12/19/2020

NetFlix Data

ETL Project report

1. Ahmad Abu Alafa
2. Hamid Zarringhalam
3. Loic Tiemani
4. Hazim Hamadneh

Contents

[Introduction 2](#_Toc59313217)

[Extract Method 2](#_Toc59313218)

[NetFlix CSV output data 5](#_Toc59313219)

[This is OMDB API extract: 6](#_Toc59313220)

[This is TMDB extracted fields 8](#_Toc59313221)

[Transform Method: 10](#_Toc59313222)

[Data Cleaning 12](#_Toc59313223)

[Database tables designed 15](#_Toc59313224)

[**ERD Diagram:** 16](#_Toc59313225)

[SQL Server identified and connected 16](#_Toc59313226)

[Loading Method 22](#_Toc59313227)

[**“Title” Table:** 24](#_Toc59313228)

[**“Director” Table:** 25](#_Toc59313229)

[**“Director\_title” Table:** 26](#_Toc59313230)

[**“Country” Table:** 26](#_Toc59313231)

[**“Title\_Country” Table:** 27](#_Toc59313232)

[**“Listed\_in” Table:** 28](#_Toc59313233)

[**“Listed\_in\_title” Table:** 28](#_Toc59313234)

[**“PG\_Rating” Table:** 29](#_Toc59313235)

[**“Players” Table:** 30](#_Toc59313236)

[**“Player\_Title” Table:** 31](#_Toc59313237)

[Final Database 32](#_Toc59313238)

[Query Examples: 33](#_Toc59313239)

# Introduction

In this project which is the continuation of Project-1, Netflix Data analysis. It is required to identify the source of Data sets, perform Extraction, Transformation and Loading (ETL) on these data sets.

ETL comprises of three methods:

1. Extraction
2. Transformation
3. Loading

# Extract Method

Data extraction includes the following tasks:

1. Extract relevant data from data sources:

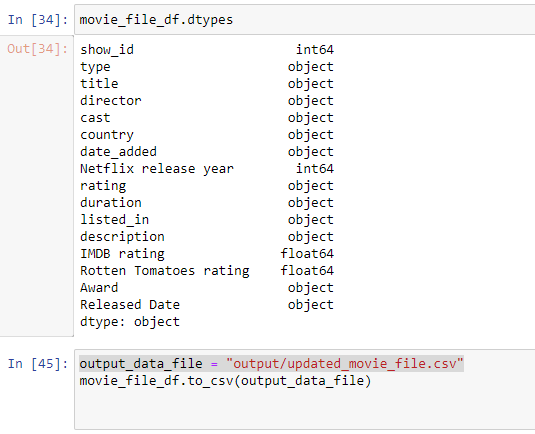
* <https://www.kaggle.com/shivamb/netflix-shows?select=netflix_titles.csv>

The result of extracted data is in the form of CSV and it has the following fields:

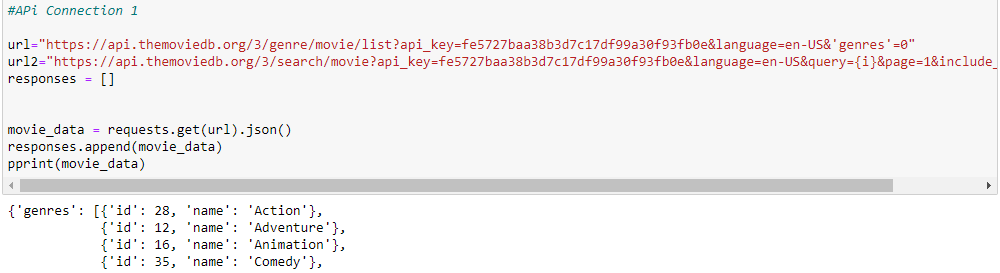


* Public movie APIs:
  + 1. OMDB - OMDB to obtain information and metadata about movies.

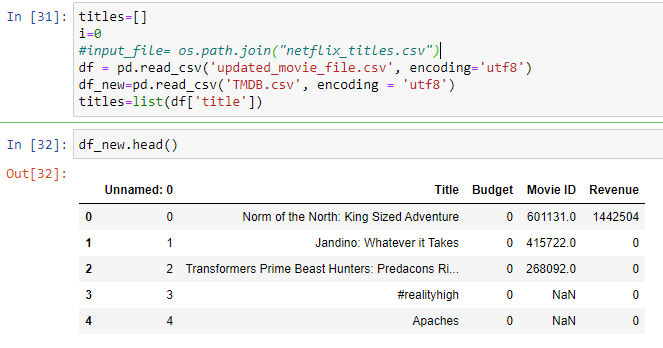




* + 1. TMDB - TMDB for movie discovery.



