**Netflix & Movies *ETL***

Dilia Yunusova

Jessica Kwon

Leonard Paul Kamara

Sapiesi Tupou

Wynne Zhang

**Project Description/Outline**

The movie industry is monumental. Watching movies is still one of the most popular ways of spending leisure time, and Netflix has become one of the most popular movie streaming services chosen by the consumer. One would wonder what kind of budget these movies have… Does it determine the success of a film? Does high budgets mean more revenue?

The purpose of the project is to extract movie data from the listed data sources below, transform them into a new dataframe and load in PostgreSQL and see whether it can answer our questions.

**aData Sets and Sources**

* Netflix\_final.csv: [https://www.kaggle.com/shivamb/netflix-show](https://www.kaggle.com/shivamb/netflix-shows)
* Movies\_revenue.csv: API call from [https //www.themoviedb.org](http://www.themoviedb.org)
* movies\_num\_final.csv: <https://www.the-numbers.com/movie/budgets/all>

**Extracted Data Type**

CSV

JSON

**Data Worked On**

Python Pandas

PostgreSQL

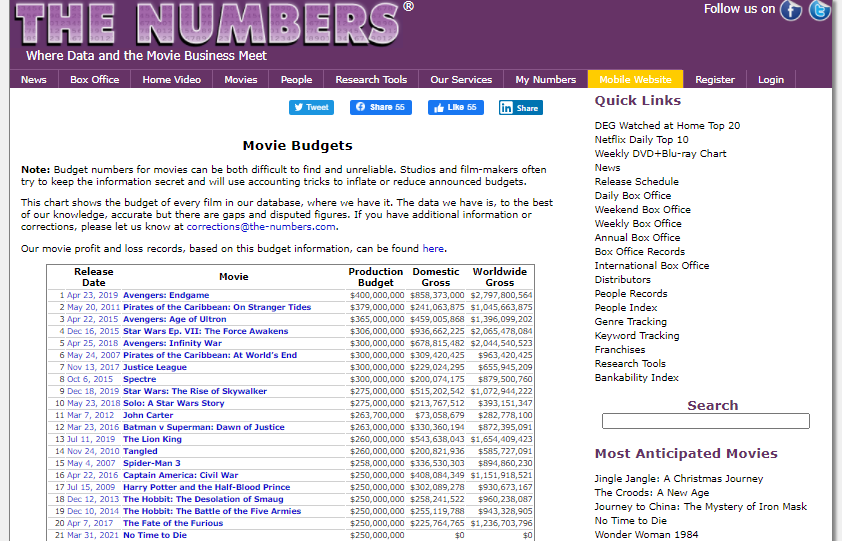
**Our Method of Action**

* How do we find the relevant data and extract it?
* Load and check: if data doesn’t satisfy needs check for another source.
* Extract new data and merge with old data.
* Clean and sort

**Extract**

Extract #1: The Numbers: [The Numbers Website](https://www.the-numbers.com/movie/budgets/all)

We began by extracting data from our first source: a website called “The Numbers.” This site is run by Nash Information Services, a premier provider of movie industry data to multi-billion dollar production companies and independent filmmakers. Using Splinter and BeautifulSoup, we extracted 6000 movie titles along with their budgets and gross profit from the web pages. This general movie data was exported into a CSV file for later analysis.

