

Project Title: Exploratory Data Analysis on Film Data Using PostgreSQL

Prepared by Thomas, Nelson Chinenyeze

Date: 5th August 2024

Table of Contents

Title Page	i
Table of Contents	ii
Introduction	1
Methodology	2
Data Retrieval and Analysis	3
Task 1: Data Retrieval	3a
Task 2: Film Analysis	3b
Task 3: People and Roles Analysis	3c
Task 4: User Reviews and Votes Analysis	3d
Visualizations	4
Conclusion	5
References	6

1. INTRODUCTION

Project Description: In this project, I applied the SQL skills I have learned throughout the course to perform exploratory data analysis on the “film” database, which contains information about films, people, reviews, and roles. The project focused on retrieving, filtering, and analyzing data to answer real-life business questions related to the film industry.

Project Objectives:

- Apply SQL queries to retrieve and analyze data from the “film” database.
- Gain practical experience in using SQL for data analysis.
- Answer real-life business questions using SQL.
- Present findings in a clear and organized manner.

2. METHODOLOGY

Tools and Databases Used:

- PostgreSQL and PgAdmin4 for SQL queries and data retrieval.
- Excel for descriptive analysis.
- Tableau for data visualization and presentation.

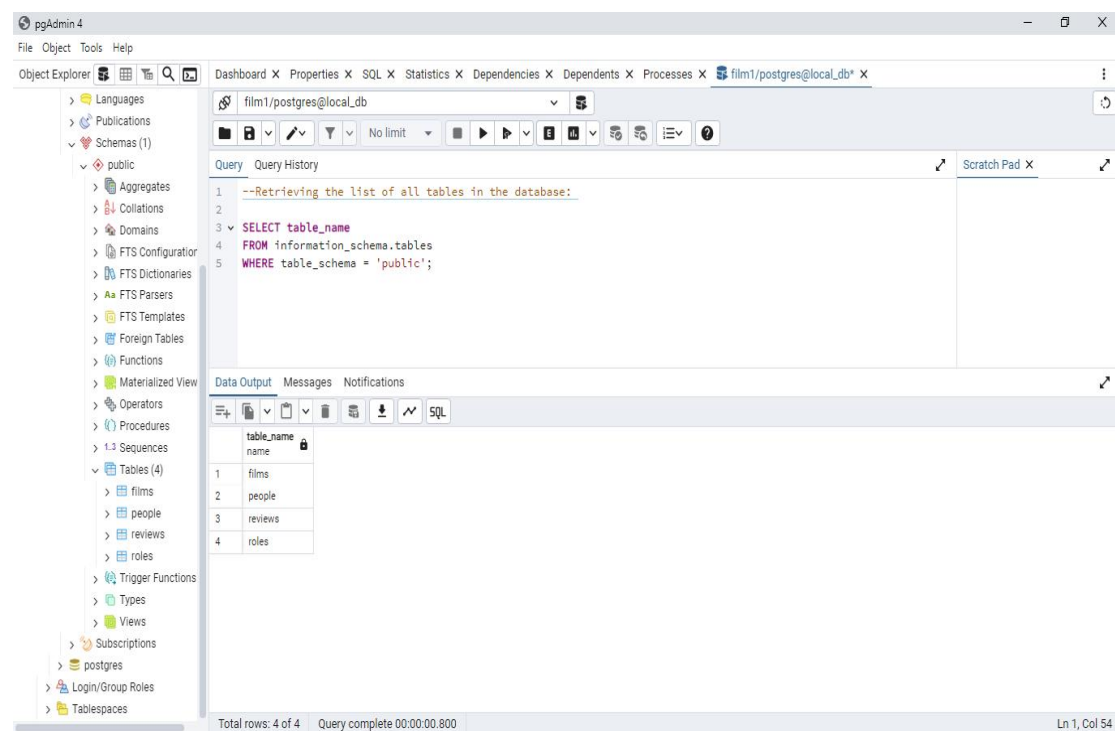
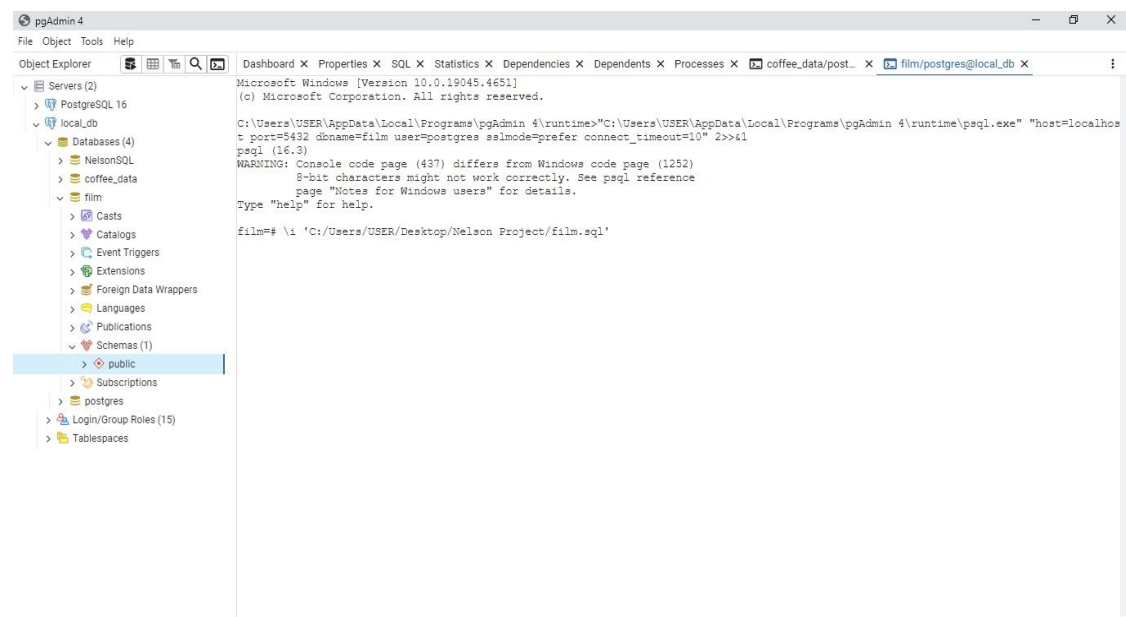
SQL Queries and Analysis Process: A series of SQL queries were used to extract data from the “film” database, followed by analysis to derive insights related to film performance, distribution, and audience engagement.

3. Data Retrieval and Analysis

Task 1: Data Retrieval

Here I connected to the “film” database using PostgreSQL and PgAdmin4 and I ensured the tables in the film database are related. Below images show the SQL code queries.

SQL Code:



Task 2: Film Analysis

In this section, I used SQL queries to retrieve, filter, and analyze data to answer real-life business questions related to the film industry. Below are the questions;

Question 2.1 : What are the top 10 highest-grossing films in the database, and when were they released?

SQL Code:

```
1  --What are the top 10 highest-grossing films in the database, and when were they released?
2
3  SELECT title, release_year, gross
4  FROM films
5  WHERE gross IS NOT NULL
6  ORDER BY gross DESC
7  LIMIT 10;
```

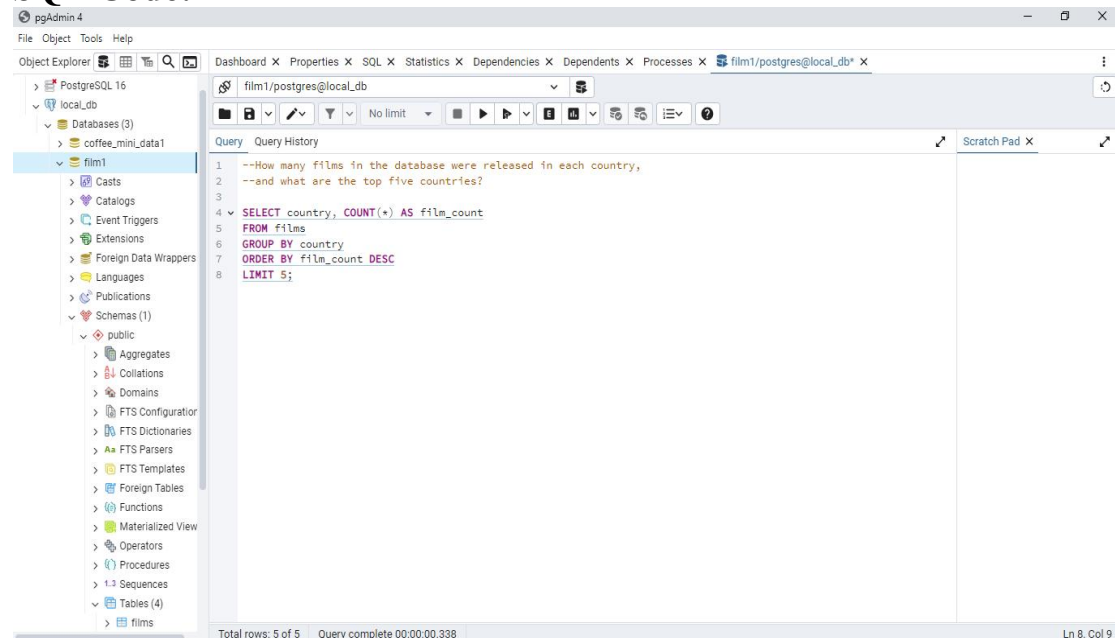
Results:

	title	release_year	gross
1	Star Wars: Episode VII - The Force Awakens	2015	936627416
2	Avatar	2009	760505847
3	Titanic	1997	68672302
4	Jurassic World	2015	652177271
5	The Avengers	2012	623279547
6	The Avengers	2012	623279547
7	The Dark Knight	2008	53516061
8	Star Wars: Episode I - The Phantom Menace	1999	474544677
9	Star Wars: Episode IV - A New Hope	1977	46935665
10	Avengers: Age of Ultron	2015	458991599

Analysis: The top 10 highest-grossing films are listed, with their release dates. This provides insights into the most financially successful films in the database.

Question 2.2: How many films in the database were released in each country, and what are the top five countries?