1. Reconcile records with the source data

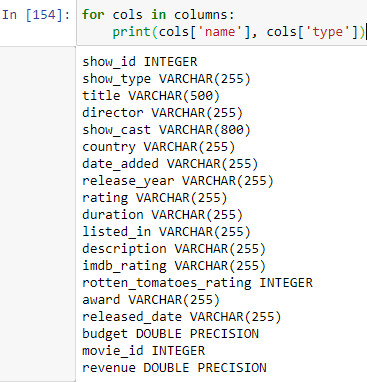


1. Make sure that no spam/unwanted data loaded





1. Data type check



Data types of all the fields were checked and identified as below:

## NetFlix CSV output data

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Field Name:** | **Description:** | **Example:** | **Data Type:** | **Unique:** | **Null:** | **Comments:** |
| show\_id | Unique ID for every Movie / Tv Show | 81145628 | String | Yes | No |  |
| type | Identifier - A Movie or TV Show | Movie | String | No | No |  |
| title | Title of the Movie / Tv Show | Norm of the North: King Sized Adventure | String | No | No |  |
| director | Director of the Movie | Richard Finn, Tim Maltby | String | No | yes | Multiple names in each field |
| cast | Actors involved in the movie / show | Alan Marriott, Andrew Toth, Brian Dobson, Cole Howard, Jennifer Cameron, Jonathan Holmes, Lee Tockar... | String | No | Yes | Multiple names in each field |
| country | Country where the movie / show was produced | United States, India, South Korea, China | String | No | yes | Multiple names in each field |
| date\_added | Date it was added on Netflix | September 9, 2019 | Date | No | yes |  |
| release\_year | Actual Release year of the move / show  text\_formatratingsort  TV Rating of the movie / | 2019 | Integer | No | No |  |
| rating | TV Rating of the movie / show | TV-PG | String | No | Yes |  |
| duration | Total Duration - in minutes or number of seasons | 90 min | String | No | No |  |
| listed\_in | Genre | Children & Family Movies, Comedies | String | No | No | Multiple names in each field |
| description | The summary description | Before planning an awesome wedding for his grandfather, a polar bear king must | String | No | No |  |

## This is OMDB API extract:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Field Name: | Description: | Example: | Data Type: | Unique: | Null: | Comments: |
| Title | Title of the Movie / Tv Show | Man of Steel | String | No | No |  |
| Year | Actual Release year of the move / show  text\_formatratingsort  TV Rating of the movie / | 2013 | Integer | No | No |  |
| Rated | TV Rating of the movie / show | PG-13 | String | No | Yes |  |
| Released | Release date | 14 Jun 2013 | String | No | Yes |  |
| Runtime | Duration | 143 min | String | No | Yes |  |
| Genre | Type of movie | Action,Adventure, Sci-Fi | String | No | No |  |
| Director | Director | Zack Snyder | String | No | No |  |
| Writer | Author | David S. Goyer (screenplay)  David S. Goyer (story) | String | No | No |  |
| Actors | Actor and Actress | Henry Cavill  Amy Adams  Michael Shannon  Diane Lane | String | No | No |  |
| Plot | Description | An alien child is evacuated from his dying world and sent to Earth to live among humans. His peace is threatened  when other survivors of his home planet invade Earth | String | Yes | No |  |
| Language |  | English | String | No | No |  |
| Country | Country of producing the product | UA, UK | String | No | No |  |
| Awards | Received Awards for the product | 7 wins & 46 nominations | String | No | Yes |  |
| Poster | create by the studio used to promote or advertise the title | N/A | String | No | No |  |
| Source | IMDb registered users who casts a vote | N/A | String | No | Yes |  |
| Value | Accurate Percentage of the Rating of respected critics | 56% | Float | No | Yes |  |
| Source | Rating of critics | Rotten Tomatoes | String | No | Yes |  |
| Value | Percentage of the Rating of respected critics | 55/100 | Float | No | Yes |  |
| Metascore | Rating of respected critics | 55 | Int | No | Yes |  |
| imdbRating | Rating on IMDB | 7.0 | Int | No | Yes |  |
| imdbVotes | Number of votes for the product | 92 | Int | No | Yes |  |
| imdbID | IMDB Movie ID Number | tt0770828 | String | Yes | No |  |
| Type | Movie or TV-Series | movie | Float | No | No |  |
| DVD | If the product is available on DVD | N/A | String | No | Yes |  |
| BoxOffice | Box Office receives data from a variety of sources, including film studios, distributors, and production companies from around the world | N/A | String | No | Yes |  |
| Production | Production company | Syncopy | String | No | Yes |  |

## This is TMDB extracted fields

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Field Name: | Description: | Example | Data Type | Unique | Null | Comments |
|
| adult: TRUE/FALSE | Restricted for adult | Yes | Boolean | No | Yes |  |
| backdrop\_path: varchar(255) |  |  | String |  |  |  |
| genre\_ids: Numbers (series) | The type of movie, Comeddy,… | Action,Adventure, Sci-Fi | String | No | No |  |
| id: varchar (10) | The Movie or Show ID | 415722 | String | Yes | No |  |
| original\_language: char (2) | Original Language that the movie is produced | en | String | No | no |  |
| original\_title: varchar (255) | The original title that is chosen | Norm of the North: King Sized Adventure | String | No | No |  |
| overview: varchar (1500) | General view of summery | An ancient Chinese artifact has been stolen by a villainous archaeologist named Dexter. With the help of his lemming friends, Norm must keep his word and embark on a journey across the world to help recover the artifact for the people of China. | String | Yes | Yes |  |
| popularity: float | Popularity based on number of votes | 10 | Float | No | Yes |  |
| poster\_path: varchar(255) | create by the studio used to promote or advertise the title | N/A | String | No | No |  |
| release\_date: date | The Release date of the product | 14 June 2013 | String | no | yes |  |
| title: varchar (255) | The title of the movie ot TV-Series | Norm of the North: King Sized Adventure | String | No | No |  |
| video: TRUE/FALSE | On Video tape or not | True | Boolean | No | Yes |  |
| vote average: float | Min Average of Votes for the movie | 10 | Float | No | Yes |  |
| vote\_count: int (edited) | Number of Votes | 10 | Int | No | Yes |  |

# Transform Method:

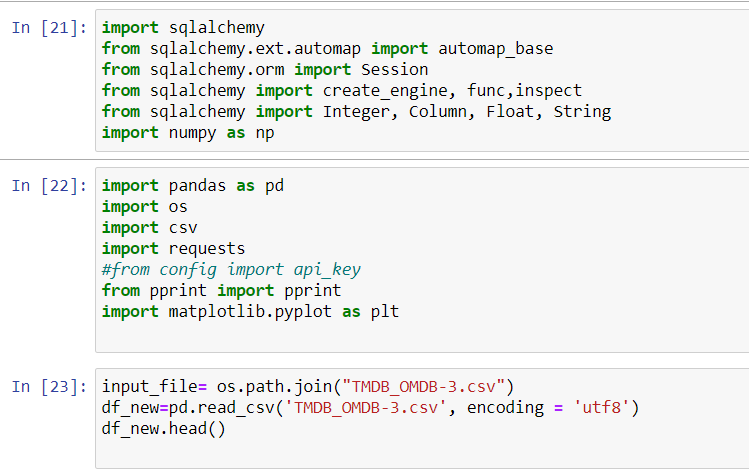
Data transformation includes the following tasks:

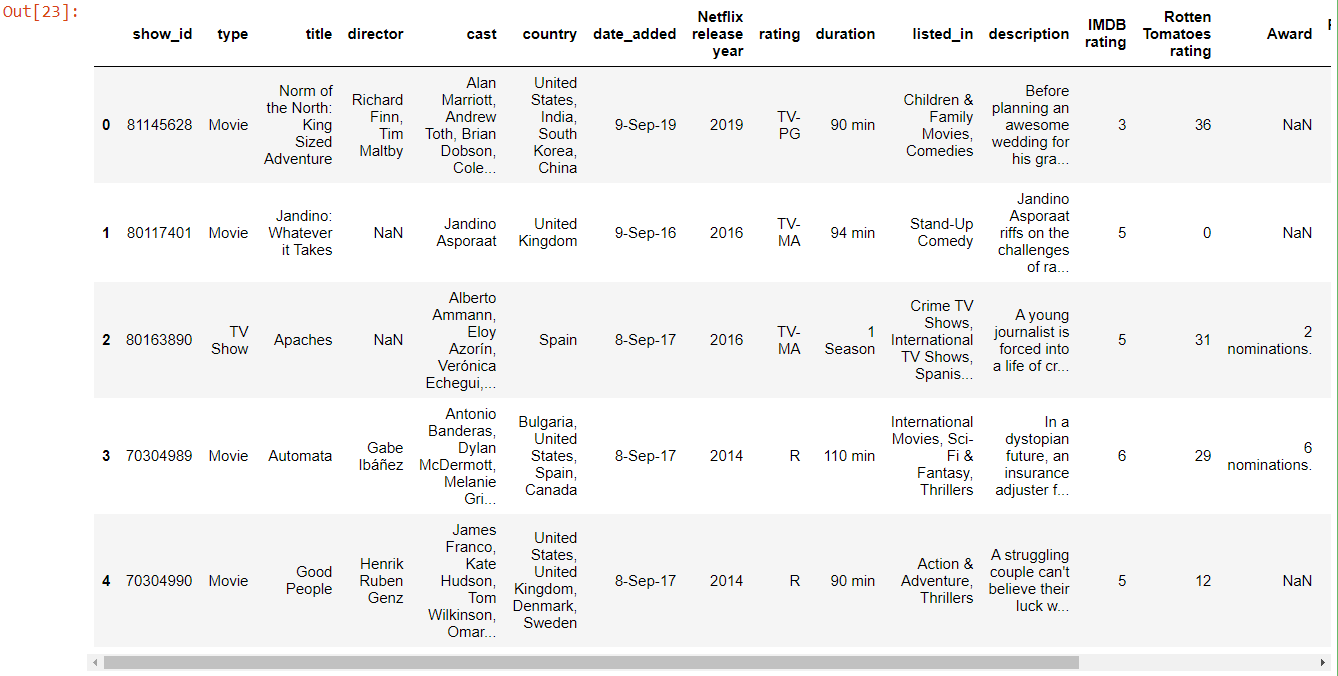
1. Data Aggregation of different sources to one:

Both API outputs and Netflix CSV merged into one CSV which conclude the following fields:









## Data Cleaning

There were some fields that are duplicate, others are no value and others have more than one information, such as Director, Cast, Country or listed in.

Example: 

In this process, we removed the duplicates, NaN and configured the proper indexing as well.



**OMDB-TMDB Table which is consolidated of the data from Netflix CSV, TMDB API and OMDB API**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Field Name:** | **Description:** | **Example** | **Data Type** | **Unique** | **Null** | **Comments** |
|
| show ID | ID number of the movie | 81145628 | Int | Yes | No |  |
| Show type | Type of product: Movie/TV Series | Movie | String | Yes | No |  |
| Title | Title of the Movie / Tv Show | Man of Steel | String | No | No |  |
| Date Added | Date added | 8-sep-2017 | String | No | Yes |  |
| Netflix release year | Release date | 14 Jun 2013 | String | No | Yes |  |
| rating | TV Rating of the movie / show | TV-PG | String | No | Yes |  |
| director | Director | Zack Snyder | String | No | No |  |
| cast | Actors involved in the movie / show | Alan Marriott, Andrew Toth, Brian Dobson, Cole Howard, Jennifer Cameron, Jonathan Holmes, Lee Tockar... | String | No | Yes | Multiple names in each field |
| duration | Total Duration - in minutes or number of seasons | 90 min | String | No | No |  |
| listed\_in | Genre | Children & Family Movies, Comedies | String | No | No | Multiple names in each field |
| description | The summary description | Before planning an awesome wedding for his grandfather, a polar bear king must | String | No | No |  |
| IMDB rating | Rating on IMDB | 7.0 | Int | No | Yes |  |
| Rotten Tomatoes rating | Rating of critics | Rotten Tomatoes | String | No | Yes |  |
| Award | Received Awards for the product | 7 wins & 46 nominations | String | No | Yes |  |
| Released Date | Release date | 14 Jun 2013 | String | No | Yes |  |
| Budget | The budget for the product | 150,000,000 | Int | No | Yes |  |
| Movie ID | The OMDB ID for the movie | 415722 | Int | yes | yes |  |
| Revenue | The income of the product | 750,000,000 | Int | No | Yes |  |

