NETFLIX ORIGINAL PROGRAMMING ANALYSIS

ETL Project

By
Echo Yu, Matt Wang

Executive Summary

Our project is inspired by the Netflix database on Kaggle. If you watch Netflix often, the Netflix original is really a hit. We want to analyze the data around Netflix original programming. We choose two dataset and manage the whole ETL process. 'Netflix Movies and TV Shows' dataset is about all the Movies and TV shows on Netflix, the content is informative. 'Netflix TV Series Dataset' is about recent original programming. Combining the two datasets, very comprehensive data can be obtained, and finally the related database is established in PGadmin. The whole process uses jupyter Notebook & SQL.

Table of contents

Introduction

This is a technical report about the Netflix Original Programming Database. Using the Dataset, you can obtain the following information:

(1) Understanding what content is available in different countries.
(2) Identifying the rating of the original series.
(3) Network analysis of different titles and actors/directors.
(4) The duration of the original series.

ETL was used in the research process, PGadmin4 was used in the final storage of the database, Jupyter Notebook was used in the analysis process, and Python & SQL was used in the language.

Methodology

Data Extraction

1. Data Source-Kaggle

(1) Netflix TV Series Dataset

This dataset is about Netflix's original programming tv series. The dataset is collected from Wikipedia.
URL: https://www.kaggle.com/datasets?search=netflix
Dataset type: 1 csv file

The content is below:

- Title- Name of the Series.
- Genre - Genre of the series.
- Premiere - Released Year
- NoofSeasons - Total number of Seasons.
- NoofEpisodes - Total number of episodes.

(2) Netflix Movies and TV Shows

This dataset is about Netflix's Movies and TV shows. (original and non-original programming). The dataset is collected from Flexible which is a third-party Netflix search engine.
URL: https://www.kaggle.com/shivamb/netflix-shows
Dataset type: 1 csv file

The content is below:

- show_id - Unique ID of every Movie/Tv show.
- type - identifier(whether it's a movie or tv shows).
- title - Name of the series.
- director - Director of the series.
- cast - Actors involved in the series.
- country - Country where the series was produced
- dated_added - Date it was added on Netflix
- release_year - Actual Release year of the series.
- rating - TV rating of the series.
- duration - Total number of seasons.

2. How to extract the data

Click the url link below each database source and click download button, these two files will save in the local computer.

Screenshot - viewing file (1)&(2) in local computer

3. Data Cleansing

Data cleaning is a very essential process involved within any project. Before reading the raw data into Jupyter Notebook, it is suggested to view the data via Excel or any other viewing platform to get a basic understanding.
From the file (2), there are 'nan' cells in some of the data columns and this could be eliminated by Pandas dataframe function '.dropna()'. However, in file (1), there are 2 main issues which could not be solved directly:

a. Duplicated column title labels in rows;
b. nan value cells are marked as 'unknown'

	A	B	C	D	E	F
1	Title	Genre	Premiere	No_of_Seasons	No_of_	isodes
127	Awaiting release	Awaiting Release	Awaiting release		unknown	
210	Title	Genre	Premiere	No_of_Seasons	No_of_Episodes	
493	Title	Genre	Premiere	No_of_Seasons	No_of_Episodes	
643						
644						

Screenshot - viewing file (1) via Excel

There are multiple ways to fix, the one that has been applied is to firstly drop nan values by using pandas '.dropna()' function. The second step is to apply a '.loc' function with multiple conditions to get rid of these unwanted data.

```
#clean up data, note that there is only one null data but there are invalied strings such as 'Awaiting R/release', 'unknown'
# and 2 rows are duplicate as the column titles
dfa = dfa.dropna()
```

```
dfb = dfa.loc[(dfa['Title'] != 'Title') & (dfa['No_of_Episodes'] != 'unknown')]

dfb
```

	Title	Genre	No_of_Seasons	No_of_Episodes
0	Stranger Things	Science Fiction Horror	3	25
1	The Crown	Historical Drama	4	40
2	Ozark	Crime Drama	3	30
3	Lost in Space	Science Fiction	2	20
4	Narcos: Mexico	Crime Drama	2	20
...	...	...	...	...
636	The Last Narc	Drug Documentary	1	4
637	All or Nothing: Tottenham Hotspur	Sports Documentary	1	9
638	Fernando	Sports Documentary	1	5
639	El Desafío: ETA	Docuseries	1	8
640	James May: Oh Cook!	Cooking Show	1	7

558 rows × 4 columns

Screenshot - codes to cleaning data in Jupyter Notebook

It is important to double check if the cleaned data or datasets are ready for further uses. By checking in Jupyter Notebook, file (1) has now 558 counts in each column from the original number of 641, where file (2) has now 2124 counts left in each column from the original 2410, after being divided into ‘TV Show’ as the type.

