**MY SQL**

**Reinforcement Project**

**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

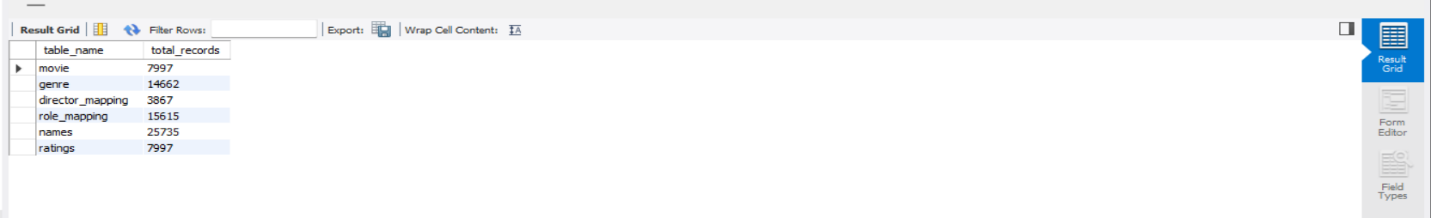
SELECT 'role\_mapping', COUNT(\*) FROM role\_mapping

UNION ALL

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

UNION ALL

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

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

SELECT

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

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

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

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

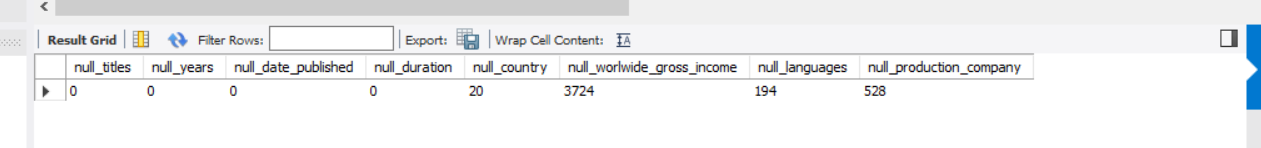
SUM(CASE WHEN country IS NULL THEN 1 ELSE 0 END) AS null\_country,

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

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

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

FROM movie;



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

SELECT

year AS release\_year,

COUNT(\*) AS total\_movies

FROM movie

GROUP BY year

ORDER BY release\_year;

**#3 (2 ) Monthwise releaed**

SELECT

YEAR(date\_published) AS release\_year,

MONTH(date\_published) AS release\_month,

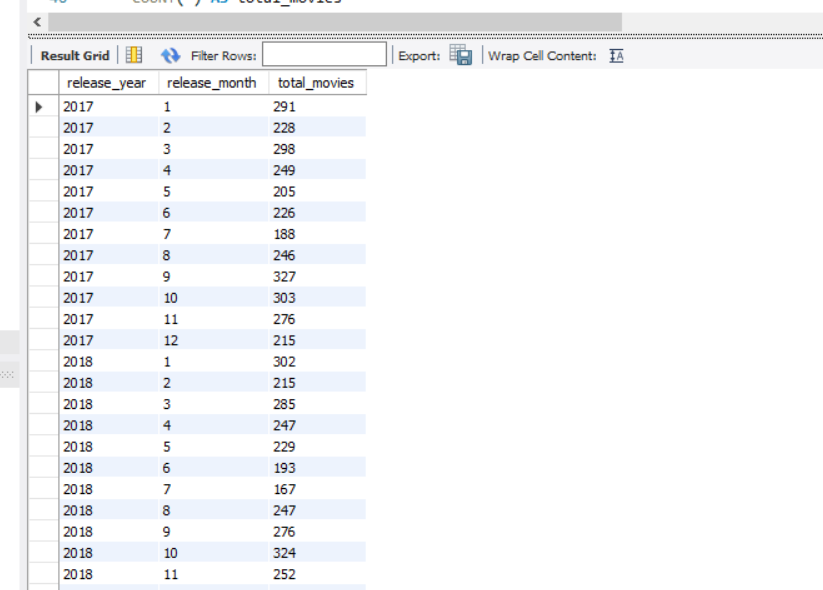
COUNT(\*) AS total\_movies

FROM movie

WHERE date\_published IS NOT NULL

GROUP BY release\_year, release\_month

ORDER BY release\_year, release\_month;



