**Extraction**

For this ETL project, I used 2 datasets from Kaggle. The 2 csv files are stated below:

CSV files used:

netflix\_shows.csv <https://www.kaggle.com/chasewillden/netflix-shows>

netflix\_titles.csv <https://www.kaggle.com/shivamb/netflix-shows>

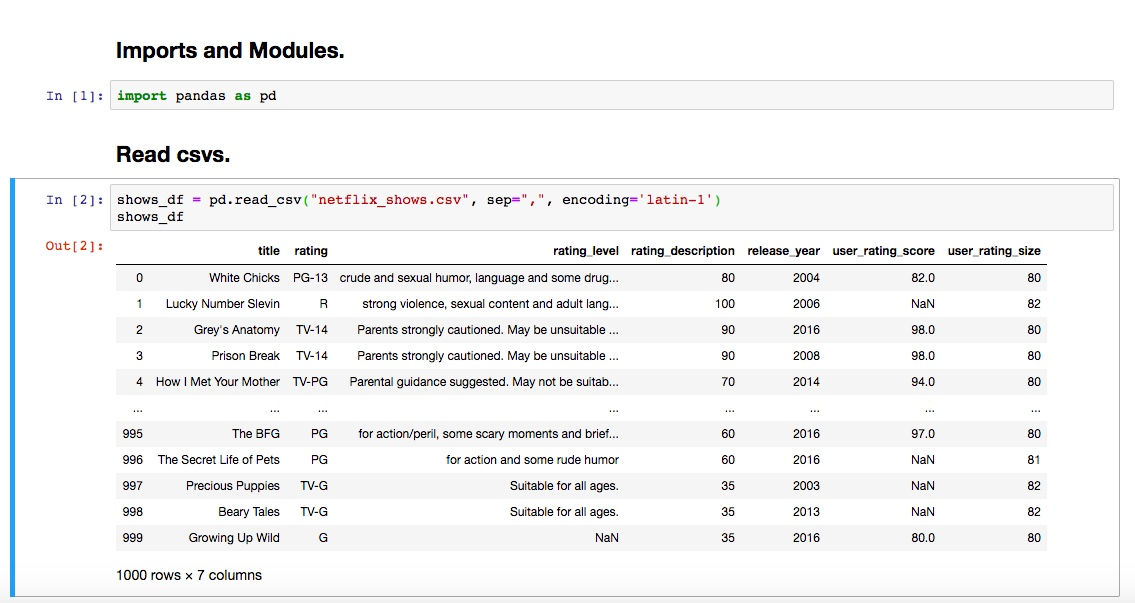
The two datasets are csvs of Netflix shows and titles. The Netflix shows csv presents data of 1000 shows located on the rows. This csv file consists of 7 columns: title, rating (in terms of content such as PG, TV-14, etc.), rating\_level (in terms of themes that require parental guidance, etc.), rating\_description (in terms of numbers ranging 0-110), release year, user rating and user rating size (with numbers ranging from 0-100 and NA).

On the other hand, the Netflix titles csv has a more broad yet precise descriptions of titles currently streaming on Netflix. It has a total of 6234 shows located on the rows. This csv file has 12 columns that consist or show id, show type (TV show or Movie), title, director, show\_cast, country (United States, South Korea, etc.), date added, release year, rating (in terms of content), duration, listed in (show category), and description (of the show).

**Extraction (continued): Read both csv files.**

The first step of reading csv files is to read the csv using pandas dataframe. It was necessary to import pandas and convert the csv into pandas dataframe. Based on trial and error, I needed to add latin -1 in order to encode the first 256 code points of the Unicode character set. By adding encoding = ‘latin -1’, it was able to execute and read through all 1000 rows and 7 columns of netflix\_shows.csv. It is depicted on Figure 1 below:

**Figure 1**



Similarly, I also had to convert the titles csv into pandas dataframe. Based on this dataset, it was not necessary to encode the file into latin -1. Although it had more rows, columns and data compared to the shows csv, it was able to execute and read through all 6234 rows and 12 columns.

