**Project 2**

**Project 2.1 - Music Data Analysis**

Student Name: Abarajithan SA

Course: Big Data Hadoop & Spark Training

Contents

[Section – 1 - Project Overview 2](#_Toc504276399)

[1.1 Fields present in the data files 2](#_Toc504276400)

[1.2 LookUp Tables 2](#_Toc504276401)

[1.3 DATASET 3](#_Toc504276402)

[1.4 Data Enrichment 4](#_Toc504276403)

[1.5 Data Analysis (SHOULD BE IMPLEMETED IN SPARK) 4](#_Toc504276404)

[1.6 Challenges and Optimizations: 4](#_Toc504276405)

[1.7 Flow of operations 5](#_Toc504276406)

[Section -2 – Design of the Project 6](#_Toc504276407)

[2.1 Low Level Design 6](#_Toc504276408)

[2.2 High Level Design 7](#_Toc504276409)

[Section-3-Hadoop Eco-System Implementation 8](#_Toc504276410)

[Section-4 –Data Ingestion, Formatting, Enrichment and Filtering 10](#_Toc504276411)

[4.1 Stage – 1 – Data Ingestion 10](#_Toc504276412)

[4.2 Stage – 2 - Data Formatting 18](#_Toc504276413)

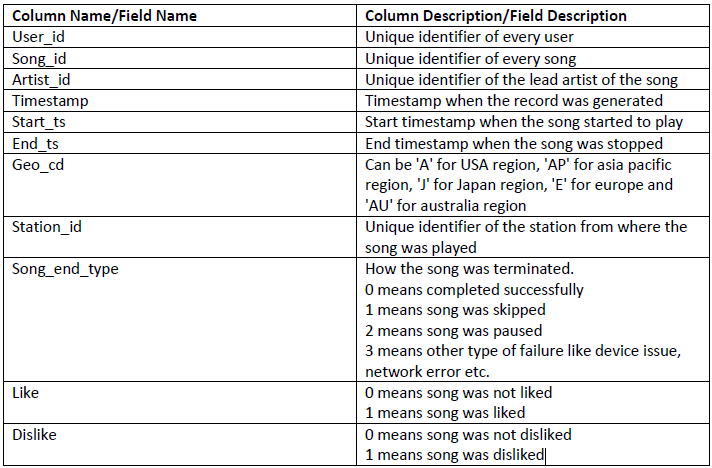
[4.3 Stage – 3 - Data Enrichment & Filtering 23](#_Toc504276414)

# Section – 1 - Project Overview

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

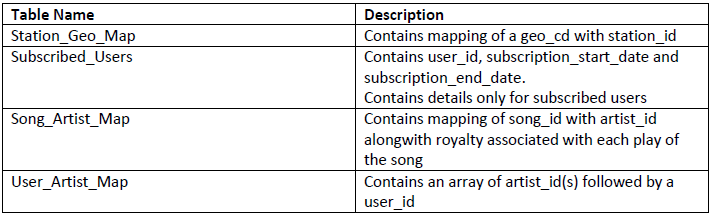
## 1.1 Fields present in the data files

Data files contain below fields.



## 1.2 LookUp Tables

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

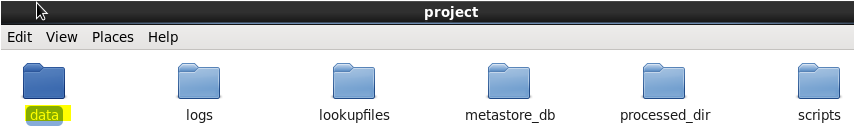


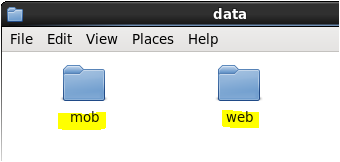
## 1.3 DATASET

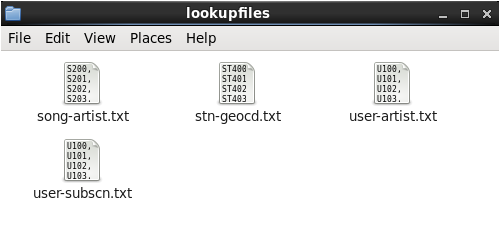
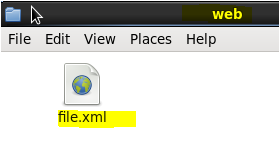
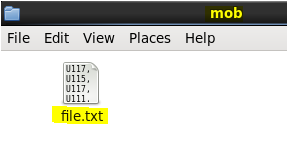
1. Data coming from web applications reside in /data/web and has xml format.
2. Data coming from mobile applications reside in /data/mob and has csv format.
3. Data present in lookup directory should be used in HBase.

Below is the link for same.

<https://drive.google.com/drive/folders/0B_P3pWagdIrrMjJGVlNsSUEtbG8?usp=sharing>







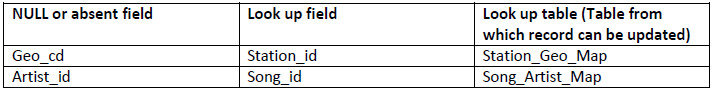
## 1.4 Data Enrichment

Rules for data enrichment,

1. If any of like or dislike is NULL or absent, consider it as 0.
2. If fields like **Geo\_cd** and **Artist\_id** are NULL or absent, consult the lookup tables for fields

**Station\_id** and **Song\_id** respectively to get the values of **Geo\_cd** and **Artist\_id**.

1. If corresponding lookup entry is not found, consider that record to be invalid.



## 1.5 Data Analysis (SHOULD BE IMPLEMETED IN SPARK)

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

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

## 1.6 Challenges and Optimizations:

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

## 1.7 Flow of operations

A schematic flow of operations is shown below,

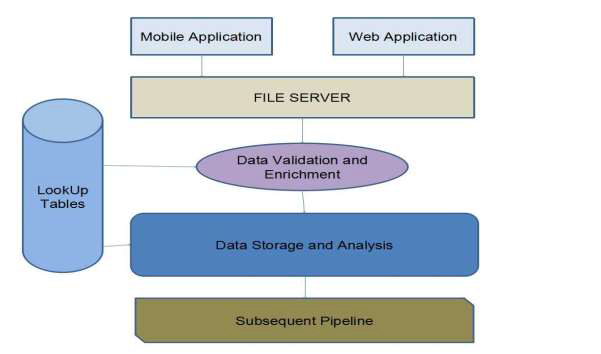


Fig-1

In the following sections, we are going to see the Music Data Analysis as per the above rules.