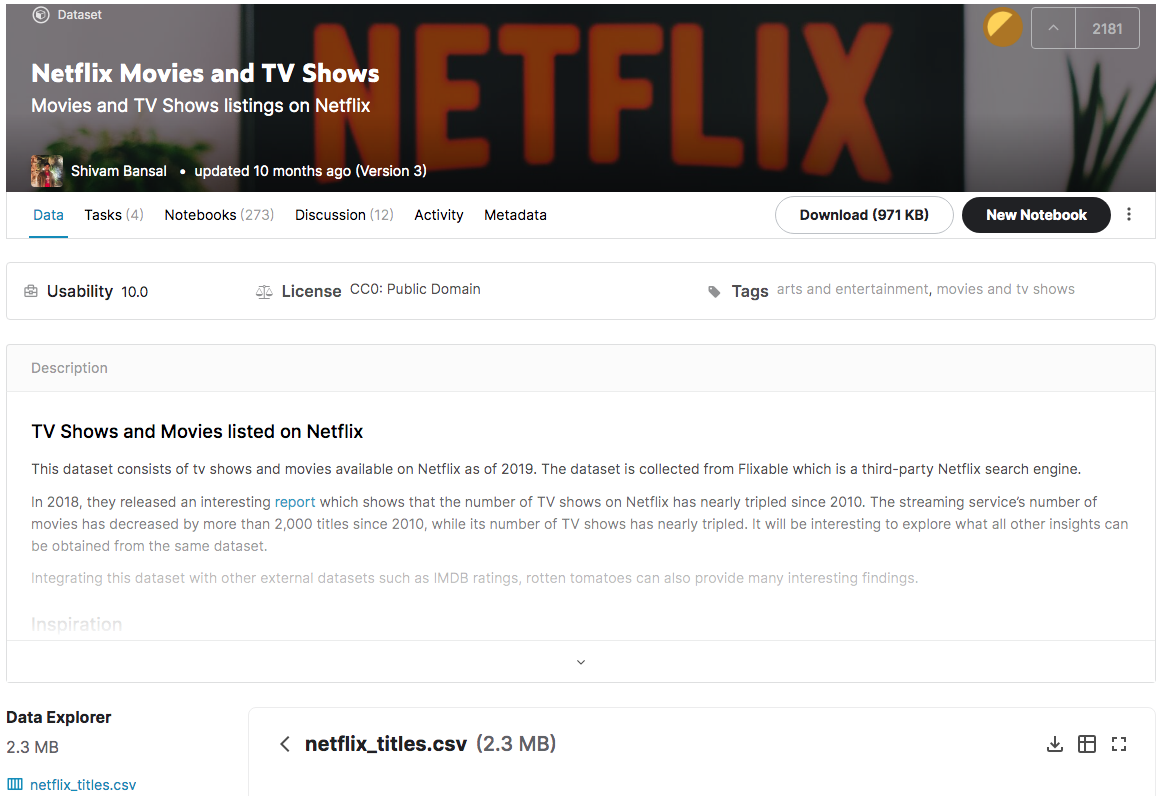
****

****

****

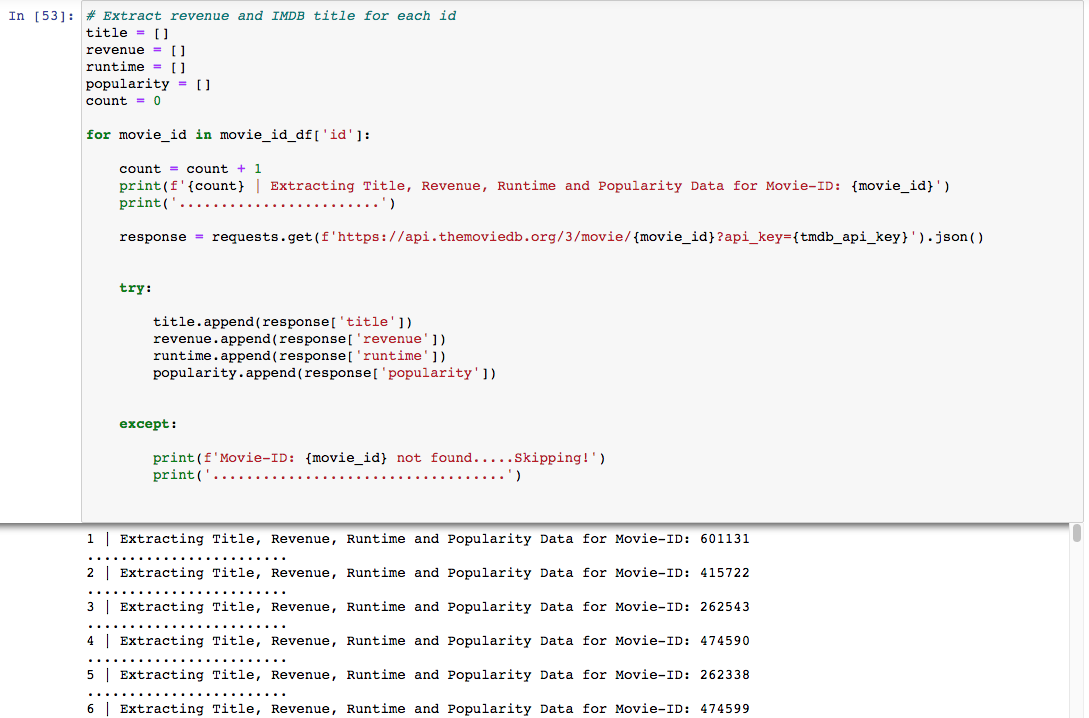
Extract #2: Netflix Movie and TV Show Data from Kaggle: [Kaggle - Netflix Movie & TV](https://www.kaggle.com/shivamb/netflix-shows)

From Kaggle we retrieved a dataset authored by Shivam Bansal consisting of TV shows and movies available on Netflix as of 2019. Bansal’s data was originally sourced from Flixable which is a third-party Netflix search engine. This dataset came in CSV format and included basic Netflix title information such as movie name, release year, rating, and country.

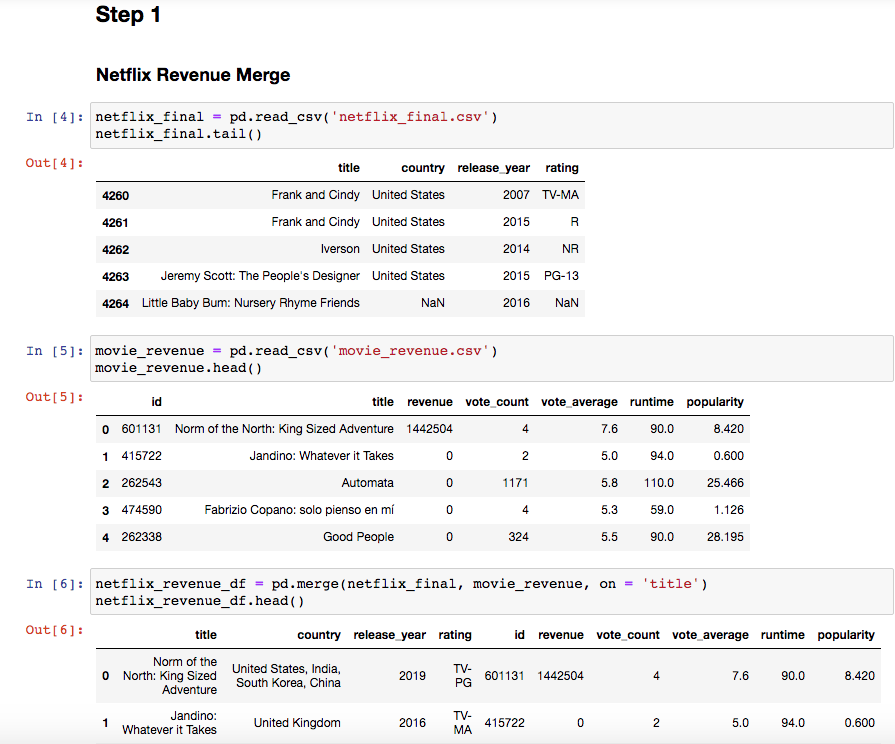
****

Extract #3: API call through The Movie Database (TMDb) API: [TMDb](https://www.themoviedb.org/)

