Extract, Transform, Load Netflix Database

Our project is based on a three-phase process where our data is extracted from a source, transformed to illustrate the data in an organised way, and loaded into our chosen database. We have chosen two Netflix datasets, one pertains Netflix shows and movie titles, and the second one showcases the movie credits.

First Step

Create Netflix Database on pgAdmin.

Extract:

We found our two Netflix datasets (titles.csv and credits.csv) on Kaggle and we imported the two csv files into dataframes using Pandas.

Transform:

The second step of the project consisted of cleaning datasets to showcase it in presentable way.

On the credits dataframe we renamed the column 'id' to 'movie_id' as it was not clear what the column 'id' represented, and to facilitate merging the credit dataframe with the titles dataframe. The datasets were presented with some null values, however we decided not to drop all of those values because some columns had values showing as N/A (e.g., movies don't have seasons). We also decided not to drop duplicates due to having multiple actors working in the same movie or tv shows. We decided to merge the two final dataframes to create a joint dataframe. We merged on movie_id, and how = left. We noticed that the genre column and production country column in the merged dataframe had values that were arrays, thus to repaired them we used numpy. (See Figure 1)

In [7]:
```
# Cleaning merged titles_credits Pandas DataFrame getting rid of certain NaN column values and keeping the relavant ones
# eg NaN for movies which don't have seasons etc
def repair_array_bound_categories(arr):
    arr = ast.literal_eval(arr)

    if len(arr) == 0:
        return np.nan

    elif len(arr) == 1:
        return arr[0]

    else:
        return random.choice(arr)
```

In [8]:
```
# DataFrame titles_credits columns Production Countries and Genre showing as arrays repaired using NumPy
titles_credits['production_countries'] = titles_credits['production_countries'].apply(repair_array_bound_categories)
titles_credits['genres'] = titles_credits['genres'].apply(repair_array_bound_categories)
titles_credits.head()
```

Figure 1

Load:

The last step consisted of loading the dataframes into a database.

We created a database connection using create engine which we imported from sqlalchemy. We created the connection from Pandas to postgres sql (pgAdmin) onto our Netflix database which we created prior on pgAdmin.

We loaded our three dataframes onto Netflix database, using to.sql, and saved our engine connection under the variable engine.

We set the index as true, in order to use the index as primary key. The movie_id and the person_id columns had duplicated values; therefore, we could not use them as primary keys as they did not have unique values.

After we finished loading the tables we confirmed that the tables were created in the Netflix database by getting the table names using engine.table_names().

Final Step

After we finished the loading step, we went to our Netflix Database that we created on pgAdmin and performed a query to confirm that our database was successfully loaded. By performing these commands on pgAdmin: SELECT * FROM titles and SELECT * FROM credits and SELECT * FROM titles_credits.

Finally, we saved the cleaned and merged DataFrame (titles_credits) as csv_file in the Resources folder.