCESS
[acadgild@localhost ~]$
```

## formatted\_hive\_load.hql

```
USE project;

CREATE TABLE IF NOT EXISTS formatted_input
(
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)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ',';

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

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

## 5) Perform Data Enrichment and Cleaning :-

The data enrichment is carried out in two steps:

- ✓ Create lookup tables in Hive and import the data from the HBase lookup tables to them. This is done by shell script [\*\*data\\_enrichment\\_filtering\\_schema.sh\*\*](#)
- ✓ Perform the data enrichment to the data in formatted\_input using the lookup tables. This is done by shell script [\*\*data\\_enrichment.sh\*\*](#)

### [\*\*data\\_enrichment\\_filtering\\_schema.sh\*\*](#)

Below is the shell script [\*\*data\\_enrichment\\_filtering\\_schema.sh\*\*](#) where the following operations are performed:

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

Add logs to the Log File signifying that the Hive lookup tables are created from the HBase lookup tables.

Run the hive script [create\\_hive\\_hbase\\_lookup.hql](#). This will create the lookup tables in Hive and import the data from the HBase lookup tables to the Hive lookup tables.

```
#!/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
```

```
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");

INSERT OVERWRITE local DIRECTORY '/home/acadgild/project/user_artists'
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
STORED AS TEXTFILE
SELECT * FROM user_artists;

create external table if not exists song_artist_map
(
song_id STRING,
artist_id STRING
)
```

**Output :**

```

Applications Places System Sun Apr 15, 3:17 PM Acadgild
acadgild@localhost:~ File Edit View Search Terminal Help
Sun Apr 15 15:16:55 IST 2018 WARN: Establishing SSL connection without server's identity verification is not recommended. According to MySQL 5.5.45+, 5.6.26+ and 5.7.6+ requirements SSL connection must be established by default if explicit option isn't set. For compliance with existing applications not using SSL the verifyServerCertificate property is set to 'false'. You need either to explicitly disable SSL by setting useSSL=false, or set useSSL=true and provide truststore for server certificate verification.
Sun Apr 15 15:16:55 IST 2018 WARN: Establishing SSL connection without server's identity verification is not recommended. According to MySQL 5.5.45+, 5.6.26+ and 5.7.6+ requirements SSL connection must be established by default if explicit option isn't set. For compliance with existing applications not using SSL the verifyServerCertificate property is set to 'false'. You need either to explicitly disable SSL by setting useSSL=false, or set useSSL=true and provide truststore for server certificate verification.
Sun Apr 15 15:16:59 IST 2018 WARN: Establishing SSL connection without server's identity verification is not recommended. According to MySQL 5.5.45+, 5.6.26+ and 5.7.6+ requirements SSL connection must be established by default if explicit option isn't set. For compliance with existing applications not using SSL the verifyServerCertificate property is set to 'false'. You need either to explicitly disable SSL by setting useSSL=false, or set useSSL=true and provide truststore for server certificate verification.
Sun Apr 15 15:17:00 IST 2018 WARN: Establishing SSL connection without server's identity verification is not recommended. According to MySQL 5.5.45+, 5.6.26+ and 5.7.6+ requirements SSL connection must be established by default if explicit option isn't set. For compliance with existing applications not using SSL the verifyServerCertificate property is set to 'false'. You need either to explicitly disable SSL by setting useSSL=false, or set useSSL=true and provide truststore for server certificate verification.
Sun Apr 15 15:17:00 IST 2018 WARN: Establishing SSL connection without server's identity verification is not recommended. According to MySQL 5.5.45+, 5.6.26+ and 5.7.6+ requirements SSL connection must be established by default if explicit option isn't set. For compliance with existing applications not using SSL the verifyServerCertificate property is set to 'false'. You need either to explicitly disable SSL by setting useSSL=false, or set useSSL=true and provide truststore for server certificate verification.
Sun Apr 15 15:17:00 IST 2018 WARN: Establishing SSL connection without server's identity verification is not recommended. According to MySQL 5.5.45+, 5.6.26+ and 5.7.6+ requirements SSL connection must be established by default if explicit option isn't set. For compliance with existing applications not using SSL the verifyServerCertificate property is set to 'false'. You need either to explicitly disable SSL by setting useSSL=false, or set useSSL=true and provide truststore for server certificate verification.
OK
Time taken: 13.667 seconds
OK
Time taken: 9.704 seconds
OK
Time taken: 0.7 seconds
OK
Time taken: 0.341 seconds
[acadgild@localhost ~]$ 

```

```

Applications Places System Mon Apr 16, 9:50 PM Acadgild
acadgild@localhost:~ File Edit View Search Terminal Help
Time taken: 13.081 seconds
OK
Time taken: 4.693 seconds
OK
Time taken: 0.429 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_20180416213648_89fc7e1c-a3dd-44e6-9e80-2198a7a1cacc
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks is set to 0 since there's no reduce operator
Starting Job = job_1523891462666_0004, Tracking URL = http://localhost:8088/proxy/application_1523891462666_0004/
Kill Command = /home/acadgild/install/hadoop/hadoop-2.9.0/bin/hadoop job -kill job_1523891462666_0004
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 0
2018-04-16 21:37:51,460 Stage-1 map = 0%, reduce = 0%
2018-04-16 21:38:52,149 Stage-1 map = 0%, reduce = 0%
2018-04-16 21:39:52,157 Stage-1 map = 0%, reduce = 0%
2018-04-16 21:40:01,957 Stage-1 map = 100%, reduce = 0%, Cumulative CPU 4.04 sec
MapReduce Total cumulative CPU time: 4 seconds 40 msec
Ended Job = job_1523891462666_0004
Moving data to local directory /home/acadgild/project/subscribed_users
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1 Cumulative CPU: 4.04 sec HDFS Read: 10812 HDFS Write: 405 SUCCESS
Total MapReduce CPU Time Spent: 4 seconds 40 msec
OK
Time taken: 202.413 seconds
OK
Time taken: 6.944 seconds
[acadgild@localhost ~]$ hive
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/acadgild/install/hive/apache-hive-2.3.2-bin/lib/log4j-slf4j-impl-2.6.2.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/acadgild/install/hadoop/hadoop-2.9.0/share/hadoop/common/lib/slf4j-log4j12-1.7.25.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