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

SELECT COUNT(\*) AS total\_movies\_usa\_india\_2019

FROM movie

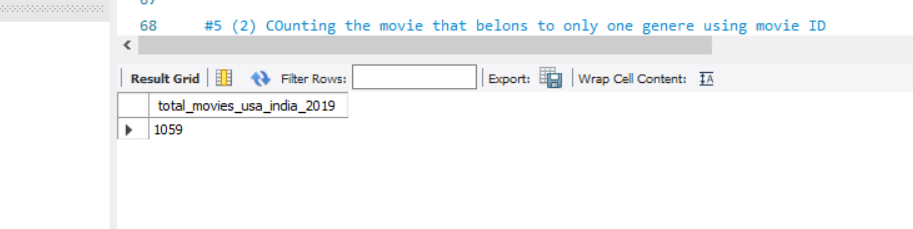
WHERE year = 2019

AND (

country LIKE '%USA%'

OR country LIKE '%India%'

);



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

SELECT DISTINCT genre

FROM genre

ORDER BY genre;

#5 (2) COunting the movie that belons to only one genere using movie ID

SELECT COUNT(\*) AS movies\_with\_one\_genre

FROM (

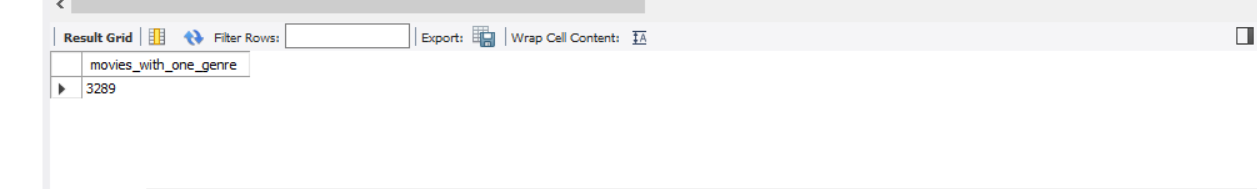
SELECT movie\_id

FROM genre

GROUP BY movie\_id

HAVING COUNT(\*) = 1

) AS single\_genre\_movies;



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

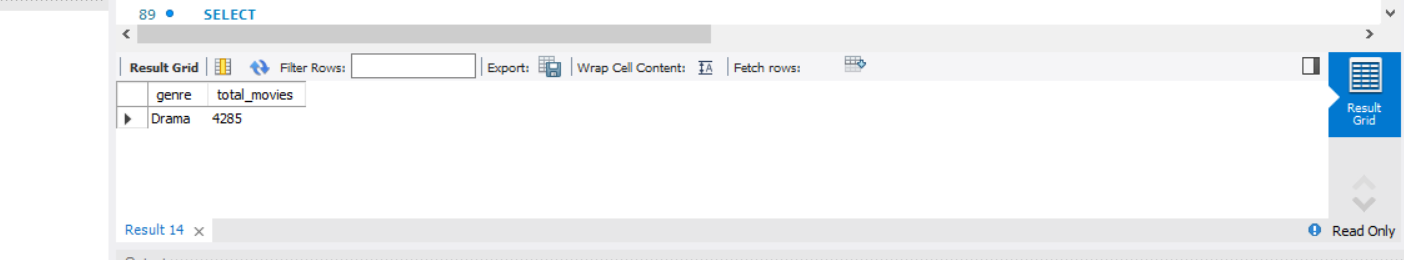
SELECT genre, COUNT(\*) AS total\_movies

