## horizontal line



Books into Movies Analysis

10.19.2019

**─**

ETL PROJECT

ALYSSIA SMITH

LETICIA SALAZAR

EILISH BOYD

# TABLE OF CONTENTS

1. [TABLE OF CONTENTS](#_xz91j2lg2xqn)
2. [EXECUTIVE SUMMARY](#_au51mny0sx6)
3. [SOURCES OF DATA](#_3at9u9s4e0vp)
4. [DATA DICTIONARY](#_4p7xi5bvhxdr)
5. [DATA CLEANING](#_yyrhu7ml5bea)
6. [ERD DIAGRAM](#_3tfs0fjvm5v6)
7. [TABLE SCHEMA](#_30u5c7oqjs29)
8. [FIVE QUERIES WITH ANALYSIS](#_nh9ulbqnrk1i)

# EXECUTIVE SUMMARY

Many of the New York Times bestsellers are turned into books. We wanted to show the correlation between books produced to movies that were created based on the book. We used both of our datasets from Kaggle.com. The first dataset we used was data pulled from the Goodread API and placed onto kaggle, it contains various details about the books on their website. The second dataset we used was data pulled from TMDB API which included 5000+ movies with the title, producer and the release date.

**----------------------------------------------------------------------------------------------------------**

Overall Findings:

* Out of the 10,000 books analyzed, close to 520 were made into movies.
  + Within the last decade there were 152 books that were made into movies.
    - The earliest book “Beowulf” was published in the year 975 and the movie was released in 2007.
    - The latest movie was “Indignation” released in 2016, whose book was published in 2008.

**----------------------------------------------------------------------------------------------------------**

Using the ETL procedure we were able to conduct our analysis:

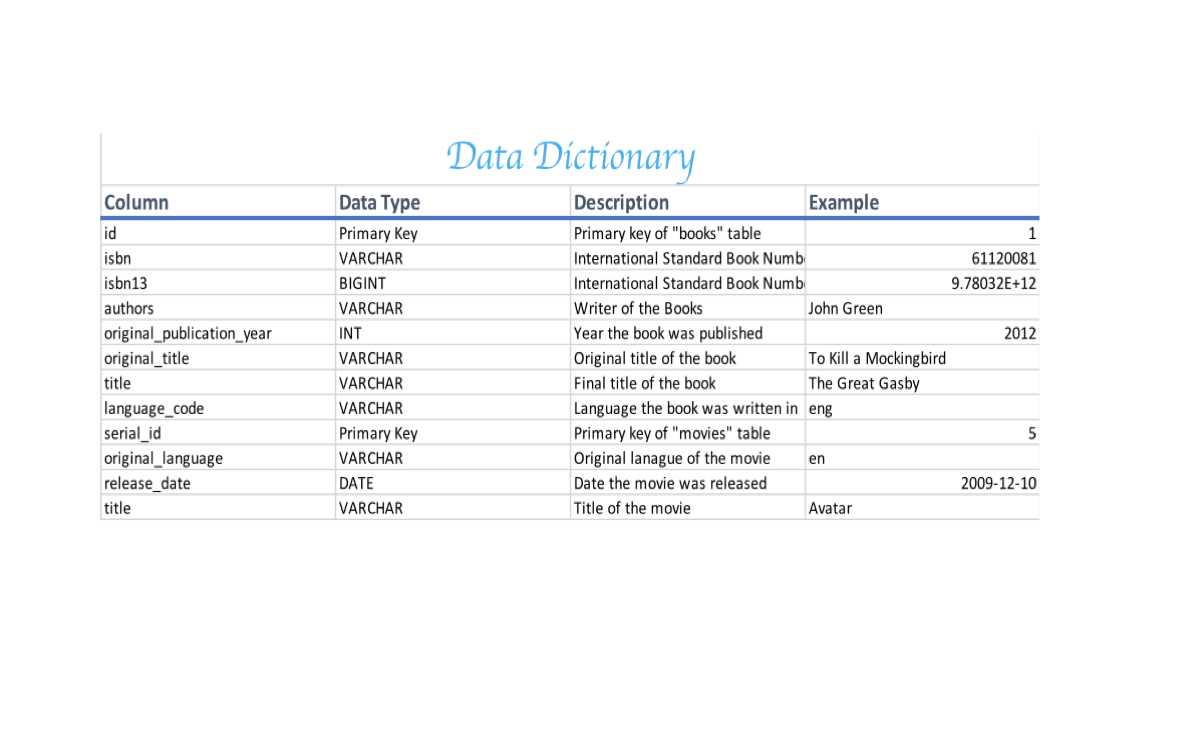
* **Extract:**
  + We looked for data sets that provided similar information on the books published and movie releases
    - For books we looked to reference “book\_id”, “title”, “author name” and “published date”
    - For movies we looked particularly for “title” and “release date”
* **Transform:**
  + We had to clean both data sets which mainly included renaming columns, dropping all columns not needed for the analysis, locate any null values and replacing them with the correct information or with an ‘unknown’ and sorting the columns to be more legible.
* **Load:** Using Postgres SQL we were able to create and join both cleaned tables into one and then proceed with our analysis.