## Database tables designed

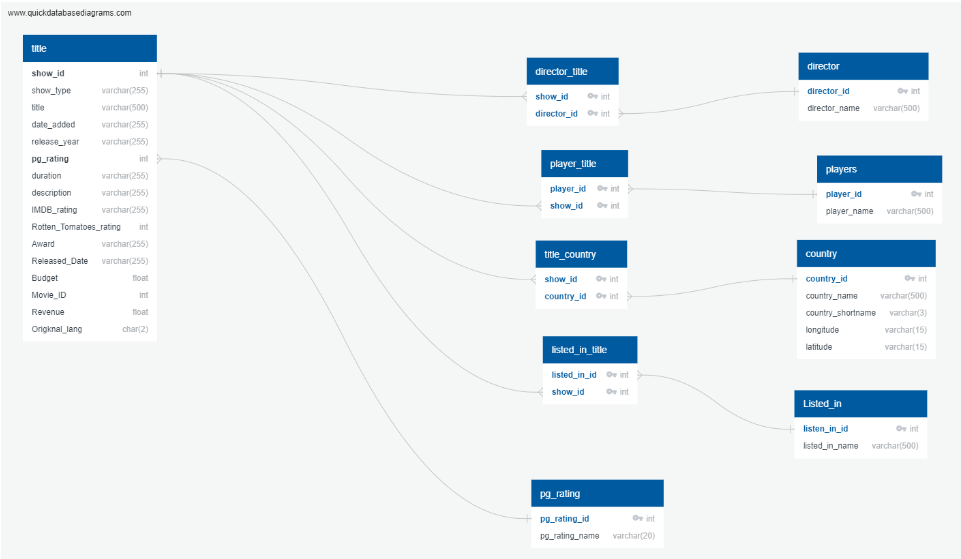
Main table is representing the show titles and its related properties.

This table has been normalized to first normal form by separating entities in different tables and relations have been created to connect each entity to its related data. For example: Titles and Players, Titles and Directors, etc.

Necessary tables and the relations between them was designed.

The following were considered to separate the related data from each other and have proper connections between the tables.

## ERD Diagram:



## SQL Server identified and connected

**Azure cloud** PostgreSQL standalone server selected and used



1. Tables created on the server

Table Schema:

CREATE TABLE "title" (

"show\_id" int NOT NULL,

"show\_type" varchar(255) NOT NULL,

"title" varchar(500) NOT NULL,

"date\_added" varchar(255) NOT NULL,

"release\_year" varchar(255) NOT NULL,

"pg\_rating" int NOT NULL,

"duration" varchar(255) NOT NULL,

"description" varchar(255) NOT NULL,

"IMDB\_rating" varchar(255) NOT NULL,

"Rotten\_Tomatoes\_rating" int NOT NULL,

"Award" varchar(255) NOT NULL,

"Released\_Date" varchar(255) NOT NULL,

"Budget" float NOT NULL,

"Movie\_ID" int NOT NULL,

"Revenue" float NOT NULL,

"Origknal\_lang" char(2) NOT NULL

);

CREATE TABLE "title\_country" (

"show\_id" int NOT NULL,

"country\_id" int NOT NULL,

CONSTRAINT "pk\_title\_country" PRIMARY KEY (

"show\_id","country\_id"

)

);

CREATE TABLE "country" (

"country\_id" int NOT NULL,

"country\_name" varchar(500) NOT NULL,

"country\_shortname" varchar(3) NOT NULL,

"longitude" varchar(15) NOT NULL,

"latitude" varchar(15) NOT NULL,

CONSTRAINT "pk\_country" PRIMARY KEY (

"country\_id"

)

);

CREATE TABLE "Listed\_in" (

"listen\_in\_id" int NOT NULL,

"listed\_in\_name" varchar(500) NOT NULL,

CONSTRAINT "pk\_Listed\_in" PRIMARY KEY (

"listen\_in\_id"

)

);

CREATE TABLE "listed\_in\_title" (

"listed\_in\_id" int NOT NULL,

"show\_id" int NOT NULL,

CONSTRAINT "pk\_listed\_in\_title" PRIMARY KEY (

"listed\_in\_id","show\_id"

)

);

CREATE TABLE "director" (

"director\_id" int NOT NULL,

"director\_name" varchar(500) NOT NULL,

CONSTRAINT "pk\_director" PRIMARY KEY (

"director\_id"

)

);

CREATE TABLE "director\_title" (

"show\_id" int NOT NULL,

"director\_id" int NOT NULL,

CONSTRAINT "pk\_director\_title" PRIMARY KEY (

"show\_id","director\_id"

)

);

CREATE TABLE "players" (

"player\_id" int NOT NULL,

"player\_name" varchar(500) NOT NULL,

CONSTRAINT "pk\_players" PRIMARY KEY (

"player\_id"

)

);

CREATE TABLE "player\_title" (

"player\_id" int NOT NULL,

"show\_id" int NOT NULL,

CONSTRAINT "pk\_player\_title" PRIMARY KEY (

"player\_id","show\_id"

)

);

CREATE TABLE "pg\_rating" (

"pg\_rating\_id" int NOT NULL,

"pg\_rating\_name" varchar(20) NOT NULL,

CONSTRAINT "pk\_pg\_rating" PRIMARY KEY (

"pg\_rating\_id"

)

);

ALTER TABLE "title" ADD CONSTRAINT "fk\_title\_pg\_rating" FOREIGN KEY("pg\_rating")

REFERENCES "pg\_rating" ("pg\_rating\_id");

ALTER TABLE "title\_country" ADD CONSTRAINT "fk\_title\_country\_show\_id" FOREIGN KEY("show\_id")

REFERENCES "title" ("show\_id");

ALTER TABLE "title\_country" ADD CONSTRAINT "fk\_title\_country\_country\_id" FOREIGN KEY("country\_id")

REFERENCES "country" ("country\_id");

ALTER TABLE "listed\_in\_title" ADD CONSTRAINT "fk\_listed\_in\_title\_listed\_in\_id" FOREIGN KEY("listed\_in\_id")

REFERENCES "Listed\_in" ("listen\_in\_id");

ALTER TABLE "listed\_in\_title" ADD CONSTRAINT "fk\_listed\_in\_title\_show\_id" FOREIGN KEY("show\_id")

REFERENCES "title" ("show\_id");

ALTER TABLE "director\_title" ADD CONSTRAINT "fk\_director\_title\_show\_id" FOREIGN KEY("show\_id")

REFERENCES "title" ("show\_id");

ALTER TABLE "director\_title" ADD CONSTRAINT "fk\_director\_title\_director\_id" FOREIGN KEY("director\_id")

REFERENCES "director" ("director\_id");

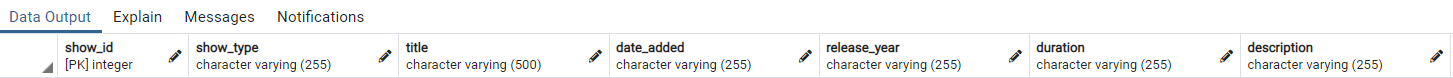
ALTER TABLE "player\_title" ADD CONSTRAINT "fk\_player\_title\_player\_id" FOREIGN KEY("player\_id")

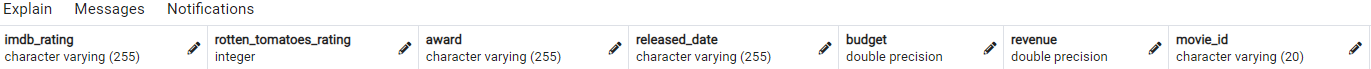
REFERENCES "players" ("player\_id");

ALTER TABLE "player\_title" ADD CONSTRAINT "fk\_player\_title\_show\_id" FOREIGN KEY("show\_id")

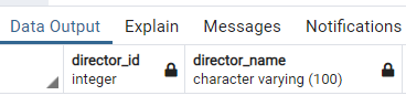
REFERENCES "title" ("show\_id");

**“Title” Table:**

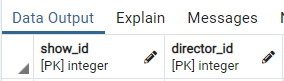




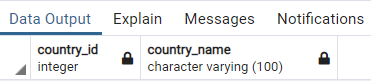
**“Director” Table:**



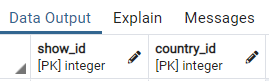
**“Director\_title” Table:**



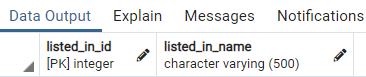
**“Country “Table:**



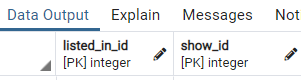
**“Title\_country” Table:**



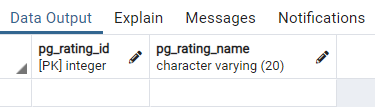
**“Listed\_in” Table:**



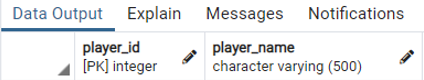
**“Listed\_in\_title” Table:**



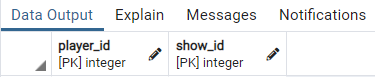
**“PG\_Rating” Table:**



**“Players” Table:**



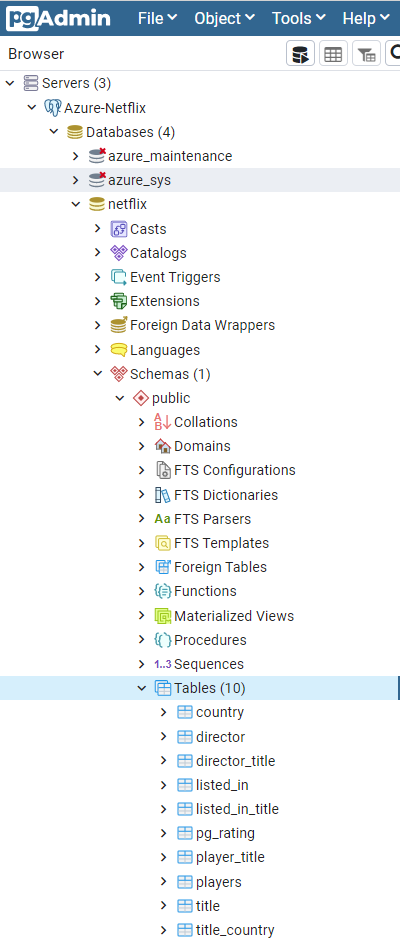
**“Players\_Title” Table:**



And as a result, we have multiple tables on the server as below:

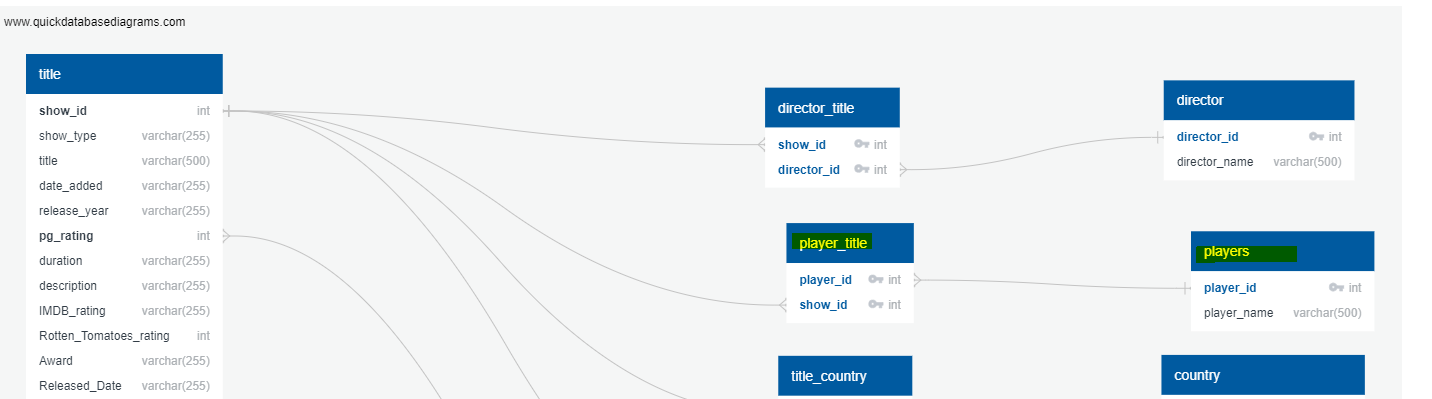
# Loading Method

After creating the tables, we imported the data to the tables based on the build design of the tables. Example of loading data into Title table:



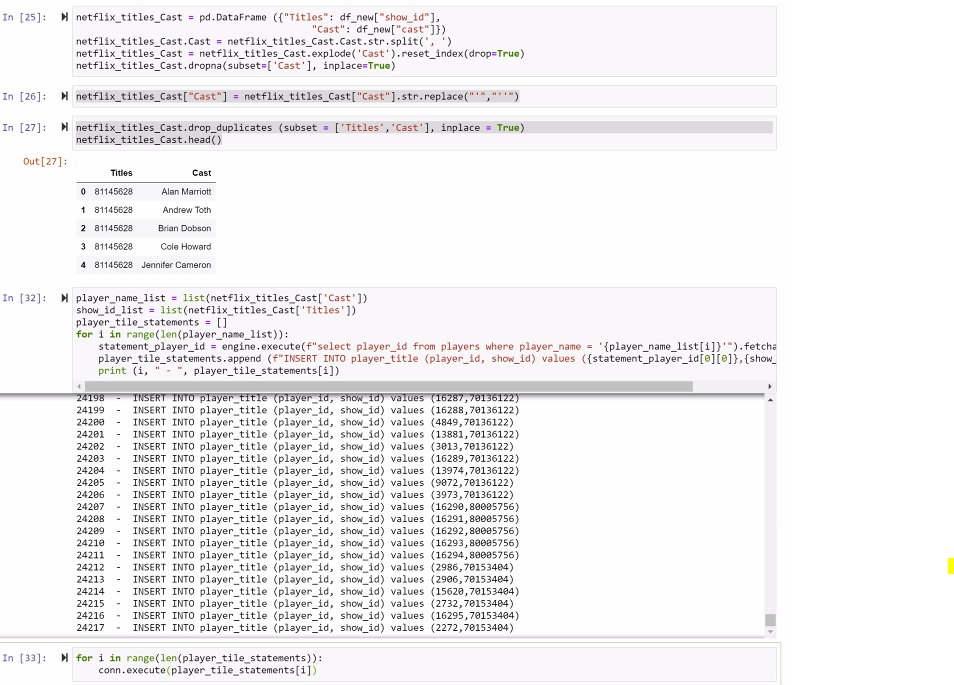


For tables that have Join table such as “Players” which has “Player\_title” as its join table. before importing the data to join table, we needed data imported to “Players” table first:

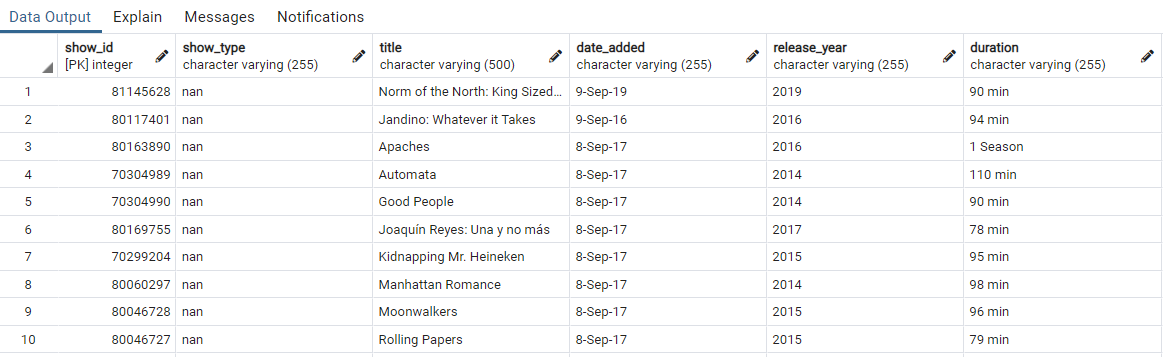


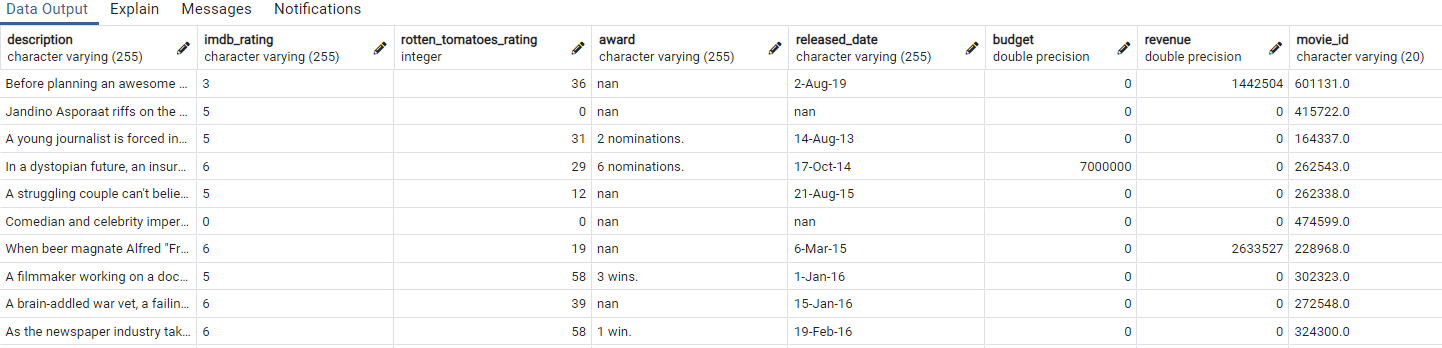
The table “player\_title” is used to resolve the M-M relationship.