```

```
Applications Places System Sun Apr 15, 5:46 PM Acadgild
acadgild@localhost:~ File Edit View Search Terminal Help
hive> select * from
> song_artist_map;
OK
S200 A300
S201 A301
S202 A302
S203 A303
S204 A304
S205 A301
S206 A302
S207 A303
S208 A304
S209 A305
Time taken: 3.751 seconds, Fetched: 10 row(s)
hive> select * from station_geo_map;
OK
ST400 A
ST401 AU
ST402 AP
ST403 J
ST404 E
ST405 A
ST406 AU
ST407 AP
ST408 E
ST409 E
ST410 A
ST411 A
ST412 AP
ST413 J
ST414 E
Time taken: 0.509 seconds, Fetched: 15 row(s)
hive> select * from subscribed_users;
OK
U100 1465230523 1465130523
U101 1465230523 1475130523
U102 1465230523 1475130523
U103 1465230523 1475130523
U104 1465230523 1475130523
acadgild@localhost:~ [batch_1 - File Browser] Solution_execution.txt...
```

```
Applications Places System Sun Apr 15, 5:47 PM Acadgild
acadgild@localhost:~ File Edit View Search Terminal Help
S209 A305
Time taken: 3.751 seconds, Fetched: 10 row(s)
hive> select * from station_geo_map;
OK
ST400 A
ST401 AU
ST402 AP
ST403 J
ST404 E
ST405 A
ST406 AU
ST407 AP
ST408 E
ST409 E
ST410 A
ST411 A
ST412 AP
ST413 J
ST414 E
Time taken: 0.509 seconds, Fetched: 15 row(s)
hive> select * from subscribed_users;
OK
U100 1465230523 1465130523
U101 1465230523 1475130523
U102 1465230523 1475130523
U103 1465230523 1475130523
U104 1465230523 1475130523
U105 1465230523 1475130523
U106 1465230523 1485130523
U107 1465230523 1455130523
U108 1465230523 1465230623
U109 1465230523 1475130523
U110 1465230523 1475130523
U111 1465230523 1475130523
U112 1465230523 1475130523
U113 1465230523 1485130523
U114 1465230523 1468130523
Time taken: 0.491 seconds, Fetched: 15 row(s)
hive> 
```

## [\*\*data\\_enrichment.sh\*\*](#)

The following operations are performed:

- ✓ Get the batch id number from the batch file and get the Log File for the batch using the batch id.
- ✓ Add logs to the Log File signifying that the data enrichment has begun.
- ✓ Run the hive script [\*\*data\\_enrichment.hql\*\*](#). This will create a Hive table enriched\_data that will hold the data that is partitioned based on given rules as pass or fail (status) and batchid.
- ✓ Copy the data from the pass and fail folders (valid & invalid) in the Hive warehouse to the Local FS.

## [\*\*data\\_enrichment.hql :\*\*](#)

### **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)
<i>Geo_cd</i>	<i>Station_id</i>	<i>Station_Geo_Map</i>
<i>Artist_id</i>	<i>Song_id</i>	<i>Song_Artist_Map</i>

```
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,
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};

INSERT OVERWRITE local DIRECTORY '/home/acadgild/project/enriched_data'
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ''
STORED AS TEXTFILE

SELECT * FROM enriched_data;

```

## Output in hive :

```

Applications Places System acadgild@localhost:~ Sat Apr 21, 6:37 PM
File Edit View Search Terminal Help
disable SSL by setting useSSL=false, or set useSSL=true and provide truststore f
or server certificate verification.
Sat Apr 21 18:37:08 IST 2018 WARN: Establishing SSL connection without server's
identity verification is not recommended. According to MySQL 5.5.45+, 5.6.26+ an
d 5.7.6+ requirements SSL connection must be established by default if explicit
option isn't set. For compliance with existing applications not using SSL the ve
rifyServerCertificate property is set to 'false'. You need either to explicitly
disable SSL by setting useSSL=false, or set useSSL=true and provide truststore f
or server certificate verification.
OK
Time taken: 11.847 seconds
hive> select * from enriched_data;
OK
S201 A301 1465130523 1465130523 1475130523 AP ST402 0 0 1 1 fail
U100 S201 A301 1494297562 1494297562 1465490556 J ST413 2 1 1 1 fail
U111 S202 A302 1465130523 1475130523 1465490556 F ST400 2 1 1 1 fail

```

```

Applications Places System acadgild@localhost:~ Sun Apr 15, 5:51 PM
File Edit View Search Terminal Help
U111 S203 A303 1494297562 1462863262 1494297562 J ST413 3 0 1 1 fail
U112 S203 A303 1465490556 1462863262 1462863262 AU ST406 3 1 1 1 fail
U105 S204 A304 1475130523 1485130523 1465130523 E ST408 3 0 0 1 fail
U114 S204 A304 1475130523 1475130523 1485130523 AP ST412 0 1 1 1 fail
U100 S205 A301 1465490556 1468094889 1465490556 E ST408 2 1 1 1 fail
U100 S207 A303 1465230523 1465230523 1465230523 J ST403 3 1 1 1 fail
U119 S207 A303 1462863262 1465490556 1468094889 NULL ST415 2 0 1 1 fail
U119 S208 A304 1462863262 1465490556 1465490556 NULL ST415 3 0 1 1 fail
U103 S209 A305 1495130523 1465230523 1465130523 E ST404 0 1 1 1 fail
U120 S210 NULL 1495130523 1465130523 1465230523 E ST414 3 1 1 1 fail
U108 S210 NULL 1495130523 1465130523 1465230523 E ST414 1 0 0 1 fail
U102 S210 NULL 1465130523 1465230523 1485130523 AP ST412 0 1 0 1 fail
U102 S210 NULL 1468094889 1465490556 1462863262 A ST410 0 0 0 1 fail
U111 S200 A300 1465490556 1462863262 1494297562 A ST400 3 0 1 1 pass
U116 S200 A300 1465490556 1462863262 1465490556 J ST413 1 0 1 1 pass
U103 S200 A300 1462863262 1468094889 AP ST412 1 1 0 1 pass
U100 S200 A300 1465230523 1465130523 1475130523 A ST410 1 0 1 1 pass
U108 S200 A300 1495130523 1475130523 1475130523 E ST408 2 1 0 1 pass
U109 S200 A300 1495130523 1465130523 1465230523 E ST408 0 0 1 1 pass
U103 S201 A301 1465130523 1465130523 1465130523 AP ST407 3 0 1 1 pass
U112 S202 A302 1475130523 1475130523 1465230523 AP ST402 2 0 0 1 pass
U108 S202 A302 1494297562 1462863262 1468094889 AU ST406 3 0 0 1 pass
U102 S202 A302 1495130523 1475130523 1475130523 AU ST406 3 1 0 1 pass
U111 S203 A303 1494297562 1494297562 1494297562 J ST413 1 0 1 1 pass
U120 S203 A303 1465490556 1468094889 1465490556 AU ST401 3 1 0 1 pass
U119 S203 A303 1465130523 1475130523 1465130523 E ST409 0 0 0 1 pass
U101 S203 A303 1475130523 1465130523 1475130523 E ST408 3 0 0 1 pass
U110 S203 A303 1465490556 1465490556 1468094889 A ST405 0 0 1 1 pass
U107 S203 A303 1494297562 1468094889 1462863262 E ST414 2 0 0 1 pass
U120 S204 A304 1468094889 1494297562 1494297562 J ST403 1 1 0 1 pass
U101 S205 A301 1494297562 1494297562 1465490556 AU ST401 0 0 0 1 pass
U102 S205 A301 1494297562 1465490556 1494297562 AU ST406 2 0 0 1 pass
U103 S205 A301 1475130523 1485130523 1465130523 E ST404 0 0 0 1 pass
U106 S205 A301 1465230523 1485130523 1485130523 AP ST407 2 1 0 1 pass
U114 S207 A303 1465230523 1465230523 1475130523 AU ST401 0 0 1 1 pass
U103 S208 A304 1462863262 1468094889 1494297562 A ST400 0 0 0 1 pass
U108 S209 A305 1465490556 1462863262 1465490556 J ST413 2 0 1 1 pass
Time taken: 3.082 seconds, Fetched: 40 row(s)
hive>
```

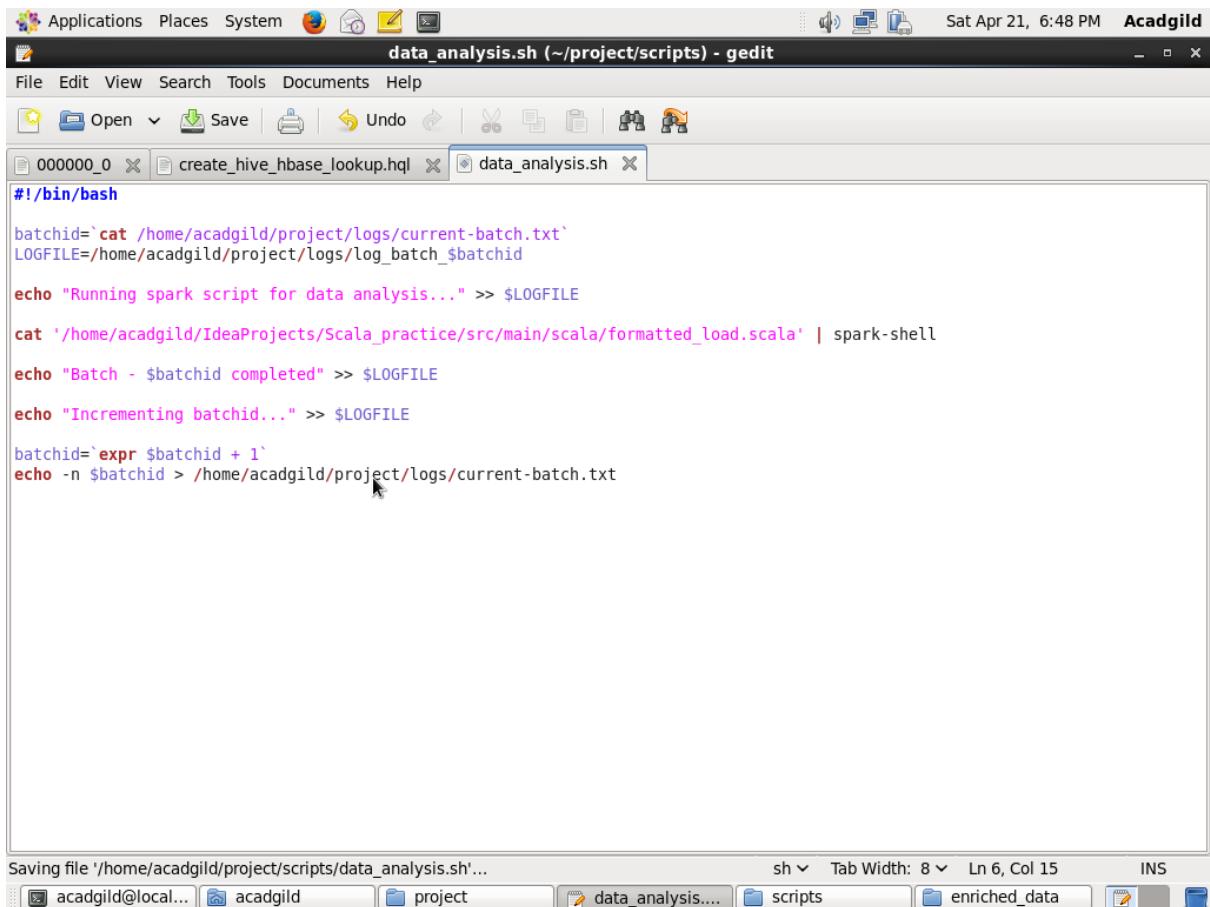
## Output in local :