FROM genre

GROUP BY genre

ORDER BY total\_movies DESC

LIMIT 1;



**7.Calculate the average movie duration for each genre.**

SELECT

g.genre,

ROUND(AVG(m.duration), 2) AS avg\_duration

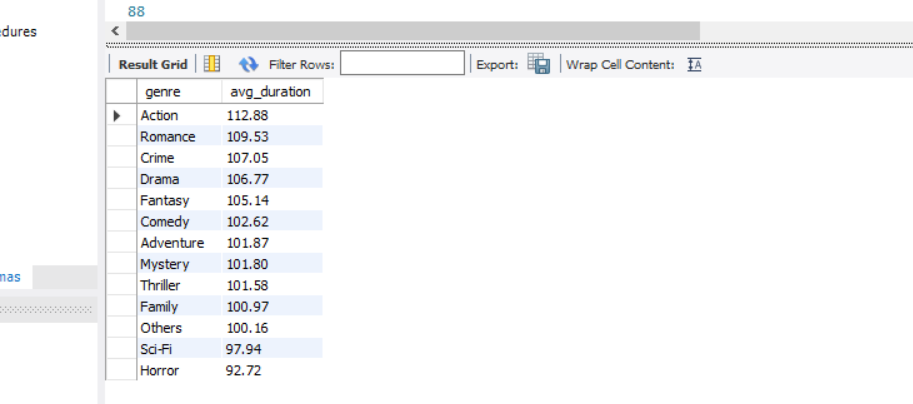
FROM genre g

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

WHERE m.duration IS NOT NULL

GROUP BY g.genre

ORDER BY avg\_duration DESC;



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

SELECT

n.name,

COUNT(\*) AS low\_rated\_movie\_count

FROM role\_mapping rm

JOIN ratings r ON rm.movie\_id = r.movie\_id

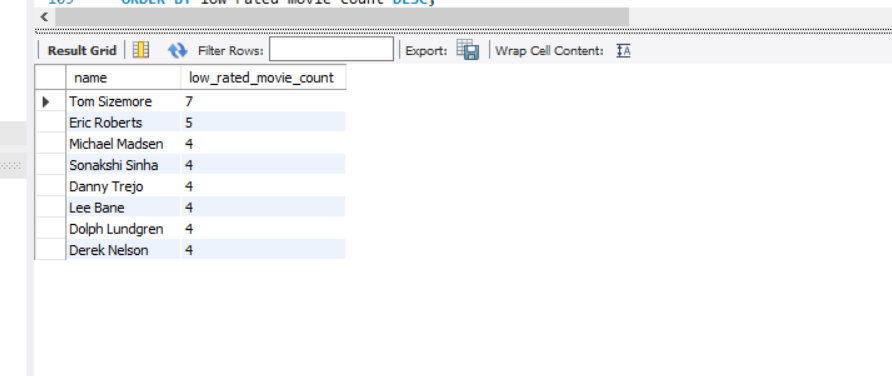
JOIN names n ON rm.name\_id = n.id

WHERE r.avg\_rating < 5

GROUP BY n.name

HAVING COUNT(\*) > 3

ORDER BY low\_rated\_movie\_count DESC;



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

SELECT

MIN(avg\_rating) AS min\_avg\_rating,

MAX(avg\_rating) AS max\_avg\_rating,

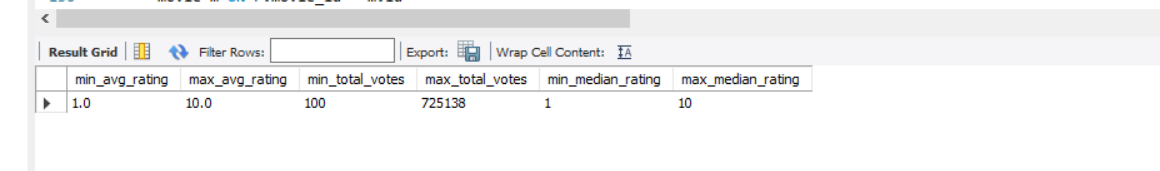
MIN(total\_votes) AS min\_total\_votes,

MAX(total\_votes) AS max\_total\_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

m.title,

r.avg\_rating

FROM

ratings r

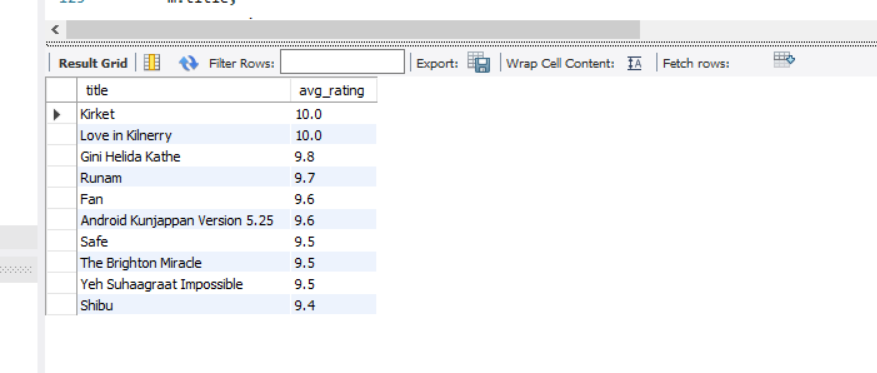
JOIN

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

ORDER BY

r.avg\_rating DESC

LIMIT 10;



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

SELECT

median\_rating,

COUNT(\*) AS movie\_count,

AVG(avg\_rating) AS average\_of\_avg\_rating,

SUM(total\_votes) AS total\_votes

FROM

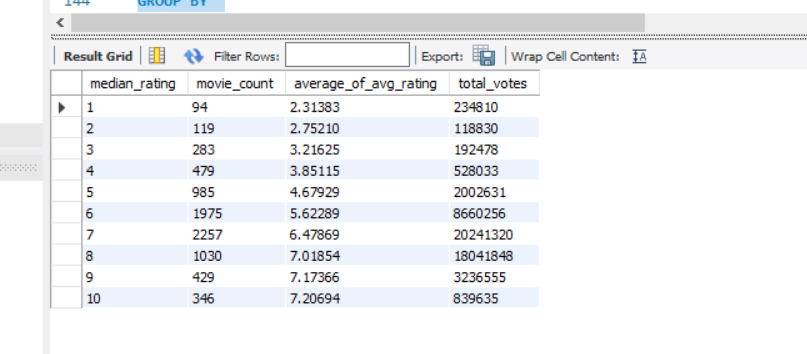
ratings

GROUP BY

median\_rating

ORDER BY

median\_rating;



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

SELECT

COUNT(\*) AS movie\_count

FROM

movie m

JOIN

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

JOIN

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

WHERE

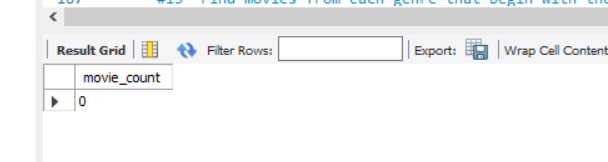
MONTH(m.date\_published) = 3

AND YEAR(m.date\_published) = 2017

AND m.country LIKE '%USA%'

AND g.genre = 'SPECIFIC\_GENRE'

AND r.total\_votes > 1000;



**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

JOIN

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

JOIN

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

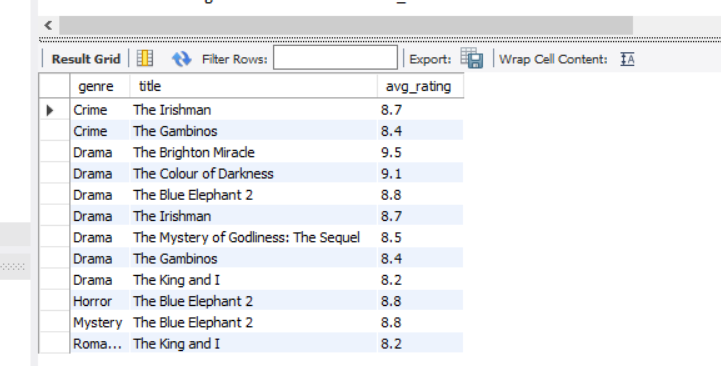
WHERE

m.title LIKE 'The %'

AND r.avg\_rating > 8

ORDER BY

g.genre, r.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 COUNT(\*) AS movie\_count

FROM movie m

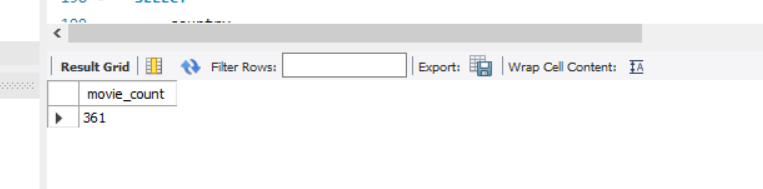
JOIN ratings r ON m.id = r.movie\_id

WHERE

m.date\_published >= '2018-04-01'

AND m.date\_published < '2019-04-02'

AND r.median\_rating = 8;



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

SELECT

country,

AVG(r.total\_votes) AS average\_votes

FROM

movie m

JOIN

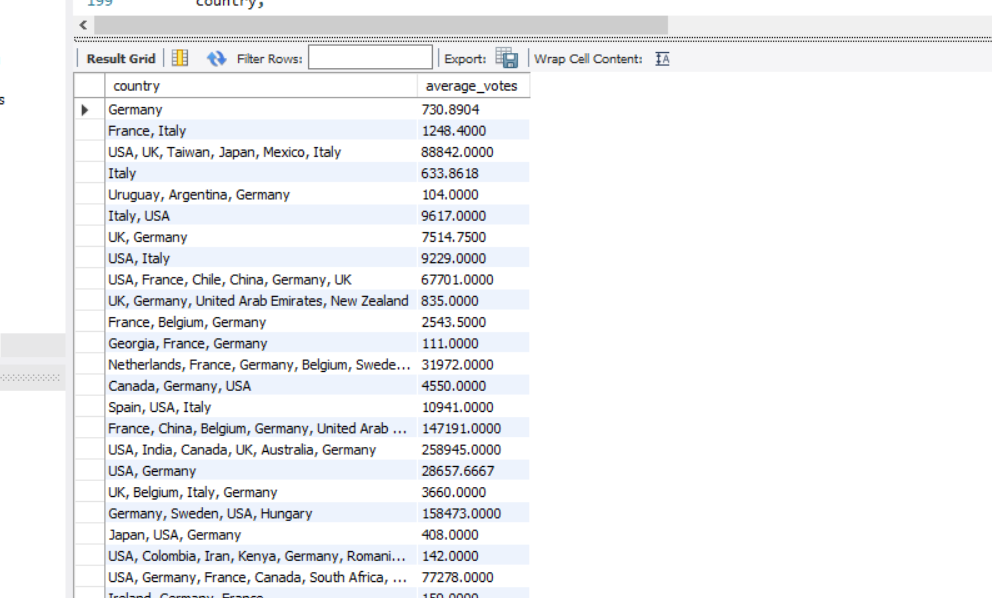
ratings r ON m.id = r.movie\_id

WHERE

country LIKE '%Germany%' OR country LIKE '%Italy%'

GROUP BY

country;



**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\_nulls,

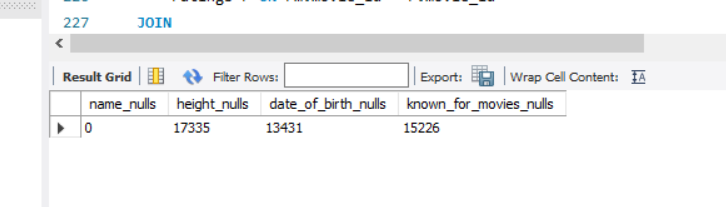
SUM(CASE WHEN height IS NULL THEN 1 ELSE 0 END) AS height\_nulls,

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

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

FROM

names;



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

SELECT

n.name,

COUNT(\*) AS high\_rated\_movies

FROM

role\_mapping rm

JOIN

ratings r ON rm.movie\_id = r.movie\_id

JOIN

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

WHERE

rm.category = 'actor'

AND r.median\_rating >= 8

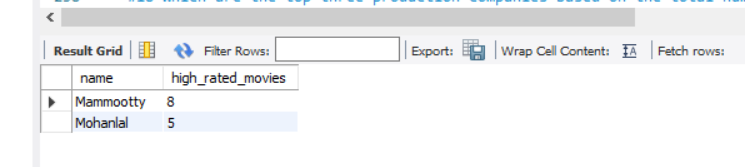
GROUP BY

n.name

ORDER BY

high\_rated\_movies DESC

LIMIT 2;



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

SELECT

m.production\_company,

SUM(r.total\_votes) AS total\_votes

FROM

movie m

JOIN

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

WHERE

m.production\_company IS NOT NULL

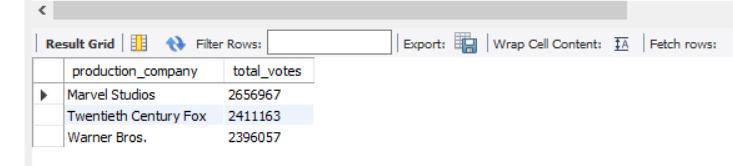
GROUP BY

m.production\_company

ORDER BY

total\_votes DESC

LIMIT 3;



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

SELECT COUNT(\*) AS director\_count

FROM (

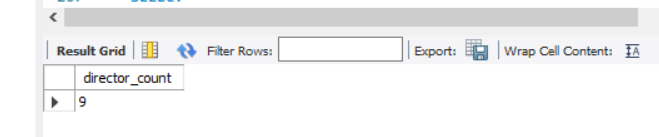
SELECT name\_id

FROM director\_mapping

GROUP BY name\_id

HAVING COUNT(movie\_id) > 3

) AS prolific\_directors;



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

SELECT

category,

AVG(height) AS average\_height

FROM

role\_mapping rm

JOIN

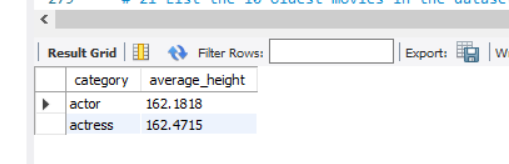
names n ON rm.name\_id = n.id

WHERE

category IN ('actor', 'actress') AND height IS NOT NULL

GROUP BY

category;



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

SELECT

m.title,

m.country,

n.name AS director\_name,

m.year

FROM

movie m

JOIN

director\_mapping dm ON m.id = dm.movie\_id

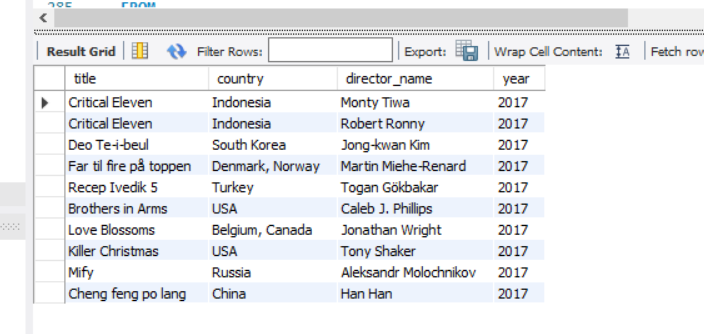
JOIN

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

ORDER BY

m.year ASC

LIMIT 10;



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

SELECT

m.title,

r.total\_votes,

GROUP\_CONCAT(g.genre ORDER BY g.genre SEPARATOR ', ') AS genres

FROM

ratings r

JOIN

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

JOIN

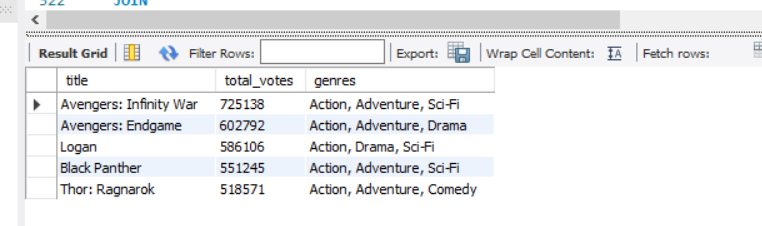
genre g ON m.id = g.movie\_id

GROUP BY

m.id, m.title, r.total\_votes

ORDER BY

r.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,

GROUP\_CONCAT(g.genre ORDER BY g.genre SEPARATOR ', ') AS genres,

m.production\_company

FROM

movie m

JOIN

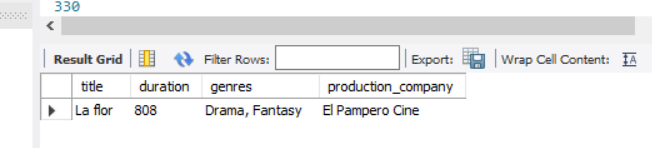
genre g ON m.id = g.movie\_id

WHERE

m.duration = (SELECT MAX(duration) FROM movie)

GROUP BY

m.id, m.title, m.duration, m.production\_company;



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

SELECT

m.title,

r.total\_votes

FROM

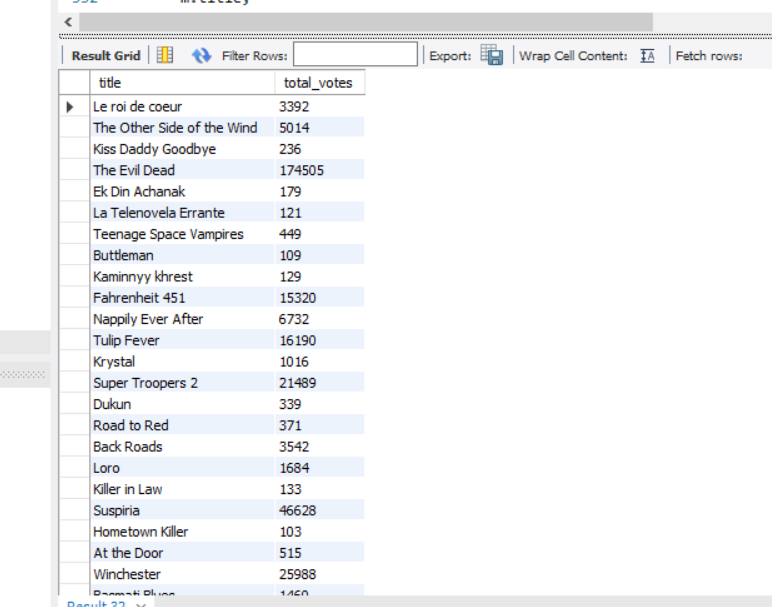
movie m

JOIN

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

WHERE

m.year = 2018;



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

SELECT

TRIM(SUBSTRING\_INDEX(languages, ',', 1)) AS primary\_language,

COUNT(\*) AS movie\_count

FROM

movie

WHERE

languages IS NOT NULL

GROUP BY

primary\_language

ORDER BY

movie\_count DESC

LIMIT 01;