# SOURCES OF DATA

1. Goodbooks-10k: <https://www.kaggle.com/zygmunt/goodbooks-10k#books.csv>
2. TMDB 5000 Movie Dataset: <https://www.kaggle.com/tmdb/tmdb-movie-metadata>

# DATA DICTIONARY

The data dictionary illustrates the type of data in our datasets:



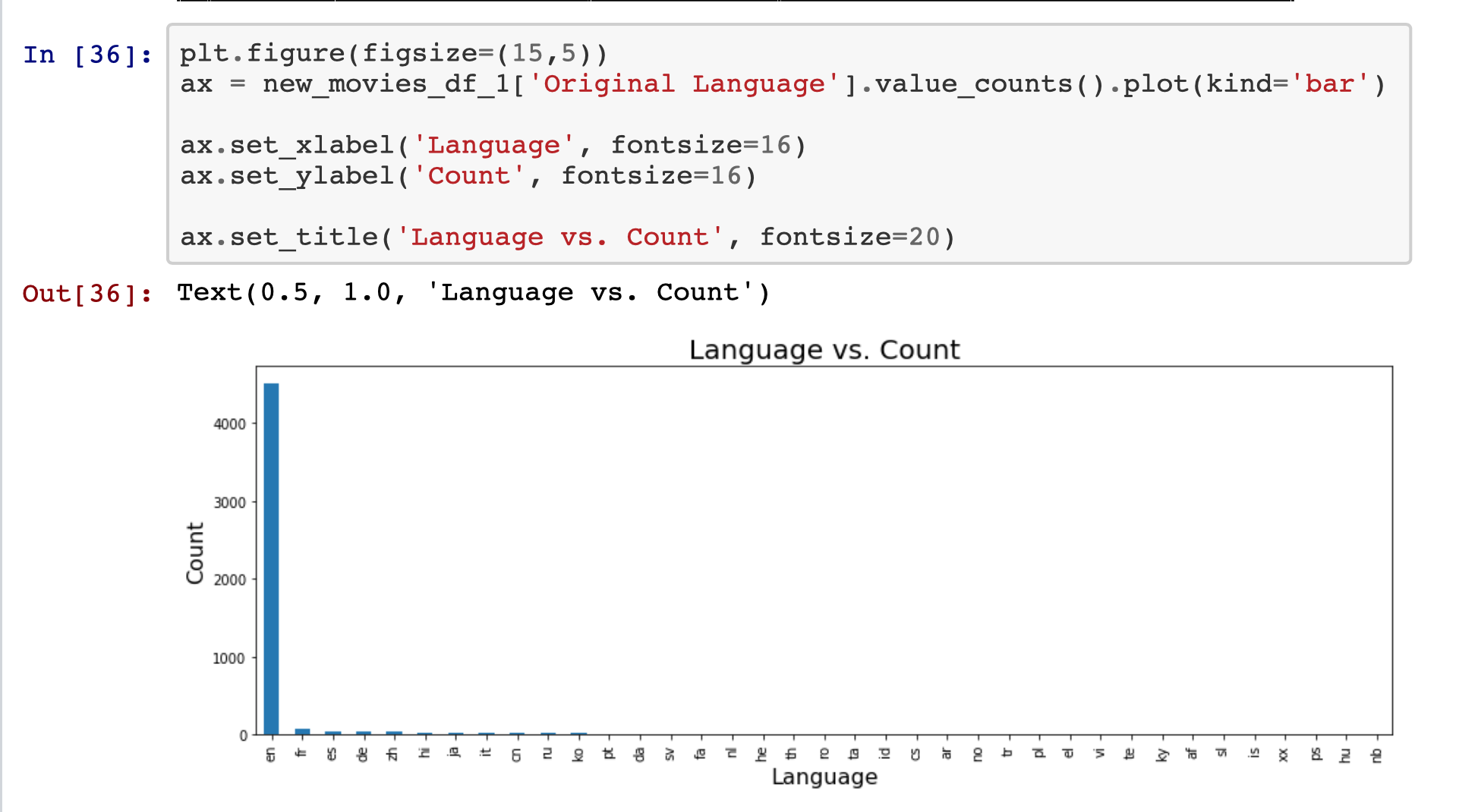
# DATA CLEANING

## Books Dataset

In our books dataset, we had many null values present. I wanted to count how many null values were present in each column so I created a matrix to create a visual.



## Movies Dataset

We wanted to see which columns made sense to drop based on the most common data present. 

# 

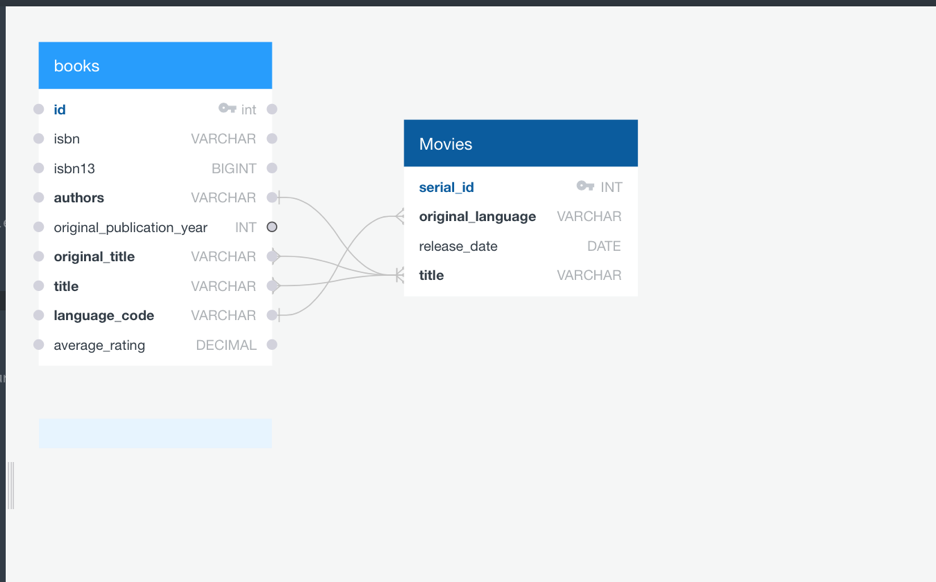
# 

# 

# 

# ERD DIAGRAM

The ERD Diagram shows the connections between both of our datasets.



# 

# TABLE SCHEMA

The table schema below shows how we were able to pull the data from both CSV files separately and then join the tables together to be able to analyze the results.

-- Drop table if exists

DROP TABLE IF EXISTS movies;

-- Create New Table

CREATE TABLE movies (