# Section -2 – Design of the Project

## 2.1 Low Level Design

The following flowchart shows the Low Level design of this project,

**Low Level Design**

**Stage-1**

**Data Ingestion**

**Stage-2**

**estion**

**Stage-2**

**Data Formatting**

**Stage-3**

**Data Enrichment and Filtering**

**Stage-5**

**Data Storage**

**Stage-4**

**Data Analysis**

Fig-2

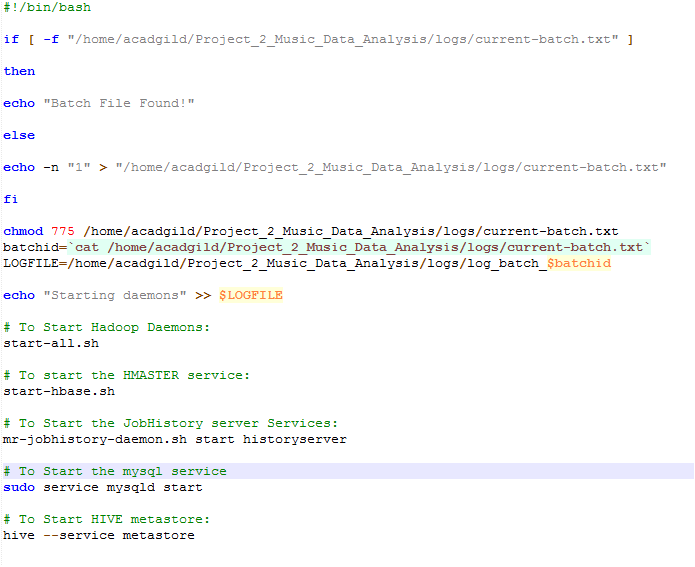
## 2.2 High Level Design

Fig-3

# Section-3-Hadoop Eco-System Implementation

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

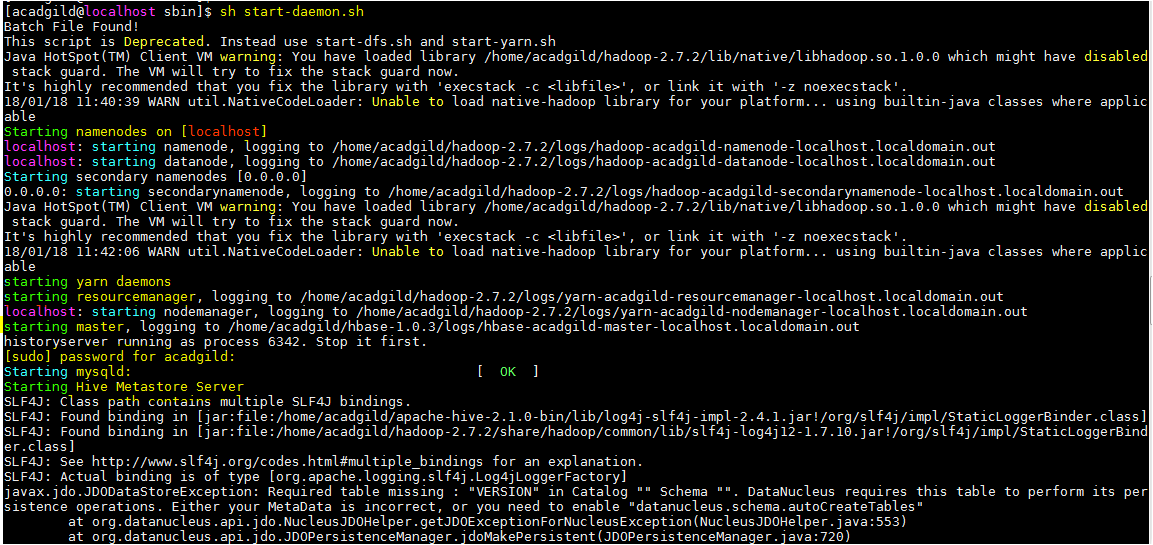
Batch file script,



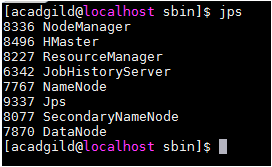
1. Starting all daemons,

* ***sh start-daemon.sh***

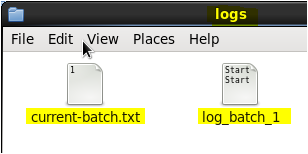
As per the batch file script all the hadoop daemons and the Hive, MySql and Hive daemons are started shown in the below screen shot,



1. We can see the list active services using the ***jps*** command, see below screen shot and also Starting the hive metastore created a metastore\_db in the location where we desired,



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



# Section-4 –Data Ingestion, Formatting, Enrichment and Filtering

## 4.1 Stage – 1 – Data Ingestion

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

* Data formatting,
* Data enrichment and
* Analysis stage

**Lookup Tables**

|  |  |  |  |
| --- | --- | --- | --- |
| Sl.no | Table Name | Description | Related File |
| 1 | station-geo-map | Contains mapping of a **geo\_cd** with **station\_id** | stn-geocd.txt |
| 2 | subscribed-users | Contains **user\_id**, **subscription\_start\_date** and  **subscription\_end\_date.**  Contains details only for subscribed users | user-subscn.txt |
| 3 | song-artist-map | Contains mapping of **song\_id** with **artist\_id**  Along with royalty associated with each play of  the song | song-artist.txt |
| 4 | user-artist-map | Contains an array of **artist\_id(**s) followed by a  **user\_id** | user-artist.txt |