SQL Code:



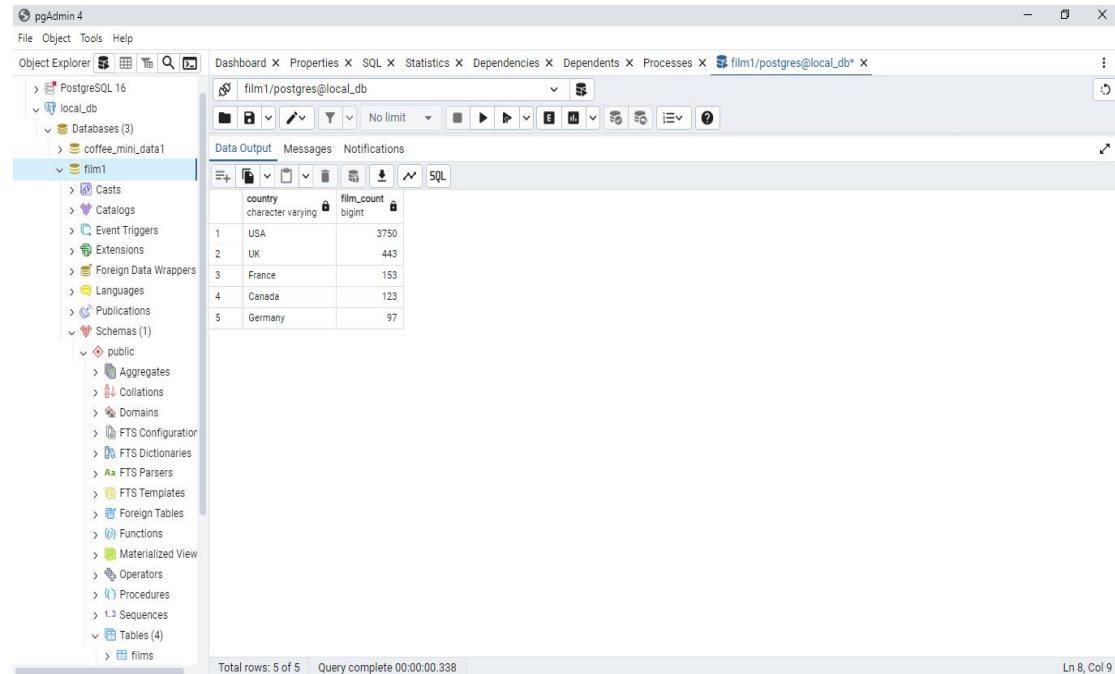
The screenshot shows the pgAdmin 4 interface. On the left, the Object Explorer shows the database structure: PostgreSQL 16, local_db, Databases (3), coffee_mini_data1, and film1. The film1 database is selected, showing its contents: Casts, Catalogs, Event Triggers, Extensions, Foreign Data Wrappers, Languages, Publications, Schemas (1), and public. The public schema contains Aggregates, Collations, Domains, FTS Configuration, FTS Dictionaries, FTS Parsers, FTS Templates, Foreign Tables, Functions, Materialized View, Operators, Procedures, Sequences, and Tables (4). The films table is selected. The main pane shows the SQL query editor with the following query:

```
--How many films in the database were released in each country,
--and what are the top five countries?

SELECT country, COUNT(*) AS film_count
FROM films
GROUP BY country
ORDER BY film_count DESC
LIMIT 5;
```

The status bar at the bottom indicates "Total rows: 5 of 5" and "Query complete 00:00:00.338".

Results:



The screenshot shows the pgAdmin 4 interface with the same database structure as the previous screenshot. The main pane shows the Results tab with the following data:

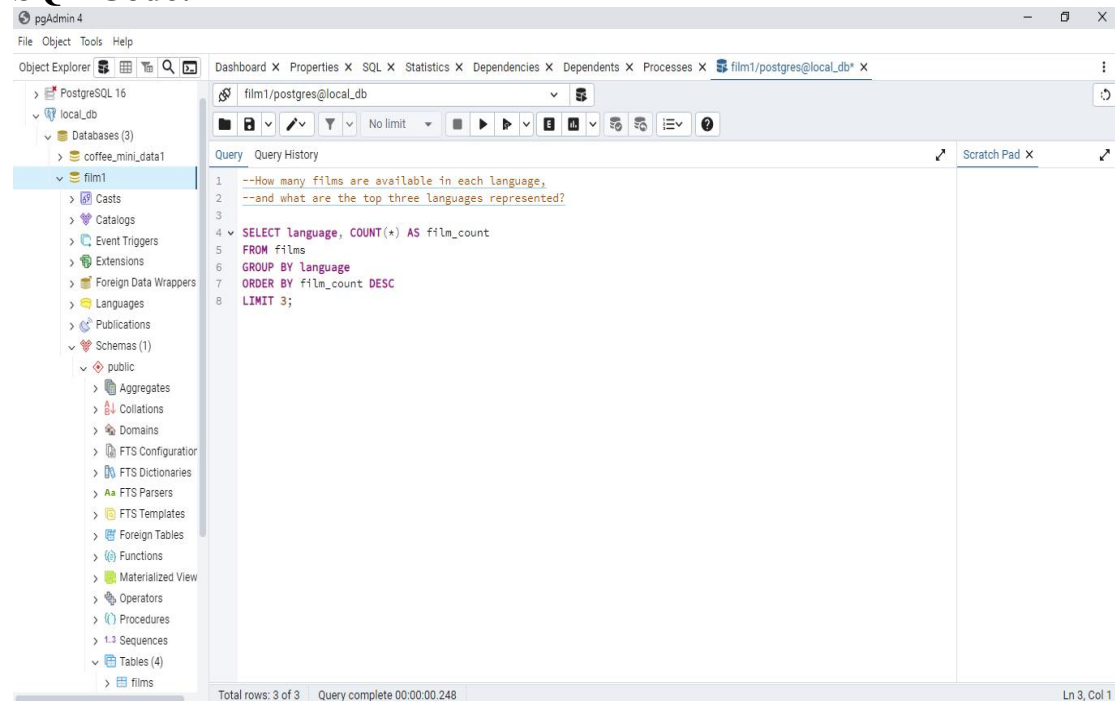
country	film_count
USA	3750
UK	443
France	153
Canada	123
Germany	97

The status bar at the bottom indicates "Total rows: 5 of 5" and "Query complete 00:00:00.338".

Analysis: The data shows the number of films released in each country, highlighting the top five countries with the most film release.

Question 2.3: How many films are available in each language, and what are the top three languages represented?

SQL Code:

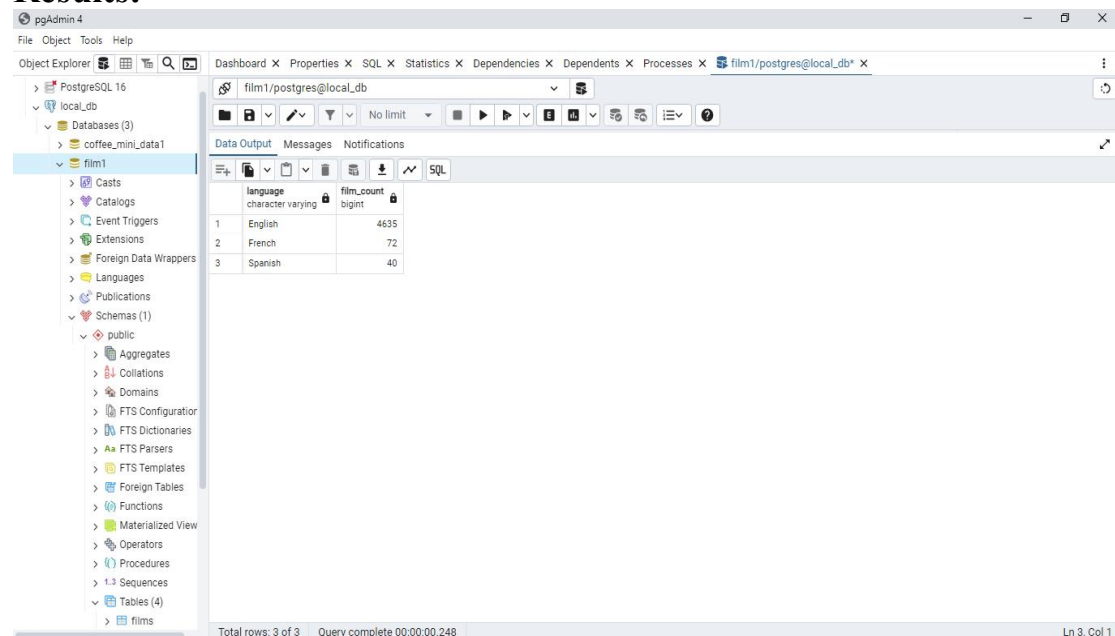


The screenshot shows the pgAdmin 4 interface. The left pane displays the database structure, including the 'public' schema and the 'films' table. The central pane shows the SQL query editor with the following query:

```
--How many films are available in each language,  
--and what are the top three languages represented?  
  
SELECT language, COUNT(*) AS film_count  
FROM films  
GROUP BY language  
ORDER BY film_count DESC  
LIMIT 3;
```

The status bar at the bottom indicates 'Total rows: 3 of 3' and 'Query complete 00:00:00.248'.

Results:



The screenshot shows the pgAdmin 4 interface with the query results displayed in the 'Data Output' pane. The results are as follows:

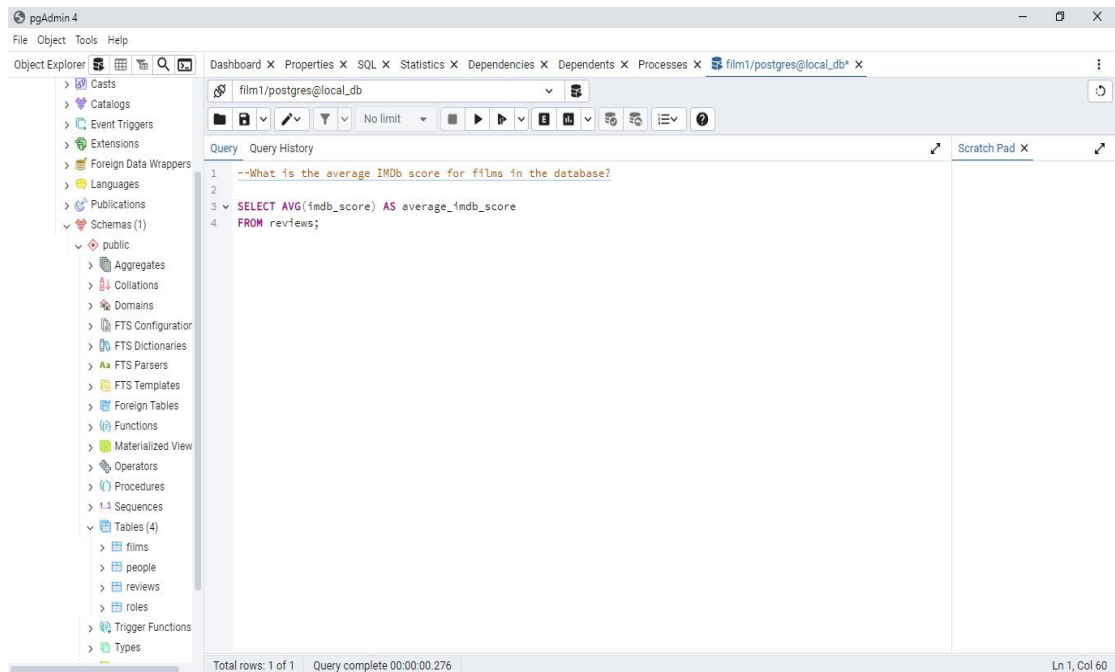
language	film_count
English	4635
French	72
Spanish	40

The status bar at the bottom indicates 'Total rows: 3 of 3' and 'Query complete 00:00:00.248'.