Data Transformation

There are 2 tables within the database. Table ‘shows_detail’ has 5 columns as ‘title’, the name of the TV show, ‘country’, the country that the show were produced, ‘publish_date’, the date in format of ‘Month D, Yr’ as in text type, ‘release_year’, the year when the show were firstly released and ‘description’, a brief introduction about the content of the show as in text. Table ‘shows_other’ has 4 columns as ‘title’, same as in the other table, ‘genre’, the type of the show in text, ‘no_of_seasons’ and ‘no_of_episodes’ to show the number of seasons and the number of episodes one show has.

Screenshot - ERD of existing database

Please note that there is no forgien key involved in the actual database, from the ERD is to show that the title will be used when joining the two tables together.

In order to transform data into the database, further data filtering and formatting will be performed in both two dataframes.

a. Select only columns that match with existing ones in the database

```
#select only interested data to match to tables in database
df_filtered = df_divd[['title','country','date_added','release_year','description']].copy()

df_filtered
```

```
#drop unwanted column
dfa = dfa.drop(['Premiere'],axis=1)

dfa
```

Screenshot - example methods to select target columns or drop unrequired ones

b. Renaming the columns to exactly match the column label in the existing database. i.e. change ‘date_added’ in file (2) to ‘publish_date’, as well as make the column labels to lowercase by using ‘Dataframe.columns.str.lower()’ function

c. Reset tables indexes and drop the original ones

```
#rename the table columns to match with the database, and reset index
df = df.rename(columns={'date_added':'publish_date'})

df1 = df.reset_index(drop=True)

df1
```

	title	country	publish_date	release_year	description
0	3%	Brazil	August 14, 2020	2020	In a future where the elite inhabit an island ...
1	46	Turkey	July 1, 2017	2016	A genetics professor experiments with a treatm...
2	1983	Poland, United States	November 30, 2018	2018	In this dark alt-history thriller, a naïve law...
3	1994	Mexico	May 17, 2019	2019	Archival video and new interviews examine Mexi...
4	SAINT SEIYA: Knights of the Zodiac	Japan	January 23, 2020	2020	Seiya and the Knights of the Zodiac rise again...
...	...	...	...	...	...
2119	Zig & Sharko	France	December 1, 2017	2016	Zig, an island-bound hyena, will do anything t...
2120	Zindagi Gulzar Hai	Pakistan	December 15, 2016	2012	Strong-willed, middle-class Kashaf and carefre...
2121	Zoids Wild	Japan	August 14, 2020	2018	A quest for freedom and legendary treasure beg...
2122	Zona Rosa	Mexico	November 26, 2019	2019	An assortment of talent takes the stage for a ...
2123	Zumbo's Just Desserts	Australia	October 31, 2020	2019	Dessert wizard Adriano Zumbo looks for the nex...