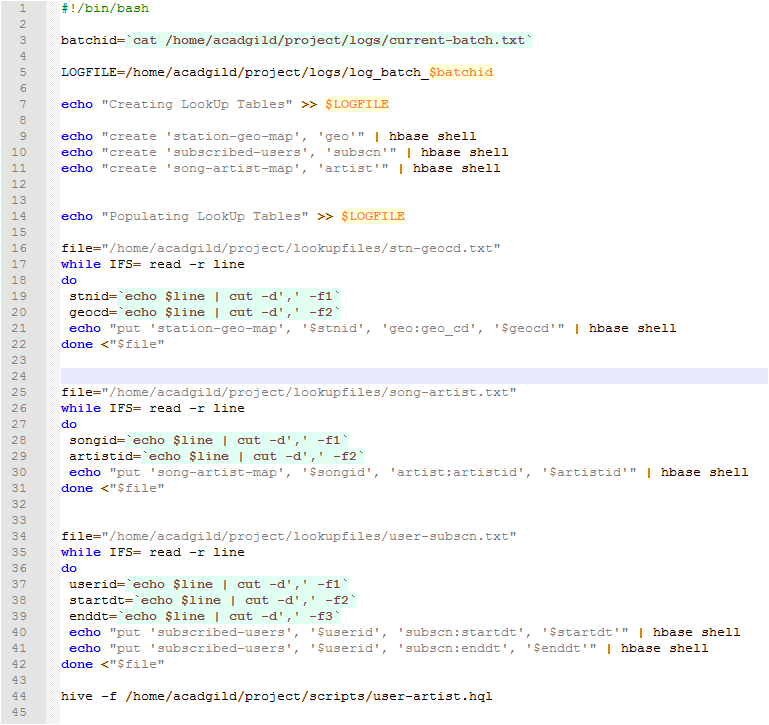
Table-1

**“populate-lookup.sh” script**

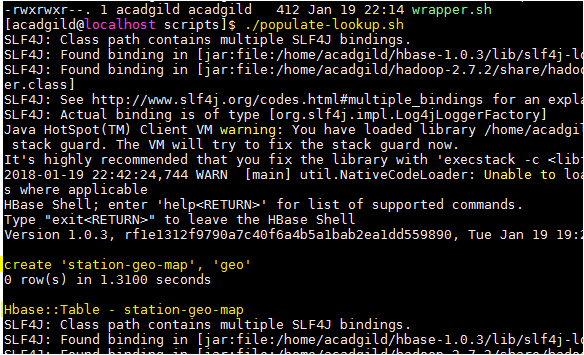
The “***populate-lookup.sh”*** shell script creates the above 4 lookup tables in the Hbase and populate the data into the lookup tables from the dataset files.

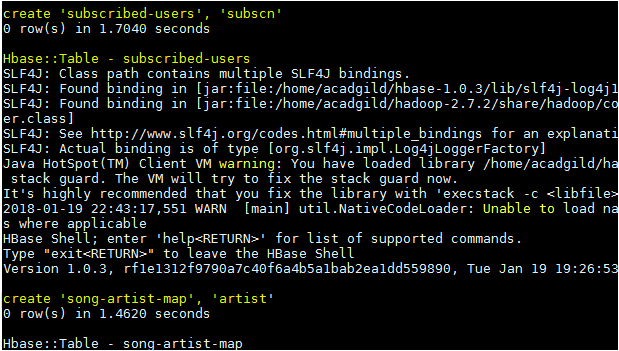
In the below screen shots, we can see the create-lookup.sh scripts and the following screen shots shows the tables creation and population of the data in the Hbase. Also, the values loaded into the Hbase Tables are also shown, please see the below screen shots.

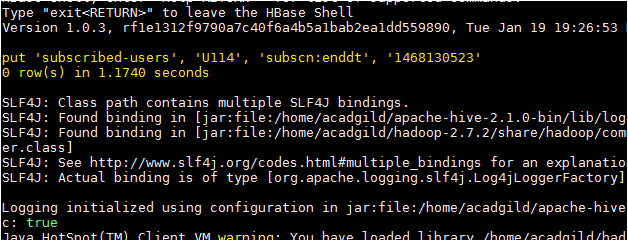
populate-lookup.sh



Run the script: ./populate-lookup.sh

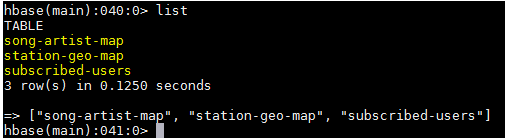






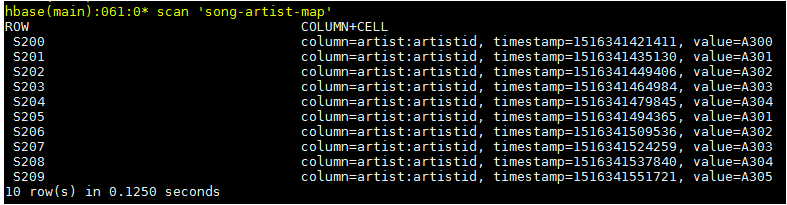
We can see the lookup tables created using the ***“populate-lookup.sh”*** in the below screen shot,

Lookup Tables,

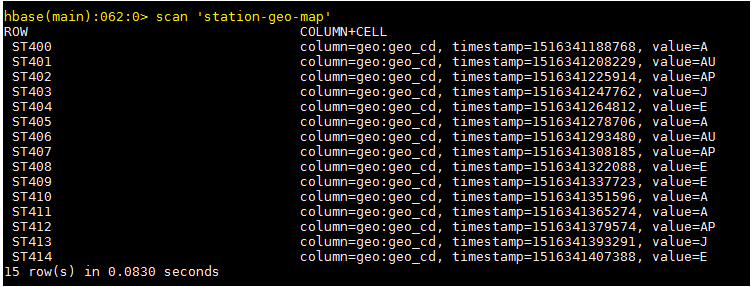


The values loaded in the Lookup tables are shown below,

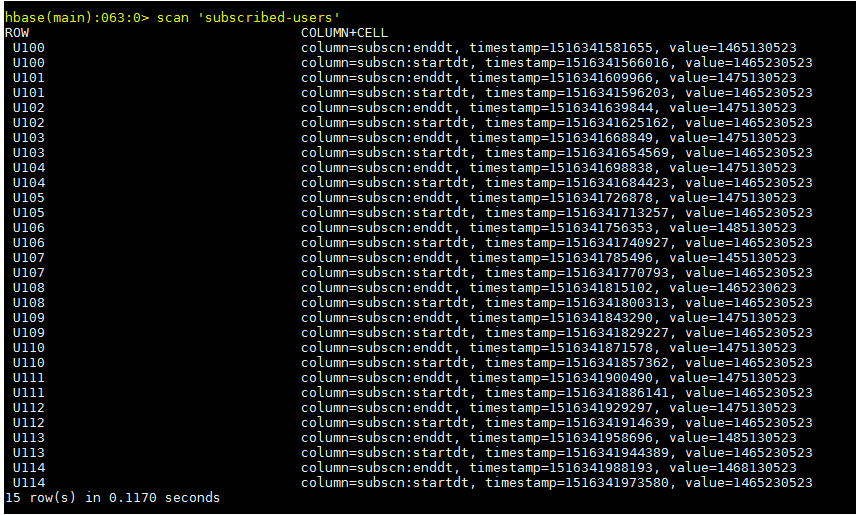
song-artist-map



station-geo-map

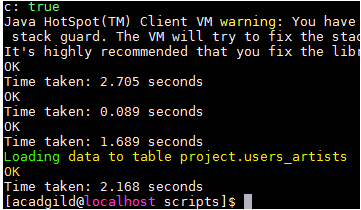


subscribed-users

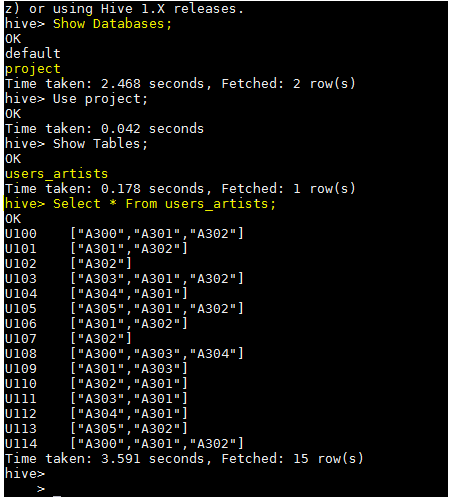


We have successfully created the lookup tables in the Hbase.

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



***hive> Select \* From users\_artists;***



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

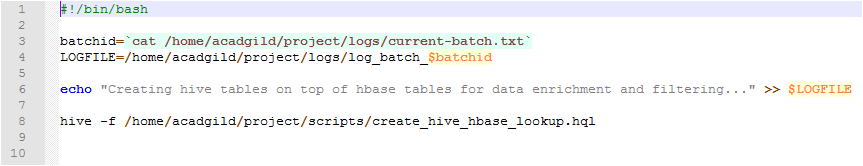
With the help of **“data\_enrichment\_filtering\_schema.sh”** file we will create hive tables on the top of Hbase tables using **“create\_hive\_hbase\_lookup.hql”.**

**Creating Hive Tables on the top of Hbase:**

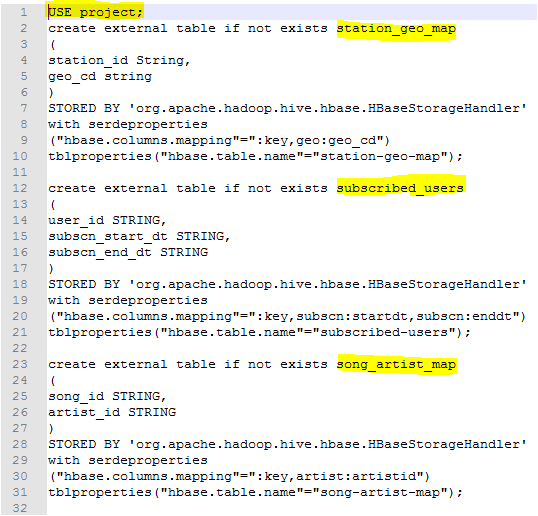
In this section with the help of Hbase storage handler & SerDe properties we are creating the hive external tables by matching the columns of Hbase tables to hive tables.

Run the script: **./data\_enrichment\_filtering\_schema.sh,**

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

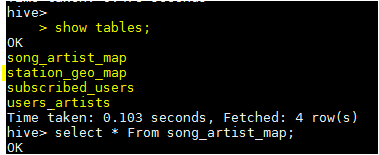


**create\_hive\_hbase\_lookup.hql**

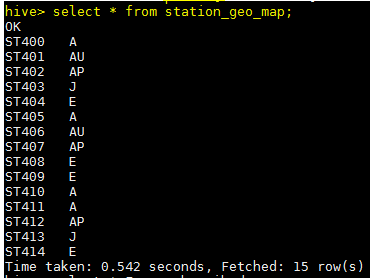


The below screenshot we can see tables getting created in hive by running the **“data\_enrichement\_filtering\_schema.sh file”**

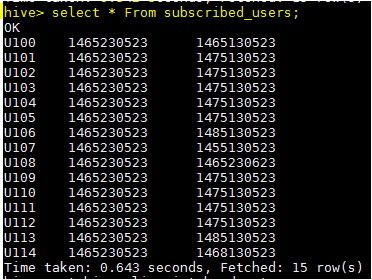
***Hive>Show Tables;***



**song\_artist\_map station\_geo\_map**

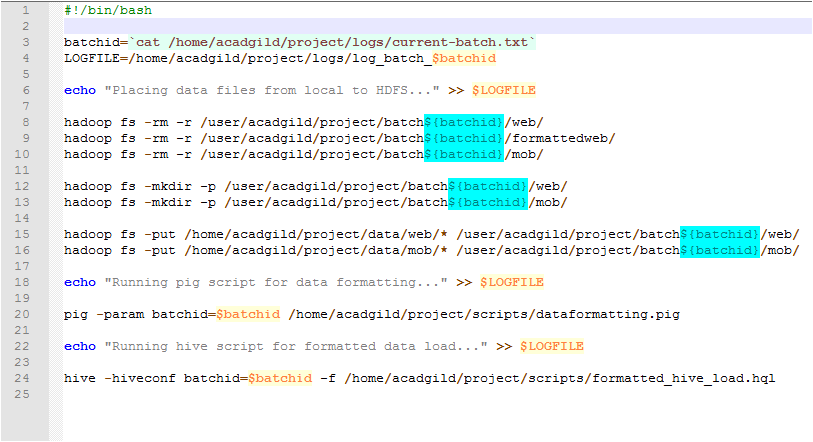
**Subscribed\_users**

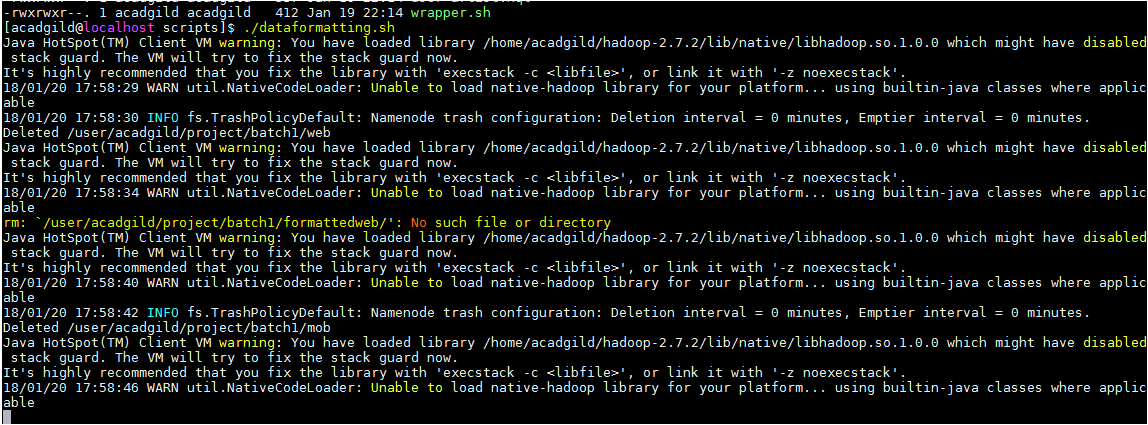


## 4.2 Stage – 2 - Data Formatting

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

Run the script: ./dataformatting.sh



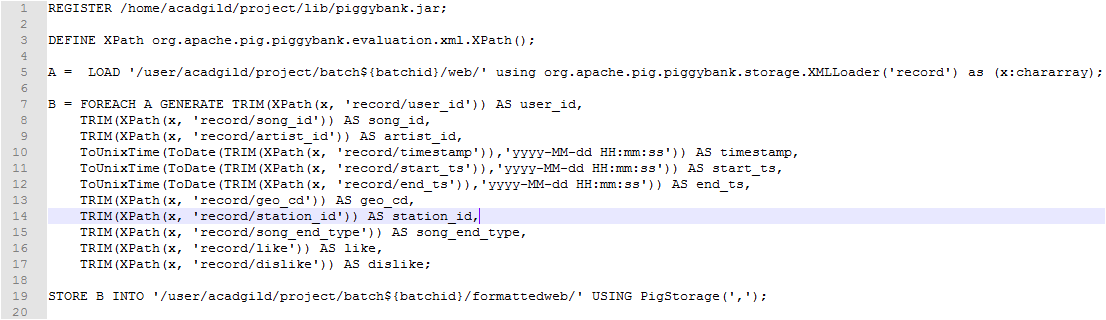


We are running two scripts to format the data. They are:

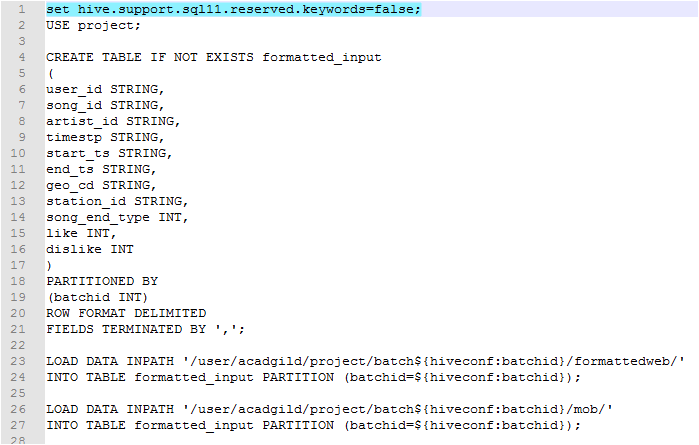
* **Dataformatting.pig**
* **Formatted\_hive\_load.hql**

