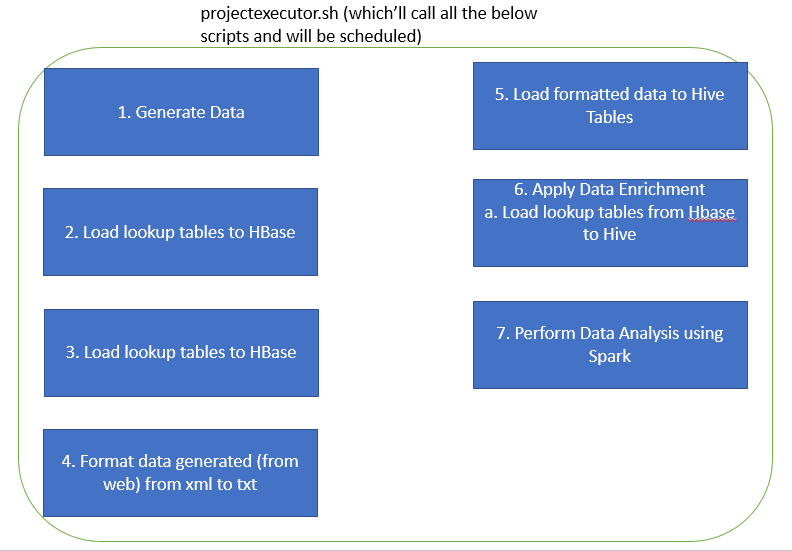
# Project – Music Data Analysis

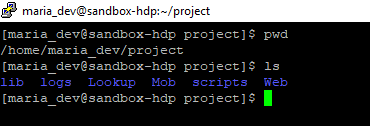
Overall Design:



# Step 1: Download the datafiles and lookup files and place them into a directory (generate\_mob\_data.py & generate\_web\_data.py)

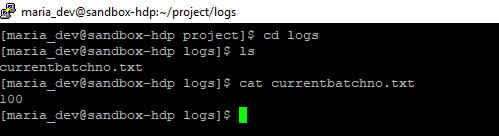
Project directory structure.

Generating input files will be through two python scripts generate\_mob\_data.py and generate\_web\_data.py. These two files will generate files with random data. Generate\_mob\_data.py will generated data in csv format while generate\_web\_data.py will generate data in xml format. Once generated they will be placed in /home/maria\_dev/project/Web & /home/maria\_dev/project/Mob



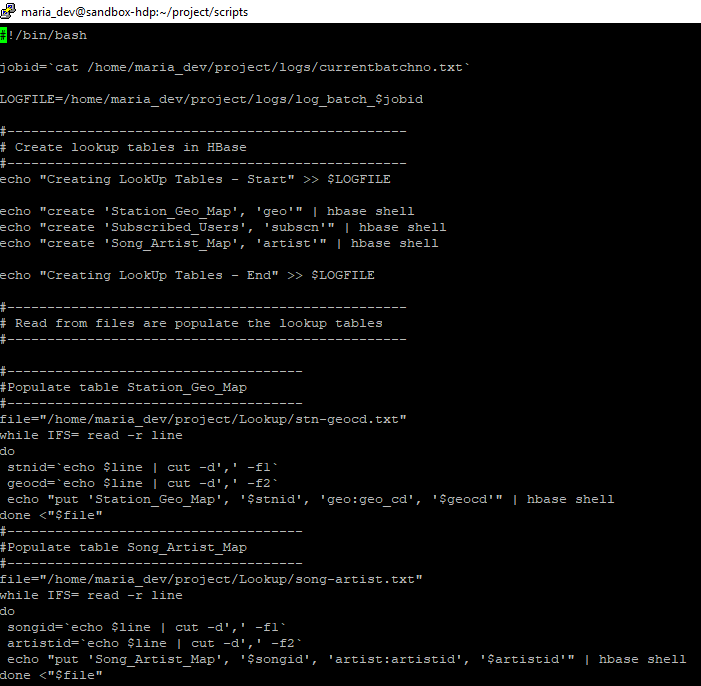
# Step 2: Setting the file for creating logs with new id’s (currentbatchno.txt)

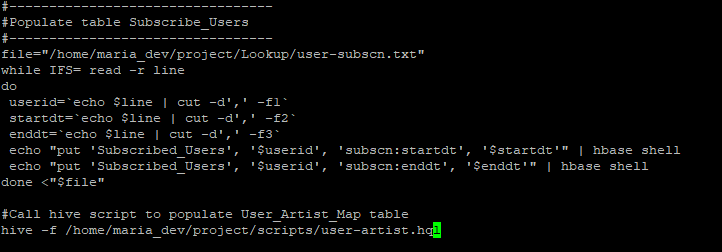
We have created a file currentbatchno.txt which holds the start id for the log file.



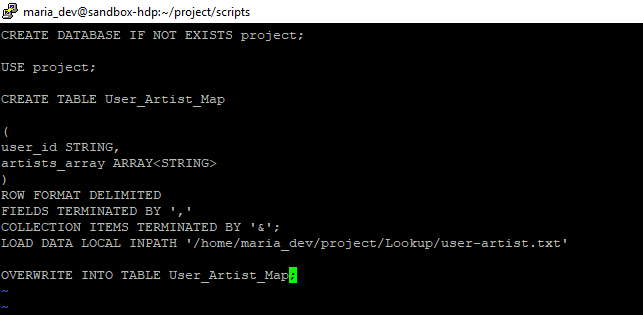
# Step 3: Load the data from the files under Lookup folder into the hbase tables. (populatelookup.sh)

populatelookup.sh will load data from lookup files to hbase tables.