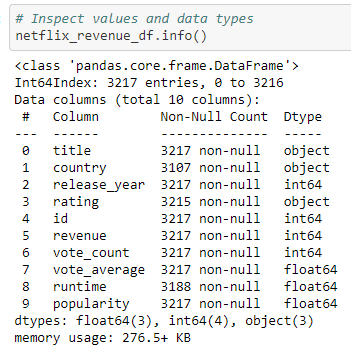
The Kaggle dataset only had very basic information regarding the Netflix movies, and we wanted to learn more about budgeting, revenue, and popularity of these movies. To do this we connected with The Movie Database’s API service, an extensive movie and TV show database. Using movie titles retrieved from Kaggle’s preliminary Netflix data, we performed an API call using TMDb’s API to get the movies’ corresponding IDs. We exported the resulting JSON response into a CSV file, and then used the newly acquired movie IDs to retrieve additional information through TMDb API. **Our end goal is to fill up a dataframe with the movie’s ID, title, country, runtime, popularity, revenue, and production budget.** With another TMDb API call, we retrieved revenue, runtime and popularity data and exported into a separate CSV file.



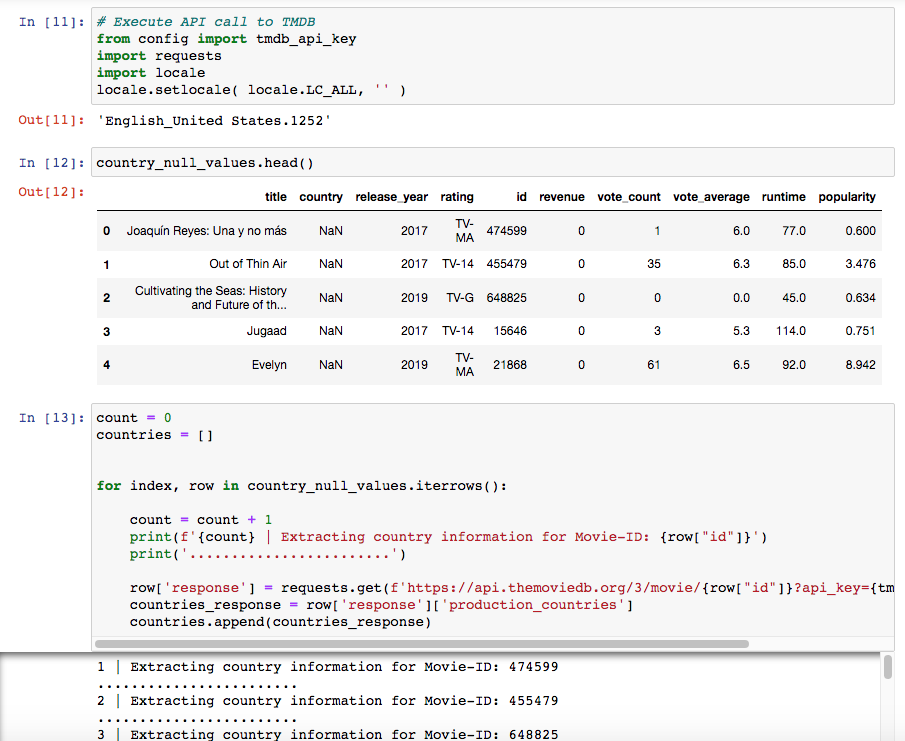
## **Transform - Merge, Clean, and Wrangle**



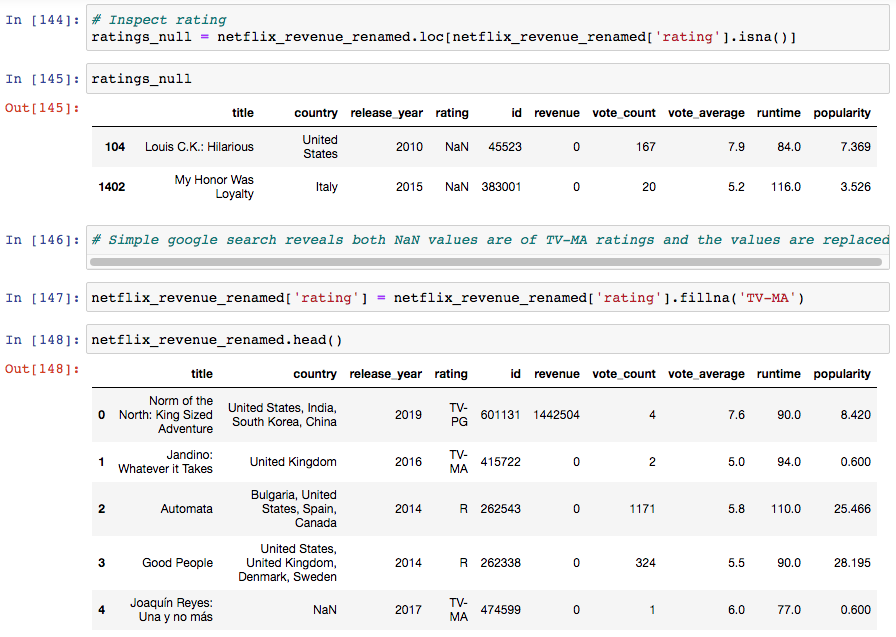
To begin putting our dataset into one main file, we merged “netflix\_final” data from Kaggle and “movies\_revenue” from TMDb API pulls using the title column to create a new data frame called “netflix\_revenue\_df” where it has both general Netflix data as well as movie revenues.We then inspected information in our newly created “netflix\_revenue\_df” where we noticed discrepancies in the count of null values indicating some of the columns in our data-frame had null values. A closer look shows that there are count discrepancies in the following columns: *country, rating,* and *runtime.*