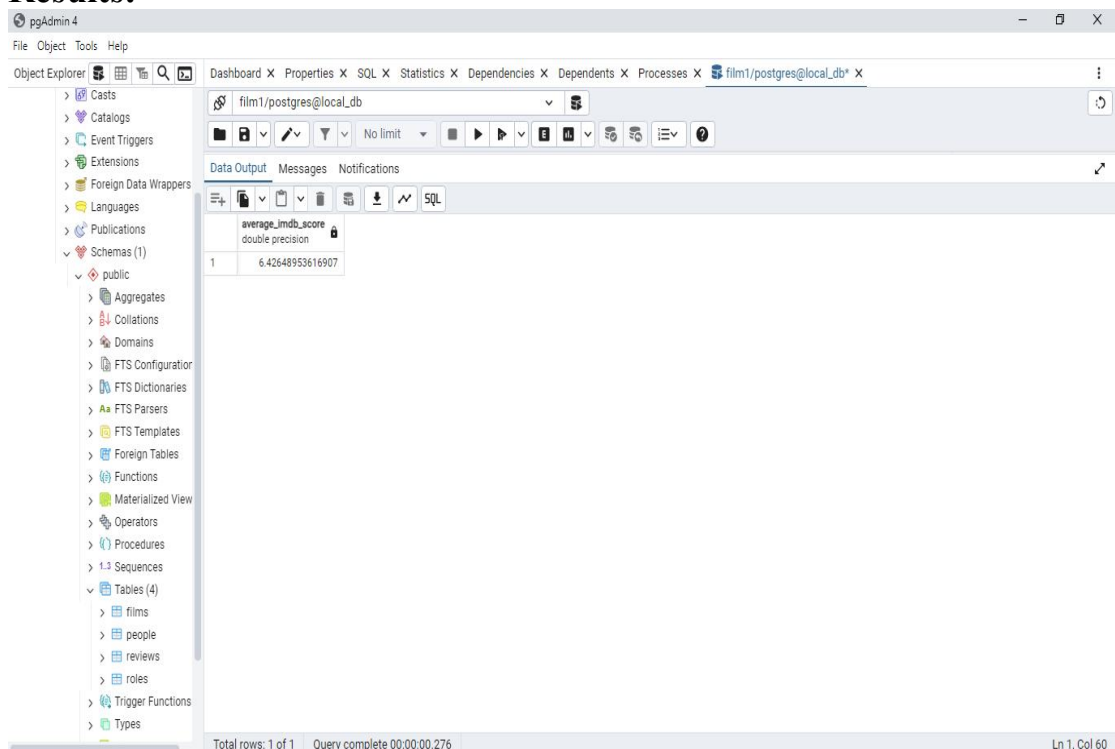
Analysis: The data shows the top three languages with the highest number of films, along with the count of films available in each of the languages.

Question 2.4: What is the average IMDb score for films in the database?

SQL Code:



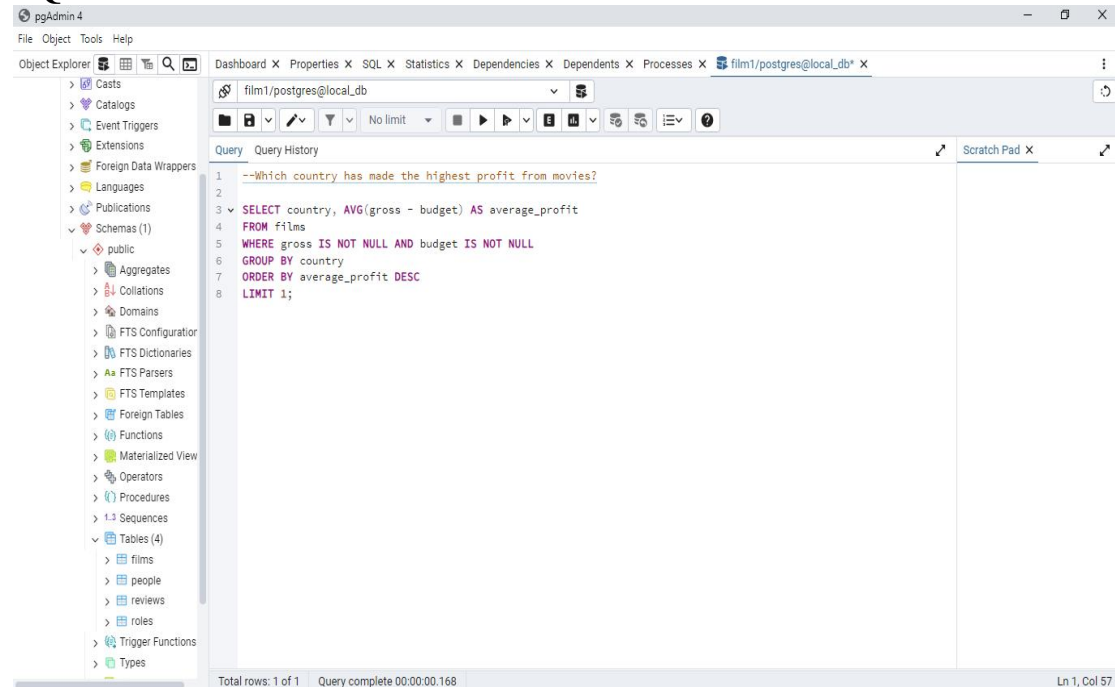
Results:



Analysis: The query result shows the average IMDb score for all the films in the database.

Question 2.5: Which country has made the highest profit from movies?

SQL Code:

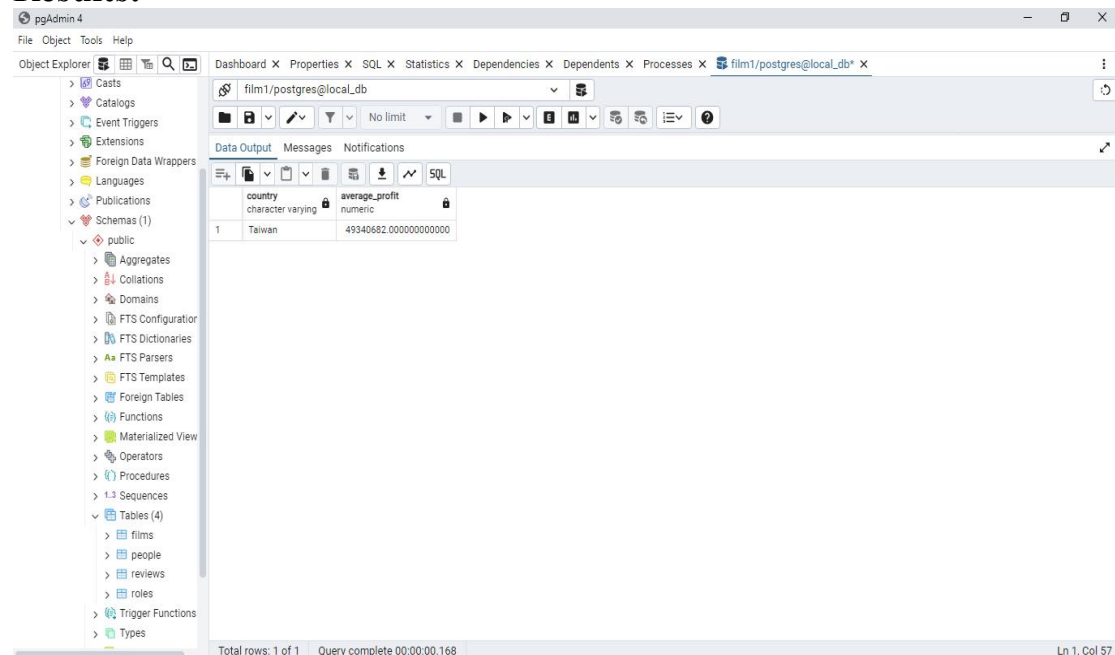


The screenshot shows the pgAdmin 4 interface. The left sidebar displays the database structure, including the 'public' schema and the 'films' table. The main pane shows the SQL query editor with the following query:

```
--Which country has made the highest profit from movies?
SELECT country, AVG(gross - budget) AS average_profit
FROM films
WHERE gross IS NOT NULL AND budget IS NOT NULL
GROUP BY country
ORDER BY average_profit DESC
LIMIT 1;
```

The status bar at the bottom indicates 'Total rows: 1 of 1' and 'Query complete 00:00:00.168'.