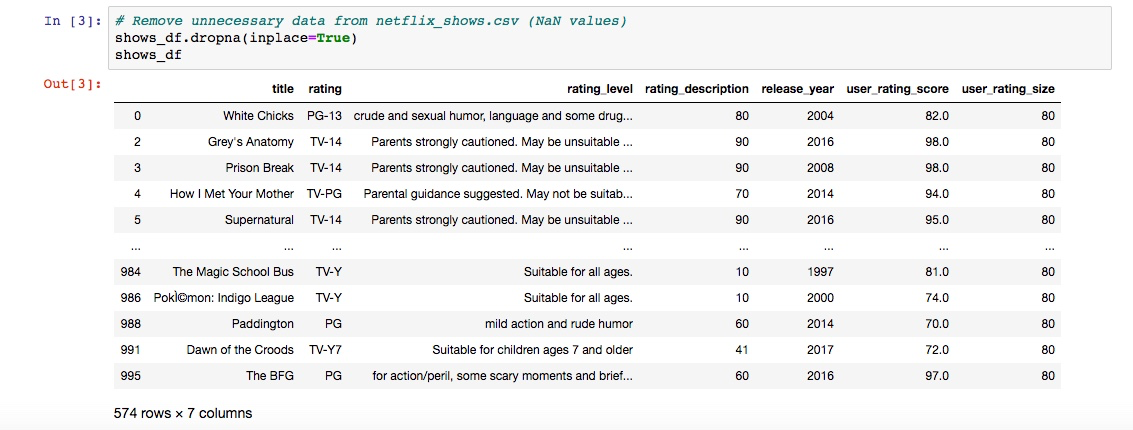
**Figure 2**



**Transform**

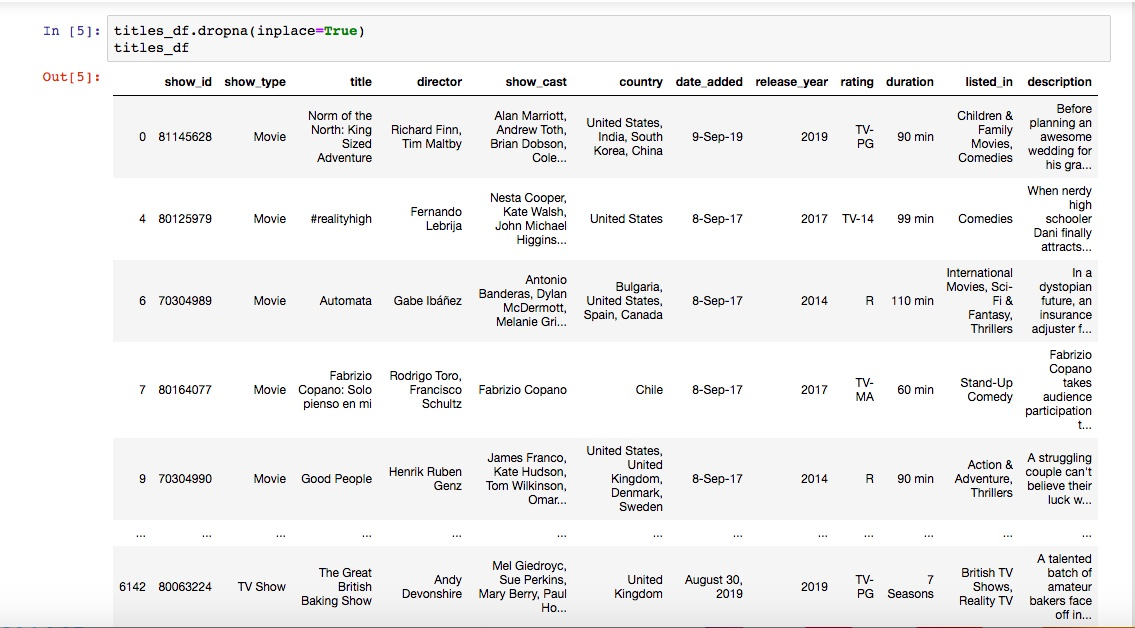
The second crucial step of cleaning up the datasets involves filtering out the unnecessary data. For both csv files, I had to eliminate all sets of data that had NaN inputted on certain titles. In order to eliminate the NaN values, I had to use dropna(inplace=True) for both data frames.

**Figure 3**

****

As a result, 426 titles on rows had NaN values. The amount of rows on shows.csv were reduced from 1000 to 574.