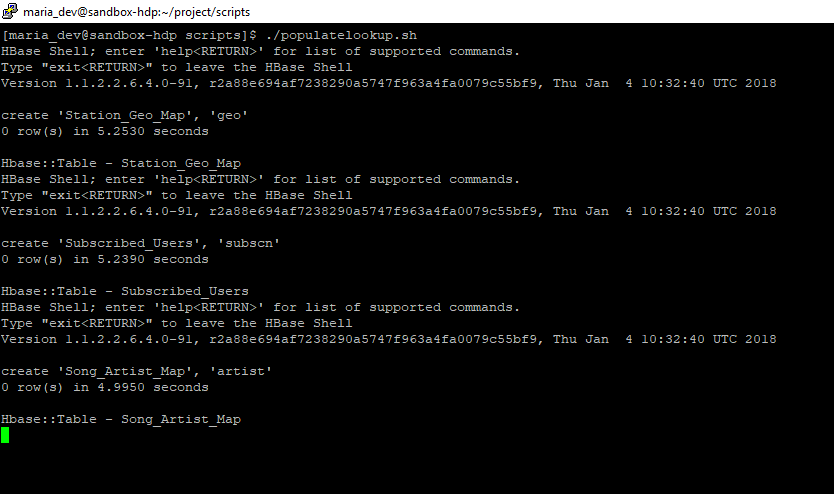


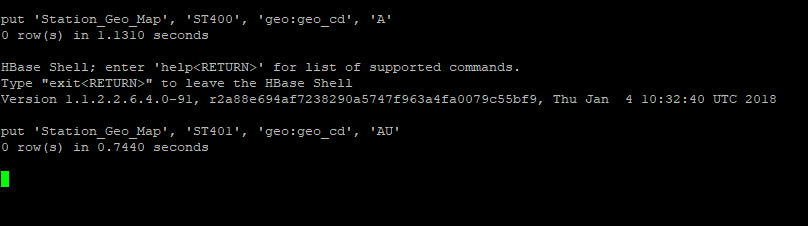


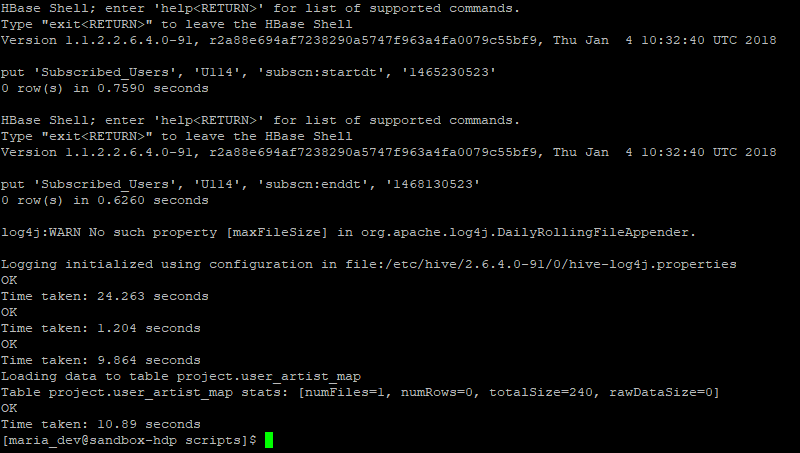
user-artist.hql will load data in User\_Artist\_Map table.



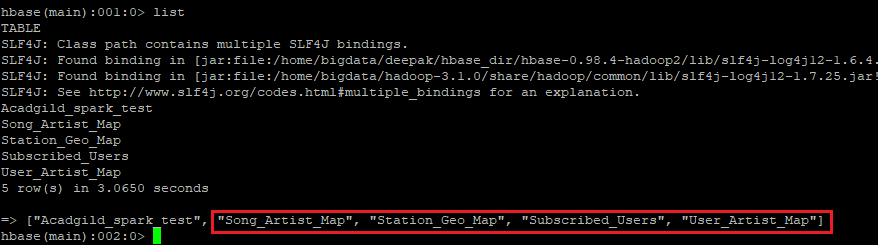
Execute script populatelookupsh



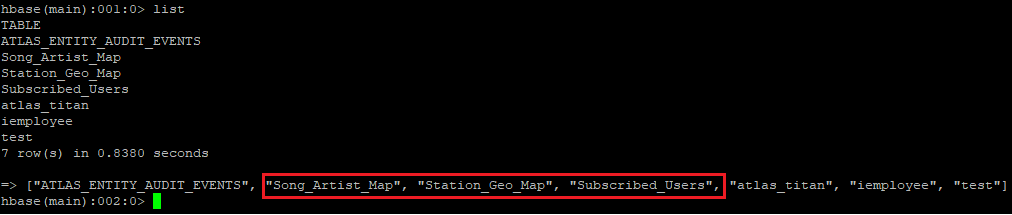




Once the script has completed successfully, verify the tables and data.



We can see that all the three tables have been created.



Verify the data.

Table – Song\_Artist\_Map

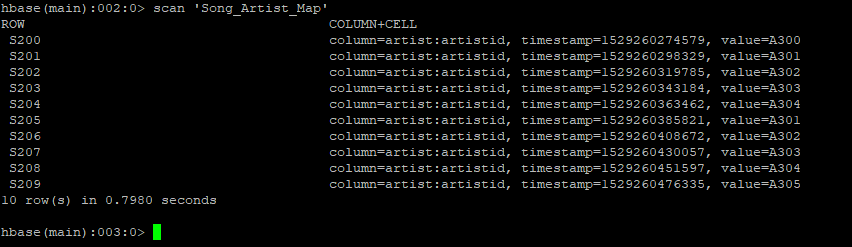


Table – Station\_Geo\_Map

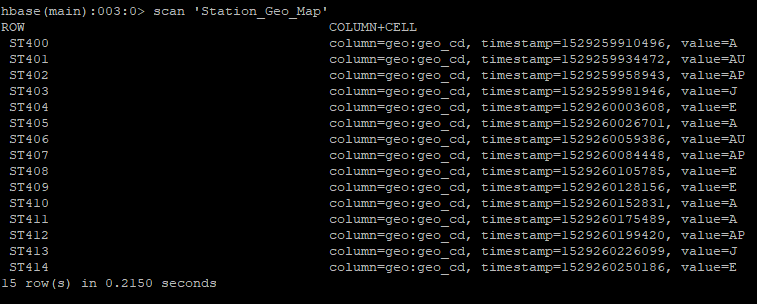


Table – Subscribed\_Users

Table – User\_Artist\_Map

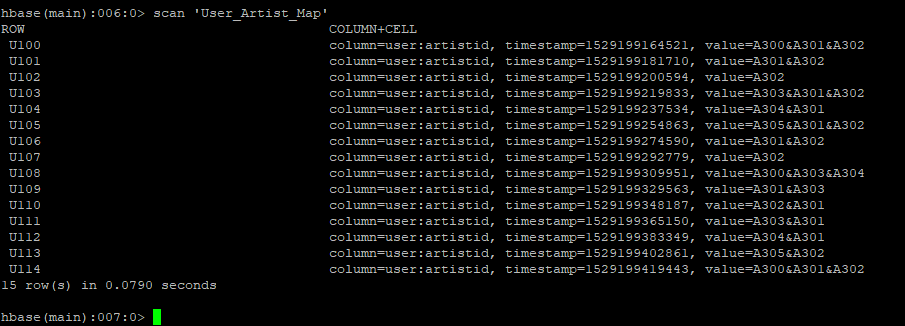
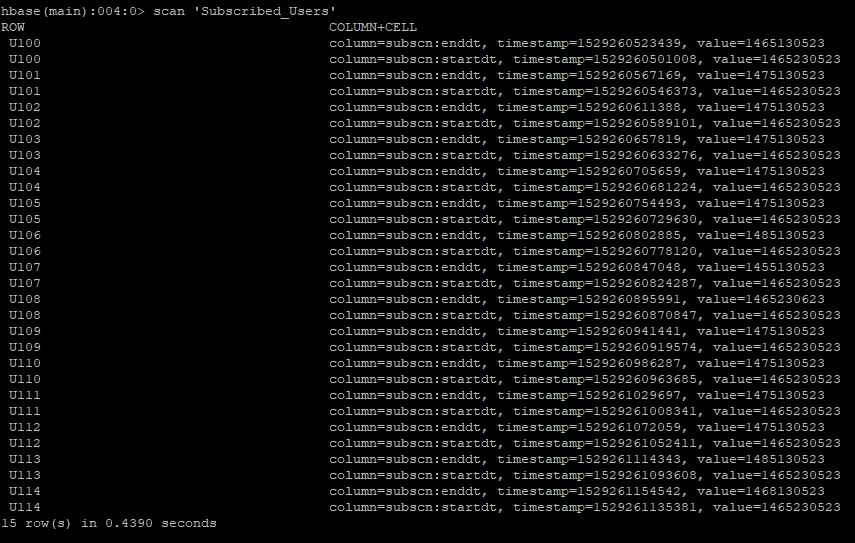
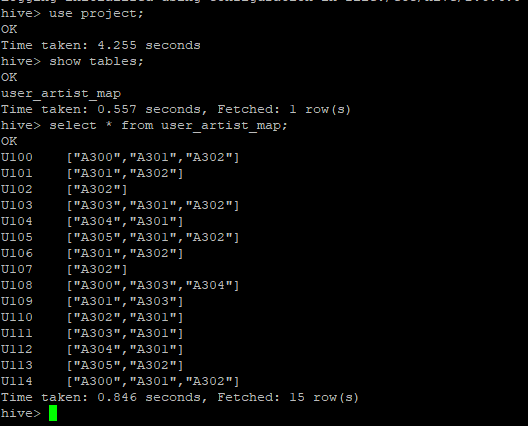


