ETL Report

Increasing Audio Streaming User Retention
via Music Recommendation System

Land’o’datalakes: Vanessa Gleason, Alistair Marsden, Olivier Rochaix, Eduard Stalmakov

ETL Process 09/22/22

Introduction

Data Sources

1. Boland, D. (n.d.). *Streamable Playlists with User Data*. Streamable playlists with User Data.

   Retrieved September 19, 2022, from http://www.dcs.gla.ac.uk/~daniel/spud/

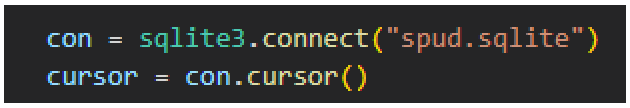
2. Ay, Y. E. (2021, April). *Spotify dataset 1921-2020, 600K+ tracks*. Kaggle. Retrieved September 26,

   2022, from https://www.kaggle.com/datasets/yamaerenay/spotify-dataset-19212020-600k-tracks

Extraction

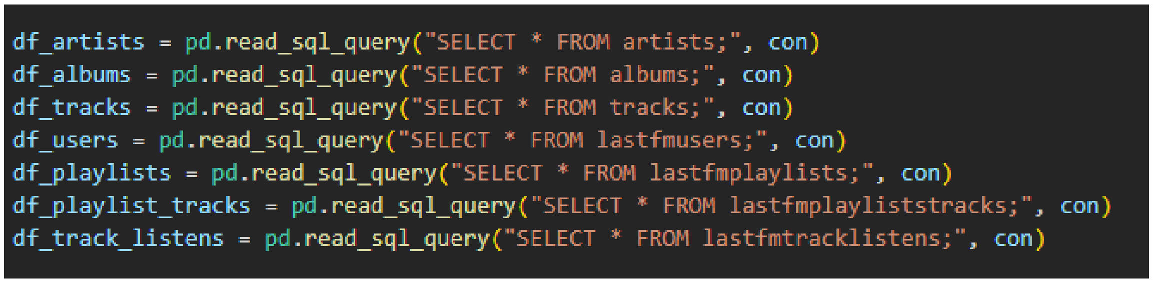
Source 1: Streamable playlists with User Data

1. Go to this link: http://www.dcs.gla.ac.uk/~daniel/spud/ and click “Download dataset”
2. Open spud.sqlite file with the following command:

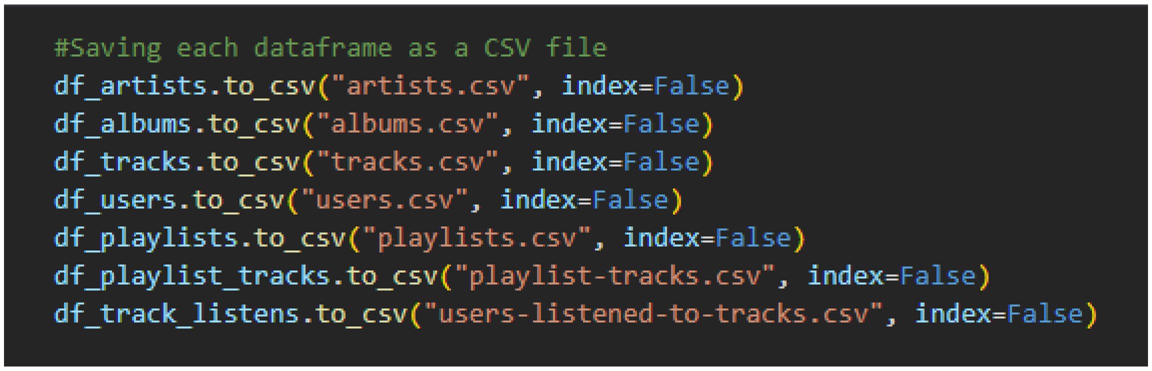


```
con = sqlite3.connect("spud.sqlite")
cursor = con.cursor()
```

3. Extract each table to a pandas dataframe and save each dataframe as a csv file to your desired directory



```
df_artists = pd.read_sql_query("SELECT * FROM artists;", con)
df_albums = pd.read_sql_query("SELECT * FROM albums;", con)
df_tracks = pd.read_sql_query("SELECT * FROM tracks;", con)
df_users = pd.read_sql_query("SELECT * FROM lastfmusers;", con)
df_playlists = pd.read_sql_query("SELECT * FROM lastfmplaylists;", con)
df_playlist_tracks = pd.read_sql_query("SELECT * FROM lastfmplayliststracks;", con)
df_track_listens = pd.read_sql_query("SELECT * FROM lastfmtracklistens;", con)
```



```
#Saving each dataframe as a CSV file
df_artists.to_csv("artists.csv", index=False)
df_albums.to_csv("albums.csv", index=False)
df_tracks.to_csv("tracks.csv", index=False)
df_users.to_csv("users.csv", index=False)
df_playlists.to_csv("playlists.csv", index=False)
df_playlist_tracks.to_csv("playlist-tracks.csv", index=False)
df_track_listens.to_csv("users-listened-to-tracks.csv", index=False)
```