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

Dataformatting.pig

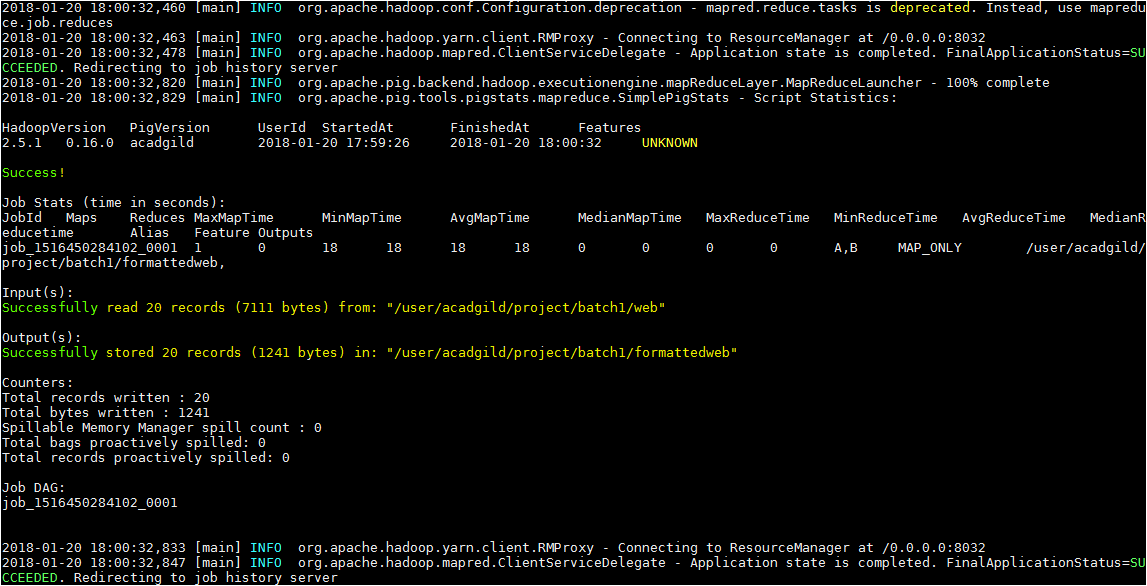


formatted\_hive\_load.hql

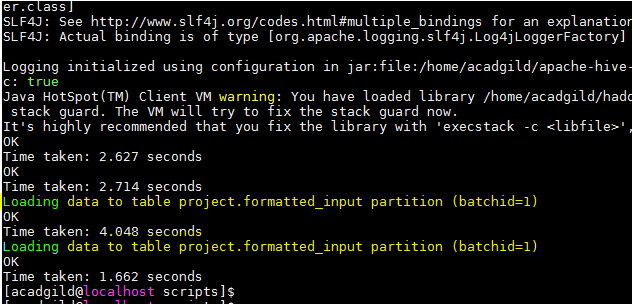


In the below screenshot we can see the data both the scripts in action, first pig script will parse the data and then hive script will load the data into hive terminal successfully.

Pig script successful completion,

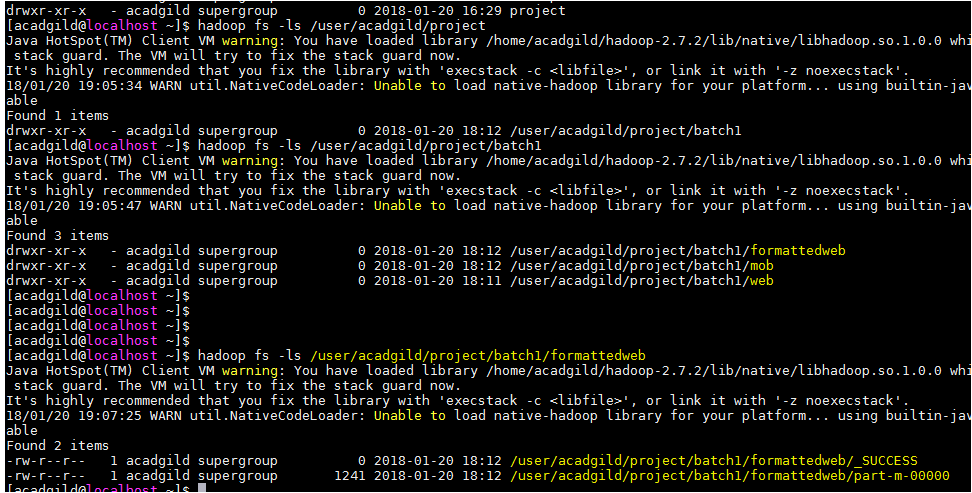


hive script successfully load the data into hive terminal,



In the above screenshot we can see the **dataformatting.pig** along with the **formatted\_hive\_load.hql** executed successfully.

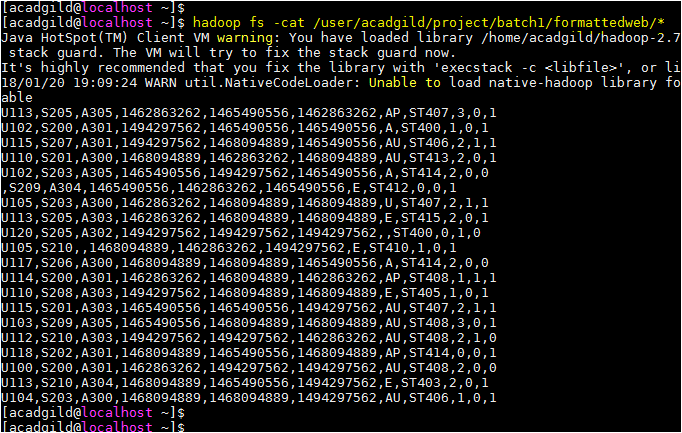
The output of **dataformatting.sh** script in HDFS folders:



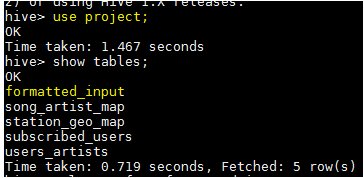
The output of the formattedweb data obtained from the Dataformatting.pig is shown in the below screen shot,

Command,

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

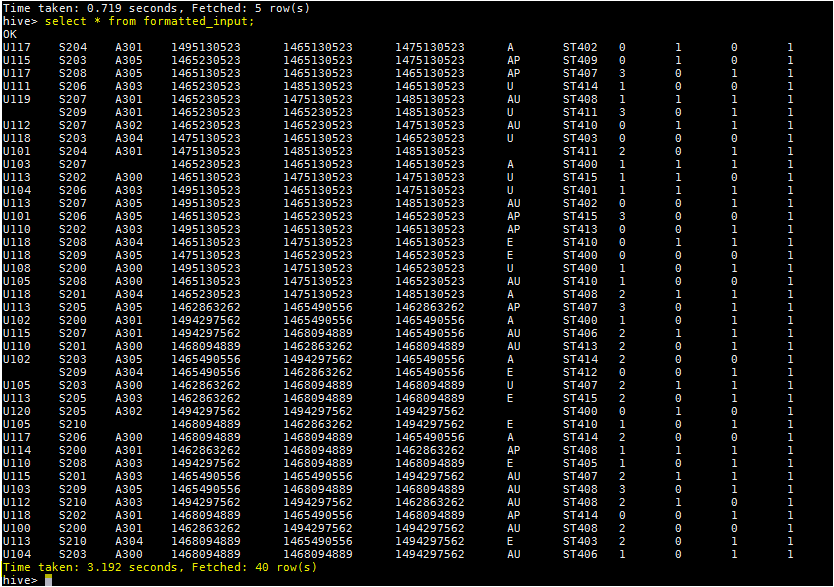


The new Tables has been created and show below,



DataFormatting.sh output in hive terminal,

***hive> select \* from formatted\_input;***



* In the above screenshot we can see the formatted input data with some null values in **user\_id,** **aritist\_id** and **geo\_cd** columns which we will fill the enrichment script based on rules of enrichment for **artist\_id** and **geo\_cd** only. We will get neglect **user\_id** because they didn’t mentioned anything about **user\_id** for enrichment purpose.
* Data formatting phase is executed successfully by loading both **mobile** and **web** data and partitioned based on **batchid**.

## 4.3 Stage – 3 - Data Enrichment & Filtering

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

**Rules for data enrichment,**

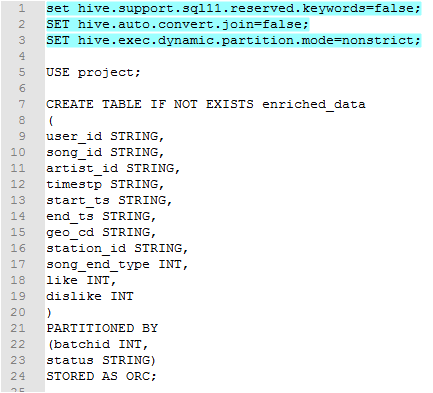
1. If any of like or dislike is **NULL** or **absent**, consider it as **0**.
2. If fields like **Geo\_cd** and **Artist\_id** are NULL or absent, consult the lookup tables for fields **Station\_id** and **Song\_id** respectively to get the values of **Geo\_cd** and **Artist\_id.**
3. If corresponding lookup entry is not found, consider that **record** to be **invalid**

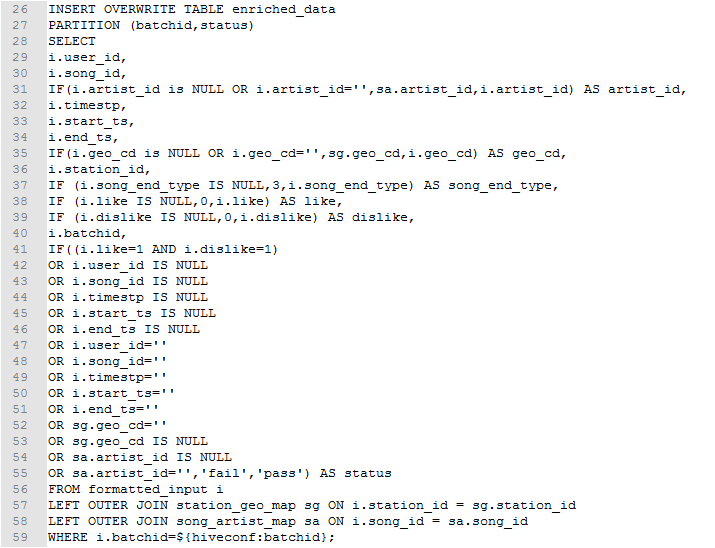
So based on the enrichment rules we will fill the null **geo\_cd** and **artist\_id** values with the help of corresponding lookup values in **song-artist-map** and **station-geo-map** tables in **Hive-Hbase** tables.