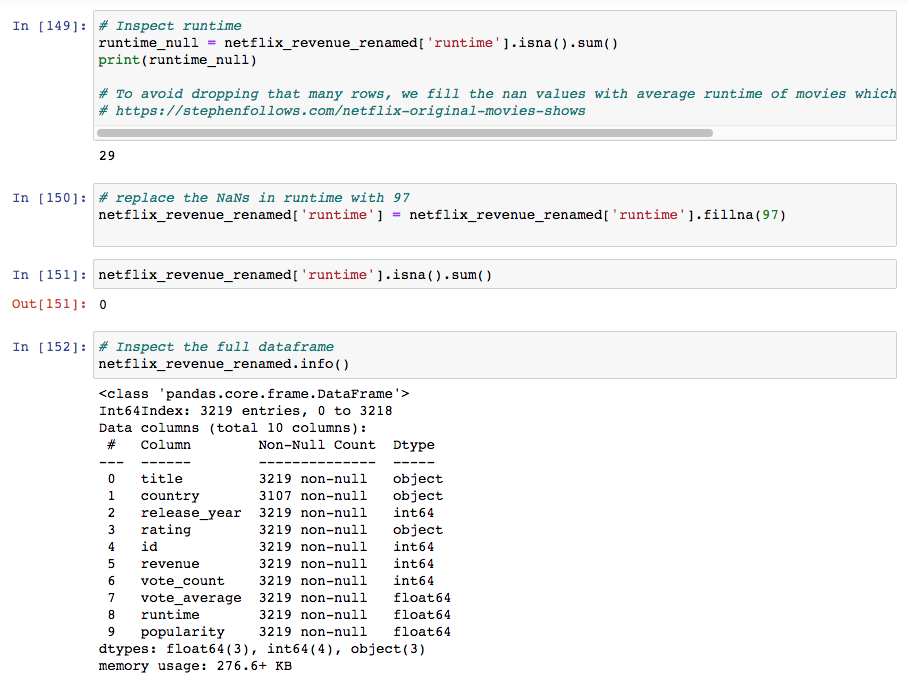
*Inspecting Country:* 110 values were NaN values on the Country column. To eliminate columns with these NaN values we extracted the country names by using titles and id to perform an API call to the TMDB database where it provided the 69 countries that were missing under country. The remaining ones are renamed as “unknown”. However, some countries did have multiple values, so we had to split them and select the first attribute to bring back only one country.



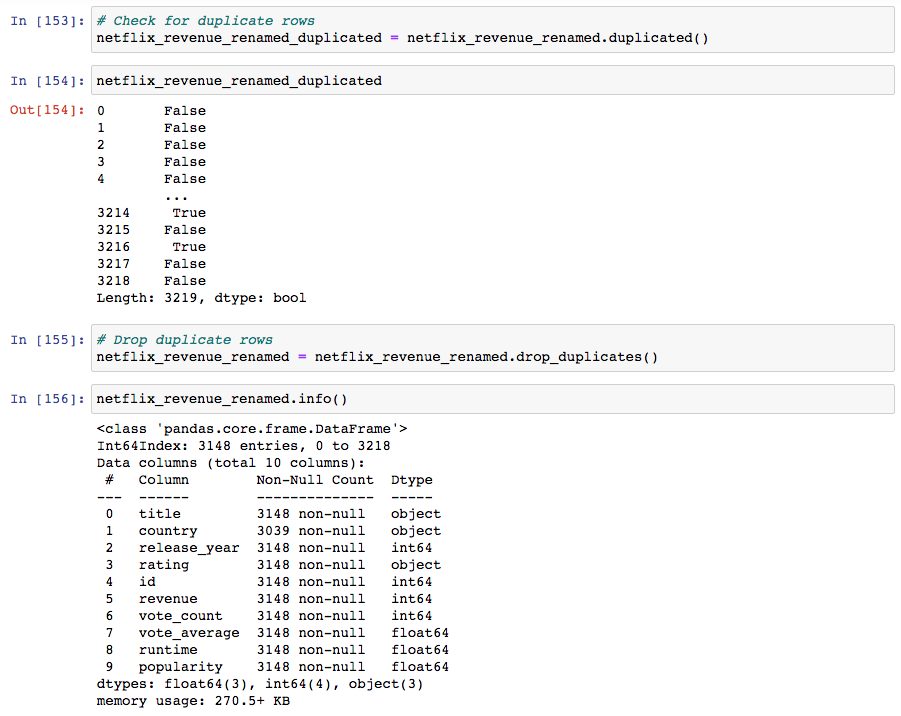
*Inspecting Rating:* Two values were NaN values on the Rating column. With a simple google search, it revealed that both NaN values are of TV-MA ratings. With the renamed function, the rating column was then updated to TV-MA.



*Inspecting Runtime:* Twenty-nine values were also NaN values on the Runtime column. Just like Rating, Google directed us to [*https://stephenfollows.com/netflix-original-movies-shows*](https://stephenfollows.com/netflix-original-movies-shows) where we found out the average runtime for netflix shows were 97 minutes. We updated those NaN values to 97.

