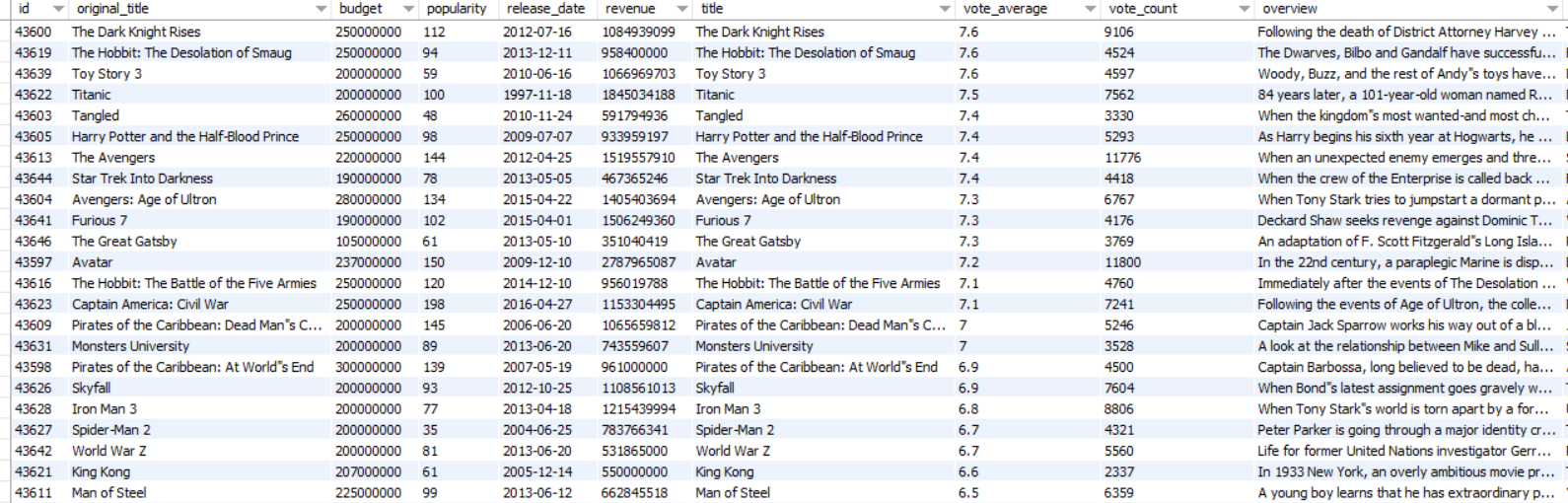
**PRSQ-02-IMDB Movies**

1 Can you get all data about movies?

**Query**: Select \* from movies;

**Explanation**: The SQL statement Select \* from movies; is a SQL query that selects all columns from the movies table.

**Output**: It returns 47 Rows and 13 Columns.



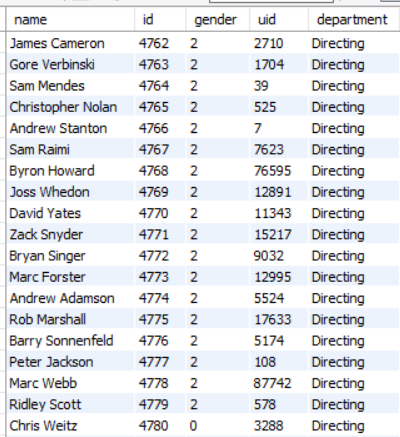
2. How do you get all data about directors?

**Query:** SELECT \* FROM directors;

**Explanation:** The SELECT keyword is used to select data from a database table. The \* wildcard is used to select all columns from the table. The FROM keyword specifies the table from which the data should be selected. In this case, the directors table is being selected.

This SQL statement will return a list of all the directors in the database, including their name, ID, and any other columns that are stored in the directors table.

**Output**



3. Check how many movies are present in IMDB.

**Query:** select count(\*) as total\_movies from movies;

**Explanation: The SQL statement counts the total number of movies in the** movies **table and assign the result to the variable** total\_movies**.**

The COUNT(\*) function returns the number of rows in a table. In this case, the COUNT(\*) function is used to count the total number of rows in the movies table. The result of the COUNT(\*) function is assigned to the variable total\_movies.

The as total\_movies clause is used to give the result of the COUNT(\*) function a more meaningful name. This makes it easier to understand the results of the query.

**Output:**



4. Find these 3 directors: James Cameron ; Luc Besson ; John Woo

**Query:** select name from directors where name in('James Cameron','Luc Besson','John Woo');

**Explanation:** The SQL statement selects the name column from the directors table where the name value is one of James Cameron, Luc Besson, or John Woo.

**Output:**

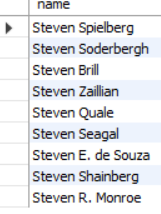


5. Find all directors with name starting with Steven.

**Query:** select name from directors where name like 'Steven%';

**Explanation:** The SQL statement select name from directors where name like 'Steven%'; selects the names of all directors whose names start with the word "Steven".

**Output:**

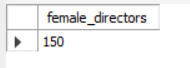


6. Count female directors

**Query:** select count(gender) as female\_directors from directors where gender=1;

**Explanation:** The SQL statement will **count the number of female directors in the** directors **table** and store the result in the variable female\_directors. The gender column in the directors table has two values: 2 for male and 1 for female. The where clause of the query ensures that only records where the gender column is equal to 1 (i.e., female) are counted. The count function counts the number of rows in a table that satisfy a given condition. In this case, the condition is that the gender column is equal to 1.

**Output:**



7. Find the name of the 10th first women directors?

**Query:** SELECT name FROM directors WHERE gender='1' ORDER BY id LIMIT 1 OFFSET 10;

**Explanation:** The SQL statement SELECT name FROM directors WHERE gender='1' ORDER BY id LIMIT 1 OFFSET 10; selects the first director with the gender '2' (male) from the directors table, after skipping the first 10 rows.

**Output:**

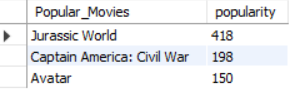


8. What are the 3 most popular movies?

**Query:** Select original\_title as Popular\_Movies, popularity from movies order by popularity desc limit 3;

**Explanation:** The SQL statement selects the original title and popularity of the 3 most popular movies from the movies table, and renames the original title column to Popular\_Movies.

**Output:**

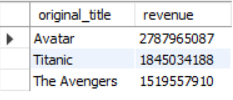


9. What are the 3 most bankable movies?

**Query:** select original\_title, revenue from movies order by revenue desc limit 3;

**Explanation:** The SQL statement selects the original title and revenue of the top 3 movies by revenue, in descending order.

**Output:**



10. What is the most awarded average vote since the January 1st, 2000?

**Query:** SELECT MAX(vote\_average) FROM movies WHERE release\_date >= '2000-01-01';

**Explanation:** The SQL statement selects the maximum vote average from all movies released on or after January 1, 2000.

**Output:**



11. Which movie(s) were directed by Brenda Chapman?

**Query:** SELECT original\_title FROM movies as m JOIN directors as d ON d.id = m.director\_id WHERE d.name = 'Brenda Chapman';

**Explanation:** The SQL statement selects the original title of all movies directed by Brenda Chapman.

**Output:** There is no movie directed by Brenda. So no output will be generated.

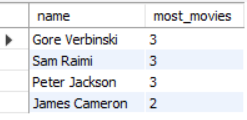


12. Whose director made the most movies?

**Query:** SELECT d.name,count(m.director\_id) as most\_movies FROM directors as d JOIN movies as m ON m.director\_id=d.id GROUP BY m.director\_id ORDER BY count(m.director\_id) DESC LIMIT 4;

**Explanation:** The SQL statement selects the name and the number of movies directed by each director from the directors and movies tables. The results are grouped by director ID and ordered by the number of movies in descending order. The top 4 directors are then returned.

**Output:**



13. Whose director is the most bankable?

**Query:** SELECT d.name,sum(m.revenue) as Total\_Revenue FROM directors as d JOIN movies as m ON m.director\_id=d.id GROUP BY m.director\_id ORDER BY sum(m.revenue) DESC LIMIT 2;

**Explanation:** The SQL statement will **select the names and total revenue of the top 2 directors**, based on the total revenue of their movies. The code first joins the directors and movies tables on the director\_id column. Then, it groups the results by the director\_id column and sums the revenue column for each group. Finally, it orders the results by the total revenue in descending order and limits the results to the top 2 directors.

**Output:**