serial\_id INT PRIMARY KEY,

original\_language VARCHAR,

release\_date DATE,

title VARCHAR

);

SELECT \* FROM movies;

SELECT title

from movies;

SELECT release\_date

FROM movies;

SELECT \* FROM movies

WHERE release\_date >='2009-05-25'

AND release\_date <'2011-05-26'

SELECT \* FROM movies where serial\_id % 2 = 0;

SELECT \* FROM movies

WHERE release\_date >='2012-12-01'

AND release\_date <'2012-12-31'

-- Drop Table if exists

DROP TABLE books;

-- Create table

CREATE TABLE books (

id SERIAL PRIMARY KEY,

book\_id INT,

best\_book\_id INT,

work\_id INt,

books\_count INT,

isbn VARCHAR(10000),

isbn13 BIGINT,

authors VARCHAR(10000),

original\_publication\_year INT,

original\_title VARCHAR(10000),

title VARCHAR(10000),

language\_code VARCHAR(10000),

average\_rating DECIMAL,

ratings\_count INT,

work\_ratings\_count INT,

work\_text\_reviews\_count INT

);

SELECT \* FROM books;

SELECT book\_id, coalesce(book\_id,0)

FROM books;

SELECT isbn, coalesce(isbn, ‘Unknown’)

FROM books;

SELECT authors, coalesce(authors, ‘Unknown’)

FROM books;

SELECT title, coalesce(title, ‘Unknown’)

FROM books;

SELECT language\_code, coalesce(language\_code, ‘Unknown’)

FROM books;

--Join tables to display Title

SELECT \* from books;

SELECT \* from movies;

SELECT original\_title, authors, original\_publication\_year, release\_date

from books

INNER JOIN movies ON

movies.title=books.original\_title;

--Joined tables to display Authors

SELECT \* FROM (

SELECT original\_title, authors, original\_publication\_year, release\_date

from books

INNER JOIN movies ON

movies.title=books.original\_title

) combined1

WHERE authors IN(

SELECT authors FROM(

SELECT authors

FROM books, movies

WHERE movies.title=books.original\_title

) combined2

Group by authors

Having count (authors) > 1)

--Earliest Published Book

SELECT \* from books;

SELECT \* from movies;

SELECT original\_title, authors, original\_publication\_year, release\_date

from books

INNER JOIN movies ON

movies.title=books.original\_title

WHERE original\_publication\_year >='0'

AND original\_publication\_year <'2019'

--Latest Movie Release Date

SELECT \* from books;

SELECT \* from movies;

SELECT original\_title, authors, original\_publication\_year, release\_date

from books

INNER JOIN movies ON

movies.title=books.original\_title

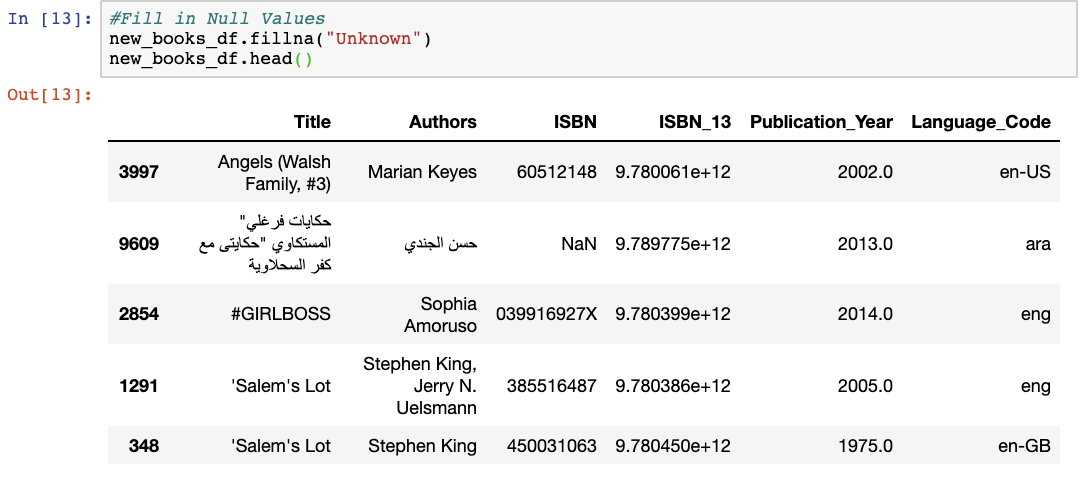
WHERE release\_date >='2016-01-01'

AND release\_date <'2016-12-31'

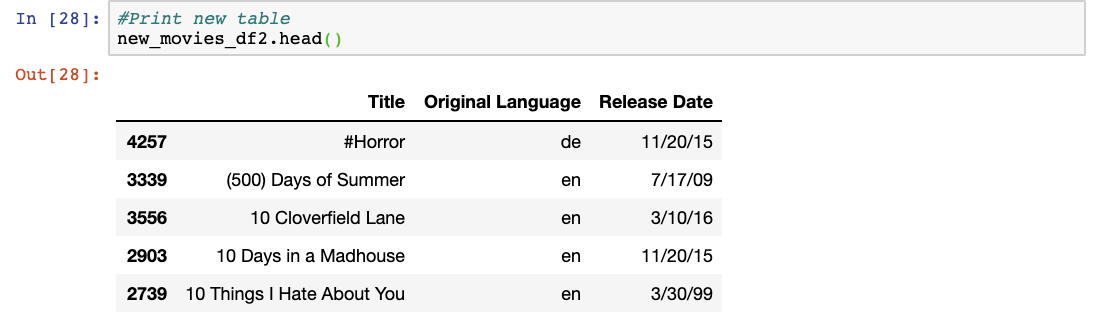
# QUERIES WITH ANALYSIS

## “Books and Movies CSV”

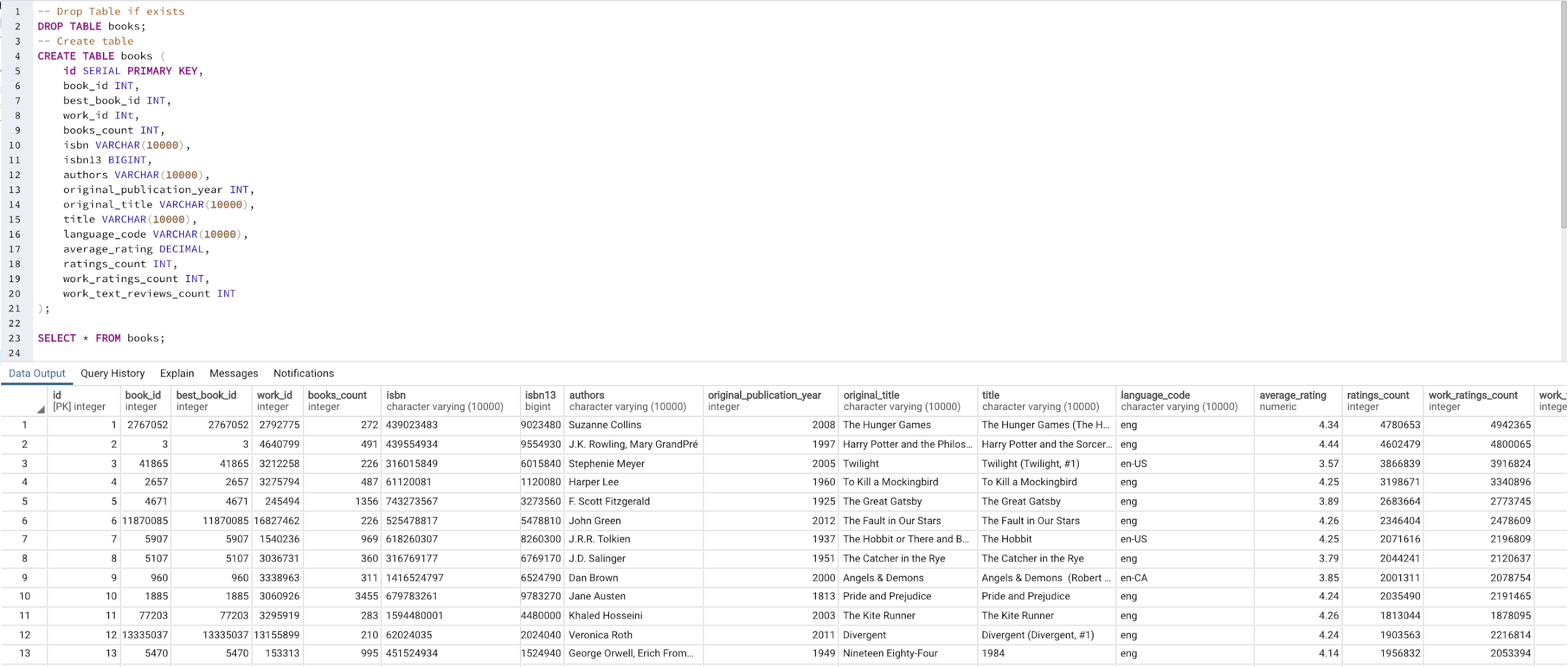
The Books CSV file cleaned and sorted with the Top 5 Books listed.

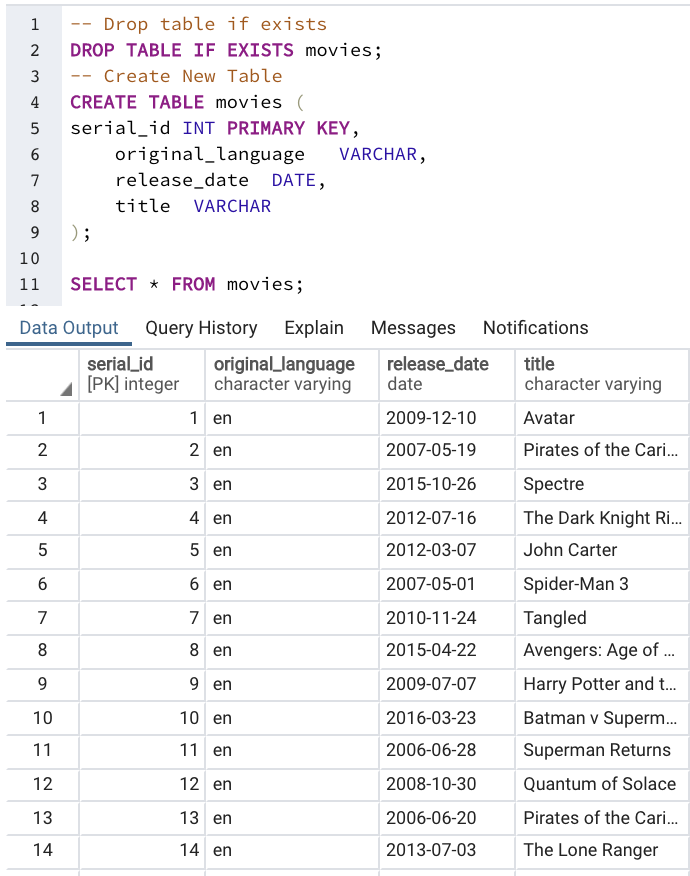


The Movies CSV file cleaned and sorted with the Top 5 Movies listed.

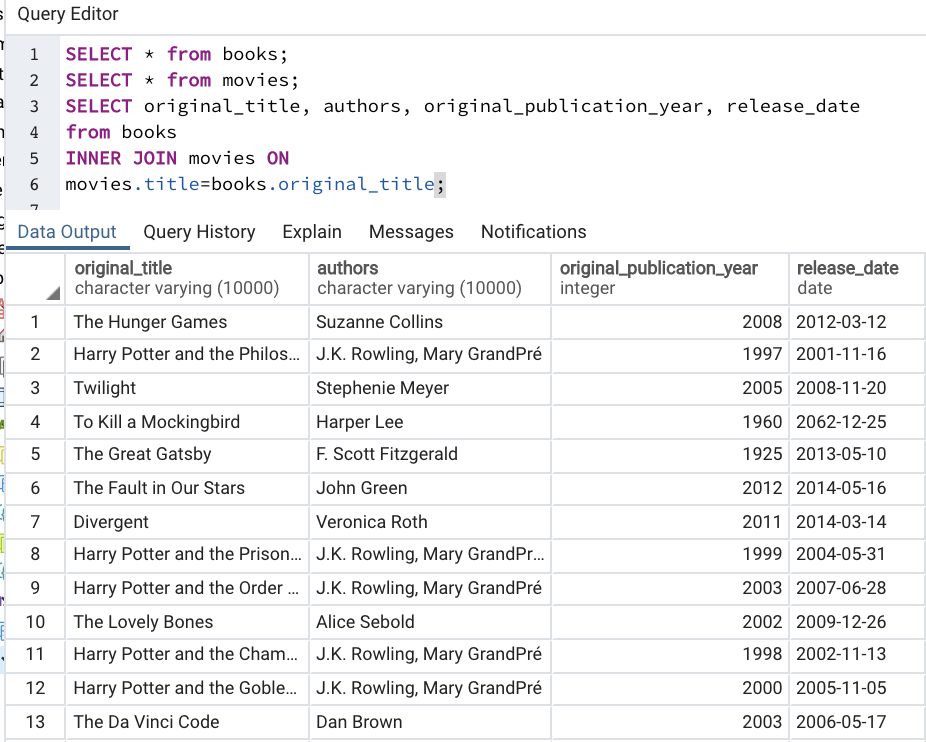


## “Books and Movies Query Table”

Using Postgres SQL, we were able to import the Books and Movies CSV file and create separate tables. Eventually these tables will be used to join the data based on Title and Author.