Table – Subscribed\_Users



user\_artist\_map table



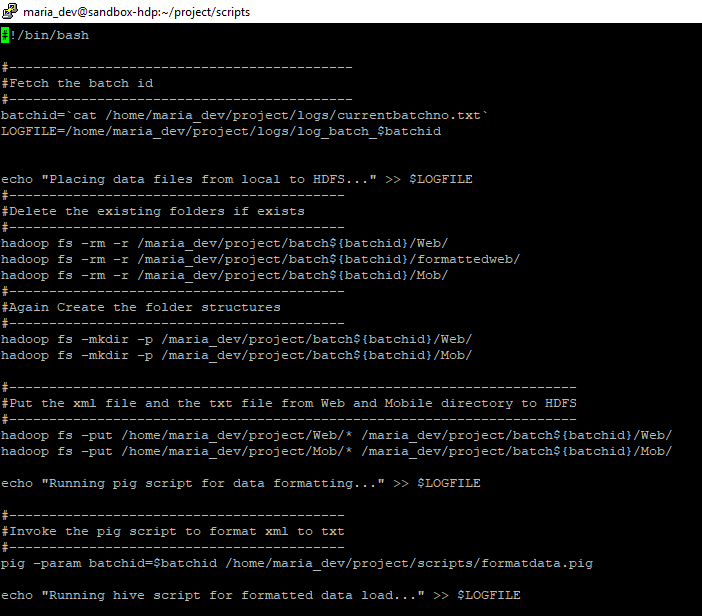
So, we can see that all the tables have been created and all the required data also have been loaded.

# Step 4 – Data Formatting (formatdata.sh)

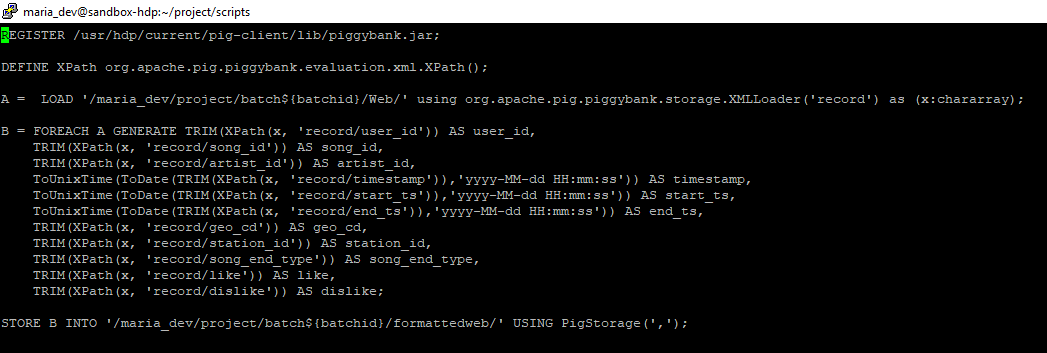
In this step we will format the file file.xml under /home/maria\_dev/project/Web to text format, which will later loaded along with /home/maria\_dev/project/Mob/file.txt for data enrichment.

We will use pig to convert the xml file to txt file.

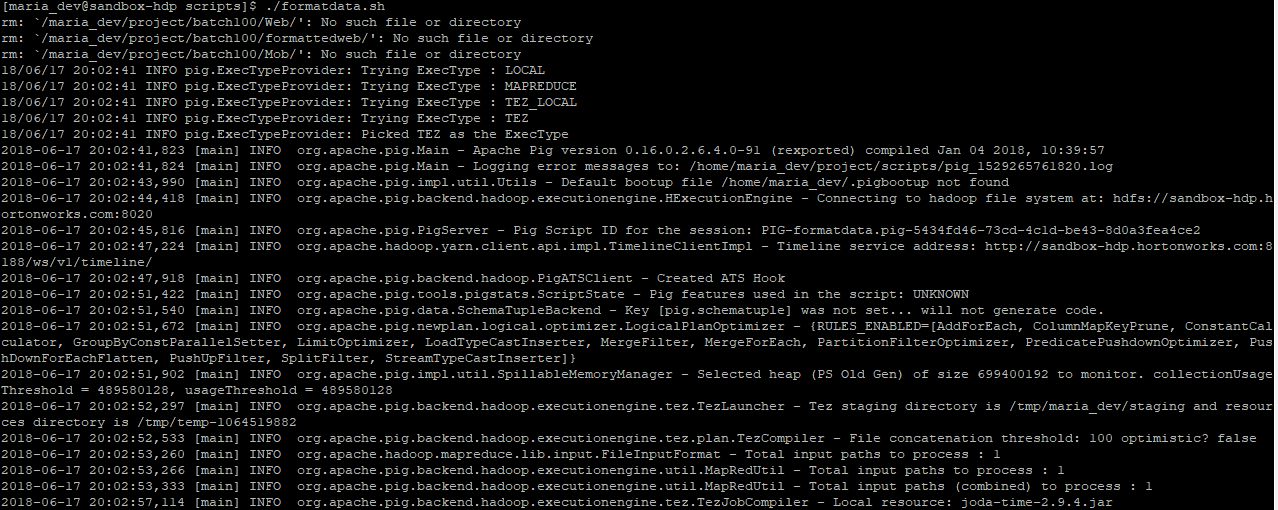
Batch script formatdata.sh to invoke the pig script formatdata.pig