Source 2: Spotify dataset 1921-2020, 600K+ tracks

1. Go to this link: https://www.kaggle.com/datasets/yamaerenay/spotify-dataset-19212020-600k-tracks

and click “Download”

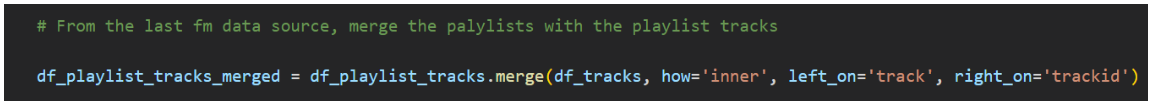
2. Save the CSV file to your desired directory

Transformation

Source 1: Streamable playlists with User Data

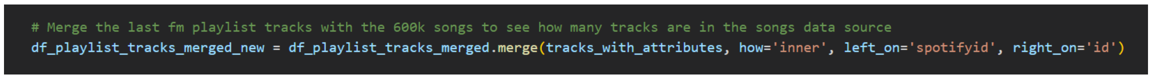
Part 1: Filtering the desired playlists

1. Open “tracks.csv”, “playlist-tracks.csv”, “users.csv”,and “playlists.csv” into a pandas dataframes called:
   - `df_tracks`
   - `df_plalylist_tracks`
   - `df_users`
   - `df_playlists`
2. Merge the `df_tracks` and `df_plalylist_tracks` dataframes



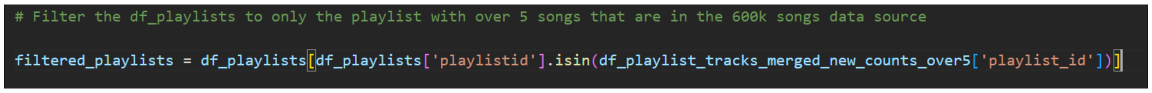
```
# From the last fm data source, merge the palylists with the playlist tracks

df_playlist_tracks_merged = df_playlist_tracks.merge(df_tracks, how='inner', left_on='track', right_on='trackid')
```

3. Open source 2 csv file into a pandas dataframe called `tracks_with_attributes`
4. Merge the previous dataframe with the `tracks_with_attributes`



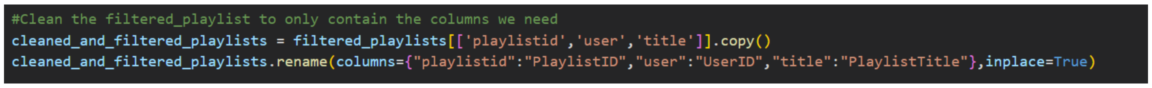
```
# Merge the last fm playlist tracks with the 600k songs to see how many tracks are in the songs data source
df_playlist_tracks_merged_new = df_playlist_tracks_merged.merge(tracks_with_attributes, how='inner', left_on='spotifyid', right_on='id')
```

5. Save only the playlists which have more than 5 songs on them (playlist value counts > 5) into a dataframe called `df_playlist_tracks_merged_new_counts_over5`
6. Filter the `df_playlists` dataframe to only those tracks



```
# Filter the df_playlists to only the playlist with over 5 songs that are in the 600k songs data source

filtered_playlists = df_playlists[df_playlists['playlistid'].isin(df_playlist_tracks_merged_new_counts_over5['playlist_id'])]
```

7. Rename the columns



```
#Clean the filtered_playlist to only contain the columns we need
cleaned_and_filtered_playlists = filtered_playlists[['playlistid','user','title']].copy()
cleaned_and_filtered_playlists.rename(columns={"playlistid":"PlaylistID","user":"UserID","title":"PlaylistTitle"},inplace=True)
```

8. Save to a csv file

```
#Save this as a csv file

#Change in databricks to overwrite the cleaned file in the blob

cleaned_and_filtered_playlists.to_csv("Filtered-Playlists.csv", index=False)
```

Part 2: Filtering the desired playlist tracks

9. Filter the `df_playlist_tracks_merged_new` `to only the tracks that show up in the cleaned_and_filtered_playlists dataframe`

```
df_tracks_in_filteredplaylists_and_600k = df_playlist_tracks_merged_new[df_playlist_tracks_merged_new['playlist'].isin(cleaned_and_filtered_playlists['PlaylistID'])]
```