**Figure 4**

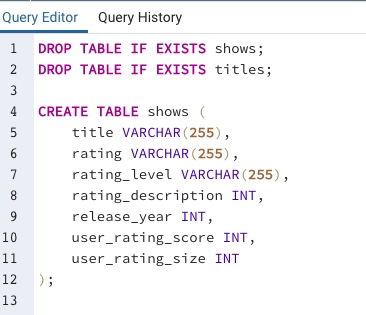
****

Similarly, the titles csv was reduced from 6234 to 3774 rows respectively.

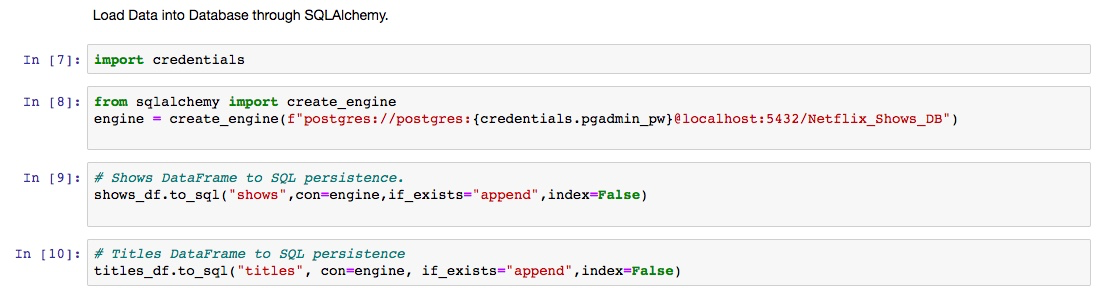
**Load**

The last final step was to load the final output into a DataBase on PostgreSQL. I had to create a schema that matched all columns and rows of both csv files.

**Figure 5: Create schema.**

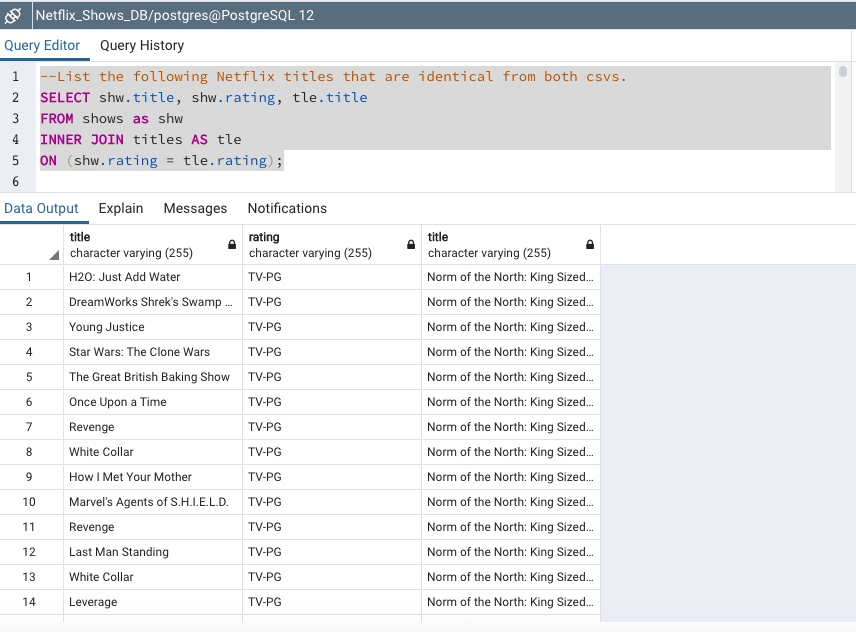


I connected the database using SQLAlchemy and utilized both data frames to SQL persistence.



