Click-to-Insights\_-ETL-SpotifyChartsAnalysis

Creating an ETL and Analyzing Spotify’s Top Tracks Using Data Visualization

SPOTIFY MUSIC - ETL PROJECT - SQL POSTGRES DB:

WHAT MAKES A MUSIC LIKABLE?



Background:

As the music industry becomes more and more data driven its critical to understand the top charts and the trends to get insights on what kind of music do people like. Here we analyze the popularity aspect of songs. To understand what really makes a song likeable to the masses, I have selected a project to use Data Visualization to find patterns.

In this ETL (Extract, Transform, Load), merge & visualize project, I will read top music charts dataset, analyze and visualize them using a Python.

Data Sources and Tools used:

Datasets to be used: a) https://spotifycharts.com/regional - Extracting data from this website that stores Top 200 songs for all countries by day and week. For this analysis we will be using the top 200 songs which are trending on a weekly basis.

b) https://developer.spotify.com/documentation/web-api/ - This is the spotify API which we will use to extract the data for the songs specifically the attributes of the song.

The data base where the data will be store is Postgres and we will using Python to clean, store , merge and analyze the data.

Goal

Get Spotify data from different variety of source such APIs, Web-scraping and Google-Scholar data-sets

Get a Large amount of data, goal is to get around 1 Million rows.

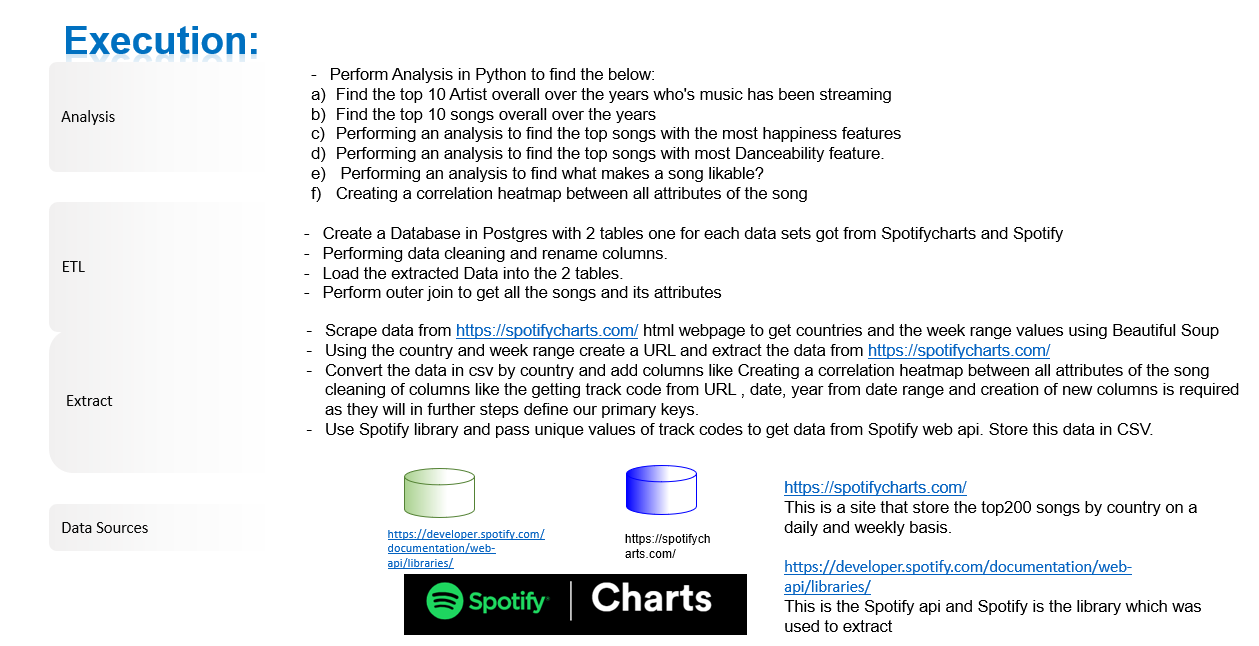
Once I have identified the datasets, I will perform ETL on the data and document the following within the jpynb.