10. `Filter to only the playlist and spotifyid columns and rename them`

```
cleaned_and_filtered_playlist_tracks = df_tracks_in_filteredplaylists_and_600k[['playlist','spotifyid']].copy()

cleaned_and_filtered_playlist_tracks.rename(columns={"playlist":"PlaylistID","spotifyid":"TrackID"}, inplace=True)
```

11. Save to a csv file

```
cleaned_and_filtered_playlist_tracks.to_csv("Filtered-Playlists-Tracks.csv", index=False)
```

Part 2: Filtering the desired users

12. Filter the `df_users` dataframe to only the users whose playlists are in the `cleaned_and_filtered_playlists` dataframe

```
users_filtered = df_users[df_users['userid'].isin(cleaned_and_filtered_playlists['UserID'])]
```

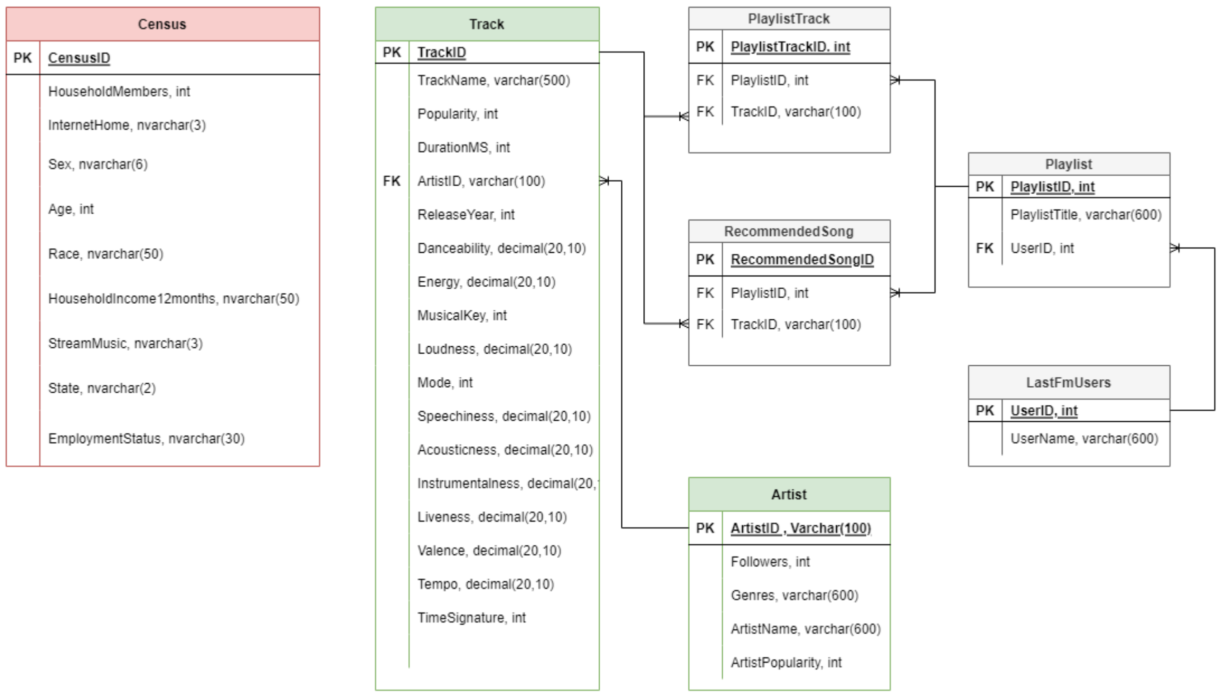
13. Rename columns and save to a csv file

```
#Clean to only the columns we need and rename them
cleaned_and_filtered_users = users_filtered.rename(columns={"userid":"UserID","lastfmuserid":"UserName"})
```

```
cleaned_and_filtered_users.to_csv("Filtered-Users.csv", index=False)
```

Load

To load the data into the SQL database, use this ERD diagram as a reference to the column names and datatypes. The data was loaded using Apache Spark within Azure Databricks.



<u>Source 1:</u> <u>Streamable playlists with User Data</u>

1. In the Azure Databrick, open “Filtered-Playlists.csv”
2. Change the PlaylistID and UserID datatypes to int
3. Load this spark dataframe into the Playlist SQL table
4. Open the “Filtered-Users.csv”
5. Change the UserID datatype to int
6. Load this spark dataframe into the LastFmUsers SQL table

Conclusion