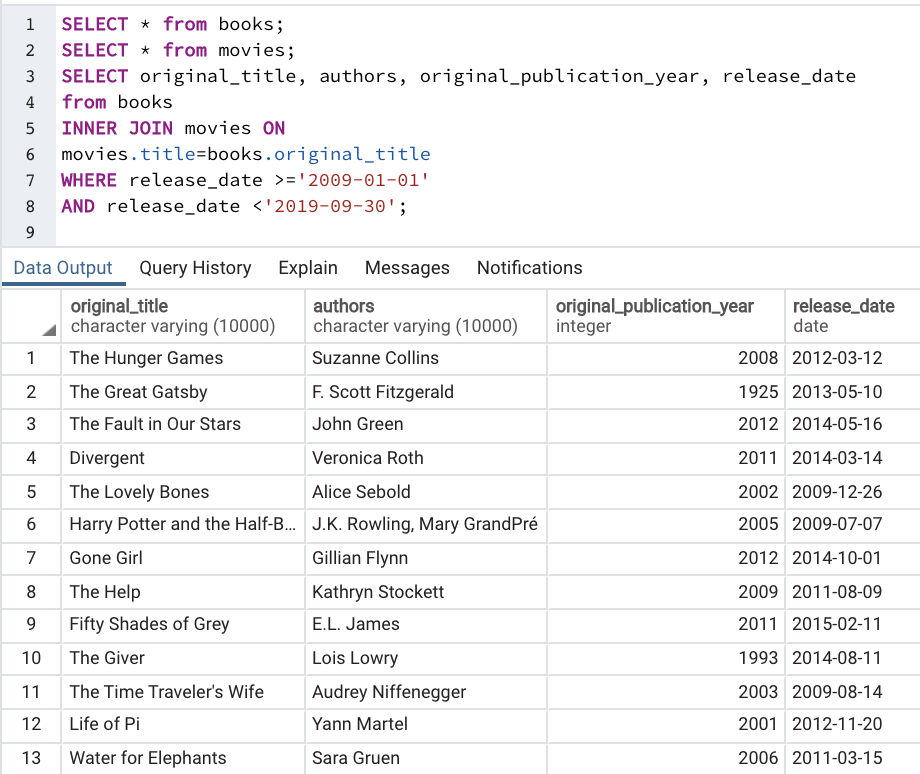
## “Joined Query Table”

Both Books and Movies tables joined together. The data below shows the titles and authors of books and movies inner joined together with the published year of the book and the release date of the movie. 

## 

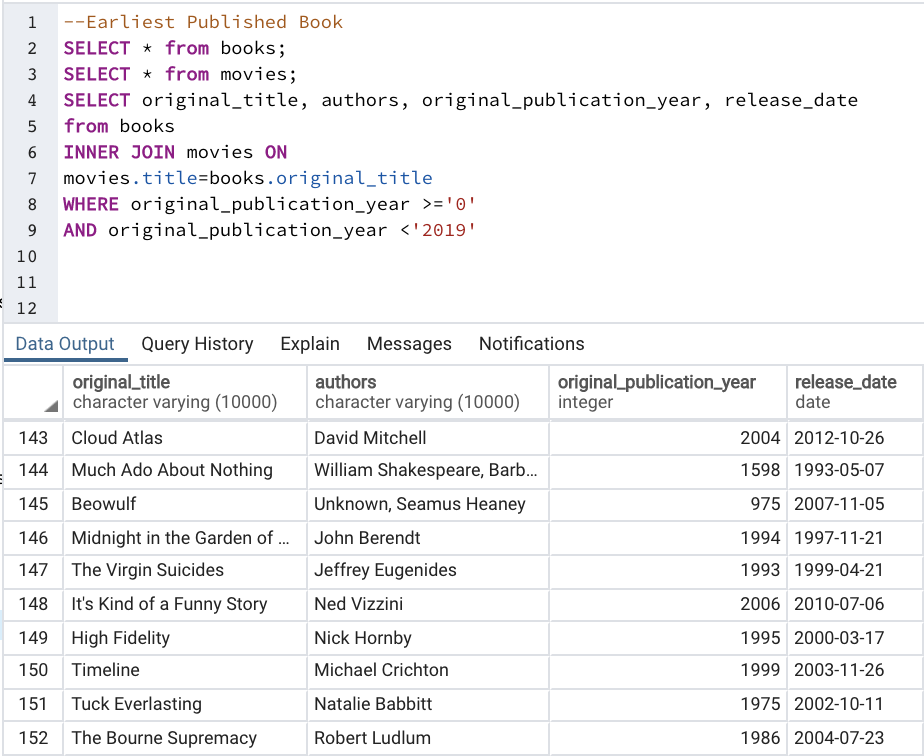
## “10 years of Books into Movies”

Showing from the years 2009 to 2019.



## “Earliest Book Published and Released into a Movie”

The earliest book was published in 975 and movie was released in 2007



## “Latest Movie Released from Published Book”

Latest movie released was in 2016.