The type of transformation needed for this data (cleaning, joining, filtering, aggregating, etc.).

The type of final production database to load the data into (relational or non-relational).

The final tables or collections that will be used in the production database.

Submit a final technical report with the above information and steps required to reproduce your ETL process.

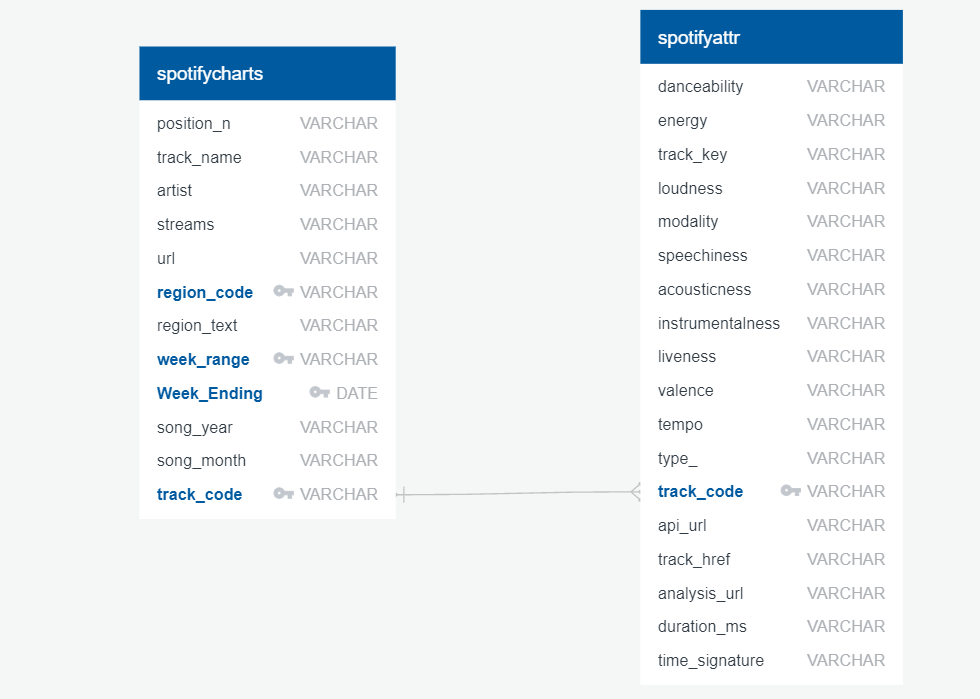


Steps:

Please follow the below steps in the document. Also can be found as markdown in the Jupiter notebook:



Postgres Database diagram:



SQL Statements to create schema:

*-- Create Table spotifycharts*

**CREATE** **TABLE** spotifycharts (

position\_n VARCHAR NOT **NULL**,

track\_name VARCHAR,

artist VARCHAR,

streams VARCHAR NOT **NULL**,

url VARCHAR,

region\_code VARCHAR NOT **NULL**,

region\_text VARCHAR NOT **NULL**,

week\_range VARCHAR NOT **NULL**,

Week\_Ending DATE NOT **NULL**,

song\_year VARCHAR NOT **NULL**,

song\_month VARCHAR NOT **NULL**,

track\_code VARCHAR NOT **NULL**,

**PRIMARY** **KEY** (region\_code, week\_range, Week\_Ending, track\_code )

);

*-- Create Table spotifyattr*

**CREATE** **TABLE** spotifyattr (

danceability VARCHAR,

energy VARCHAR,

track\_key VARCHAR,

loudness VARCHAR,

modality VARCHAR,

speechiness VARCHAR,

acousticness VARCHAR,

instrumentalness VARCHAR,

liveness VARCHAR,

valence VARCHAR,

tempo VARCHAR,

type\_ VARCHAR,

track\_code VARCHAR NOT **NULL** **PRIMARY** **KEY**,

api\_url VARCHAR,

track\_href VARCHAR,

analysis\_url VARCHAR,

duration\_ms VARCHAR,

time\_signature VARCHAR

);

Final results:

