**Advanced SQL – Reinforcement Project – IMDB dataset**

**Introduction**:

This dataset provides structured information about movies, including details on production, genres, ratings, and key contributors like directors and actors. It consists of multiple tables covering movie details, genres, ratings, cast, and crew, enabling in-depth analysis of industry trends, audience preferences, and movie performance.

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

**2.** **Genre:** Describes the genres associated with each movie.

**3.** **Director Mapping:** Maps movies to their directors.

**4.** **Role Mapping:** Maps actors/actresses to movies and specifies the role category (e.g.,actor, director, producer).

**5. Names:** Stores information about people (actors, directors, etc.), including their birthdates, heights, and known movies.

**6. Ratings:** Contains ratings information for movies, including the average rating, total votes, and median rating.

**Project Objective**:

The primary objective of this project is to:

• Reinforce key SQL concepts such as joins, aggregation, and grouping.

• Analyze and extract meaningful insights from a real-world movie dataset.

**Queries:**

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

SELECT 'movie' AS table\_name, COUNT(\*) AS total\_records FROM movie

UNION ALL

SELECT 'genre', COUNT(\*) FROM genre

UNION ALL

SELECT 'director\_mapping', COUNT(\*) FROM director\_mapping

UNION ALL

SELECT 'role\_mapping', COUNT(\*) FROM role\_mapping

UNION ALL

SELECT 'names', COUNT(\*) FROM names

UNION ALL

SELECT 'ratings', COUNT(\*) FROM ratings;

A screenshot of a computer

AI-generated content may be incorrect.

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

SELECT

SUM(CASE WHEN title IS NULL THEN 1 ELSE 0 END) AS title\_null\_count,

SUM(CASE WHEN year IS NULL THEN 1 ELSE 0 END) AS year\_null\_count,

SUM(CASE WHEN date\_published IS NULL THEN 1 ELSE 0 END) AS date\_published\_null\_count,

SUM(CASE WHEN duration IS NULL THEN 1 ELSE 0 END) AS duration\_null\_count,

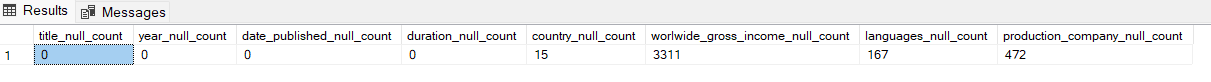
SUM(CASE WHEN country IS NULL THEN 1 ELSE 0 END) AS country\_null\_count,

SUM(CASE WHEN worlwide\_gross\_income IS NULL THEN 1 ELSE 0 END) AS worlwide\_gross\_income\_null\_count,

SUM(CASE WHEN languages IS NULL THEN 1 ELSE 0 END) AS languages\_null\_count,

SUM(CASE WHEN production\_company IS NULL THEN 1 ELSE 0 END) AS production\_company\_null\_count

FROM movie;



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

SELECT YEAR(date\_published) AS release\_year,

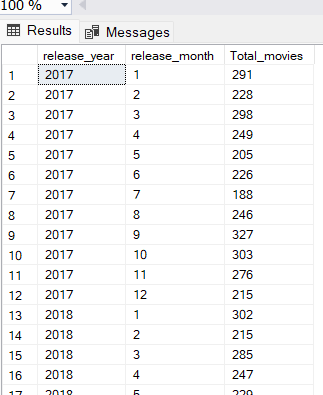
MONTH(date\_published) AS release\_month,

COUNT(\*) AS Total\_movies

FROM movie

GROUP BY YEAR(date\_published), MONTH(date\_published)

ORDER BY YEAR(date\_published), MONTH(date\_published);

 A table of numbers with numbers on it

AI-generated content may be incorrect.

A white grid with black numbers

AI-generated content may be incorrect.

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

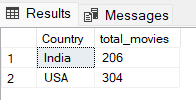
SELECT Country, COUNT(\*) AS total\_movies

FROM movie

WHERE Country IN ('USA', 'India')

AND YEAR(date\_published) = 2019

GROUP BY Country;



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

SELECT DISTINCT(genre), COUNT(movie\_id) AS total\_movies

FROM genre

WHERE movie\_id IN (

SELECT movie\_id

FROM genre

GROUP BY movie\_id

HAVING COUNT(genre) = 1)

GROUP BY genre

ORDER BY COUNT(movie\_id) DESC;



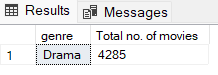
6. Which genre has the highest total number of movies produced?

SELECT TOP 1 genre, COUNT(movie\_id) AS 'Total no. of movies'

FROM genre

GROUP BY genre

ORDER BY COUNT(movie\_id) DESC;



7. Calculate the average movie duration for each genre.

SELECT g.genre, AVG(m.duration) AS 'Avg movie duration'

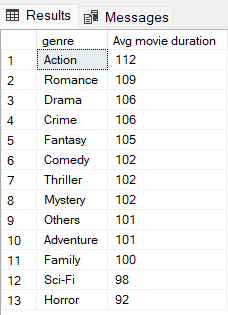
FROM movie m

INNER JOIN genre g

ON g.movie\_id = m.id

GROUP BY g.genre

ORDER BY AVG(m.duration) DESC ;



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

SELECT r.name\_id, n.name, COUNT(r.movie\_id) AS Movie\_count

FROM role\_mapping r

INNER JOIN ratings r1

ON r.movie\_id = r1.movie\_id

INNER JOIN names n

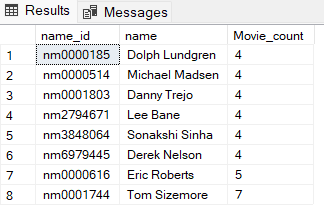
ON r.name\_id = n.id

WHERE r1.avg\_rating < 5

GROUP BY r.name\_id, n.name

HAVING COUNT(r.movie\_id) > 3

ORDER BY COUNT(r.movie\_id);



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

SELECT MIN(avg\_rating) AS 'Minimum Avg-rating',

MAX(avg\_rating) AS 'Maximum Avg-rating',

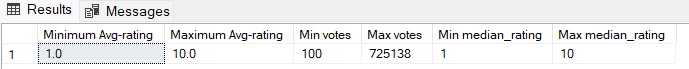
MIN(total\_votes) AS 'Min votes',

MAX(total\_votes) AS 'Max votes',

MIN(median\_rating) AS 'Min median\_rating',

MAX(median\_rating) AS 'Max median\_rating'

FROM ratings;



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

SELECT TOP 10 m.title, r1.avg\_rating

FROM movie m

INNER JOIN ratings r1 ON m.id = r1.movie\_id

ORDER BY r1.avg\_rating DESC;

A screenshot of a computer

AI-generated content may be incorrect.

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

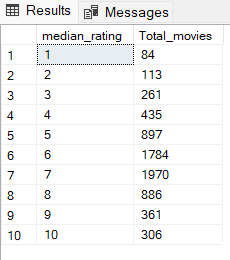
SELECT r1.median\_rating,COUNT(m.id) AS 'Total\_movies'

FROM movie m

INNER JOIN ratings r1 ON m.id = r1.movie\_id

GROUP BY r1.median\_rating

ORDER BY r1.median\_rating;



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

SELECT g.genre, COUNT(m.id) AS 'Total Movies'

FROM movie m

INNER JOIN Ratings r1 ON m.id = r1.movie\_id

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

WHERE YEAR(m.date\_published) = 2017

AND MONTH(m.date\_published) = 3

AND m.country = 'USA'

AND r1.total\_votes > 1000

GROUP BY g.genre

ORDER BY COUNT(m.id) DESC;

A screen shot of a computer

AI-generated content may be incorrect.

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

SELECT m.title, g.genre, r1.avg\_rating

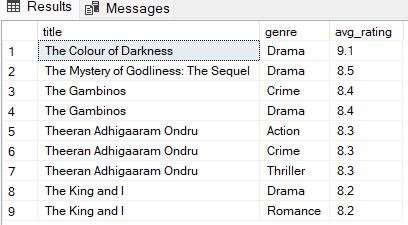
FROM movie m

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

INNER JOIN ratings r1 ON m.id = r1.movie\_id

WHERE m.title LIKE 'The%' AND r1.avg\_rating > 8

ORDER BY r1.avg\_rating DESC ;



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

SELECT r1.median\_rating, COUNT(m.id) AS total\_movies

FROM movie m

INNER JOIN ratings r1 ON m.id = r1.movie\_id

WHERE (Date\_published BETWEEN '2018-04-01' AND '2019-04-01') AND r1.median\_rating = 8

GROUP BY r1.median\_rating ;

A screenshot of a message

AI-generated content may be incorrect.

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

SELECT m.Country, AVG(r1.total\_votes) AS Avg\_votes

FROM movie m

INNER JOIN ratings r1 ON m.id = r1.Movie\_id

WHERE m.Country IN ('Germany', 'Italy')

GROUP BY m.country;

A screenshot of a computer

AI-generated content may be incorrect.

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

SELECT

SUM(CASE WHEN name IS NULL THEN 1 ELSE 0 END) AS Name\_nullCount,

SUM(CASE WHEN height IS NULL THEN 1 ELSE 0 END) AS height\_nullCount,

SUM(CASE WHEN date\_of\_birth IS NULL THEN 1 ELSE 0 END) AS DOB\_nullcount,

SUM(CASE WHEN known\_for\_movies IS NULL THEN 1 ELSE 0 END) AS Known\_for\_movies\_nullCount

FROM names;

A screenshot of a computer

AI-generated content may be incorrect.

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

SELECT TOP 2 n.name, COUNT(r1.median\_rating) AS total\_high\_rated\_movies

FROM names n

INNER JOIN role\_mapping rm ON rm.name\_id = n.id

INNER JOIN ratings r1 ON r1.Movie\_id = rm.movie\_id

WHERE rm.category = 'actor' AND r1.median\_rating >= 8

GROUP BY n.name

ORDER BY COUNT(r1.median\_rating) DESC;

A screenshot of a computer

AI-generated content may be incorrect.

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

SELECT TOP 3 m.production\_company, SUM(r1.total\_votes) AS total\_votes

FROM movie m

INNER JOIN ratings r1 ON m.id = r1.movie\_id

WHERE m.production\_company IS NOT NULL

GROUP BY m.production\_company

ORDER BY SUM(r1.total\_votes) DESC;

A screenshot of a computer

AI-generated content may be incorrect.

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

SELECT n.name, COUNT(dm.movie\_id) AS total\_movies

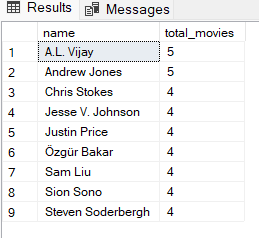
FROM director\_mapping dm

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

GROUP BY n.name

HAVING COUNT(dm.movie\_id) > 3

ORDER BY COUNT(dm.movie\_id) DESC;



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

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

FROM role\_mapping rm

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

WHERE rm.category IN ('actor', 'actress')

GROUP BY rm.category;

A screenshot of a computer

AI-generated content may be incorrect.

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

SELECT TOP 10 m.title, m.date\_published, m.country, n.name AS director\_name

FROM movie m

INNER JOIN director\_mapping dm ON m.id = dm.Movie\_id

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

ORDER BY m.date\_published ASC;

A screenshot of a computer

AI-generated content may be incorrect.

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

SELECT TOP 5 m.title, SUM(r1.total\_votes) AS total\_votes, STRING\_AGG(g.genre, ',') AS genres -- used string agg since multiple genre have same movies

FROM movie m

INNER JOIN ratings r1 ON m.id = r1.movie\_id

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

GROUP BY m.title

ORDER BY SUM(r1.total\_votes) DESC;

A screenshot of a computer

AI-generated content may be incorrect.

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

SELECT TOP 1 m.title, m.duration, m.production\_company, g.genre

FROM movie m

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

ORDER BY m.duration DESC;

A screenshot of a computer

AI-generated content may be incorrect.

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

SELECT m.title, SUM(r1.total\_votes) AS total\_votes

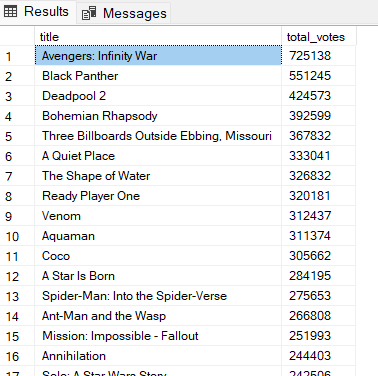
FROM movie m

INNER JOIN ratings r1 ON m.id = r1.movie\_id

WHERE year = 2018

GROUP BY m.title

ORDER BY SUM(r1.total\_votes) DESC;



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

SELECT TOP 1 languages, COUNT(\*) AS Total\_movies

FROM movie

WHERE languages IS NOT NULL

GROUP BY languages

ORDER BY COUNT(\*) DESC;

A white rectangular box with black text

AI-generated content may be incorrect.

**Conclusion & Key Insights**

1. **Movie Trends**
   * The number of movies made each year goes up and down, with some years having a big spike in releases.
   * The USA and India are the top movie-producing countries.
2. **Genres & Ratings**
   * Drama is the most common genre, followed by Action and Comedy.
   * Crime and Thriller movies tend to get higher ratings.
3. **Directors & Actors**
   * A few directors have worked on more than three movies, showing their influence in the industry.
4. **Production & Popularity**
   * Some production companies get way more votes, meaning their movies attract larger audiences.
   * The longest movie in the dataset is significantly longer than the rest.
5. **Other Findings**
   * English is the most commonly used language in movies.
   * A small number of movies get a huge amount of votes, making them stand out.

**Documented by**

**Kavitha Balasubramaniam**

**DA & DS (Dec Batch)**

**28-03-2025**