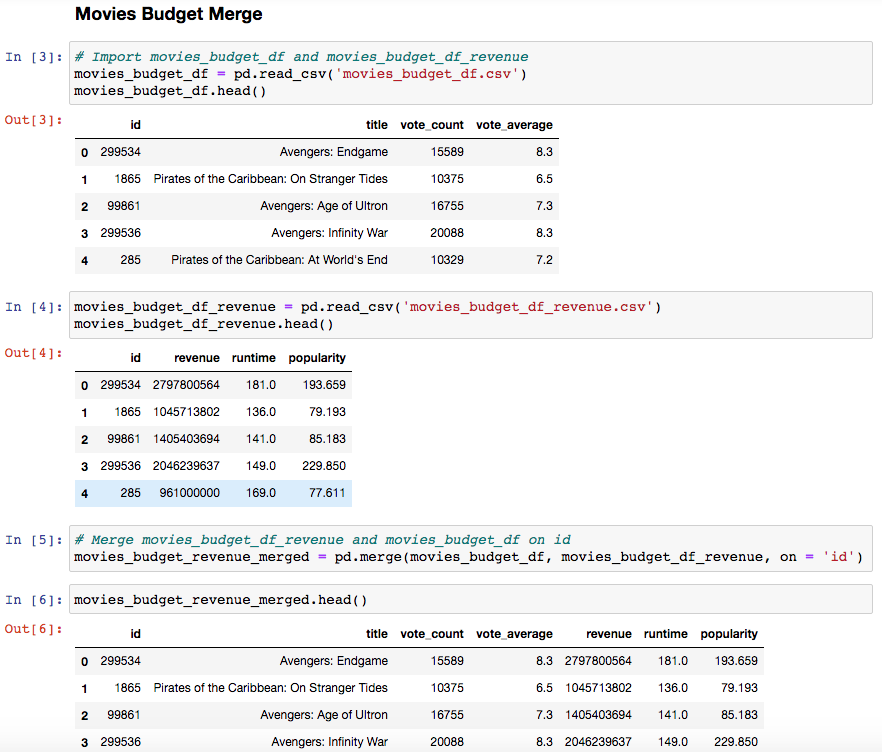


To make sure our data is unique, we used the duplicated function to identify duplicates and then dropped them.

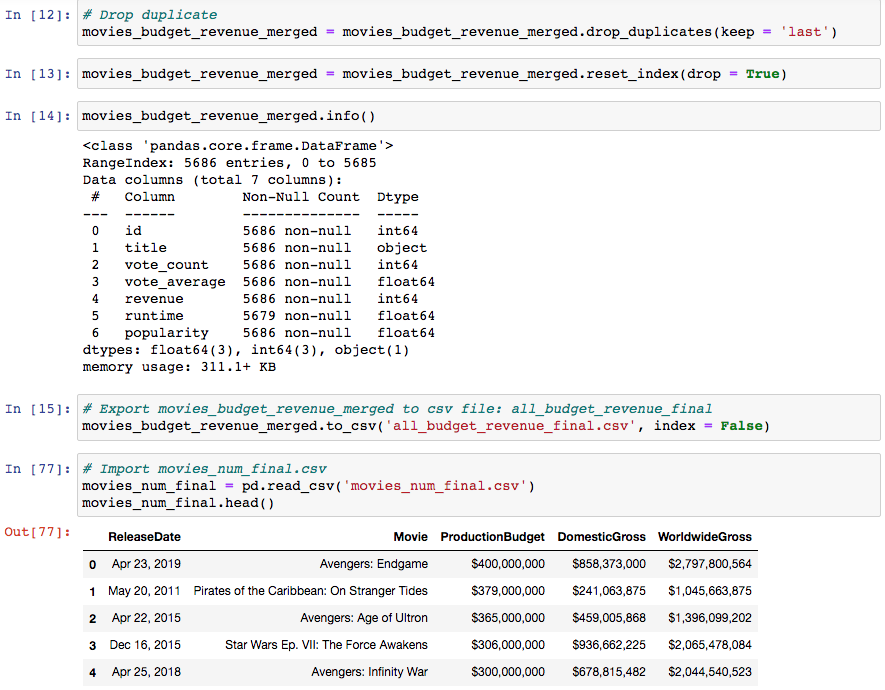


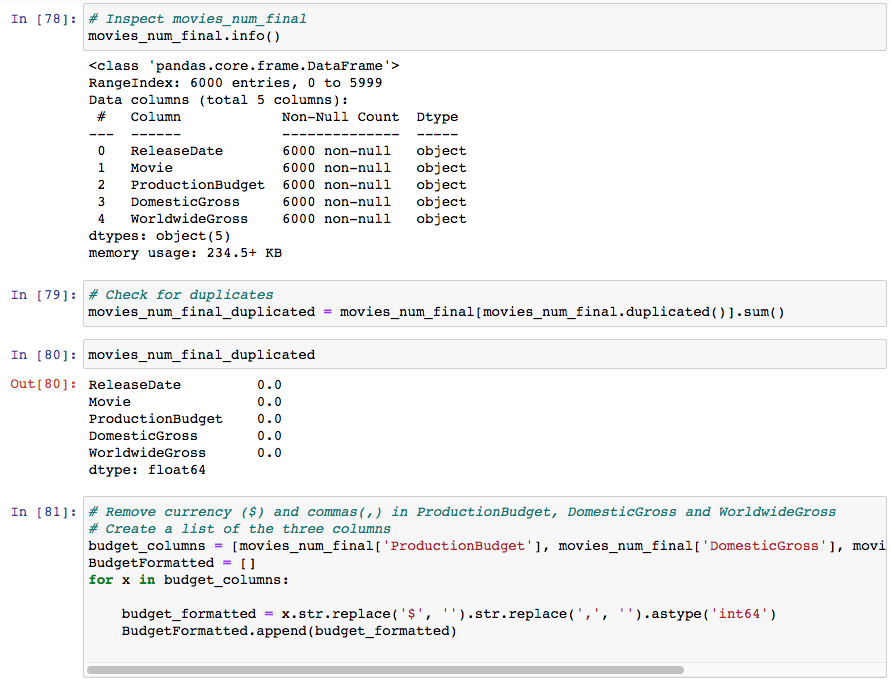
So to create our main netflix-revenue file, we then rename the columns to make it cleaner, and then export it to csv as **netflix\_movies\_revenue.csv**.

### 

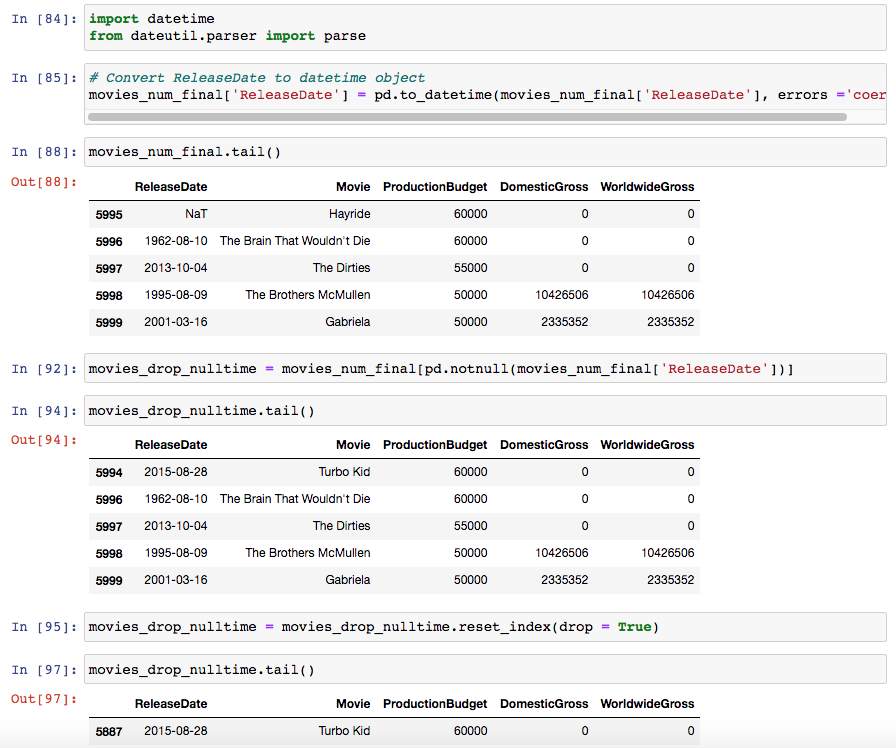


From The Numbers website where we created *movies\_budget\_df.csv* and *movies\_budget\_df\_revenue*, we mergedthese two using the id column to create movies\_budget\_revenue\_df data frame where we can view the popularity stats of a film. These would include a film’s vote\_count, it’s vote\_average as well as revenue, runtime and popularity.   
As we have done with the previous datasets, we inspected movies\_budget\_revenue\_merged using info as well as check for duplicates. Once the data set was cleaned, we then exported it to a csv file- **all\_budget\_revenue\_final.csv**.





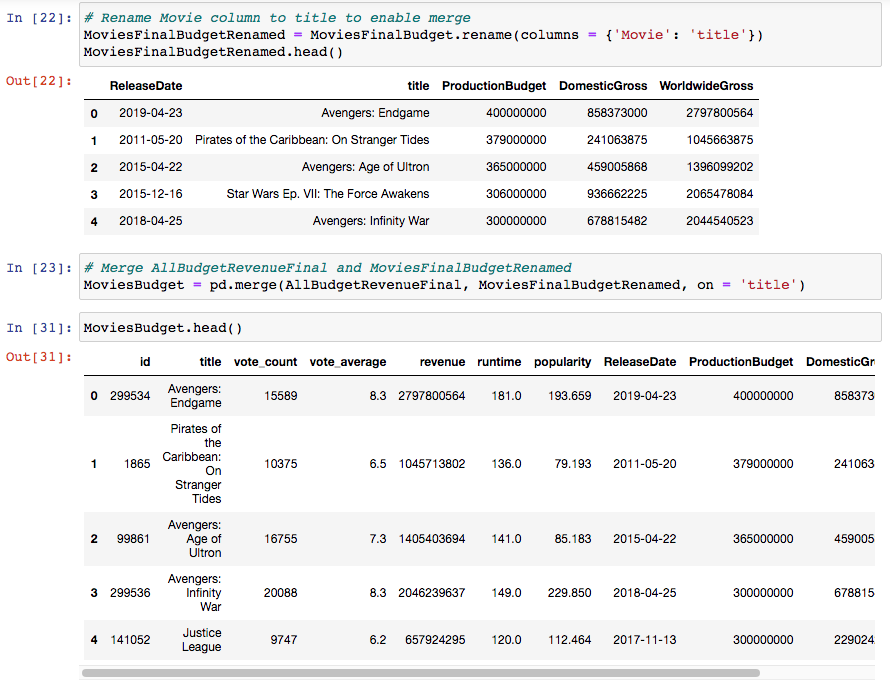
To get the budgets for our movies list, we imported over the movies\_num\_final.csv where it had the ProductionBudget, the DomesticGross as well as their WorldwideGross. All 6000 records were accounted for and when checked for duplicates, none were found. However to make sure our dataset across all csv files matched, we had to make sure that our data types would match therefore we removed currencies and comma values as well as convert ReleaseDate to a datetime format. Errors did appear since some of the movie's ReleaseDate were in 2021, hence these null values were then dropped and our final file was exported to csv as **MoviesFinalBudget.csv.**



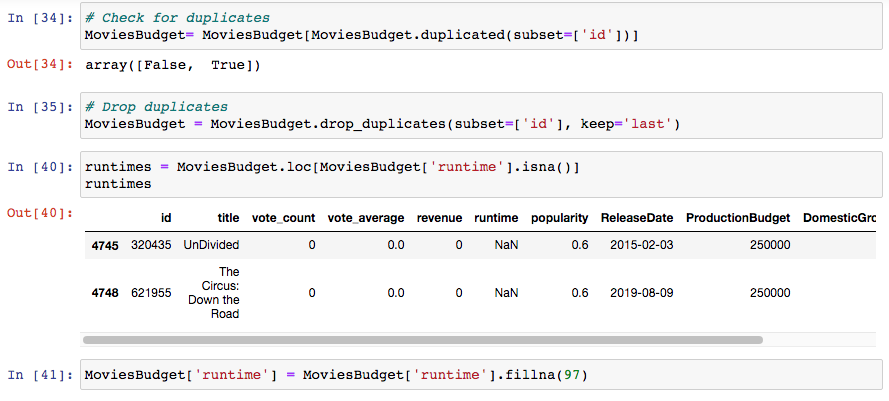
**Final transformations using:**

* Netflix\_revenue\_final.csv
* all\_budget\_revenue\_final.csv
* MoviesFinalBudget.csv

To put netflix movies, its revenues, its budgets and popularities, we had to merge AllBudgetRevenueFinal and MoviesFinalBudget. But prior to it, we had to rename our “Movie” column from MoviesFinalBudget.csv into “title” so it can match our netflix file and thereby renaming MoviesFinalBudget as MoviesFinalBudgetRenamed.



To continue having a clean set of data, we checked for duplicates once again, dropped what was duplicated and then exported into a new csv file called **FinalMoviesBudget.csv** where our movies budget, revenue and popularity were saved together with movie titles.

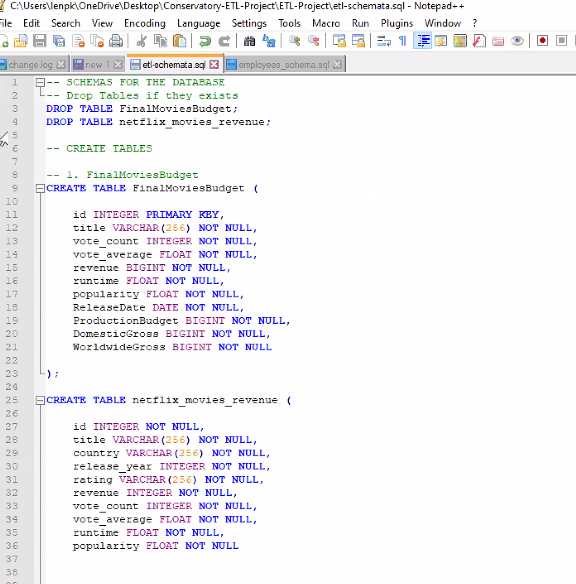


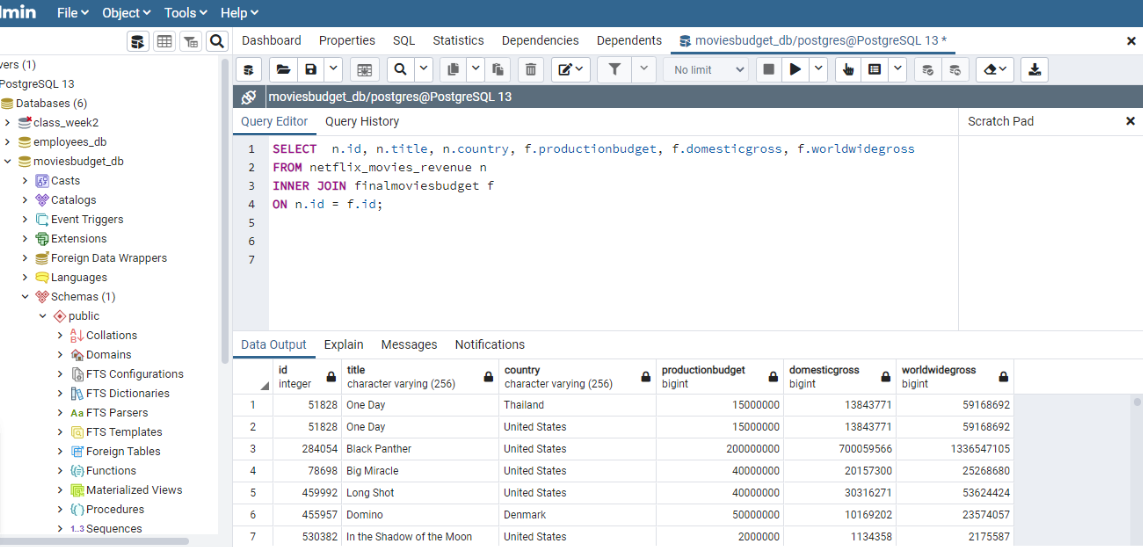
## 

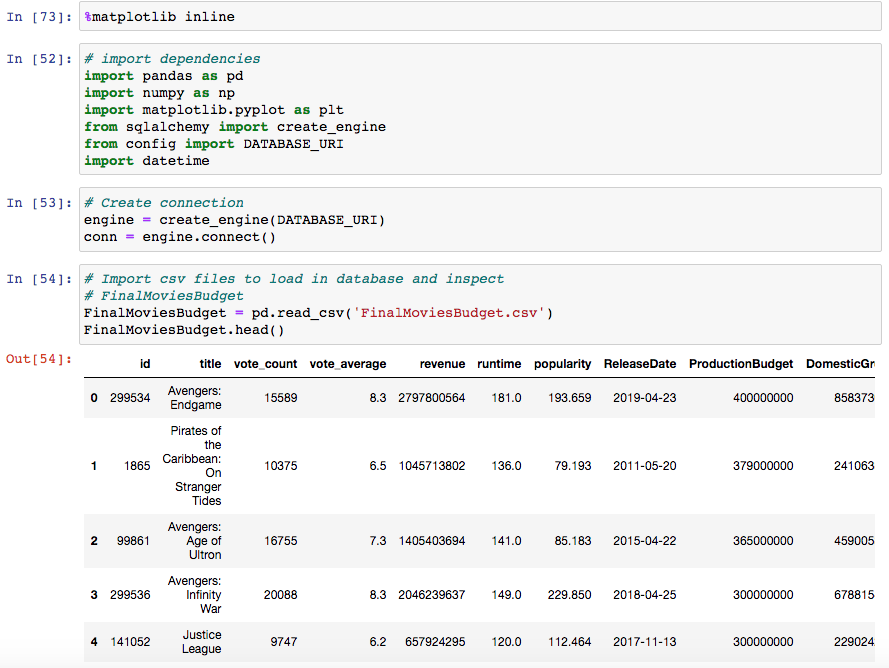
## **Load**

After transforming our datasets, we are left with two tables: a dataset from The Numbers and one with a combination of data from Kaggle and TMDb. First we created a database in PostgreSQL called moviesbudget\_db, then we created the schemata for the two tables. We created the tables in PostgreSQL using the schemata. Then we moved to pandas to create a connection to our database and imported our final CSV files to load for further inspection. We initially got an error while loading in the CSV because the column title was capitalized, and we had to convert all titles to lowercase to avoid this error.

After loading the files in the database, we query the database in pandas to confirm that the tables were loaded correctly. We also went into PGAdmin to do a merge to confirm tables could load properly. We then returned to pandas to do the merge in pandas as well. After merging, we decided to do some analysis. We analyzed the top 10 production budgets and revenues.

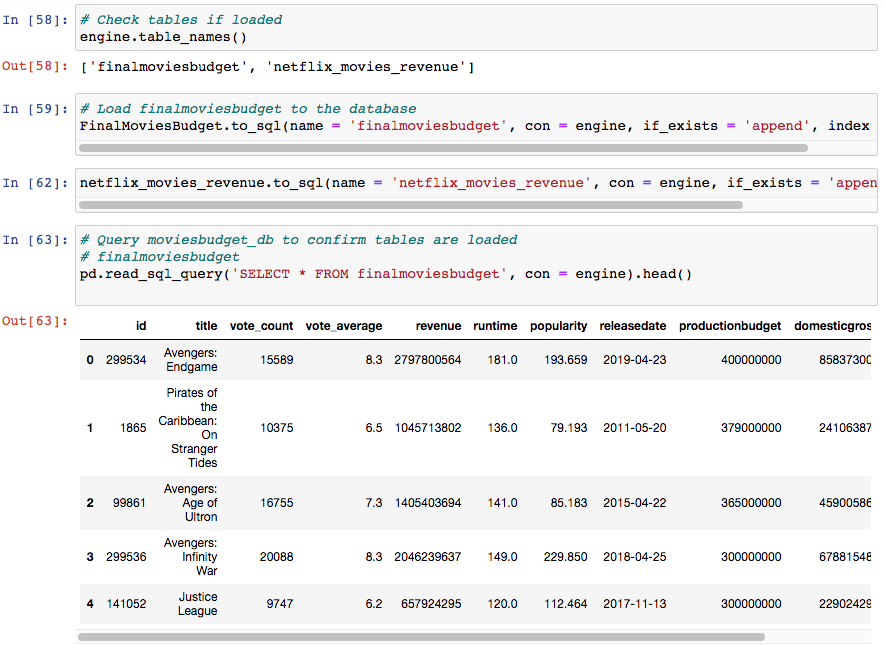






-> using SQL to query tables,

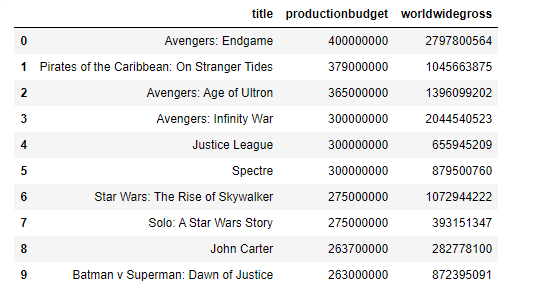
Used PGadmin to load tables & test database tables

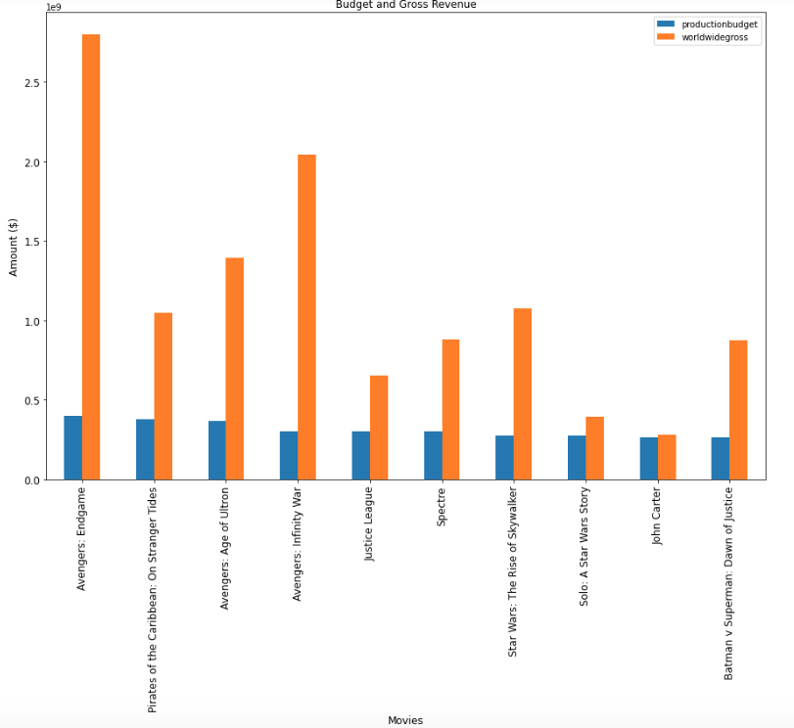


# **Conclusion**

While we started out with Netflix and the wonderment of how much they are budgeted, how much they collect in revenues and whether big budgets means more revenues, our extract, transform and load processes lead us into more than just Netflix movies and tv shows. We began to wonder overall, in the movie industry as a whole- - do highest budgeted movies make the most revenues?

Our group is able to analyze the data frame and find out top 10 movies budgeted and top 10 movies with highest revenue gathered in the movie industry overall and not just netflix. On the graphics presented below, our data table and bar graph shows that highest budgeted movies are not always the highest revenue gathered films.





Overall, we learned that the Extract, Transform and Load process is a powerful way to collect data and compare. By extracting our netflix datasets (from Kaggle) and movies datasets (from The Numbers), we were able to transform them into a main file where we can load into postgresql and would’ve done a lot of analysis with them.