The screenshot shows a Linux desktop environment. At the top, there is a panel with icons for Applications, Places, System, and a date/time indicator (Sat Apr 21, 6:27 PM). Below the panel is a terminal window titled "000000\_0 (~/project/enriched\_data) - gedit". The terminal contains a large amount of text output, which appears to be a log or data dump. The text includes numerous entries with fields like S#, A#, and various status codes (AP, ST402, ST413, etc.). To the right of the terminal is a file browser window titled "enriched\_data". The file browser shows a single file named "000000\_0" with a size of 2.7 KB. The bottom of the screen shows a dock with several icons, including "acadgild@local...", "acadgild", "project", "000000\_0 (~/p...", "scripts", and "enriched\_data".

## 6) Perform Data Analysis :-

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

- ✓ Get the batch id number from the batch file and get the Log File for the batch using the batch id.
- ✓ 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 **formatted\_load.scala**. This will perform the data analysis required in the problem statement given and save the result to the Local FS.
- ✓ Add logs to the Log File signifying that the data analysis has completed and that the batch is being incremented. Here from 1 to 2.



```
#!/bin/bash

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

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

cat '/home/acadgild/IdeaProjects/Scala_practice/src/main/scala/formatted_load.scala' | spark-shell

echo "Batch - $batchid completed" >> $LOGFILE

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

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

In the file **formatted\_load.scala** that will perform data analysis and store each result into local drive by creating a new directory with every new batch id.

Steps :

- ✓ Importing row, Sparksession, RDD, functions and types which are need for analysis.
- ✓ Then, we will retrieve the batch id from current\_batch file.
- ✓ We will now import data stored in local FS i.e. enriched\_data, subscribed\_user and user\_artists and perform the following operations for each :
  - Create the schema for the data
  - Create dataframe
  - Create temporary table

Now we are ready for analysis.

## formatted\_load.scala

The screenshot shows a Scala IDE interface with the following details:

- Top Bar:** Applications, Places, System, File, Edit, View, Search, Tools, Documents, Help.
- Title Bar:** \*formatted\_load.scala (~/IdeaProjects/Scala\_practice/src/main/scala) - gedit
- Toolbar:** Open, Save, Undo, Redo, Cut, Copy, Paste, Find, Select All, Find Next, Find Previous.
- Code Editor:** The main window displays Scala code for reading data from various sources and creating DataFrames. The code includes imports for RDD, Row, SparkSession, functions, and types. It handles extracting batch numbers, migrating enriched data, defining schemas for music and user data, and creating temporary views for further processing.

```
import org.apache.spark.rdd.RDD
import org.apache.spark.sql.{Row, SparkSession}
import org.apache.spark.sql.functions._
import org.apache.spark.sql.types._

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

//Migrating Enriched Data to DataFrame
val enr_data = sc.textFile("/home/acadgild/project/enriched_data/000000_0")

val enr_schema = StructType(Array(StructField("User_id", StringType, true), StructField("Song_id", StringType, true), StructField("Artist_id", StringType, true), StructField("u_Timestamp", StringType, true), StructField("Start_ts", StringType, true),
  StructField("End_ts", StringType, true), StructField("Geo_cd", StringType, true), StructField("Station_id", StringType, true), StructField("Song_end_type", IntegerType, true),
  StructField("u_Like", IntegerType, true), StructField("Dislike", IntegerType, true), StructField("BatchID", IntegerType, true), StructField("status", StringType, true)))
))

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

val music_data = spark.createDataFrame(enr_row_rdd, enr_schema)
music_data.createOrReplaceTempView("music_data")

//Subscribed user
val subs_data = sc.textFile("/home/acadgild/project/subscribed_users/000000_1")

val subs_schema = StructType(Array(StructField("User_id", StringType, true),
  StructField("start_dt", StringType, true),
  StructField("end_dt", StringType, true)))
)

val subs_row_rdd = subs_data.map(_.split(",")).map(r => Row(r(0), r(1), r(2)))

val subs_usr_data = spark.createDataFrame(subs_row_rdd, subs_schema)
subs_usr_data.createOrReplaceTempView("subscribed_user")

//val usr_data = sc.textFile("/home/acadgild/project/user_artists/000000_0")

val usr_schema = StructType(Array(StructField("User_id", StringType, true),
  StructField("artists", StringType, true)))
)

val usr_row_rdd = usr_data.map(_.split(",")).map(r => Row(r(0), r(1)))

val usr_artist_data = spark.createDataFrame(usr_row_rdd, usr_schema)
usr_artist_data.createOrReplaceTempView("users_artists")

//***** PROBLEM 1 *****
//***** PROBLEM 1 *****
//***** PROBLEM 1 *****

val prob_1 = spark.sql("select station_id, count(distinct song_id) Distinct_Songs, " +
  " count(distinct user_id) Distinct_users, batchid from music_data " +
  " where status='fail' and batchid = $batch_id " +
  " and u like=1 group by station_id,batchid order by Distinct_Songs " +
  " DESC limit 10").toDF("station_id", "Distinct_Songs", "Distinct_users", "batchid")
```

```

prob_1.write.format("csv").option("header","true").save(s"/home/acadgild/project/output/top_10_station/batch_$batch_id")

//*****
// PROBLEM 2
//*****

val prob_2 = spark.sql("select CASE WHEN ( subusr.user_id is null or " +
  "cast(music.u timestamp as decimal(20,0) > cast(subusr.end_dt as decimal(20,0)) " +
  ") then 'UNSUBSCRIBED' WHEN (subusr.user_id is NOT null or " +
  "cast(music.u timestamp as decimal(20,0) <= cast(subusr.end_dt as decimal(20,0)) " +
  ") then 'SUBSCRIBED' END AS USER_TYPE, " +
  "SUM(ABS(CAST(music.end_ts as decimal(20,0)) - CAST(music.start_ts as decimal(20,0))))" +
  " as duration ,batchid" +
  " from music_data music left outer join " +
  "subscribed user subusr on music.user_id = subusr.user_id " +
  "s"where music.status = 'pass' and music.batchid = $batch_id " +
  "s"group by music.batchid, CASE WHEN ( subusr.user_id is null or " +
  "cast(music.u.timestamp as decimal(20,0) > cast(subusr.end_dt as decimal(20,0)) " +
  ") then 'UNSUBSCRIBED' WHEN (subusr.user_id is NOT null or " +
  "cast(music.u.timestamp as decimal(20,0) <= cast(subusr.end_dt as decimal(20,0)) " +
  ") then 'SUBSCRIBED' END ".toDF()

prob_2.repartition(1).write.format("csv").option("header","true").save(s"/home/acadgild/project/output/user_response/
batch_$batch_id")

//*****
// PROBLEM 3
//*****



val prob_3 = spark.sql("select ua.artists, count(distinct ua.user_id) as count_of_users, " +
  "md.batchid from users_artists ua inner join (select artist_id,song_id,user_id,batchid" +
  "s" from music_data where status = 'pass' and batchid = $batch_id)md " +
  "s" on ua.artists = md.artist_id and ua.user_id = md.user_id group by ua.artists," +
  "s" batchid order by count_of_users DESC limit 10").toDF("user_id","count_of_users","batchid")

prob_3.write.format("csv").option("header","true").save(s"/home/acadgild/project/output/connected_artist/batch_
$batch_id")

//*****
// PROBLEM 4
//*****



val prob_4 = spark.sql("select song_id, " +
  "SUM(ABS(CAST(end_ts as decimal(20,0)) - " +
  " " CAST(start_ts as decimal(20,0)))) as duration,batchid" +
  "s" from music_data where status = 'pass' and batchid = $batch_id " +
  "and (u like= 1 or song_end_type = 0) group by song_id, batchid " +
  "order by duration desc limit 10").toDF("song_id","duration","batch_id")

prob_4.write.format("csv").option("header","true").save(s"/home/acadgild/project/output/royalty_songs/batch_$batch_id")

//*****
// PROBLEM 5
//*****



val prob_5 = spark.sql("select md.user_id, " +
  "SUM(ABS(CAST(end_ts as decimal(20,0)) - " +
  " " CAST(start_ts as decimal(20,0)))) as duration,batchid" +
  "s" from music_data md LEFT OUTER JOIN subscribed_user su " +
  "s"on md.user_id = su.user_id where md.status = 'pass' and md.batchid = $batch_id " +
  "and (md.user_id IS NULL or (CAST(md.u_timestamp as DECIMAL(20,0)) > " +
  " " cast(su.end_dt as DECIMAL(20,0))))" +
  " " group by md.user_id, batchid " +
  "order by duration desc limit 10").toDF()

prob_5.write.format("csv").option("header","true").save(s"/home/acadgild/project/output/unsubscribed_users/batch_$batch_id")

```

Plain Text ▾ Tab Width: 8 ▾ Ln 110, Col 122 INS



## Problem Statement :-

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.

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

Solution :