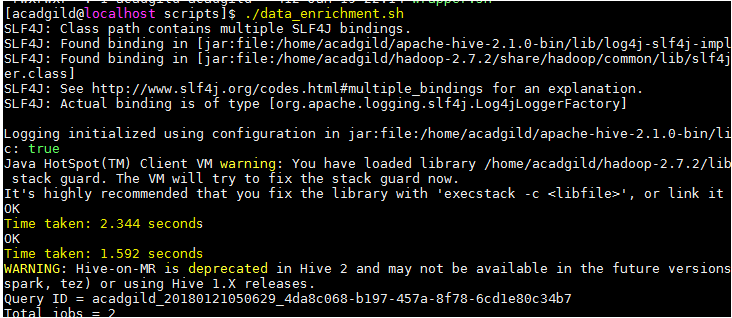
**data\_enrichment.sh**

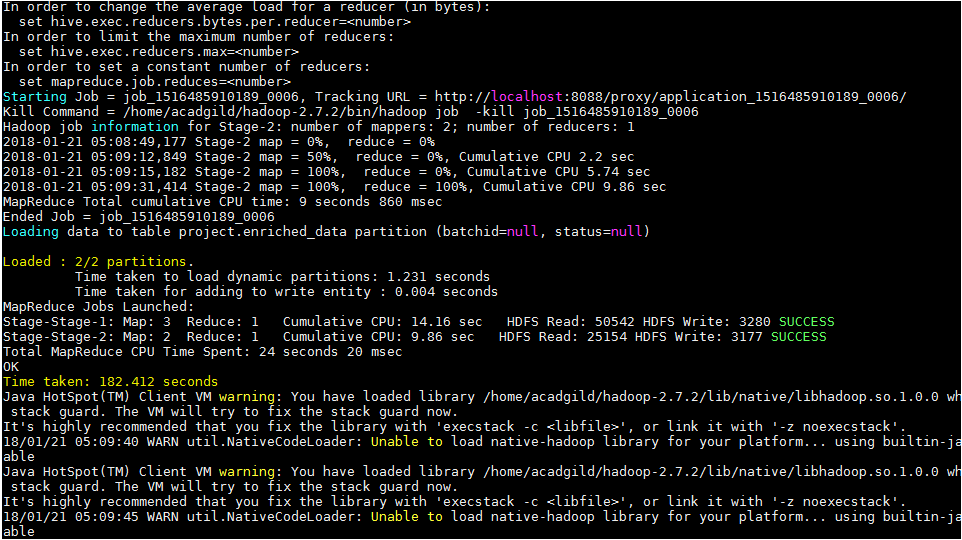


**data\_enrichment.hql**

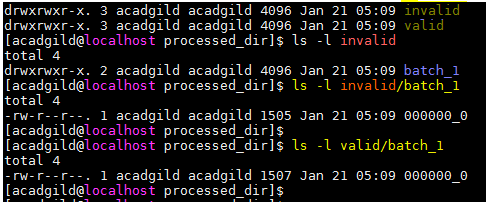


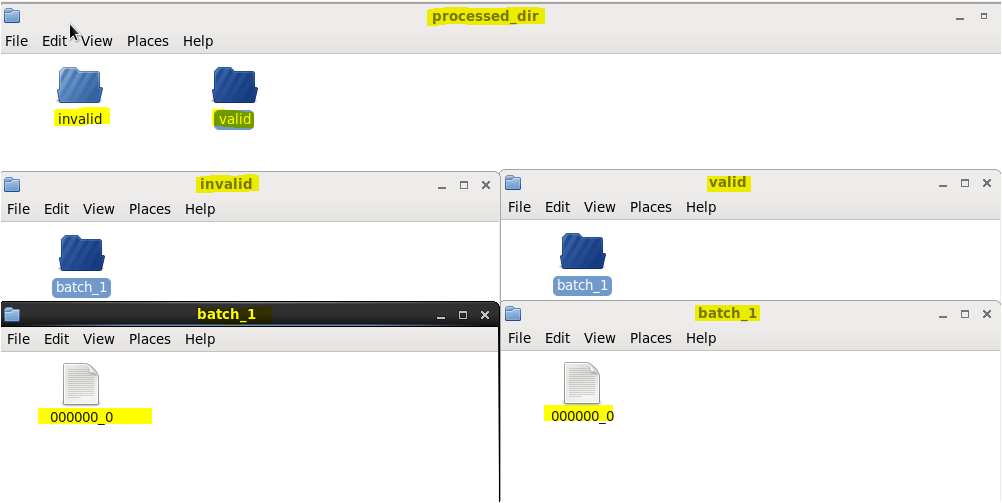




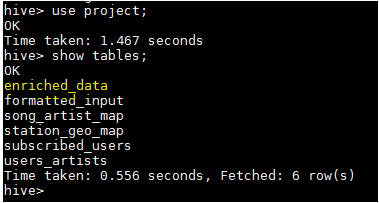


At the end script will automatically divide the records based on status **pass** & **fail** and dump the result into **processed\_dir** folder with valid and invalid folders.



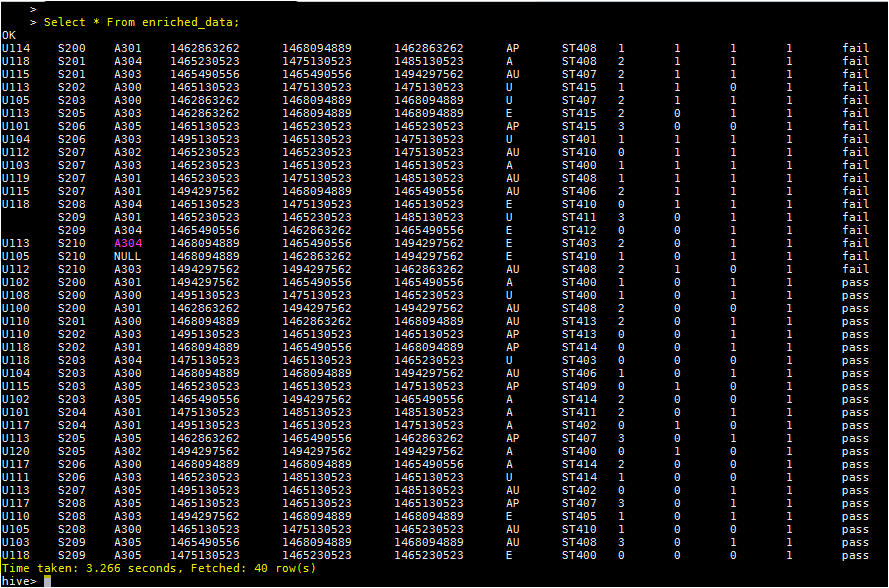


Now we can check whether the data properly loaded in the hive terminal or not.



In the below screenshot we have data for **enriched\_data** table where we filled the null values of **artist\_id** and **geo\_cd** of formatted input with the help of lookup tables,

***hive>select \* From enriched\_data;***



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