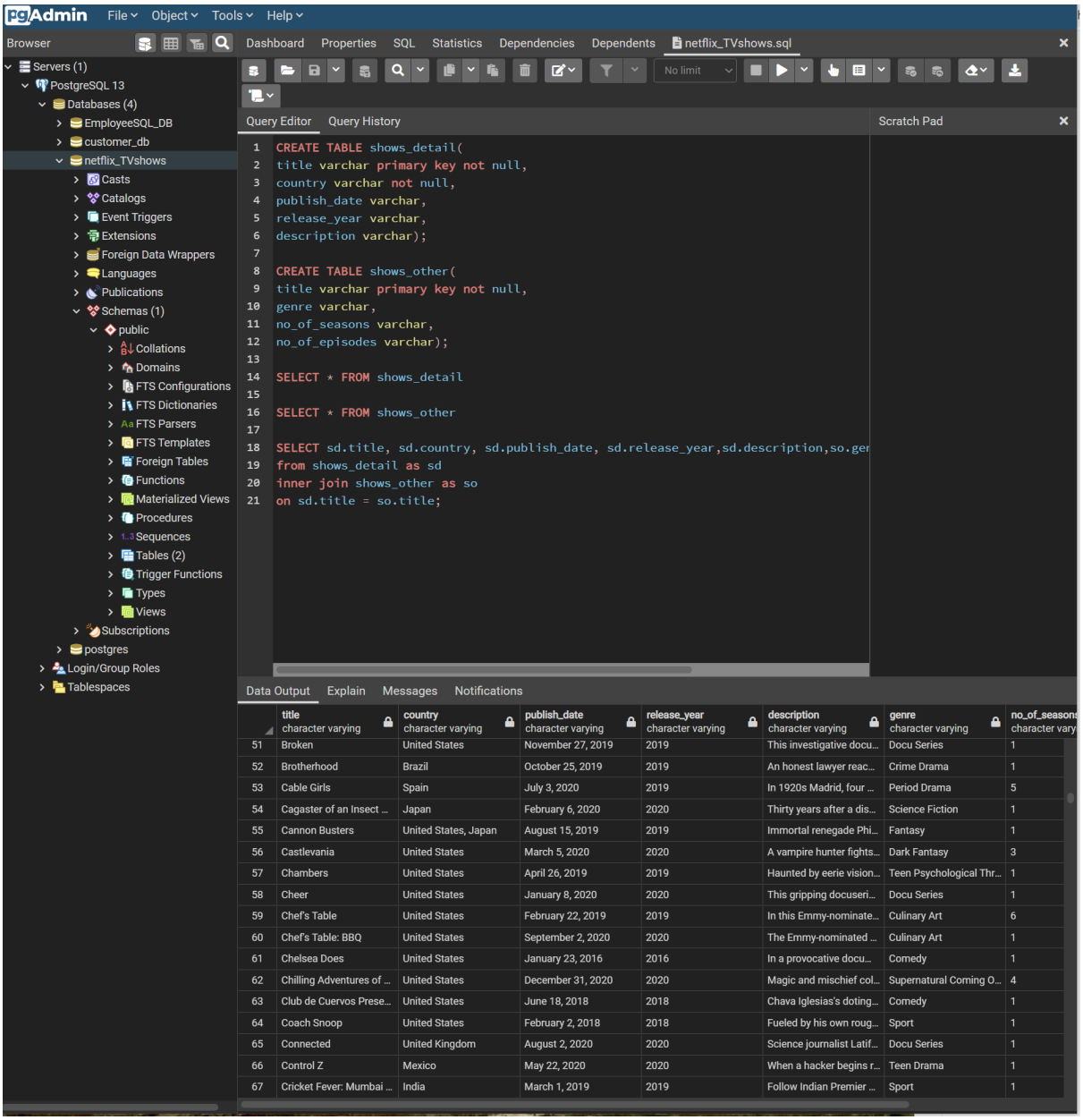
```
#change columns name to match in database and reset index
dfb.columns = dfb.columns.str.lower()

df2 = dfb.reset_index(drop=True)

df2
```

	title	genre	no_of_seasons	no_of_episodes
0	Stranger Things	Science Fiction Horror	3	25
1	The Crown	Historical Drama	4	40
2	Ozark	Crime Drama	3	30
3	Lost in Space	Science Fiction	2	20
4	Narcos: Mexico	Crime Drama	2	20
...	...	...	...	...
553	The Last Narc	Drug Documentary	1	4
554	All or Nothing: Tottenham Hotspur	Sports Documentary	1	9
555	Fernando	Sports Documentary	1	5
556	El Desafío: ETA	Docuseries	1	8
557	James May: Oh Cook!	Cooking Show	1	7

Screenshot - examples of rename or reformat and reset indexes

Data Loading

Within this project, a relational database, PostgreSQL will be used to store all datasets. First of all, the connection should be built from Jupyter Notebook to the local PostgreSQL database. This could be done by importing 'sqlalchemy' lib. Please note that the user needs to prepare a username and password pair within a pre-filled .py file as 'config.py'. Once the connection is arranged, test it by querying for table names from the database.
The next step is to assign the relevant dataframe to the existing tables from the database. It is suggested to remove the dataframes' index when doing so. They could be rebuilt after loading and for further data analysis processes.
It is also suggested to check the tables from both python and the SQL database.



Screenshot - Check tables and performing 'JOIN' within PgAdmin

Conclusions

Following the instructions in this report to obtain the dataset related with Netflix original programming.
Tools required: PGadmin4, Jupyter notebook, Excel, QuickDBD.
Languages required: Python & SQL.

References

1. https://medium.com/hashmapinc/etl-understanding-it-and-effectively-using-it-f827a5b3e54d
2. https://www.vistaprojects.com/blog/4-easy-sections-to-structure-engineering-reports-effectively/