USE imdb;

/\* Now that you have imported the data sets, let's explore some of the tables.

To begin with, it is beneficial to know the shape of the tables and whether any column has null values. Further in this segment, you will take a look at 'movies' and 'genre' tables.\*/

### -- Q1. Find the total number of rows in each table of the schema?

using database metadata from information\_schema
SELECT table\_name,
table\_rows
FROM information\_schema.tables
WHERE table schema = 'imdb';

### -- Q2. Which columns in the movie table have null values?

\*\*\*Column names with at least one null value

WITH null info

AS (SELECT 'id' AS 'Column\_Name', Count(\*) AS Null\_Values

FROM movie

WHERE id IS NULL

**UNION ALL** 

SELECT 'title' AS 'Column\_Name', Count(\*) AS Null\_Values

FROM movie

WHERE title IS NULL

**UNION ALL** 

SELECT 'year' AS 'Column Name', Count(\*) AS Null Values

FROM movie

WHERE year IS NULL

**UNION ALL** 

SELECT 'date published' AS 'Column Name', Count(\*) AS Null Values

FROM movie

WHERE date\_published IS NULL

**UNION ALL** 

SELECT 'duration' AS 'Column\_Name', Count(\*) AS Null\_Values

FROM movie

WHERE duration IS NULL

**UNION ALL** 

SELECT 'country' AS 'Column\_Name', Count(\*) AS Null\_Values

FROM movie

WHERE country IS NULL

**UNION ALL** 

SELECT 'worlwide\_gross\_income' AS 'Column\_Name', Count(\*) AS Null\_Values

FROM movie

WHERE worlwide\_gross\_income IS NULL

**UNION ALL** 

SELECT 'languages' AS 'Column\_Name', Count(\*) AS Null\_Values

FROM movie

WHERE languages IS NULL

**UNION ALL** 

SELECT 'production\_company' AS 'Column\_Name', Count(\*) AS Null\_Values

FROM movie

WHERE production\_company IS NULL)

SELECT column name

FROM null info

WHERE Null Values > 0

ORDER BY null values DESC;

\*\*\* worlwide\_gross\_income, production\_company, languages, country have null values.

# -- Q3. Find the total number of movies released each year? How does the trend look month wise? (Output expected)

/\* Output format for the first part:

++							
Year		I	number_of_movies				
+		+					
	2017	1	2134				
	2018	1	•				
	2019	1	•				
+							

Output format for the second part of the question:

++							
1	month_num	number_of_movies					
+	_ +						
	1	134	I				
1	2	231					
				- 1			
+		+ */					

### -- code below:

 total number of movies released each year SELECT year, Count(id) AS number\_of\_movies FROM movie GROUP BY year;

-- number of movies released month-wise SELECT Month(date published) AS month released,

Count(id) AS number\_of\_movies FROM movie GROUP BY month\_released ORDER BY month\_released;

-- Q4. How many movies were produced in the USA or India in the year 2019??

### -- code below:

-- used regular expression to find strings containing USA or India

SELECT Count(id) AS Movie count

FROM movie

WHERE country REGEXP 'USA | India'

AND year = 2019;

/\* USA and India produced more than a thousand movies(you know the exact number!) in the year 2019.

Exploring table Genre would be fun!!

-- Q5. Find the unique list of the genres present in the data set? code below:

SELECT genre

FROM genre

**GROUP BY genre** 

ORDER BY genre;

/\* So, RSVP Movies plans to make a movie of one of these genres.

Now, wouldn't you want to know which genre had the highest number of movies produced in the last year?

Combining both the movie and genres table can give more interesting insights. \*/

- -- Q6. Which genre had the highest number of movies produced overall?
- -- code below:
- -- Top genre based on highest number of movies

SELECT genre,

Count(m.id) AS Number\_of\_Movies

FROM genre g

INNER JOIN movie m

ON g.movie\_id = m.id

GROUP BY genre

ORDER BY Number\_of\_Movies desc

LIMIT 1;

-- Drama with 4285 movies.

/\* So, based on the insight that you just drew, RSVP Movies should focus on the â€~Drama' genre. But wait, it is too early to decide. A movie can belong to two or more genres. So, let's find out the count of movies that belong to only one genre.\*/

-- Q7. How many movies belong to only one genre?

```
code below:
```

```
WITH one_genre
AS (SELECT movie_id,
Count(DISTINCT genre) AS n_genre
FROM genre
GROUP BY movie_id
HAVING n_genre = 1
)
SELECT Count(*) AS 'Number of movies with one genre'
FROM one_genre;
-- 3289
```

/\* There are more than three thousand movies which has only one genre associated with them. So, this figure appears significant.

#### -- Q8.What is the average duration of movies in each genre?

-- (Note: The same movie can belong to multiple genres.)

#### /\* Output format:

-- average duration of movies in each genre
SELECT genre,
 Round(Avg(duration), 2) AS avg\_duration
FROM genre g
 INNER JOIN movie m
 ON g.movie\_id = m.id
GROUP BY genre
ORDER BY avg\_duration DESC;

-- Movies in Action genre have relatively longer duration than other genres, closely followed by Romance.