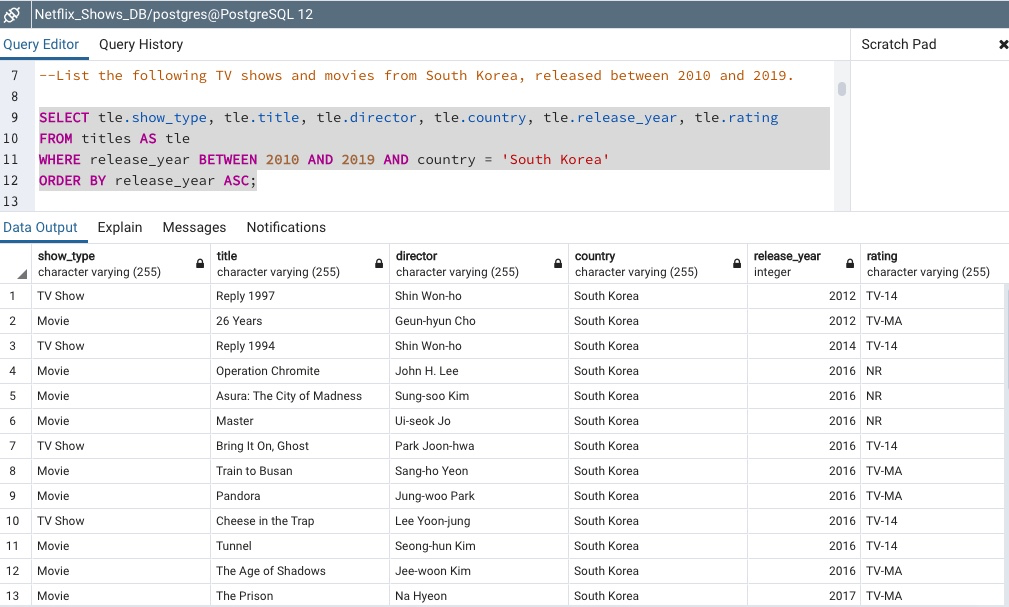
In order to examine the effectiveness of database on PostgreSQL, I used my knowledge upon creating queries that load data from both csv files. I started by joining segments from shows and titles csv, as depicted on Figure 6. More queries are demonstrated via screenshots and on the query file itself on the github repository of my project.

**Figure 6**

****

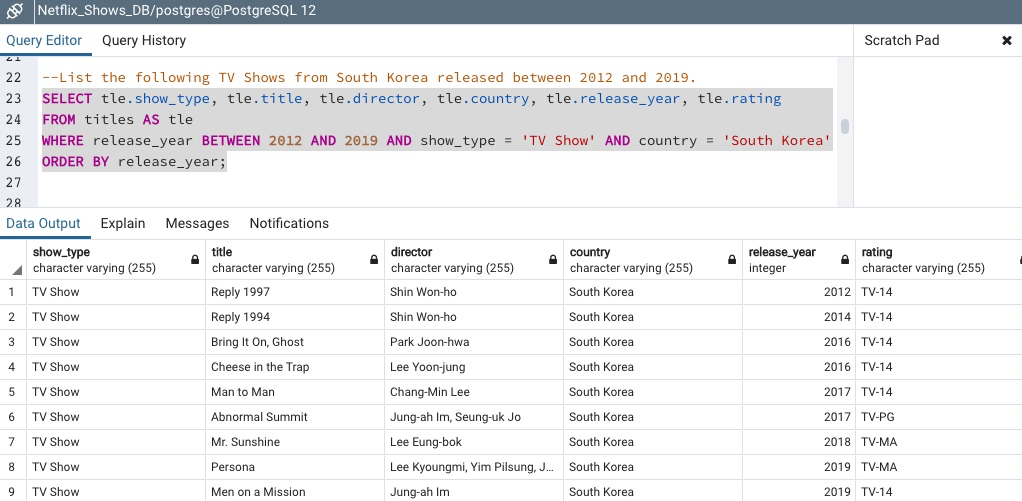
**Figure 7**

I also experimented by extracting TV shows and Movies on Netflix that were produced and released in South Korea between 2012 and 2019.



**Figure 8**

Furthermore, I consolidated the data by focusing exclusively on TV shows that were produced and released in South Korea between 2012 and 2019.



**Summary**

I used the two datasets to identify if there are any similarities on both data sets. Both datasets have ratings based on theme (PG, TV-14, etc.) as well as certain movie titles. However, the main difference between the shows and titles csv is the fact that the titles csv has included both TV shows and movies and offers a wider range of selections on Netflix. Furthermore, the dataset included shows and movies from other countries including South Korea. With the rising trend of Korean movies and TV shows, I extracted such data by executing a query that exclusively focused on Korean movies and TV shows that were produced and released between 2012 and 2019.