Results:



The screenshot shows the pgAdmin 4 interface with the query results displayed in the 'Data Output' pane. The results are as follows:

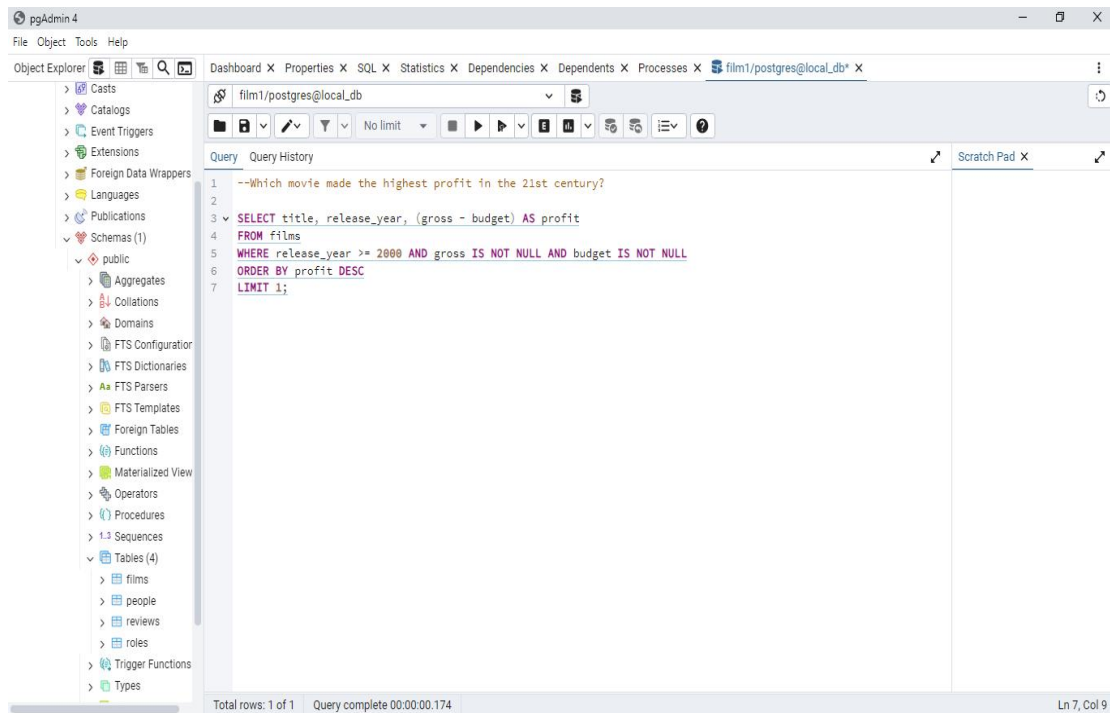
country	average_profit
Taiwan	49340682.00000000000000

The status bar at the bottom indicates 'Total rows: 1 of 1' and 'Query complete 00:00:00.168'.

Analysis: This query filters out films where the gross or budget values are null, before computing the average profit, ensuring that only valid data is used in the calculation.

Question 2.6: Which movie made the highest profit in the 21st century?

SQL Code:

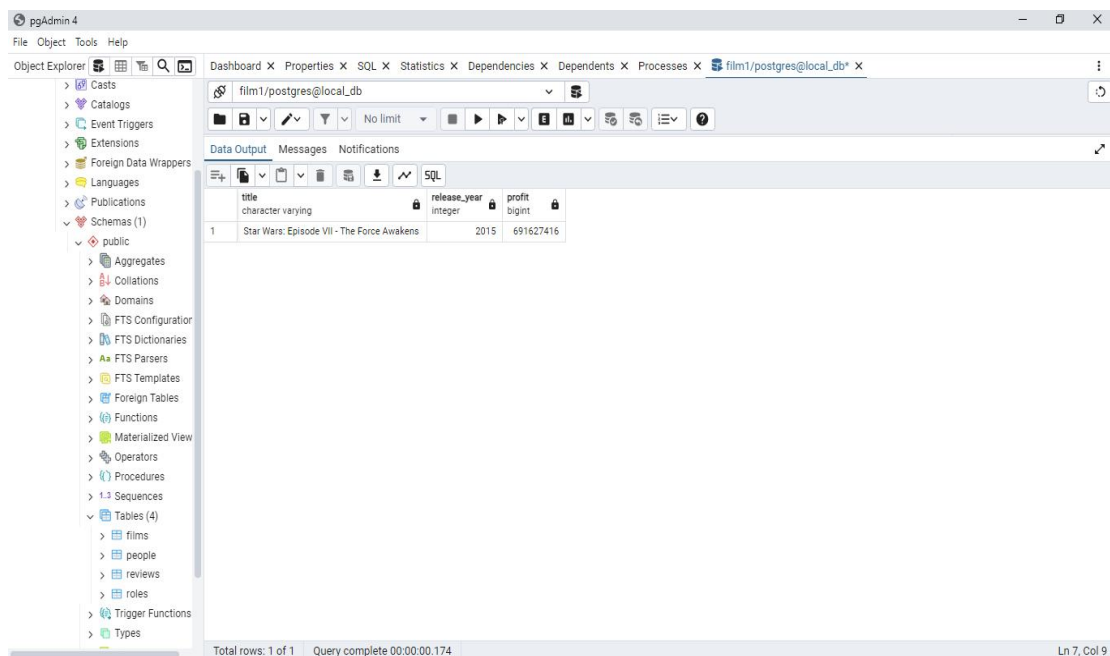


A screenshot of the pgAdmin 4 interface. The left sidebar shows the 'Object Explorer' with a tree view of database objects, including 'public' schema and 'Tables (4)' containing 'films', 'people', 'reviews', and 'roles'. The main pane displays a SQL query in the 'Query' tab. The query is as follows:

```
1 --Which movie made the highest profit in the 21st century?
2
3 SELECT title, release_year, (gross - budget) AS profit
4 FROM films
5 WHERE release_year >= 2000 AND gross IS NOT NULL AND budget IS NOT NULL
6 ORDER BY profit DESC
7 LIMIT 1;
```

The status bar at the bottom indicates 'Total rows: 1 of 1' and 'Query complete 00:00:00.174'. The bottom right corner shows 'Ln 7, Col 9'.

Results:



A screenshot of the pgAdmin 4 interface showing the results of the SQL query. The 'Data Output' tab is active, displaying a table with the following data:

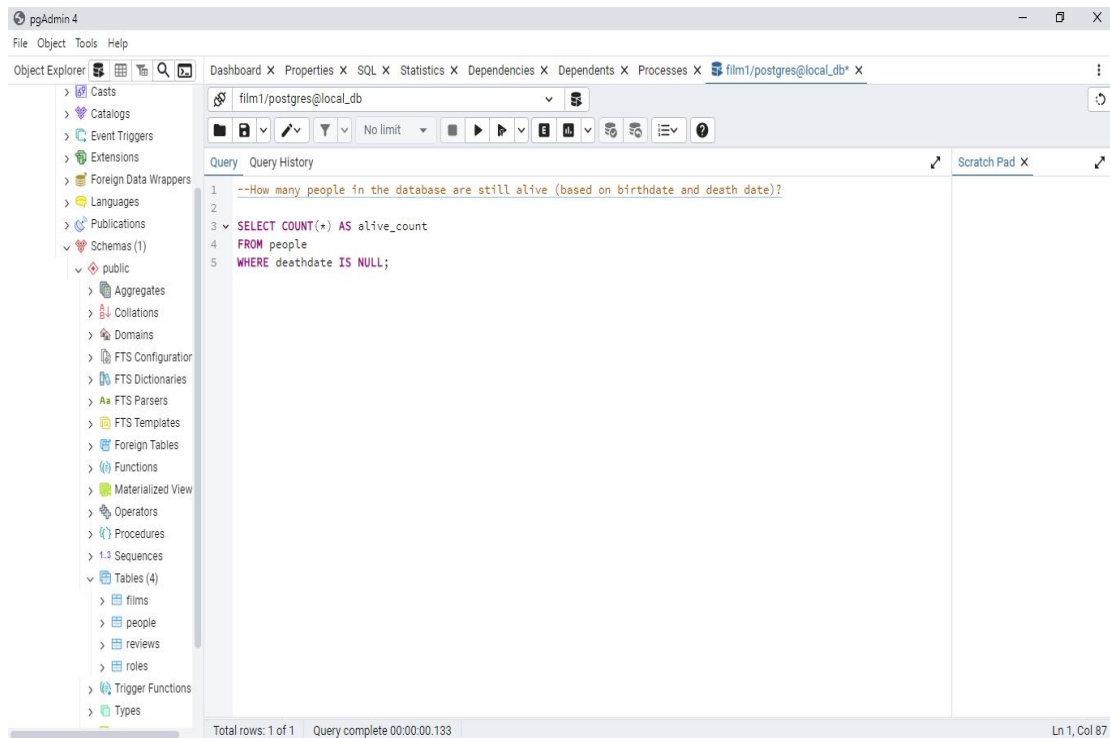
	title	release_year	profit
	character varying	integer	bigint
1	Star Wars: Episode VII - The Force Awakens	2015	691627416

The status bar at the bottom indicates 'Total rows: 1 of 1' and 'Query complete 00:00:00.174'. The bottom right corner shows 'Ln 7, Col 9'.

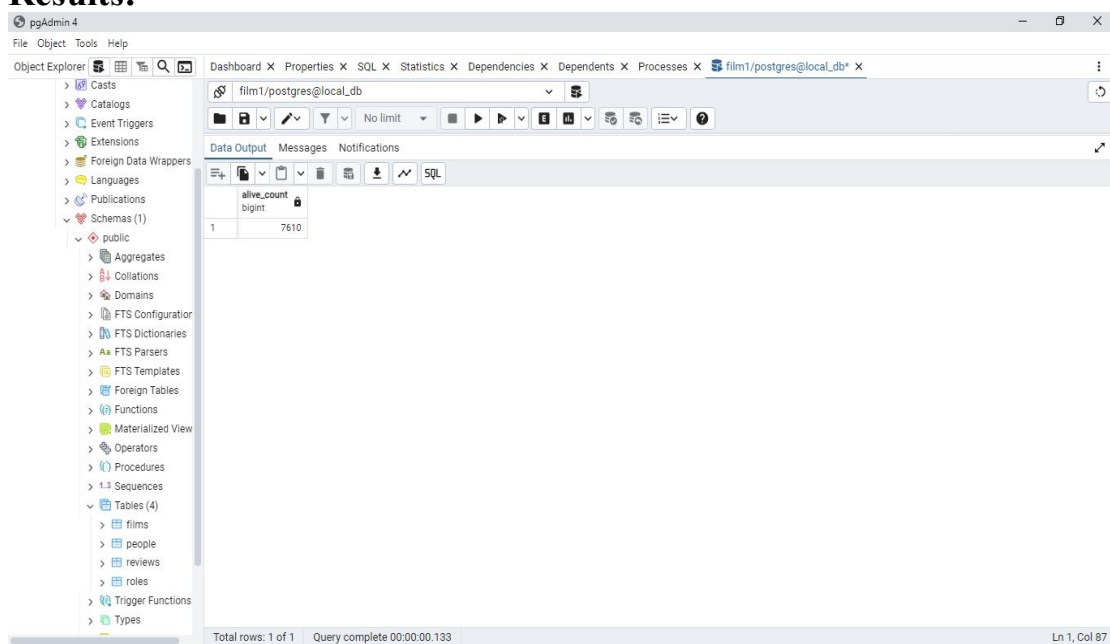
Analysis: The data shows the movie with the highest profit released in the 21st century.