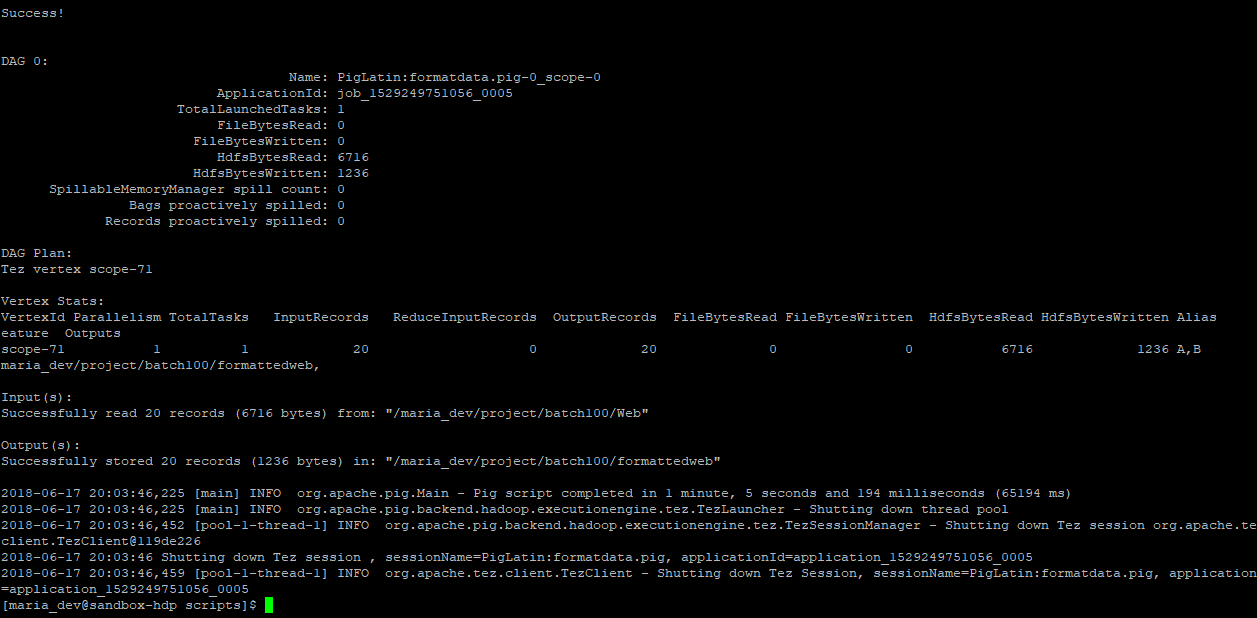


Pig Script formatdata.pig

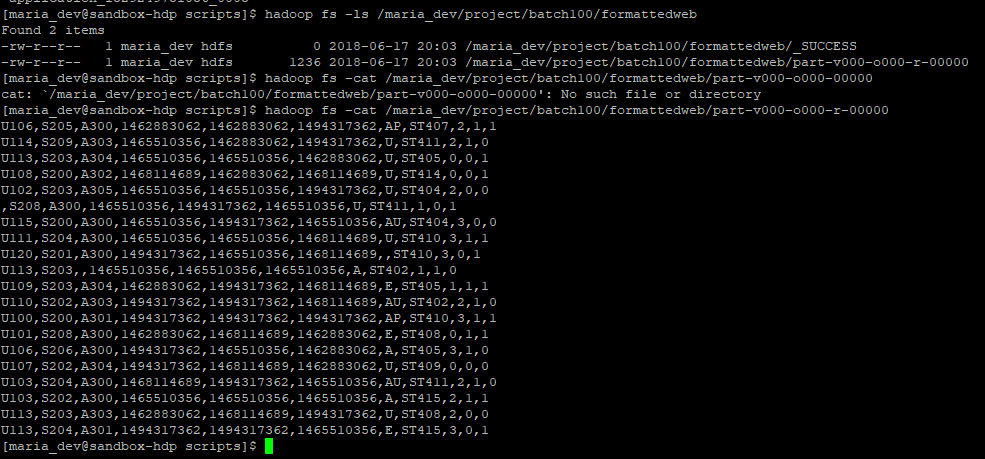


Execute the batch script formatdata.sh which invoked formatdata.pig to format the xml file.

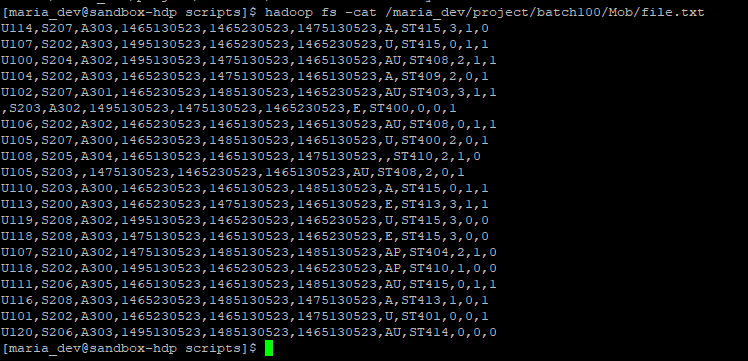




Now, we can see that the file.xml has been converted to txt file

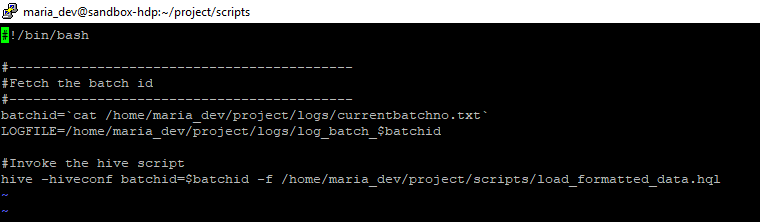


Also, we have just pasted the file.txt from local Mob directory to HDFS Mob directory

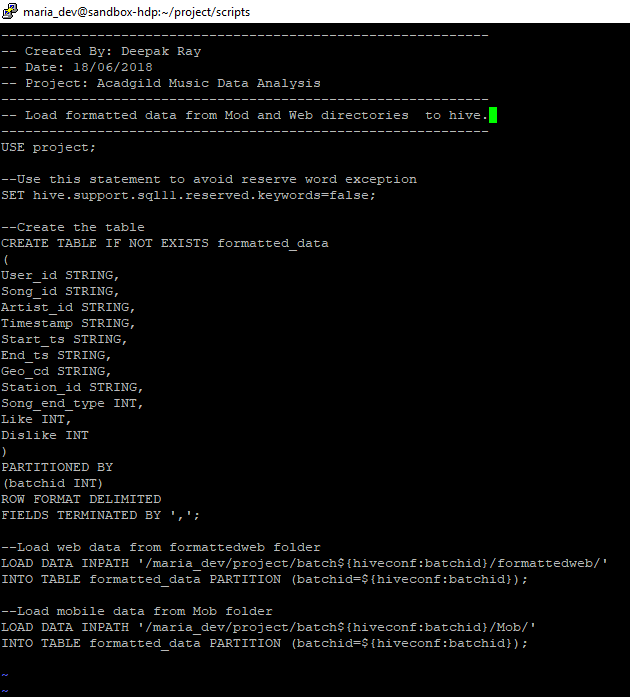


Now, we will invoke hive sql script to load these data into hive table so that data enrichment rules can be applied to this data.

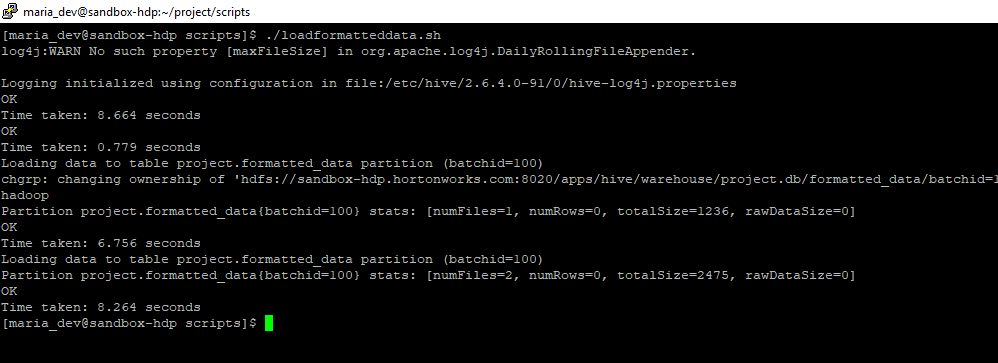
Batch script (loadformatteddata.sh) to invoke hive script.



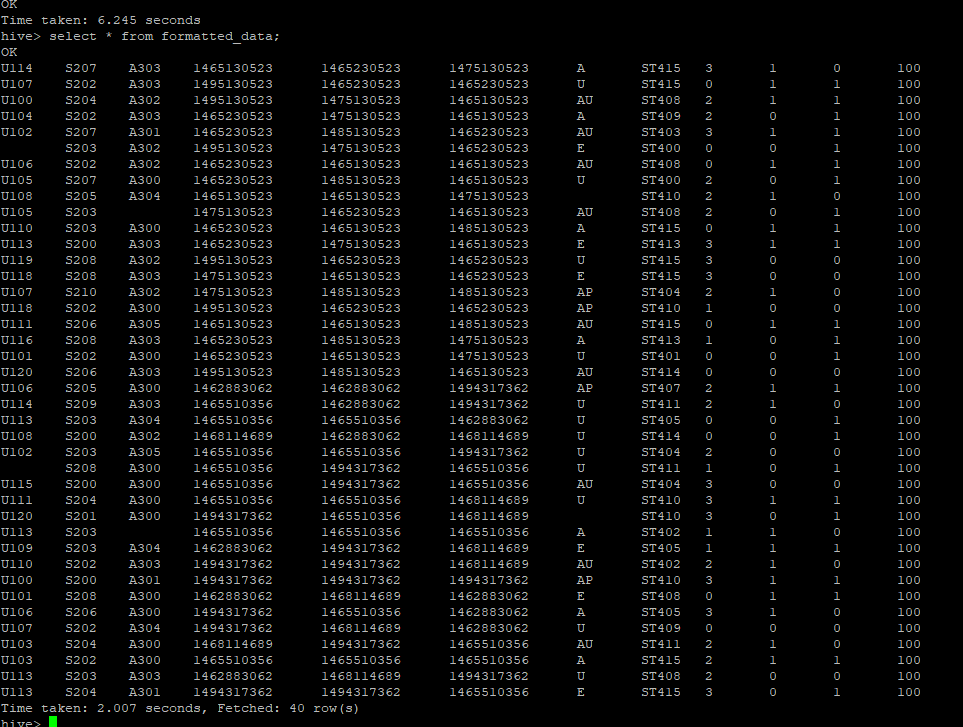
Hive Script – load\_formatted\_data.hql



Execute the script ./foadformatteddata.sh



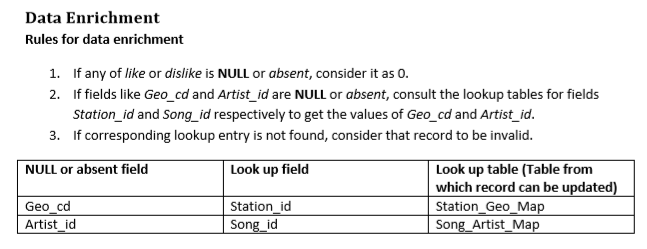
Verify the data in hive



In the next step we will apply data enrichment rules.

# Step 5 – Data Enrichment (loadformatteddata.sh)

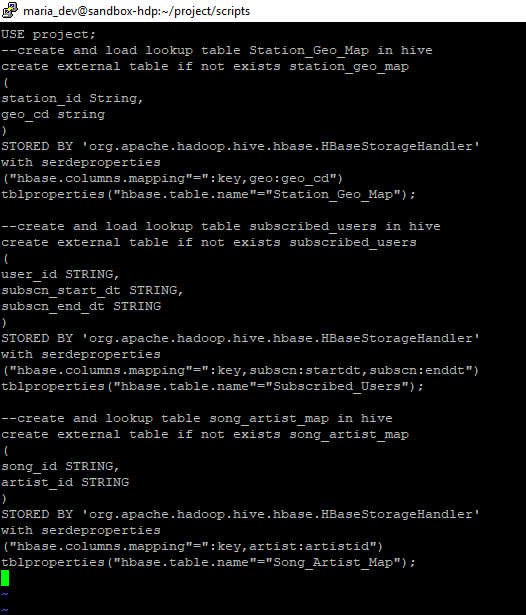
Now, once data is uploaded into the tables, we will perform the following formatting.

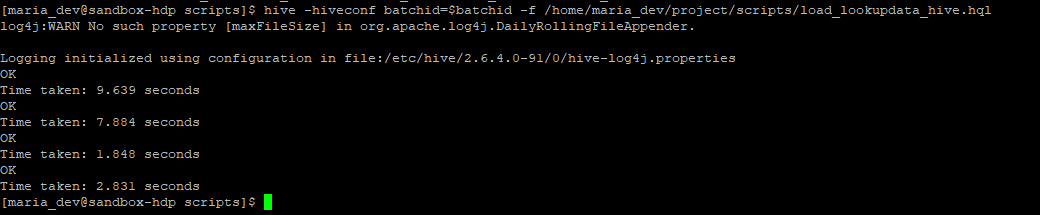


We can see in the previous step that many of the values are blank. So, in order to correct those we have to apply the above rules.

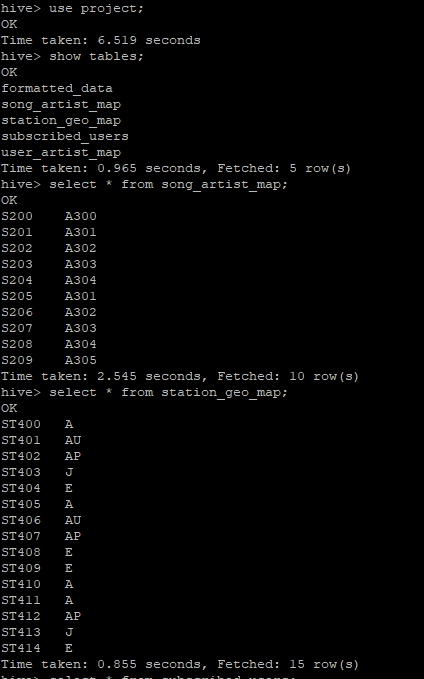
But, before that, we need to load the lookup tables from hbase to hive. So, that these tables can be used in hive to carry out data enrichment.

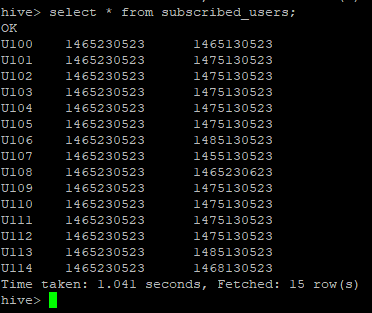
Script – load\_lookupdata.hql





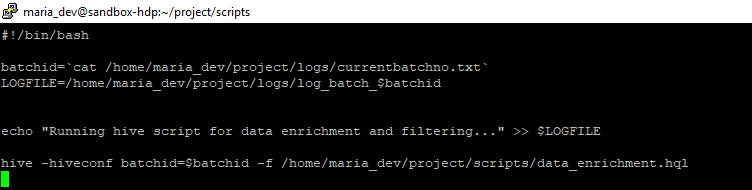
Verify data of the lookup tables in hive





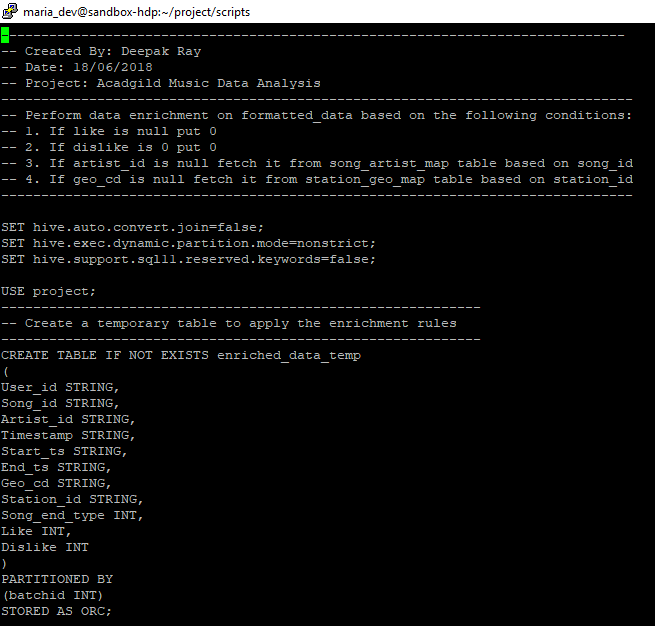
Now, we will perform the data enrichment rules.

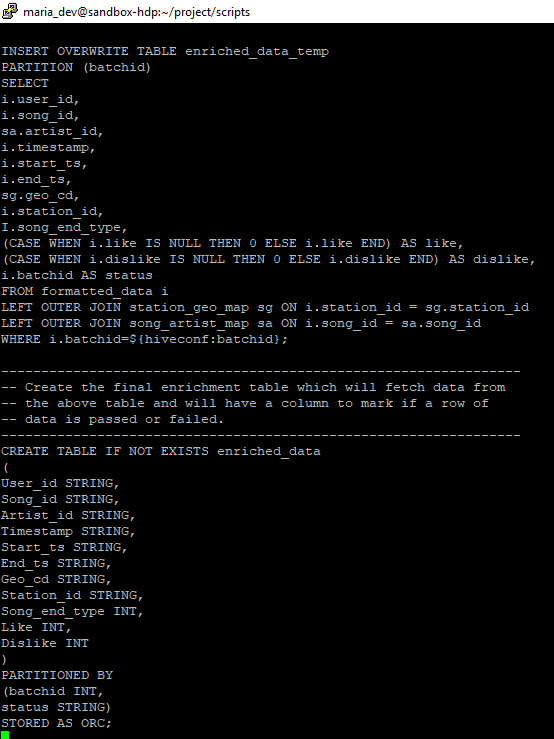
Batch script apply\_data\_enrichmemnt.sh to trigger the data\_enrichment.hql file

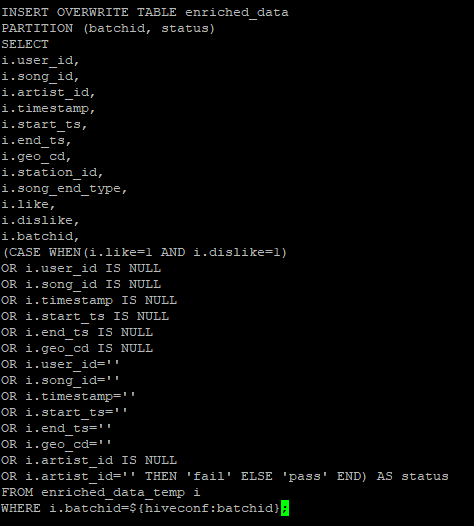


Hive Script

Here, in the data enrichment script we are first applying the data enrichment rules and storing into a table. Then, we are again inserting the data into final enrichment table and we are maintaing a column which says pass or fail for the rows where data is still not correct even after applying data enrichment.



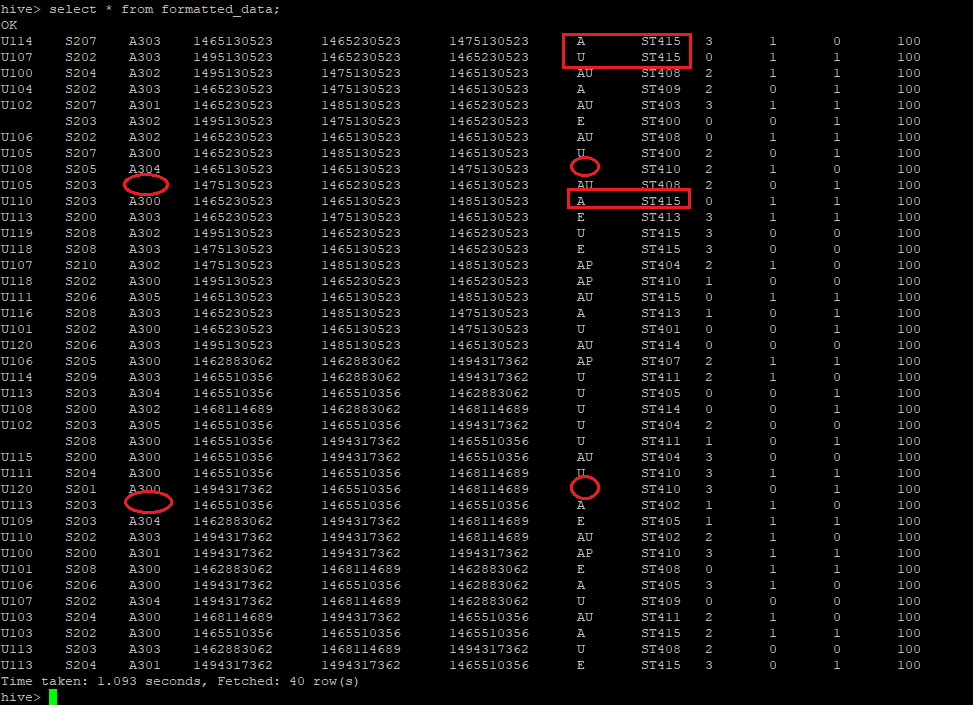




Execute the script to perform data enrichment.

Verify the data.

Before data enrichment.

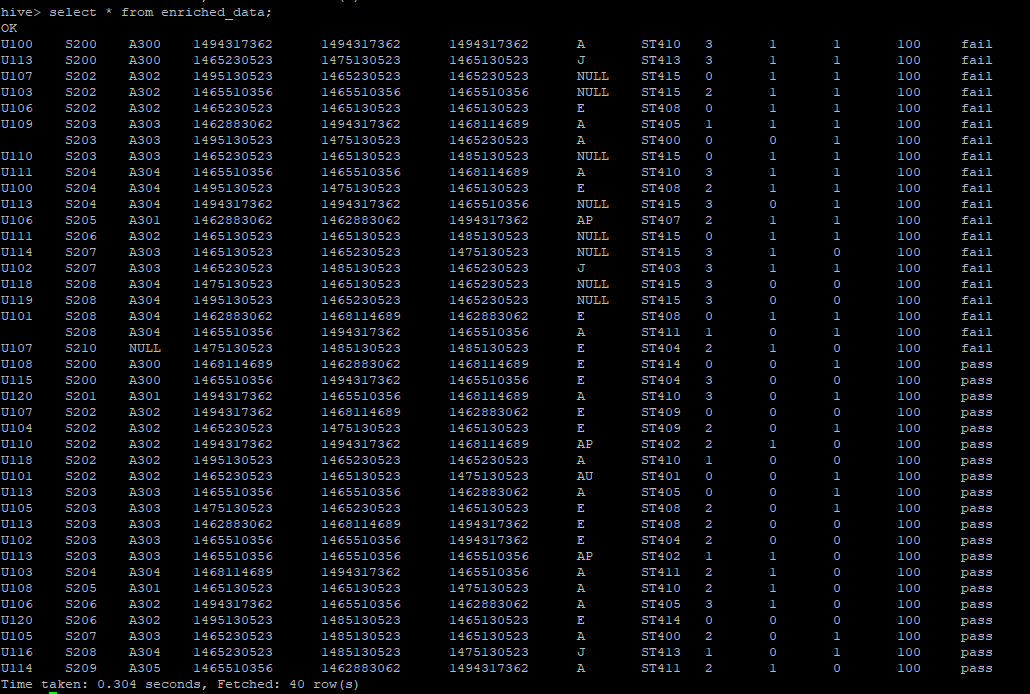


We can see that some artist\_id and geo\_cd are null. Also, the data enrichment script will correct the incorrect mappings of artist id and geo cd based on song id and station id respectively.

Also, it has corrected

After data enrichment.

Final, data enrichment table

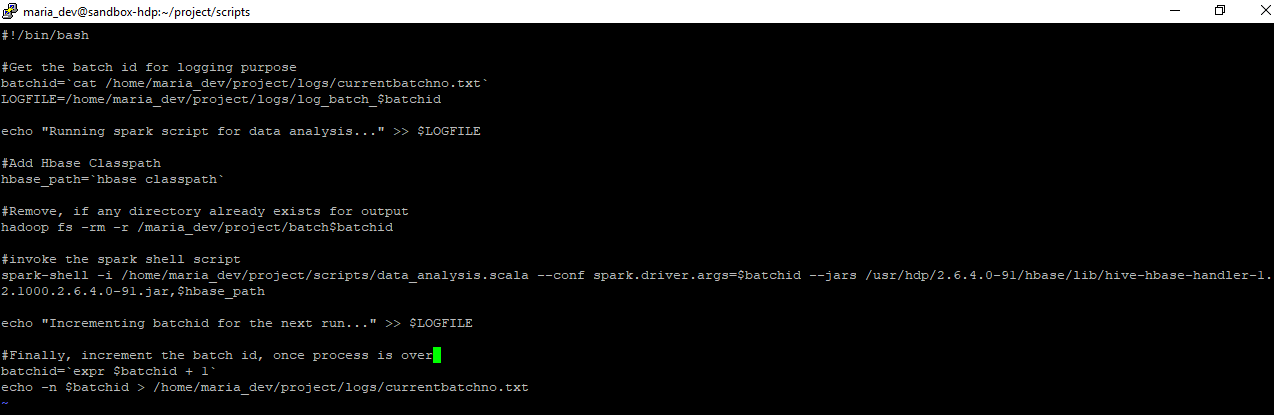


# Step 6 – Data Analysis (data\_analysis.sh)

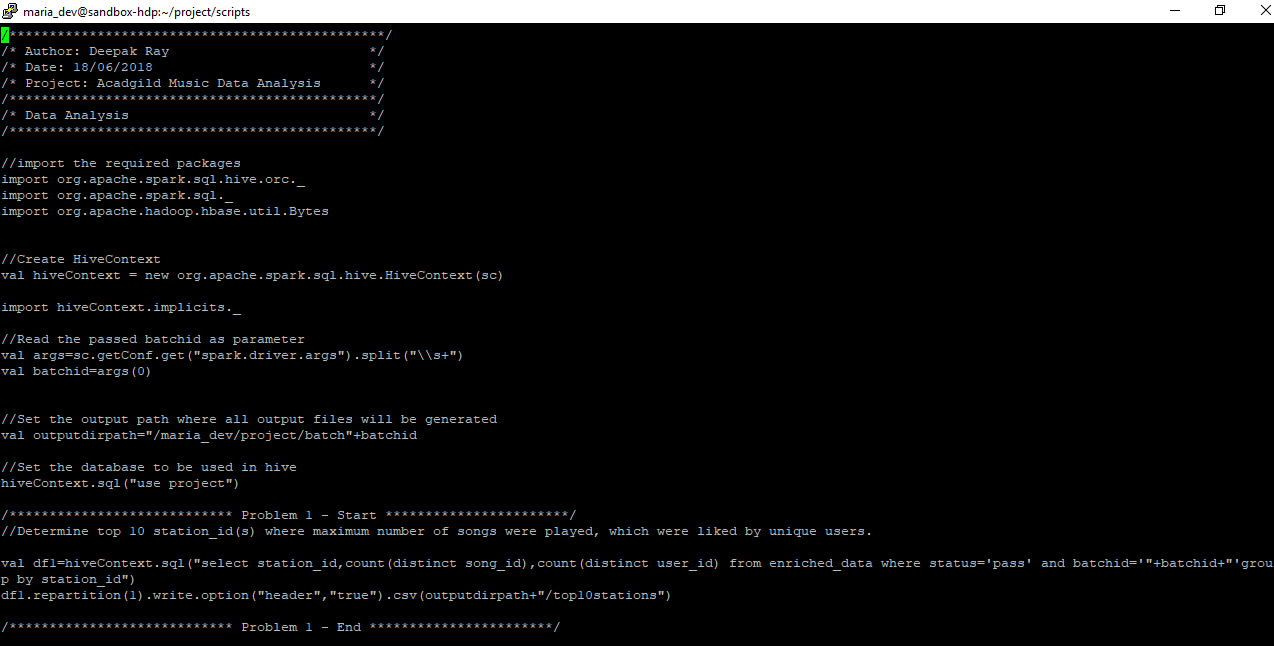
So, now we are all set to perform data analysis.

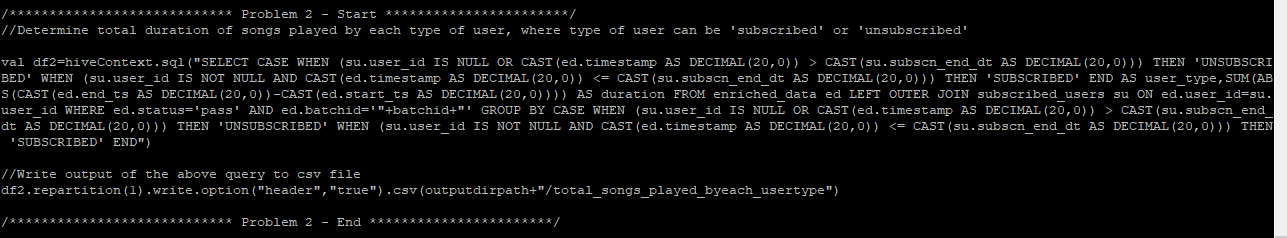
We will perform data analysis on spark.

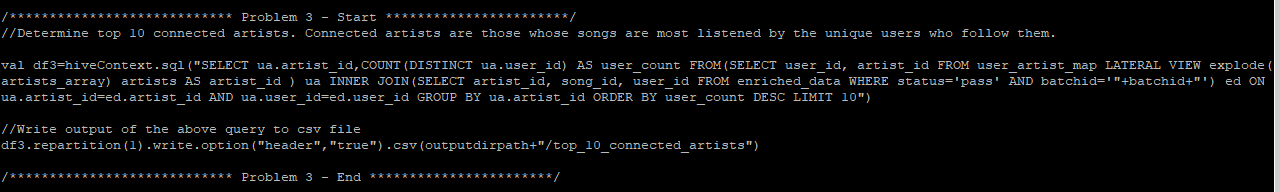
Shell script to invoke the spark script.

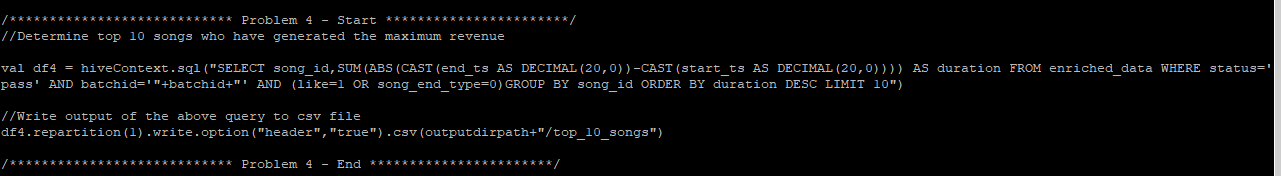


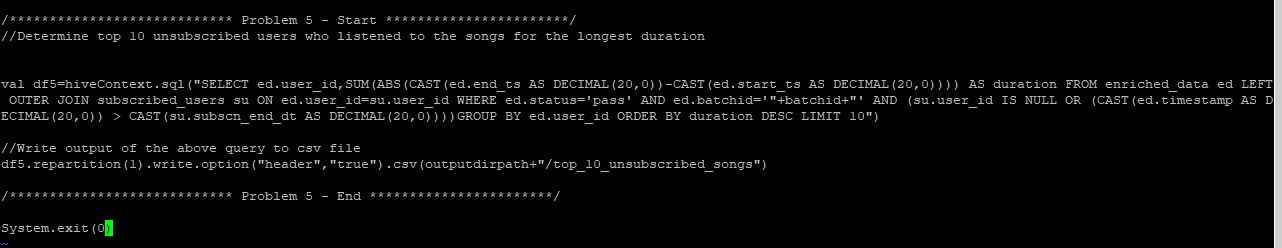
Spark Script for Data Analysis:





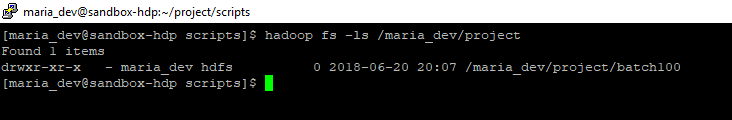




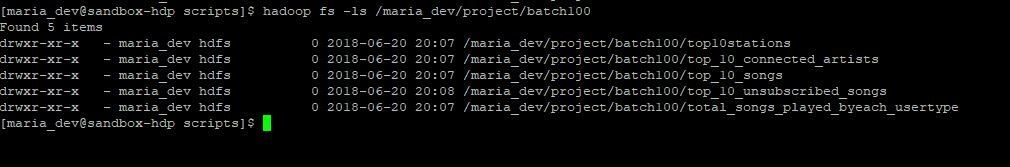


Output:

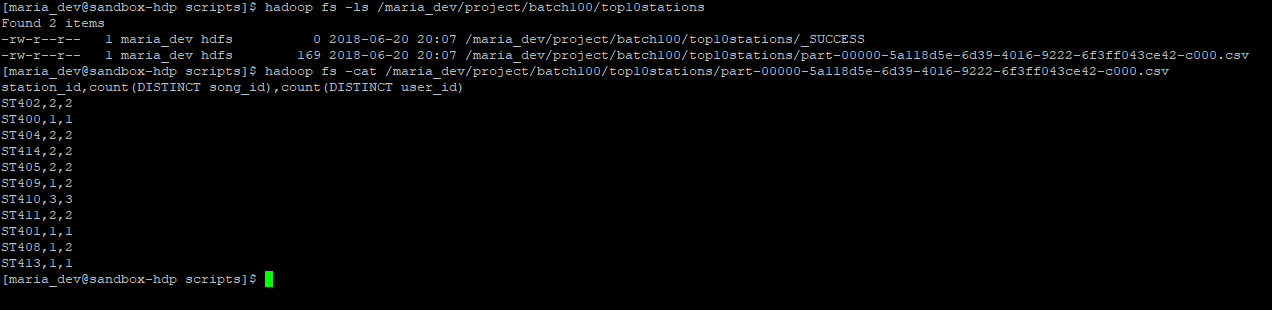
Once the script is executed, all the output will be written to /maria\_dev/project/batch100 (this will vary based on batch id)



Inside this we will have five folder, one for each analysis question.

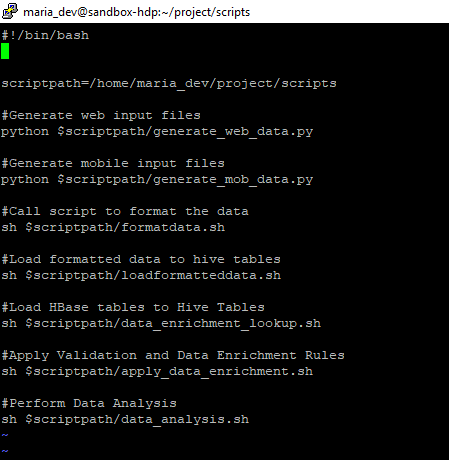


Within each of this folder, there will be csv file holding the results.

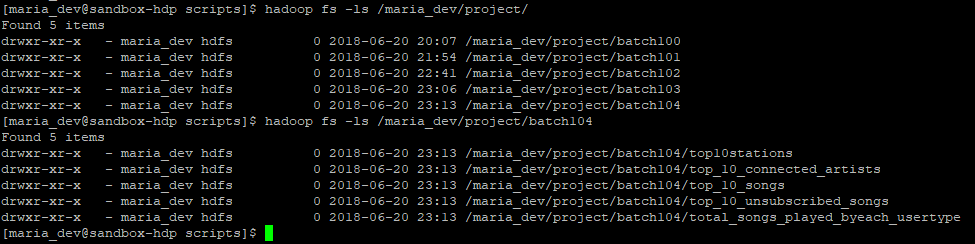


# Step 7 – Wrapping All Scripts to Once (projectexecutor.sh)

Now, we will wrap all the above scripts to one projectwrapper.sh, so that it can be scheduled.



With every run a new log file will get generated and new output folder will get generated.



# Step 8 – Scheduling the Script



