Bethzaida Guerra

Alisha Hill-McElroy

Thi Q

ETL-Project

Inspiration

We began the project with the idea of analyzing the airline industry. However, the information didn't pan out to our satisfaction. After additional brainstorming, we remembered the Week 6, Day 1, Activity 6 OMDB Request class assignment from the SMU-DAL-DATA-PT-06-2021 Boot Camp. We then decided to tackle that topic and add our creative twist to it. It is very common for analysts to analyze the best of the best, so we decided to analyze the worst of the worst. The Bottom 100 Movies!

Sources of Data

Our first thought was to check a familiar source of Kaggle.com, however we didn't find a csv to meet our needs. Our next stop was IMDB.com, a very creditable movie critic website that would provide all sorts if data for any movie you could think of. Lastly, we scrolled to OMDB.com to retrieve additional data using API keys to complete our data set.

- https://www.imdb.com/chart/bottom
- http://www.omdbapi.com/

Extract

The primary extraction came from IMDB.com in which web scraping was performed and bottom 100 movies, rank and at first the movie title was collected. During this process we noticed several values were NA. This is because we used the movie titles instead of the ID to perform the request. There were discrepancies in the titles for example: IMDB Kirk Cameron's Saving Christmas Vs OMDB's Saving Christmas

Using the movie ID we pulled from IMDB, we used an API key to retrieve additional information on the movies: Movie ID, Movies, Rank, Movie Rating, Rated, Year, Genre, Actors Directors,Gross. A loop was written in order to pull all 100 movies, saving us time.

Transform

After the data was extracted, a movie data meta table was created with the columns: Movie ID, Movies, Rank, Movie Rating, Rated, Year, Genre, Actors, Directors,Gross.

The next step of the process was to clean the data. The data was cleaned by dropping the columns: Movies_Y and IMDB Rating from the OMDB table as it was identical to the Movie rating and the Movies X column. Another thing that was decided was to not separate the genre and director's column into separate columns. This was decided because most movies had multiple genres and directors and we wanted to keep the keys the same.

We then created smaller tables: a genre table, a director's table, and a rated table were created in order for us to be able to assign primary and foreign keys. The primary key being the movie id, and the foreign keys coming from the genre, rated and director tables.

In order for the meta table to be joined with the smaller table, some data cleaning had to be done which included dropping certain columns that were interfering with SQL reading our data, after this everything was consolidated and was ready to load.

Load

Tables were created via https://app.quickdatabasediagrams.com/. There the primary and foreign keys were established. In order to create foreign keys, we needed to establish a serial id for the genre, rated and director tables. We were able to create and join the tables on the ID.

After which we began to load the data via Pandas into a CSV which could be read by SQL. We then merged all of our clean tables into our main meta table. Once they were all merged, we wrote our final data back into SQL, finally we were able to write our queries in SQL.

Limitations

With infinite time and resources, we would change the gross by removing the "$" and making it an INT instead of a VARCHAR.