Question 2.7: How many people in the database are still alive (based on birth date and death date)?

SQL Code:



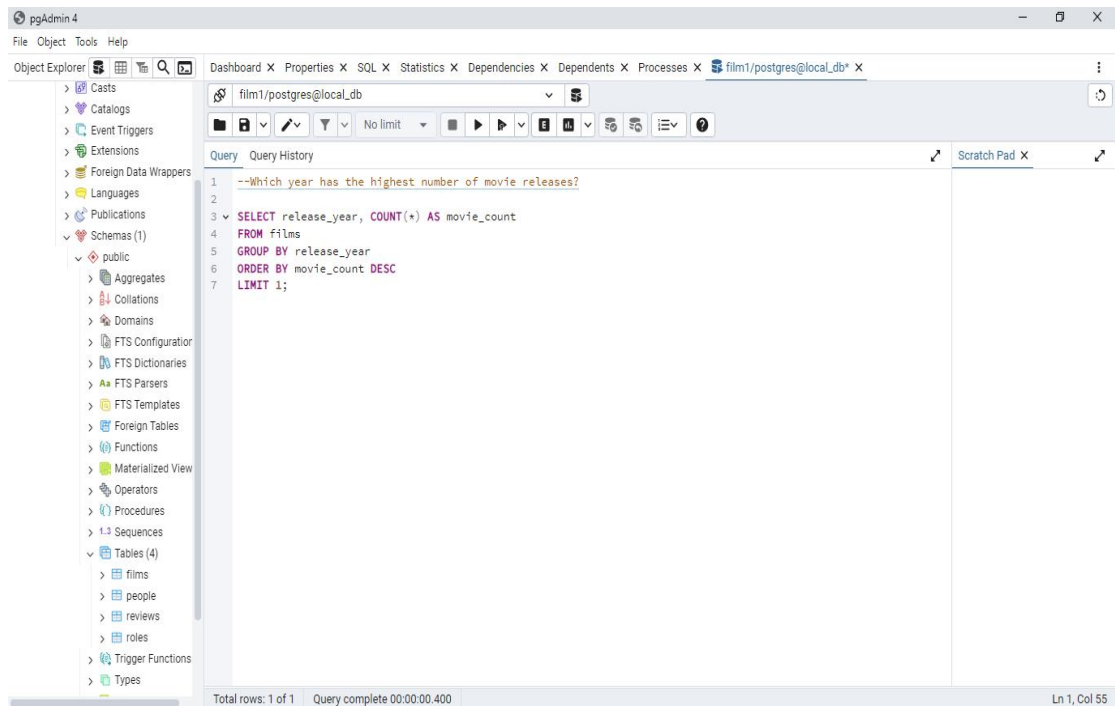
Results:



Analysis: The query shows the count of people who are still alive in the database.

Question 2.8: Which year has the highest number of movie releases?

SQL Code:

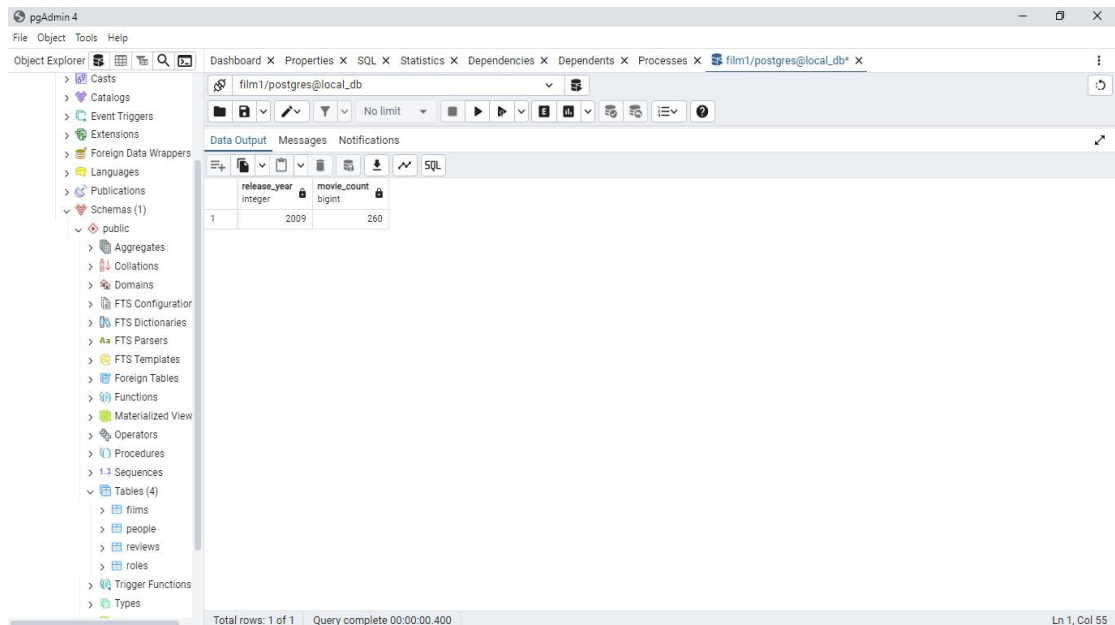


A screenshot of the pgAdmin 4 interface. The left sidebar shows the 'Object Explorer' with a tree view of database objects, including 'public' schema and 'Tables (4)' containing 'films', 'people', 'reviews', and 'roles'. The main pane shows a SQL query editor with the following code:

```
--Which year has the highest number of movie releases?
1
2
3 SELECT release_year, COUNT(*) AS movie_count
4 FROM films
5 GROUP BY release_year
6 ORDER BY movie_count DESC
7 LIMIT 1;
```

The status bar at the bottom indicates 'Total rows: 1 of 1' and 'Query complete 00:00:00.400'.

Results:



A screenshot of the pgAdmin 4 interface showing the results of the SQL query. The 'Data Output' tab is active, displaying a table with the following data:

release_year	movie_count
2009	260

The status bar at the bottom indicates 'Total rows: 1 of 1' and 'Query complete 00:00:00.400'.

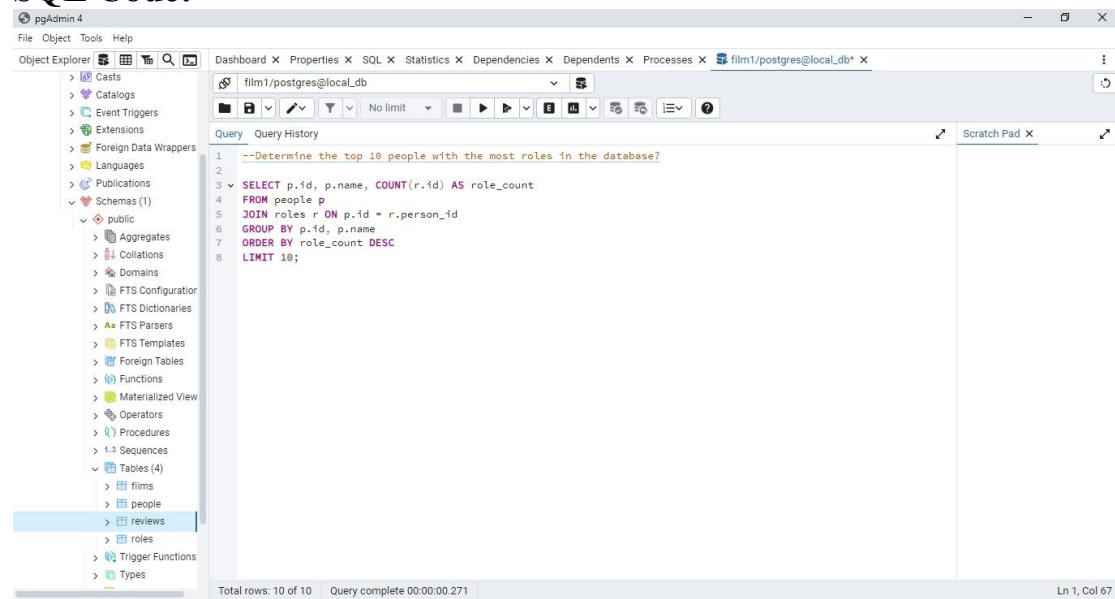
Analysis: This query result shows, the movies grouped by their release year, counts the number of movies for each year, orders the results by the count in descending order, and then limits the result to just the top year.

Task 3: People and Roles Analysis

For a better understanding of role distributions, identification of key personnel, and optimization of resource allocation, I leveraged SQL to answer the real-life business questions related to the film industry. Below are the questions;

Question 3.1: Determine the top 10 people with the most roles in the database.

SQL Code:

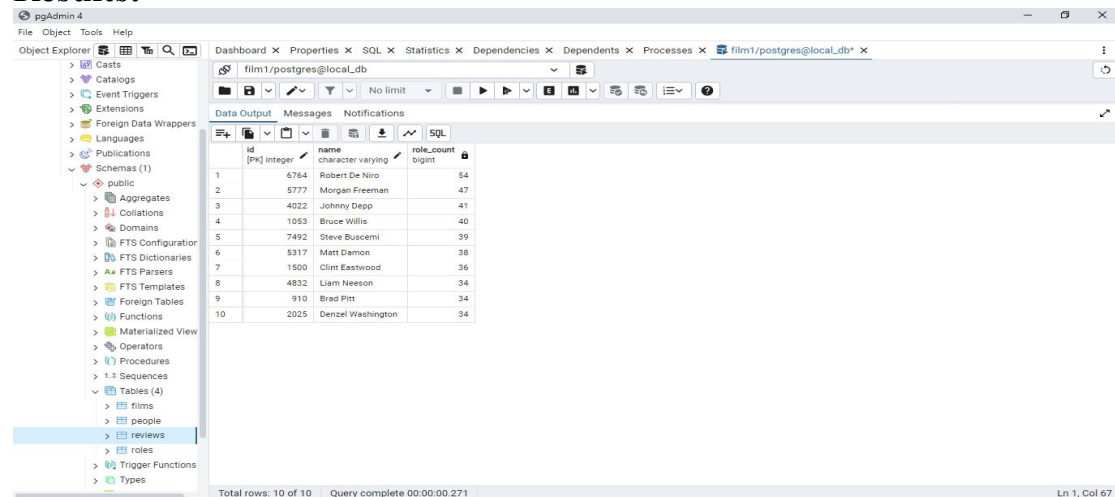


The screenshot shows the pgAdmin 4 interface. The left sidebar displays the database structure, including the 'public' schema and the 'people' table. The main pane shows the SQL query editor with the following query:

```
--Determine the top 10 people with the most roles in the database?
1
2
3 SELECT p.id, p.name, COUNT(r.id) AS role_count
4 FROM people p
5 JOIN roles r ON p.id = r.person_id
6 GROUP BY p.id, p.name
7 ORDER BY role_count DESC
8 LIMIT 10;
```

The status bar at the bottom indicates 'Total rows: 10 of 10' and 'Query complete 00:00:00.271'.

Results:



The screenshot shows the pgAdmin 4 interface with the results of the query displayed in a table. The table has three columns: 'id', 'name', and 'role_count'. The results are as follows:

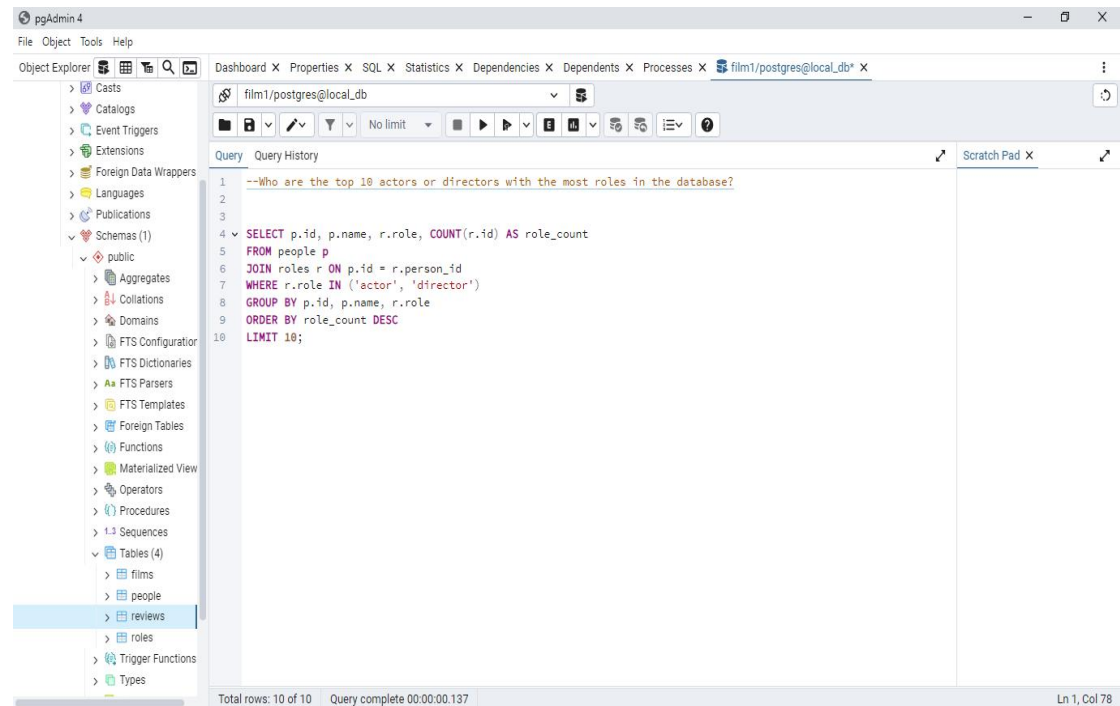
id	name	role_count
6764	Robert De Niro	54
5777	Morgan Freeman	47
4022	Johnny Depp	41
1053	Bruce Willis	40
7492	Steve Buscemi	39
5317	Matt Damon	38
1500	Clint Eastwood	36
4832	Liam Neeson	34
910	Brad Pitt	34
2025	Denzel Washington	34

The status bar at the bottom indicates 'Total rows: 10 of 10' and 'Query complete 00:00:00.271'.

Analysis: p.id and p.name are selected from the people table. COUNT (r.id) counts the number of roles each person has. The JOIN clause connects the people table with the roles table using person_id. GROUP BY groups the results by person. ORDER BY role_count DESC sorts the people by their role count in descending order. LIMIT 10 shows only the top ten people.

Question 3.2: Who are the top 10 actors or directors with the most roles in the database?

SQL Code:



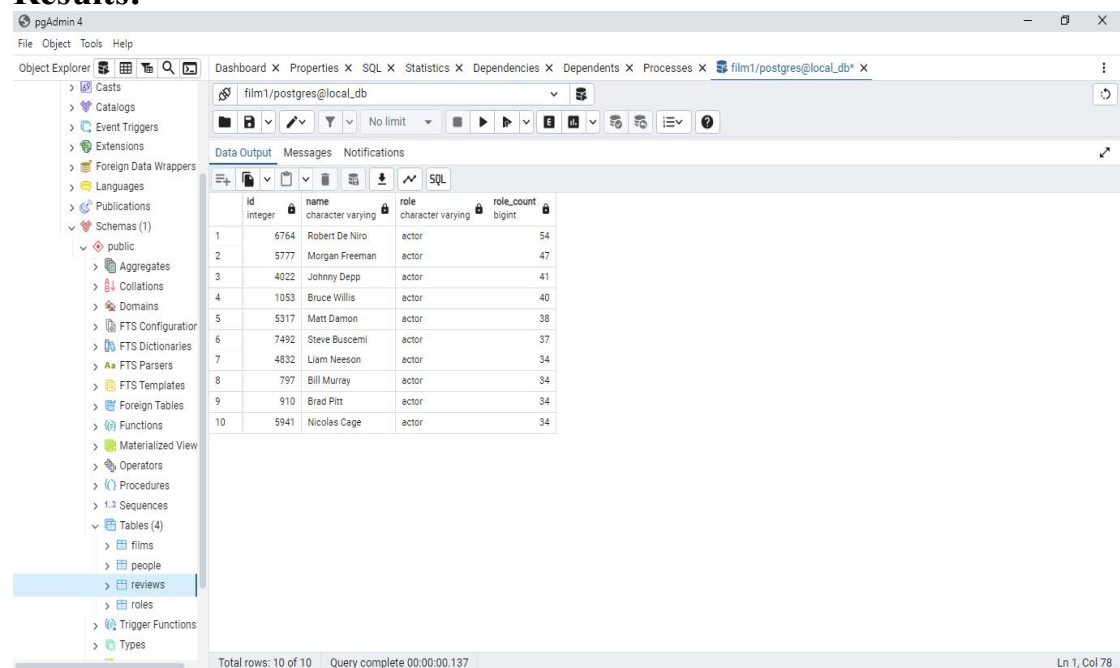
The screenshot shows the pgAdmin 4 interface. The left sidebar displays the database structure, including the 'public' schema and its tables. The main pane shows the SQL query editor with the following query:

```
--Who are the top 10 actors or directors with the most roles in the database?

SELECT p.id, p.name, r.role, COUNT(r.id) AS role_count
FROM people p
JOIN roles r ON p.id = r.person_id
WHERE r.role IN ('actor', 'director')
GROUP BY p.id, p.name, r.role
ORDER BY role_count DESC
LIMIT 10;
```

The status bar at the bottom indicates "Total rows: 10 of 10" and "Query complete 00:00:00.137".

Results:



The screenshot shows the pgAdmin 4 interface with the results of the query displayed in the "Data Output" pane. The results are as follows:

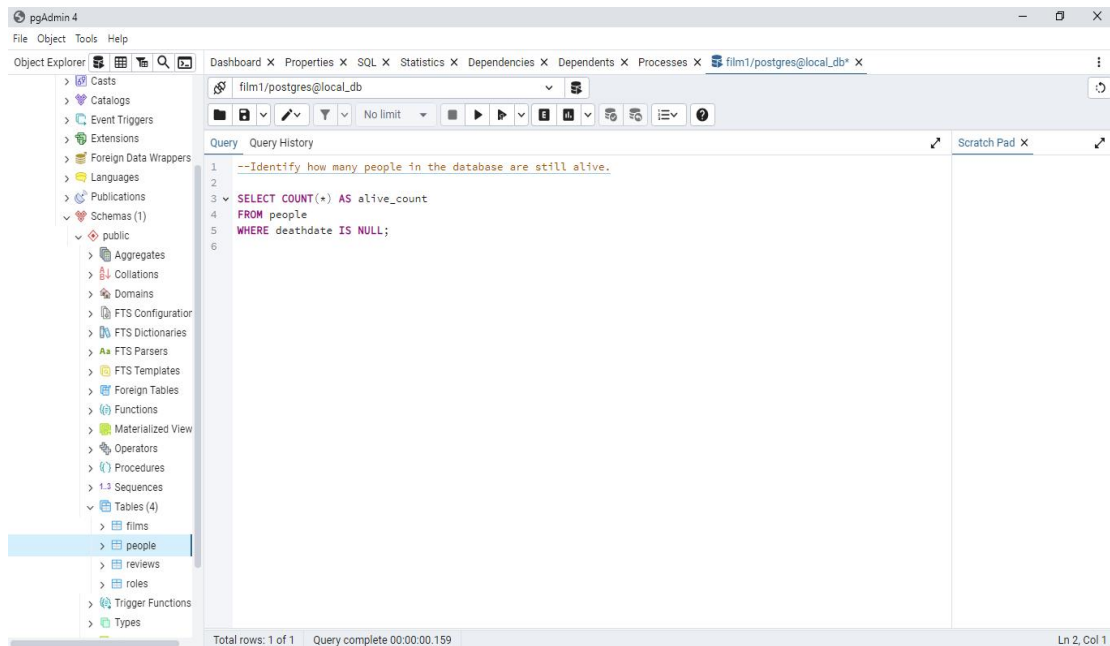
id	name	role	role_count
1	Robert De Niro	actor	54
2	Morgan Freeman	actor	47
3	Johnny Depp	actor	41
4	Bruce Willis	actor	40
5	Matt Damon	actor	38
6	Steve Buscemi	actor	37
7	Liam Neeson	actor	34
8	Bill Murray	actor	34
9	Brad Pitt	actor	34
10	Nicolas Cage	actor	34

The status bar at the bottom indicates "Total rows: 10 of 10" and "Query complete 00:00:00.137".

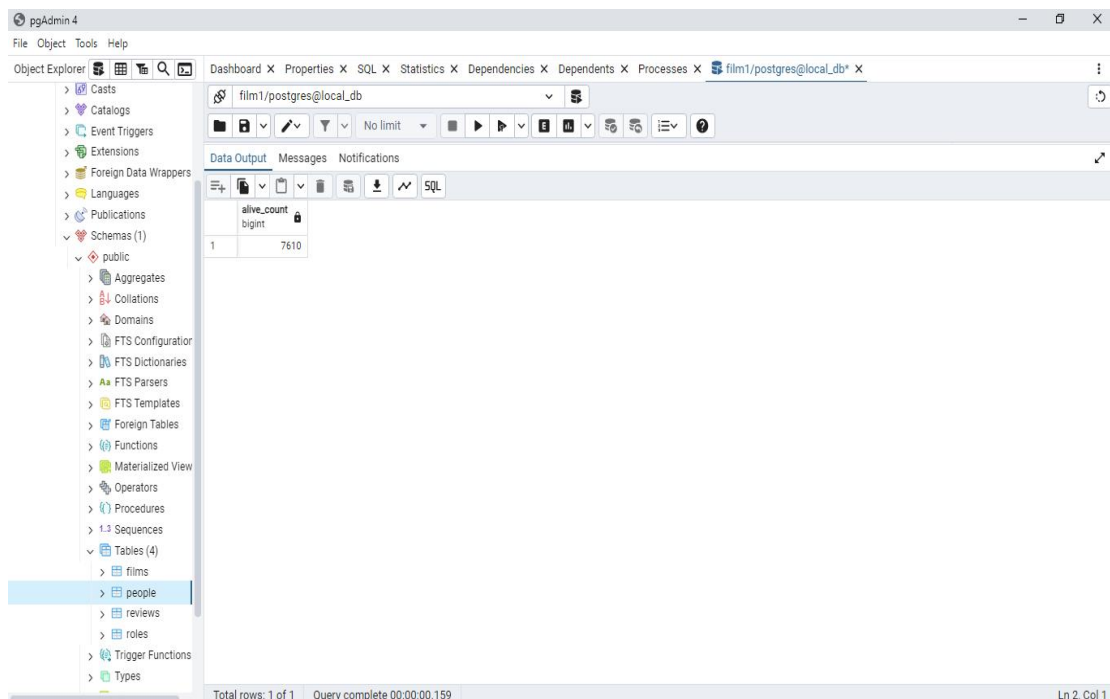
Analysis: This data shows the list of the top 10 people with the most roles.

Question 3.3: Identify how many people in the database are still alive.

SQL Code:



Results:



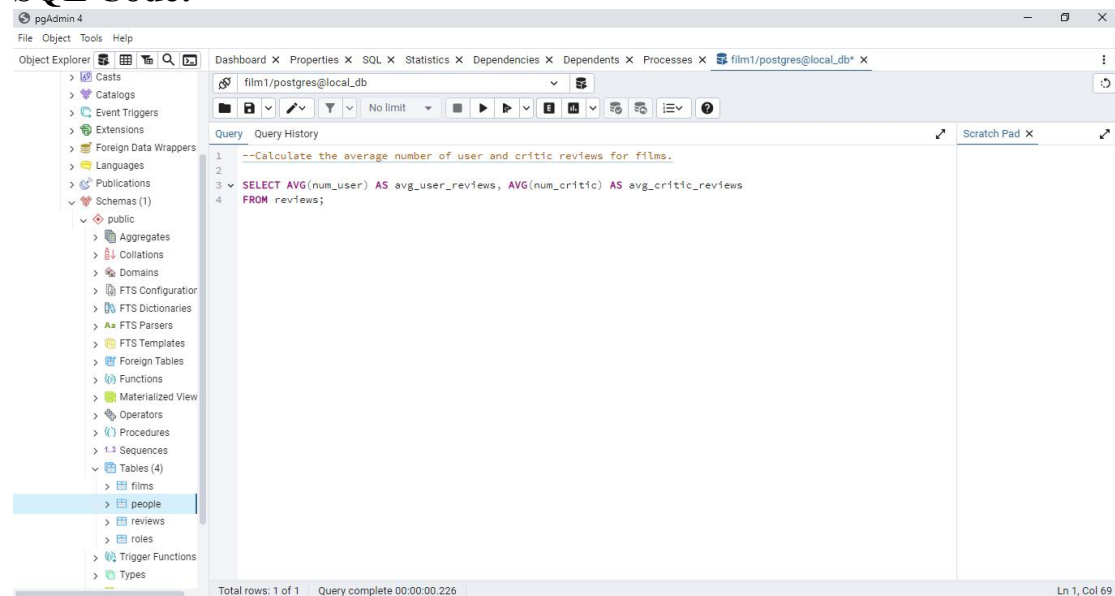
Analysis: This query counts the number of people alive in the database.

Task 4: User Reviews and Votes Analysis

In order to gain insights into customer satisfaction, preferences, and sentiments, I utilized SQL to analyze user reviews and voting patterns. This analysis helps in understanding customer behaviour, identifying key trends, and optimizing product or service offerings. Below are the questions addressed:

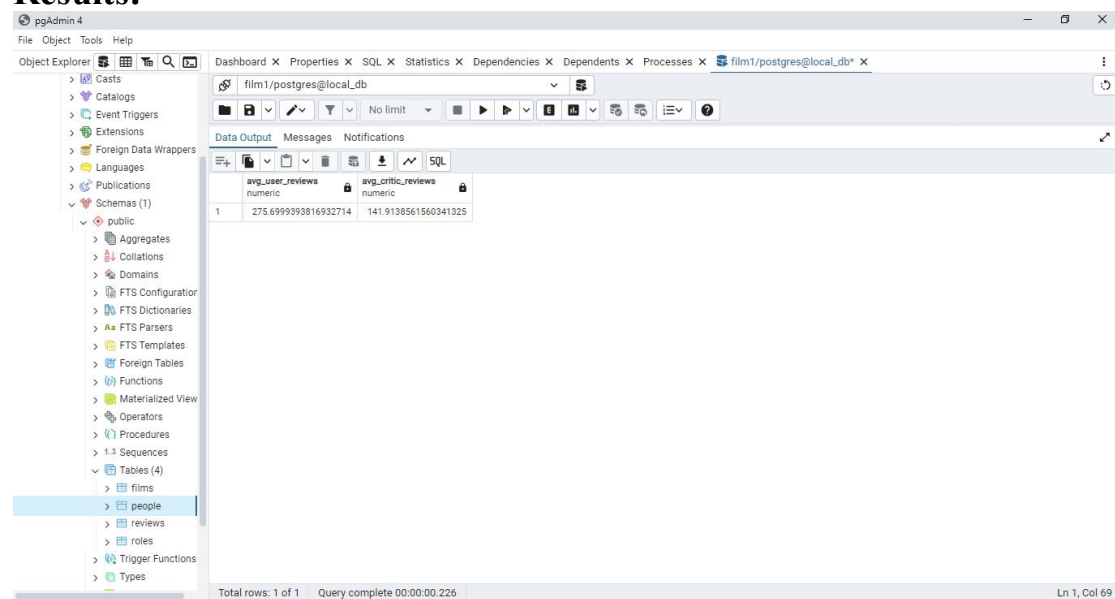
Question 4.1: Calculate the average number of user and critic reviews for films?

SQL Code:



```
--Calculate the average number of user and critic reviews for films.
SELECT AVG(num_user) AS avg_user_reviews, AVG(num_critic) AS avg_critic_reviews
FROM reviews;
```

Results:

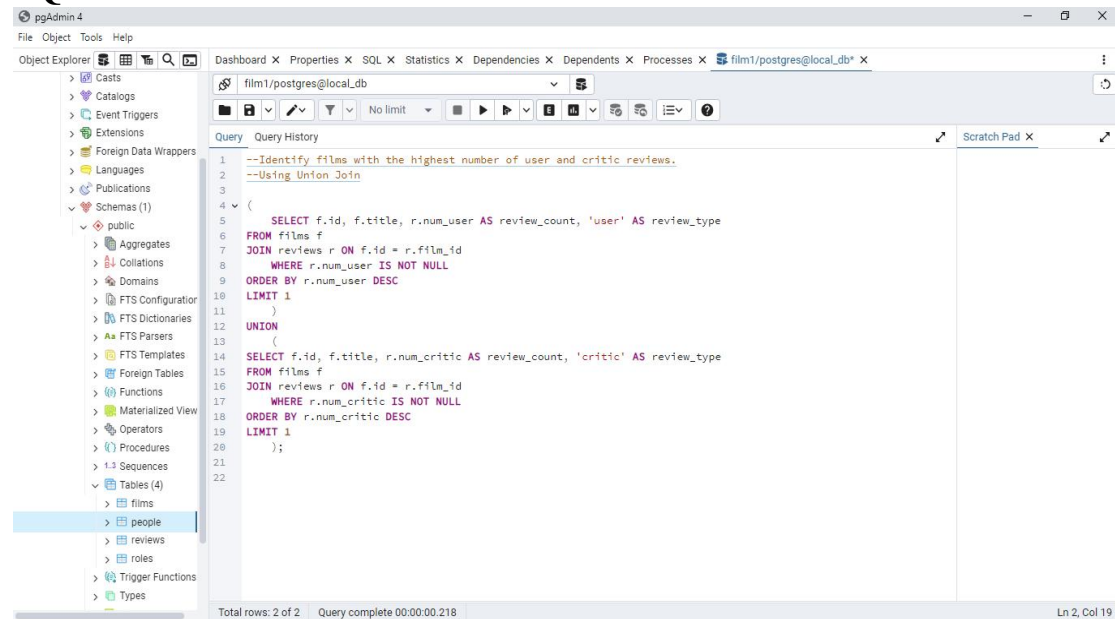


	avg_user_reviews	avg_critic_reviews
1	275.6999993816932714	141.9138561560341325

Analysis: The query result shows the average number of user and critic reviews for films in the database.

Question 4.2: Identify films with the highest number of user and critic reviews?

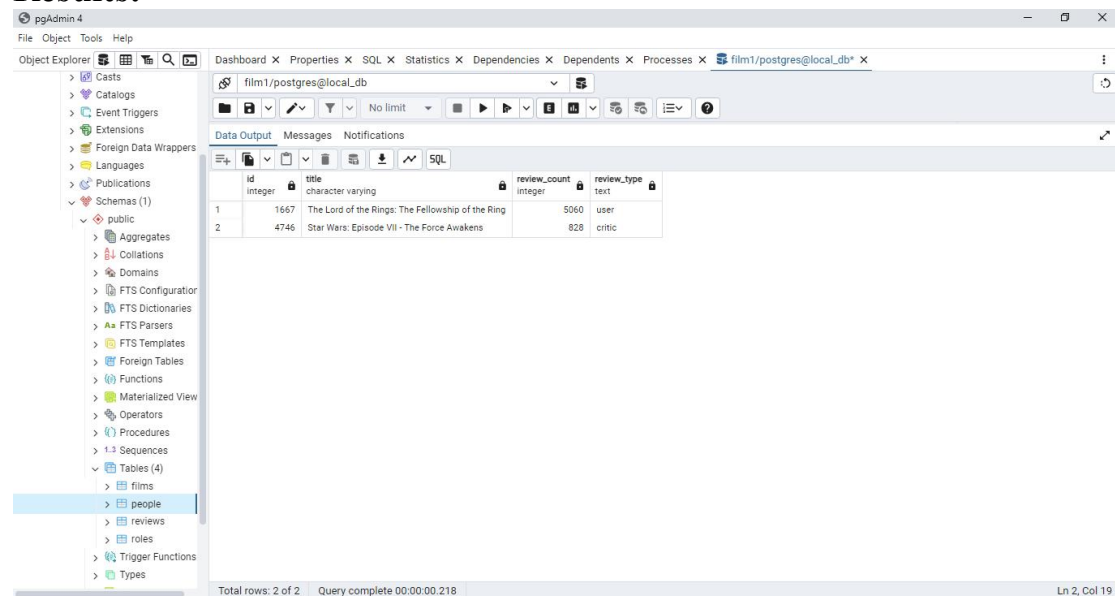
SQL Code:



The screenshot shows the pgAdmin 4 interface. The left sidebar displays the 'public' schema with tables 'films', 'people', 'reviews', 'roles', and 'types'. The 'films' table is selected. The main pane shows a SQL query in the 'Query' tab. The query is a UNION of two SELECT statements, each identifying a film with the highest number of reviews from a specific user or critic. The status bar at the bottom indicates 'Total rows: 2 of 2' and 'Query complete 00:00:00.218'.

```
1 --Identify films with the highest number of user and critic reviews.
2 --Using Union Join
3
4 (
5     SELECT f.id, f.title, r.num_user AS review_count, 'user' AS review_type
6 FROM films f
7 JOIN reviews r ON f.id = r.film_id
8 WHERE r.num_user IS NOT NULL
9 ORDER BY r.num_user DESC
10    LIMIT 1
11 )
12 UNION
13 (
14     SELECT f.id, f.title, r.num_critic AS review_count, 'critic' AS review_type
15 FROM films f
16 JOIN reviews r ON f.id = r.film_id
17 WHERE r.num_critic IS NOT NULL
18 ORDER BY r.num_critic DESC
19    LIMIT 1
20 );
21
22
```

Results:



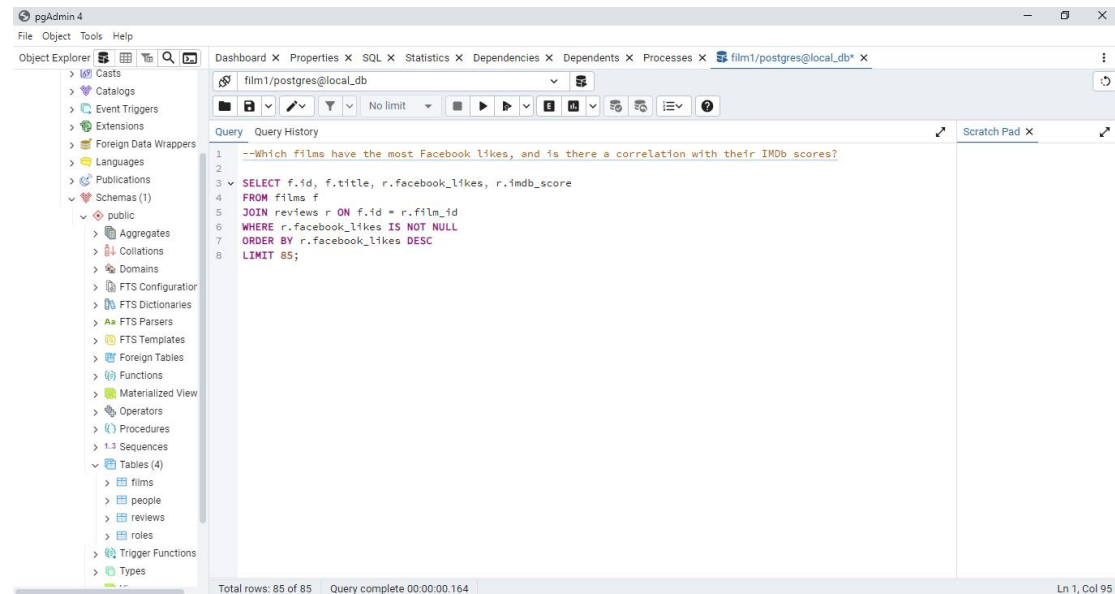
The screenshot shows the pgAdmin 4 interface with the 'Data Output' tab selected. It displays the results of the SQL query as a table with 2 rows. The status bar at the bottom indicates 'Total rows: 2 of 2' and 'Query complete 00:00:00.218'.

id	title	review_count	review_type
1	1667 The Lord of the Rings: The Fellowship of the Ring	5060	user
2	4746 Star Wars: Episode VII - The Force Awakens	828	critic

Analysis: The WHERE clauses `r.num_user IS NOT NULL` and `r.num_critic IS NOT NULL` filter out rows where the number of user or critic reviews is null. The data identifies the films with the highest number of user and critic reviews.

Question 4.3: Which films have the most Facebook likes, and is there a correlation with their IMDb scores?

SQL Code:



Results:

id	title	facebook_likes	imdb_score
66	4294 The Hobbit: The Desolation of Smaug	83000	7.9
67	4107 12 Years a Slave	83000	8.1
68	4240 Pacific Rim	83000	7
69	4296 The Hunger Games: Catching Fire	82000	7.6
70	4593 X-Men: Days of Future Past	82000	8
71	4756 Terminator Genisys	82000	6.6
72	4421 Fury	82000	7.6
73	3925 Dark Shadows	82000	6.2
74	4335 Warm Bodies	81000	6.9
75	3709 Drive	81000	7.8
76	4892 Suicide Squad	80000	6.9
77	4745 Spotlight	80000	8.1
78	4023 Skyfall	80000	7.8
79	4022 Skyfall	80000	7.8
80	2597 Idiocracy	79000	6.6
81	3767 Midnight in Paris	78000	7.7
82	3872 Warrior	77000	8.2
83	4410 Edge of Tomorrow	77000	7.9
84	4749 Straight Outta Compton	76000	7.9
85	2827 Into the Wild	76000	8.2

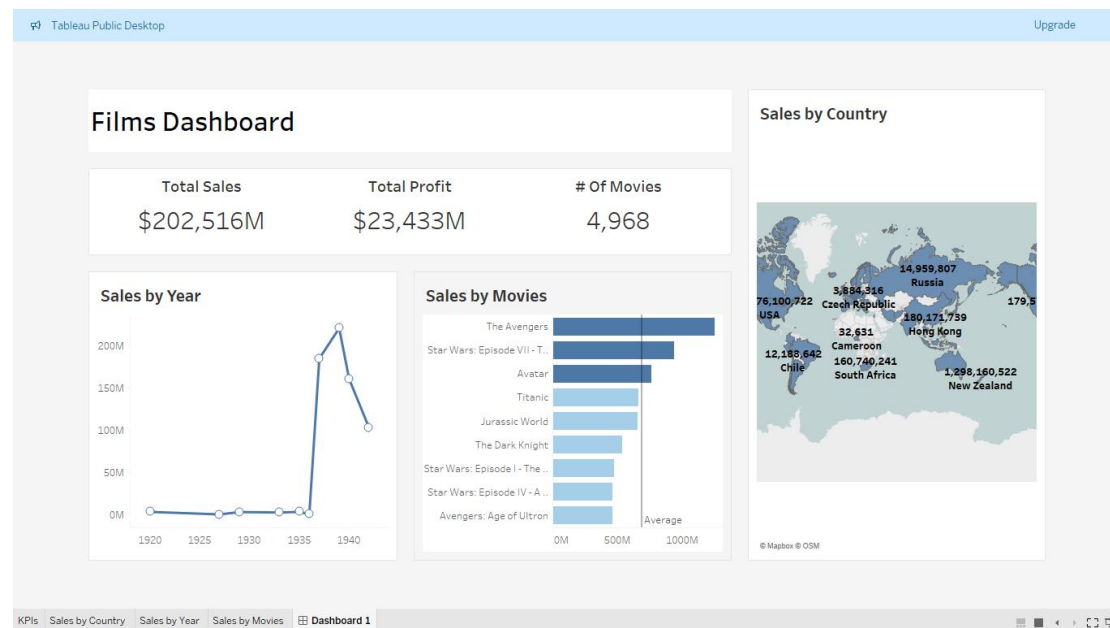
Analysis:

Here the data result shows the films which have the most Facebook likes and IMDb scores. In order to get the correlation between Facebook likes and IMDb scores, I took a limit of 85 records as a sample size out of 4968 records of the both variables, to get a medium effect size.

The datasets was imported to Excel, where I further used the correlation function to get the correlation between Facebook likes and IMDb scores, and my result was 0.16220513.

The value 0.16220513 is relatively close to 0, which indicates a weak correlation. This Suggests that the relationship between Facebook likes and IMDb scores is not very strong. While there is a positive trend, it is not pronounced, and many other factors likely influence IMDB scores independently of Facebook likes.

4. Visualizations



In this section, I created a film dashboard using Tableau to present key performance indicators (KPIs) and visualizations based on the film database given. The dashboard includes:

1) KPIs:

- Total sales (Gross) for movies
- Total profit
- Number of Movies

2) Visualizations:

- Sales by Year (Trend Line)
- Sales by Movie (Bar Chart)
- Sales by Country (Map)

5. Conclusion

Using Tableau, I developed this dashboard to provide clear insights into film performance. Key observations include:

- **Sales Trends Over Time:** The trend line reveals fluctuations in yearly sales, indicating varying periods of revenue.
- **Top-Grossing Films:** The bar chart highlights the most financially successful movies.
- **Geographical Revenue Distribution:** The map shows revenue by country, aiding in strategic planning.

Overall, this dashboard helps productive decisions towards optimizing production, marketing, and distribution strategies in the film industry.

6. References

TechWay Consult. (2024). Film database. TechWay Consult.