Example of loading data from “Players” table to “Player\_title” table

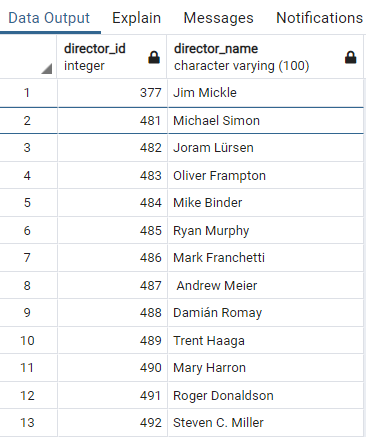


### **“Title” Table:**

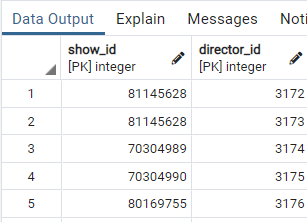




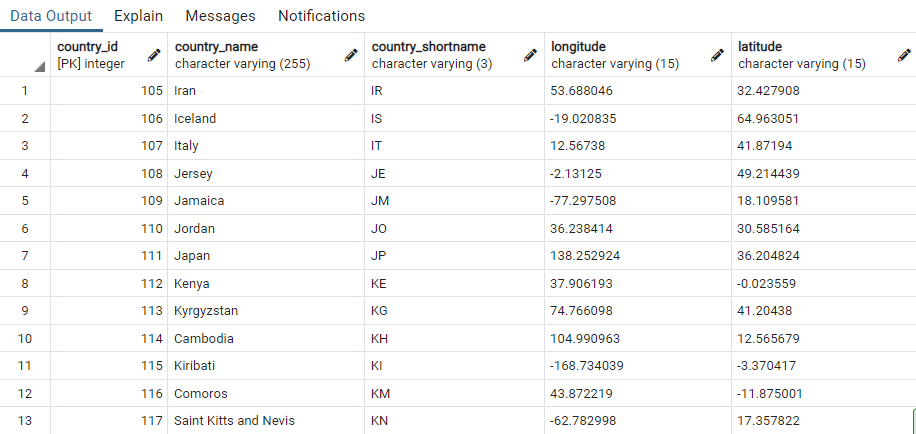
### **“Director” Table:**



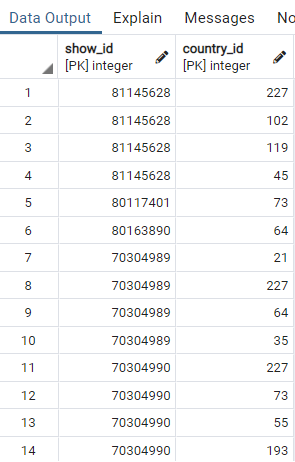
### **“Director\_title” Table:**



### **“Country” Table:**



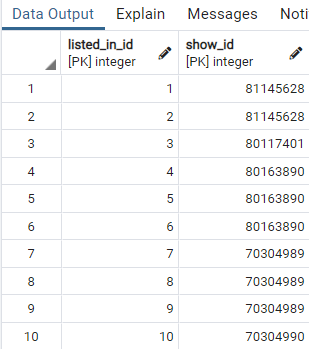
### **“Title\_Country” Table:**



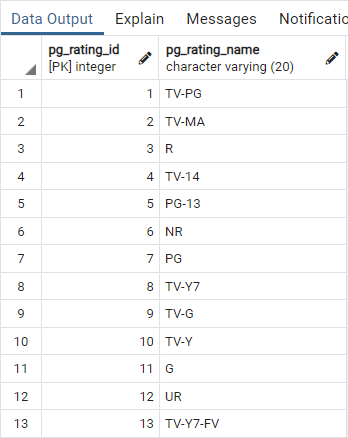
### **“Listed\_in” Table:**



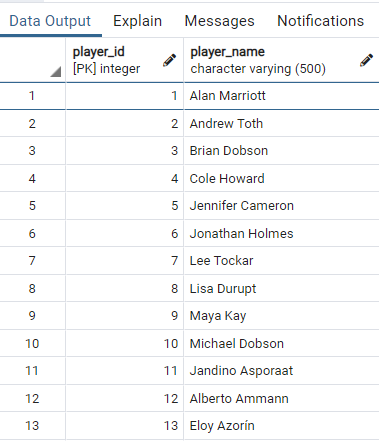
### **“Listed\_in\_title” Table:**



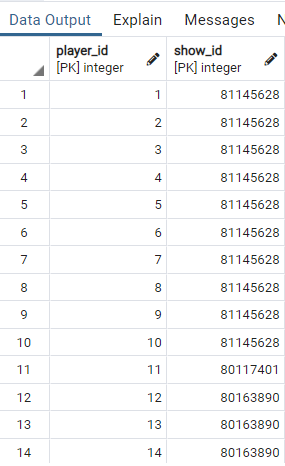
### **“PG\_Rating” Table:**



### **“Players” Table:**

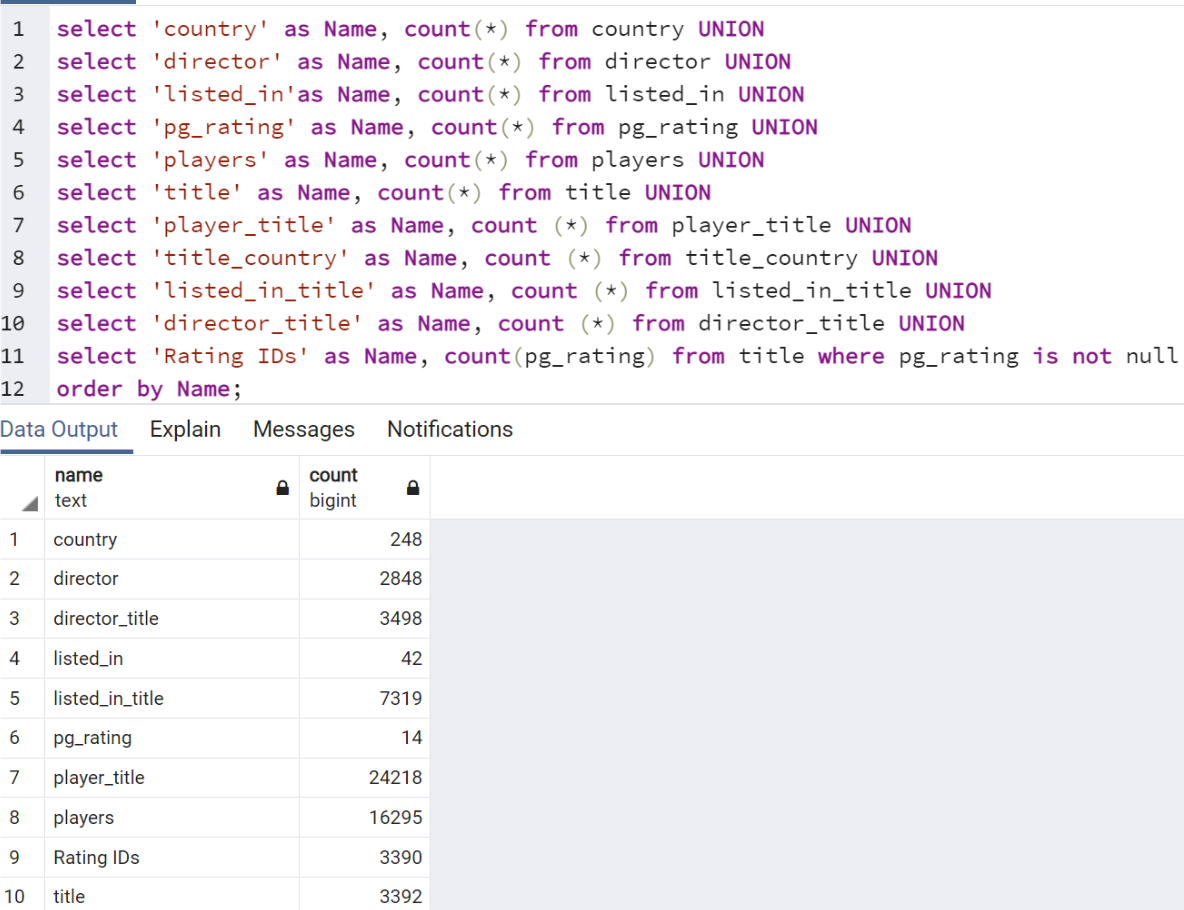


### **“Player\_Title” Table:**



# Final Database

The final database contains 10 tables with the specified relations. The data base contained more than 24000 records of Netflix data in total.



# Query Examples:

