**IMDB DATASET ANALYSIS**

**-ADVANCED** SQL PROJECT

**JOTHEESWARAN S**

**28/03/2025**

**DATA ANALYTICS & DATA SCIENCE**

**FEB 25’**

DATASETDESCRIPTION**:**

* The dataset provided is a simplified version of the IMDb database, structured to capture essential information about movies, their genres, actors, directors, ratings, and more.

AVAILABLEINFORMATION**:**

1. Movie: (7997 records)
2. Genre: (14662 records)
3. Director Mapping: (3867 records)
4. Role Mapping: (151615 records)
5. Names :(25735 records)
6. Ratings: (7997 records)

DATA DESCRIPTION:

* **Movie**:

Contains basic information about each movie, including title, release year, duration, country, income, languages, and production companies.

* **Genre**:

Describes the genres associated with each movie.

* **Director Mapping**:

Maps movies to their directors.

* **Role Mapping**:

Maps actors/actresses to movies and specifies the role category

* **Names**:

Stores information about people including their birthdates, heights, and known movies.

* **Ratings**:

Contains ratings information for movies, including the average rating, total votes, and median rating.

QURIES AND OUTPUTS

1. Count the total number of records in each table of the database.

select count(\*) as count\_of\_directormapping from director\_mapping;

select count(\*) as count\_of\_genre from genre;

select count(\*) as count\_of\_movie from movie;

select count(\*) as count\_of\_names from names;

select count(\*) as count\_of\_ratings from ratings;

select count(\*) as count\_of\_rolemapping from role\_mapping;



2. Identify which columns in the movie table contain null values

SELECT

CASE WHEN COUNT(\*) - COUNT(title) > 0 THEN 'title has nulls'

ELSE'title has no nulls'

END AS title,

CASE WHEN COUNT(\*) - COUNT(year) > 0 THEN 'year has nulls'

ELSE'year has no nulls'

END AS year,

CASE WHEN COUNT(\*) - COUNT(duration) > 0 THEN 'duration has nulls'

ELSE'duration has no nulls'

END AS duration,

CASE WHEN COUNT(\*) - COUNT(country) > 0 THEN 'country has nulls'

ELSE'country has no nulls'

END AS country,

CASE WHEN COUNT(\*) - COUNT(worlwide\_gross\_income) > 0 THEN 'worlwide\_gross\_income has nulls'

ELSE'worlwide\_gross\_income has no nulls'

END AS worlwide\_gross\_income,

CASE WHEN COUNT(\*) - COUNT(date\_published) > 0 THEN 'date\_published has nulls'

ELSE'date\_published has no nulls'

END AS date\_published,

CASE WHEN COUNT(\*) - COUNT(languages) > 0 THEN 'languages has nulls'

ELSE'languages has no nulls'

END AS languages,

CASE WHEN COUNT(\*) - COUNT(production\_company) > 0 THEN 'production\_company has nulls'

ELSE'production\_company has no nulls'

END AS production\_company

FROM movie;



3. Determine the total number of movies released each year, and analyze how the trend changes month-wise.

SELECT

release\_year,

total\_movies,

RANK() OVER (ORDER BY total\_movies DESC) AS ranking

FROM (

SELECT

YEAR(date\_published) AS release\_year,

COUNT(title) AS total\_movies

FROM

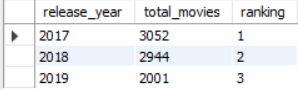
movie

GROUP BY

YEAR(date\_published)

) AS subquery

ORDER BY total\_movies DESC;



SELECT

RANK() OVER (ORDER BY total\_movies DESC) AS ranking,

ELT(release\_month, 'January', 'February', 'March', 'April', 'May', 'June',

'July', 'August', 'September', 'October', 'November', 'December') AS release\_month\_name,

release\_month,

total\_movies

FROM (

SELECT

MONTH(date\_published) AS release\_month,

COUNT(title) AS total\_movies

FROM

movie

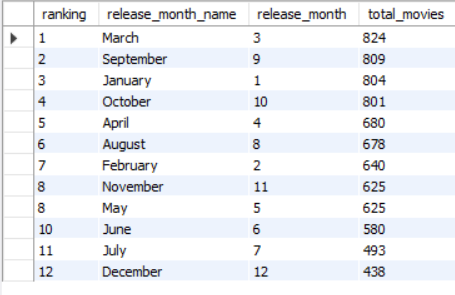
GROUP BY

MONTH(date\_published)

) AS subquery

ORDER BY

total\_movies DESC;



4. How many movies were produced in either the USA or India in the year 2019?

SELECT COUNT(title) AS number\_of\_movies\_in\_usa\_or\_india\_from\_2019

FROM movie

where (country="USA" OR country="INDIA") AND (YEAR="2019");



5. List the unique genres in the dataset, and count how many movies belong exclusively to one genre.

SELECT RANK()OVER(ORDER BY COUNT(movie\_id) DESC) AS ranking,COUNT(movie\_id) AS no\_movie,genre

FROM genre GROUP BY genre ORDER BY

COUNT(movie\_id) DESC;



6. Which genre has the highest total number of movies produced?

SELECT genre,COUNT(movie\_id) AS no\_movie

FROM genre

GROUP BY genre

ORDER BY COUNT(movie\_id) DESC LIMIT 1;



7. Calculate the average movie duration for each genre.

SELECT RANK()OVER(ORDER BY AVG(m.duration) DESC)AS ranking,g.genre,AVG(m.duration) FROM

genre g INNER JOIN movie m ON

g.movie\_id=m.id GROUP BY genre;



8. Identify actors or actresses who have appeared in more than three movies with an average rating below 5.

SELECT

m.id,

COUNT(m.title) AS movie\_count,

AVG(rs.median\_rating) AS avg\_ratings

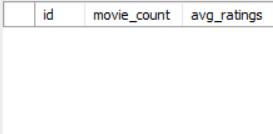
FROM movie m

INNER JOIN ratings rs ON m.id = rs.movie\_id

GROUP BY m.id

HAVING COUNT(m.title) > 3 AND AVG(rs.median\_rating) < 5;

NO OUTPUT



9. Find the minimum and maximum values for each column in the ratings table, excluding the movie\_id column.

SELECT MAX(avg\_rating) AS max\_averagerating,MIN(avg\_rating) AS min\_averagerating,

MAX(total\_votes) AS maxvotes,MIN(total\_votes) AS minvotes,

MAX(median\_rating) AS max\_medianrating,MIN(median\_rating) AS min\_medianrating

FROM ratings;



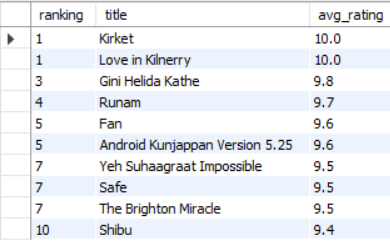
10. Which are the top 10 movies based on their average rating?

SELECT RANK()OVER (ORDER BY r.avg\_rating DESC)AS ranking,m.title,r.avg\_rating

FROM movie m inner join ratings r

ON m.id=r.movie\_id GROUP BY m.title,m.id

ORDER BY avg\_rating DESC LIMIT 10;



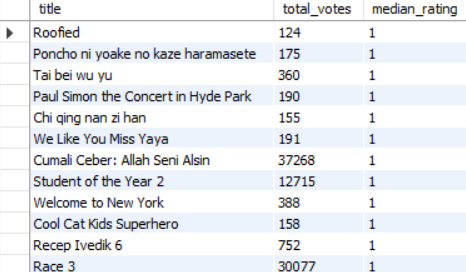
11. Summarize the ratings table by grouping movies based on their median ratings.

SELECT m.title,r.total\_votes,r.median\_rating

FROM movie m INNER JOIN ratings r

ON m.id=r.movie\_id GROUP BY r.movie\_id

ORDER BY median\_rating ASC;



12. How many movies, released in March 2017 in the USA within a specific genre, had more than 1,000 votes?

SELECT COUNT(m.title),m.year,m.country,g.genre

FROM movie m INNER JOIN genre g

ON m.id=g.movie\_id

INNER JOIN ratings r

ON r.movie\_id=m.id

WHERE year="2017" AND country="usa" AND genre="romance"AND total\_votes>"1000"

GROUP BY

m.year, m.country, g.genre;



13. Find movies from each genre that begin with the word “The” and have an average rating greater than 8.

SELECT g.genre,m.title,r.avg\_rating

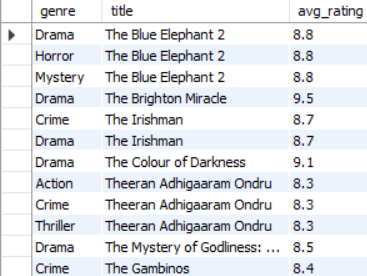
FROM movie m INNER JOIN genre g ON

m.id=g.movie\_id

INNER JOIN ratings r

ON r.movie\_id=m.id

HAVING title LIKE"the%" AND avg\_rating>"8";



14. Of the movies released between April 1, 2018, and April 1, 2019, how many received a median rating of 8?

SELECT COUNT(m.title) AS number\_movies\_relesed\_in\_from\_18to19,r.median\_rating

FROM movie m INNER JOIN ratings r ON

m.id=r.movie\_id

WHERE median\_rating="8" AND (date\_published >= '2018-04-01'

AND date\_published< '2019-04-01');



15. Do German movies receive more votes on average than Italian movies?

SELECT

CASE

WHEN (

SELECT SUM(r.total\_votes)

FROM movie m

INNER JOIN ratings r ON m.id = r.movie\_id

WHERE m.country = "germany"

) > (

SELECT SUM(r.total\_votes)

FROM movie m

INNER JOIN ratings r ON m.id = r.movie\_id

WHERE m.country = "italy"

) THEN "GERMANY HAS HIGHEST TOTAL"

ELSE "ITALY HAS HIGHEST TOTAL"

END AS RESULT;



16. Identify the columns in the names table that contain null values.

SELECT

CASE WHEN COUNT(\*) - COUNT(id) > 0 THEN 'id has nulls'

ELSE 'id has no nulls'

END AS id,

CASE WHEN COUNT(\*) - COUNT(name) > 0 THEN 'name has nulls'

ELSE 'name has no nulls'

END AS name,

CASE WHEN COUNT(\*) - COUNT(height) > 0 THEN 'height has nulls'

ELSE 'height has no nulls'

END AS height,

CASE WHEN COUNT(\*) - COUNT(date\_of\_birth) > 0 THEN 'D\_O\_B has nulls'

ELSE 'D\_O\_B has no nulls'

END AS D\_0\_B,

CASE WHEN COUNT(\*) - COUNT(known\_for\_movies ) > 0 THEN 'K\_F\_M has nulls'

ELSE 'K\_F\_M has no nulls'

END AS K\_F\_M

FROM names;



17. Who are the top two actors whose movies have a median rating of 8 or higher?

SELECT

n.name,rm.category,m.title,r.median\_rating

FROM

ratings r

INNER JOIN movie m ON m.id = r.movie\_id

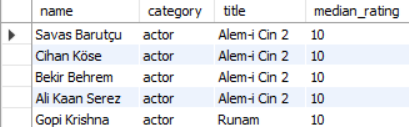
INNER JOIN role\_mapping rm

ON m.id = rm.movie\_id

INNER JOIN names n ON rm.name\_id = n.id

WHERE

r.median\_rating >= 8 ORDER BY median\_rating DESC LIMIT 5;

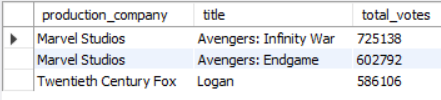


18. Which are the top three production companies based on the total number of votes their movies received?

SELECT m.production\_company,m.title,r.total\_votes

FROM movie m INNER JOIN ratings r

ON m.id=r.movie\_id ORDER BY total\_votes DESC LIMIT 3;

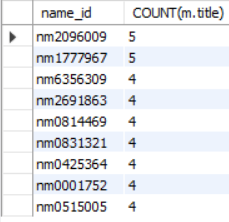


19. How many directors have worked on more than three movies?

SELECT d.name\_id,COUNT(m.title)

FROM movie m INNER JOIN director\_mapping d

ON m.id=d.movie\_id GROUP BY d.name\_id HAVING COUNT(m.title)>3 ORDER BY COUNT(m.title) DESC;

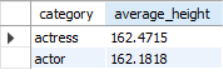


20. Calculate the average height of actors and actresses separately.

SELECT rm.category,AVG(n.height) AS average\_height

FROM names n INNER JOIN role\_mapping rm

ON n.id=rm.name\_id GROUP BY category ORDER BY AVG(n.height) DESC;



21. List the 10 oldest movies in the dataset along with their title, country, and director.

SELECT m.title,m.country,d.name\_id AS director\_id,m.year

FROM movie m INNER JOIN director\_mapping d

ON m.id=d.movie\_id ORDER BY year ASC LIMIT 10;



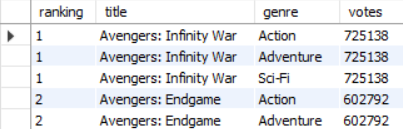
22. List the top 5 movies with the highest total votes, along with their genres.

SELECT DENSE\_RANK()OVER(ORDER BY total\_votes DESC) AS ranking,m.title,g.genre,r.total\_votes AS votes

FROM movie m INNER JOIN ratings r

ON m.id=r.movie\_id INNER JOIN genre g

ON m.id=g.movie\_id ORDER BY total\_votes DESC LIMIT 5;

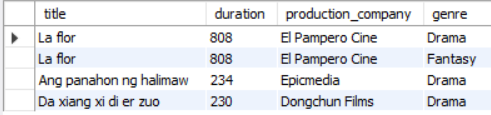


23. Identify the movie with the longest duration, along with its genre and production company.

SELECT m.title,m.duration,m.production\_company,g.genre

FROM movie m INNER JOIN genre g

ON m.id=g.movie\_id ORDER BY duration DESC LIMIT 4;



24. Determine the total number of votes for each movie released in 2018.

SELECT m.title,m.year,r.total\_votes

FROM movie m INNER JOIN ratings r

ON m.id=r.movie\_id WHERE year="2018" ORDER BY total\_votes DESC;



25. What is the most common language in which movies were produced?

SELECT DENSE\_RANK()OVER (ORDER BY COUNT(title) DESC) AS rabking,COUNT(title) AS number\_of\_movies,languages FROM movie

GROUP BY languages ORDER BY COUNT(TITLE) DESC LIMIT 5;