```
/*
PROBLEM 1
*/
val prob_1 = spark.sql("select station_id, count(distinct song_id) Distinct_Songs," +
  " count(distinct user_id) Distinct_Users, batchid from music_data " +
  "where status='fail' and batchid = $batch_id " +
  "and u like=1 group by station_id,batchid order by Distinct_Songs " +
  "DESC limit 10").toDF("station_id","Distinct_Songs","Distinct_Users","batchid")

prob_1.write.format("csv").option("header","true").save(s"/home/acadgild/project/output/top_10_station/batch_$batch_id")
```

The screenshot shows a Linux desktop environment with several windows open:

- Terminal Window:** Displays the Scala code for Problem 1, which reads data from a DataFrame named 'music\_data', filters for failed status and a specific batch ID, groups by station ID, and orders by the number of distinct songs in descending order to find the top 10.
- LibreOffice Calc Window:** Shows a spreadsheet with the following data:

	A	B	C	D	E
1	station_id	Distinct_Songs	Distinct_Users	batchid	
2	ST408	2	2	1	
3	ST412	2	2	1	
4	ST404	1	1	1	
5	ST413	1	1	1	
6	ST406	1	1	1	
7	ST414	1	1	1	
8	ST403	1	1	1	

- File Explorer Window:** Shows the directory structure for the output of Problem 1. It contains three main folders: 'connected\_artist', 'royalty\_songs', and 'top\_10\_station'. The 'top\_10\_station' folder contains a sub-folder 'batch\_1'.
- File Explorer Window (batch\_1):** Shows the contents of the 'batch\_1' folder, which includes a CSV file named 'part-00000-7217b7f0-fed0-419f-ab97-78aa006d6df8-c000.csv' and a file named '\_SUCCESS'.

Problem 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.

Solution :

```
/*
***** PROBLEM 2 *****
*****
val prob_2 = spark.sql("select CASE WHEN ( subusr.user_id is null or " +
    "cast(music.u_timestamp as decimal(20,0)) > cast(subusr.end_dt as decimal(20,0)) " +
    ") then 'UNSUBSCRIBED' WHEN (subusr.user_id is NOT null or " +
    "cast(music.u_timestamp as decimal(20,0)) <= cast(subusr.end_dt as decimal(20,0)) " +
    ") then 'SUBSCRIBED' END AS USER_TYPE, " +
    "SUM(ABS(CAST(music.end_ts as decimal(20,0)) - CAST(music.start_ts as decimal(20,0))))" +
    "as duration ,batchid" +
    " from music_data music left outer join " +
    "subscribed_user subusr on music.user_id = subusr.user_id " +
    "s"where music.status = 'pass' and music.batchid = $batch_id " +
    "s"group by music.batchid, CASE WHEN ( subusr.user_id is null or " +
    "cast(music.u_timestamp as decimal(20,0)) > cast(subusr.end_dt as decimal(20,0)) " +
    ") then 'UNSUBSCRIBED' WHEN (subusr.user_id is NOT null or " +
    "cast(music.u_timestamp as decimal(20,0)) <= cast(subusr.end_dt as decimal(20,0)) " +
    ") then 'SUBSCRIBED' END " ).toDF()

prob_2.repartition(1).write.format("csv").option("header","true").save(s"/home/acadgild/project/output/user_response/
batch_$batch_id")
```

A	B	C	D	E
1	USER TYPE	duration	batchid	
2	SUBSCRIBED	83838633		1
3	UNSUBSCRIBED	95936187		1
4				
5				
6				
7				
8				
9				
10				
11				
12				
13				
14				
15				
16				
17				
18				
19				
20				
21				
22				
23				
24				
25				
26				
27				
28				

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

Solution :

```
*****
// PROBLEM 3
*****

val prob_3 = spark.sql("select ua.artists, count(distinct ua.user_id) as count_of_users, " +
  "md.batchid from users_artists ua inner join (select artist_id,song_id,user_id,batchid" +
  "s" from music_data where status = 'pass' and batchid = $batch_id)md " +
  "s" on ua.artists = md.artist_id and ua.user_id = md.user_id group by ua.artists," +
  "s" batchid order by count_of_users DESC limit 10").toDF("user_id","count_of_users","batchid")

prob_3.write.format("csv").option("header","true").save(s"/home/acadgild/project/output/connected_artist/batch_
$batch_id")
```

The screenshot shows a Linux desktop environment with several windows open:

- Terminal Window:** The title bar says "part-00000-797829db-f3c8-47b8-91fa-". The content area displays the Scala code for Problem 3, which reads data from a database, performs a join with another dataset, groups by artist, and orders by the count of unique users. It then saves the results to a CSV file named "batch\_id.csv".
- File Manager Window:** The title bar says "output". It shows three folders: "connected\_artist", "royalty\_songs", and "top\_10\_station".
- File Browser Window:** The title bar says "Acadgild". It shows a folder structure under "connected\_artist/batch\_1". Inside "batch\_1", there is a file named "part-00000-797829db-f3c8-47b8-91fa-e9966fcdd1dc-c000.csv" and a file named "\_SUCCESS". The status bar at the bottom indicates "2 items, Free space: 2.0 GB".
- Bottom Taskbar:** Shows the current workspace with tabs for "part-00000-797829db-f3c8-47b8-91fa-e9966fcdd1dc-c000", "output", "connected\_artist", and "batch\_1".

Problem 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.

Solution :

```
*****  
// PROBLEM 4  
*****  
  
val prob_4 = spark.sql("select song_id, " +  
  "SUM(ABS(CAST(end_ts as decimal(20,0)) - " +  
    " CAST(start_ts as decimal(20,0)))) as duration,batchid" +  
  "s" from music_data where status = 'pass' and batchid = $batch_id " +  
  "and (u like= 1 or song_end type = 0) group by song_id, batchid " +  
  "order by duration desc limit 10").toDF("song_id","duration","batch_id")  
  
prob_4.write.format("csv").option("header","true").save(s"/home/acadgild/project/output/royalty_songs/batch_$batch_id")
```

The screenshot shows a desktop environment with several windows open:

- Spreadsheet Window:** A LibreOffice Calc window titled "part-00000-b69c369c-efa0-4ad1-b851-c000". It displays a table with columns A, B, C, D, E, F. The first few rows of data are:

	A	B	C	D	E	F
1	song_id	duration	batch_id			
2	S205	48807006	1			
3	S208	26202673	1			
4	S203	15208666	1			
5	S207	9900000	1			
6	S200	5331627	1			
7	S202	0	1			
8	S204	0	1			
- File Explorer Window:** A Nautilus window titled "output" containing folders: "connected\_artist", "royalty\_songs", "top\_10\_station", "unsubscribed\_users", and "user\_response".
- Terminal Window:** A terminal window titled "batch\_1" showing the command "part-00000-b69c369c-efa0-4ad1-b851-b0855fdac837-c000.csv" and a file named "\_SUCCESS".
- Bottom Taskbar:** Shows the current workspace, a terminal icon, and several file icons for "output", "part-00000-8a...", "royalty\_songs", "batch\_1", and "part-00000-b69...".

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

Solution :

```
/*
***** PROBLEM 5 *****
***** */

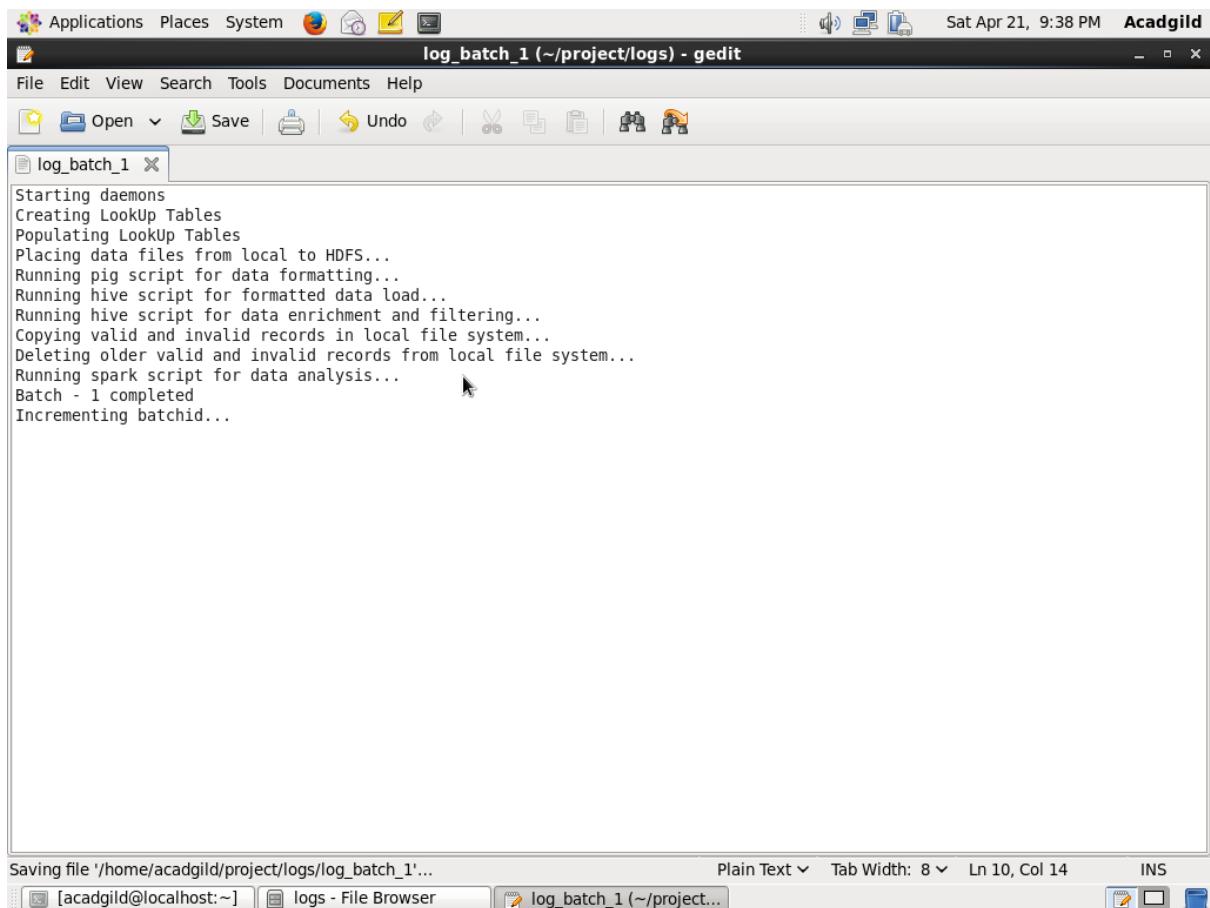
val prob_5 = spark.sql("select md.user_id, " +
  "SUM(ABS(CAST(end_ts as decimal(20,0)) - " +
  "CAST(start_ts as decimal(20,0)))) as duration,batchid" +
  "s" from music_data md LEFT OUTER JOIN subscribed_user su " +
  "on md.user_id = su.user_id where md.status = 'pass' and md.batchid = $batch_id " +
  "and (md.user_id IS NULL or (CAST(md.u_timestamp as DECIMAL(20,0)) > " +
  "cast(su.end_dt as DECIMAL(20,0))))" +
  " group by md.user_id, batchid " +
  "order by duration desc limit 10").toDF()

prob_5.write.format("csv").option("header","true").save(s"/home/acadgild/project/output/unsubscribed_users/batch_$batch_id")
```

The screenshot shows a desktop environment with several windows open:

- A terminal window titled "part-00000-074d3639-3db6-4aaa-9c29-491132ebba52-c000" containing the Scala code for Problem 5.
- A file browser window titled "output" showing directories for "connected\_artist", "royalty\_songs", "top\_10\_station", "unsubscribed\_users", and "user\_response".
- A file browser window titled "unsubscribed\_users" showing a folder named "batch\_1".
- A terminal window titled "batch\_1" showing the output of the command: "part-00000-074d3639-3db6-4aaa-9c29-491132ebba52-c000.csv".
- A file browser window titled "batch\_1" showing files "user\_id", "duration", "batchid", and "SUCCESS".
- A terminal window at the bottom showing the command "part-00000-074d3639-3db6-4aaa-9c29-491132ebba52-c000" and a list of tabs including "Sheet 1 / 1", "Default", "Sum=0", "100%", "[acadgild@localhost...]", "output", "unsubscribed\_users", "batch\_1", and "part-00000-074d3639-3db6-4aaa-9c29-491132ebba52-c000".

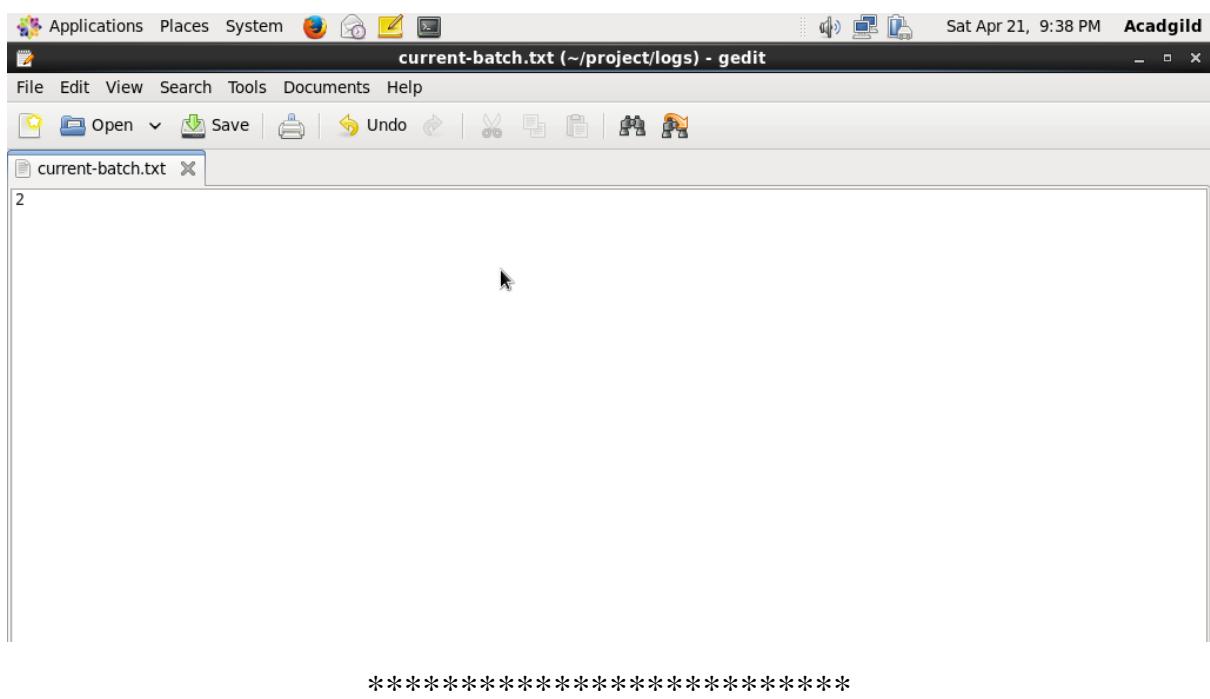
## View of Log File after Data Analysis :



The screenshot shows a Gedit text editor window titled "log\_batch\_1 (~/project/logs) - gedit". The status bar indicates it's saving the file to "/home/acadgild/project/logs/log\_batch\_1..." and shows "Plain Text" mode, tab width 8, line 10, column 14, and "INS" (insert mode). The main text area contains the following log entries:

```
Starting daemons
Creating LookUp Tables
Populating Lookup Tables
Placing data files from local to HDFS...
Running pig script for data formatting...
Running hive script for formatted data load...
Running hive script for data enrichment and filtering...
Copying valid and invalid records in local file system...
Deleting older valid and invalid records from local file system...
Running spark script for data analysis...
Batch - 1 completed
Incrementing batchid...
```

Incremented Batch id to 2



The screenshot shows a Gedit text editor window titled "current-batch.txt (~/project/logs) - gedit". The status bar indicates it's saving the file to "/home/acadgild/project/logs/current-batch.txt..." and shows "Plain Text" mode, tab width 8, line 10, column 14, and "INS" (insert mode). The main text area contains the number "2" followed by a series of asterisks at the bottom.

```
2
*****

```