/\* Now you know, movies of genre 'Drama' (produced highest in number in 2019) has the average duration of 106.77 mins.

Lets find where the movies of genre 'thriller' on the basis of number of movies.\*/

## -- Q9.What is the rank of the â€~thriller' genre of movies among all the genres in terms of number of movies produced?

-- (Hint: Use the Rank function)

```
/* Output format:
+-----+
                movie_count | genre_rank |
+-----+
               2312
+-----+*/
code below:
WITH genre_info
AS (SELECT genre,
Count(DISTINCT movie id) AS movie count,
Rank() OVER(ORDER BY Count(movie_id) DESC) AS genre_rank
FROM genre
GROUP BY genre
)
SELECT *
FROM genre_info
WHERE genre = 'Thriller';
-- Thriller genre ranks third based on number of movies.
/*Thriller movies is in top 3 among all genres in terms of number of movies
```

In the previous segment, you analysed the movies and genres tables. In this segment, you will analyse the ratings table as well.

To start with lets get the min and max values of different columns in the table\*/

-- Segment 2:

# -- Q10. Find the minimum and maximum values in each column of the ratings table except the movie id column?

```
/* Output format:
+-----+
| min_avg_rating | max_avg_rating | min_total_votes | max_total_votes
|min_median_rating|min_median_rating|
+-----+
```

1	0	1	5		177	
2000		0	8			
+		+	+	+	++*/	
Type your co	do bolove					

-- Type your code below:

FROM ratings;

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

/\* So, the minimum and maximum values in each column of the ratings table are in the expected range. This implies there are no outliers in the table.

# -- Q11. Which are the top 10 movies based on average rating?

-- code below:

-- It's ok if RANK() or DENSE\_RANK() is used too

-- top 10 movies based on avg\_ratingWITH movie\_rankingAS (SELECT title,

avg rating,

Rank() OVER(ORDER BY avg\_rating DESC) AS movie\_rank

FROM ratings r

INNER JOIN movie m

ON r.movie id = m.id)

SELECT \*
FROM movie\_ranking
WHERE movie rank <= 10;

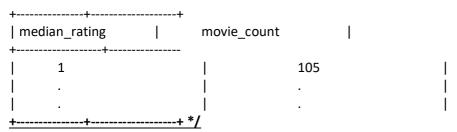
/\* Do you find you favourite movie FAN in the top 10 movies with an average rating of 9.6? If not, please check your code again!!

So, now that you know the top 10 movies, do you think character actors and filler actors can be from these movies?

Summarising the ratings table based on the movie counts by median rating can give an excellent insight.\*/

### -- Q12. Summarise the ratings table based on the movie counts by median ratings.

/\* Output format:



### -- code below:

-- Order by is good to have

SELECT median\_rating,

Count(movie\_id) AS movie\_count

FROM ratings

GROUP BY median rating

ORDER BY median\_rating;

-- sorting done on median\_rating

/\* Movies with a median rating of 7 is highest in number.

Now, let's find out the production house with which RSVP Movies can partner for its next project.\*/

# -- Q13. Which production house has produced the most number of hit movies (average rating > 8)??

/\* Output format:

+-----+
| production\_company|movie\_count | prod\_company\_rank|
+------+
| The Archers | 1 | 1 |
+-----+\*/

#### code below:

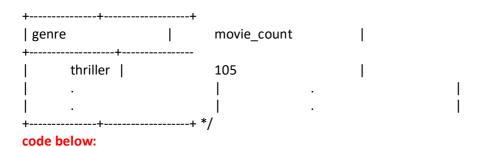
-- Ranking production companies based on total number of hit-movies produced WITH prod\_ranking

```
AS (SELECT production_company,
Count(movie_id) AS movie_count,
Rank() OVER(ORDER BY Count(movie_id) DESC) AS prod_company_rank
FROM ratings r
INNER JOIN movie m
ON r.movie_id = m.id
WHERE avg_rating > 8
AND production_company IS NOT NULL
GROUP BY production_company)
SELECT *
FROM prod_ranking
WHERE prod company rank = 1;
```

- -- Dream Warrior Pictures and National Theatre Live are both ranked as top production company.
- -- It's ok if RANK() or DENSE\_RANK() is used too
- -- Answer can be Dream Warrior Pictures or National Theatre Live or both

# -- Q14. How many movies released in each genre during March 2017 in the USA had more than 1,000 votes?

/\* Output format:



-- Number of movies in each genre satisfying the said conditions

SELECT genre,

Count(m.id) AS movie count

FROM genre g

INNER JOIN movie m

ON g.movie\_id = m.id

INNER JOIN ratings r

ON m.id = r.movie\_id

WHERE Month(date\_published) = 3

AND year = 2017

AND country = 'USA'

AND total\_votes > 1000

GROUP BY genre

ORDER BY movie count DESC;

- -- Drama genre has the most number of movies satisfying the given conditions.
- -- Lets try to analyse with a unique problem statement.
- -- Q15. Find movies of each genre that start with the word â€~The' and which have an average rating > 8?

### /\* Output format:

title		avg_rating	1	genre	I	
Theeran   .	   	8.3	1	I	Thriller	
1 .	1			l		
1 .	I			I		
+		+*/				

#### code below:

SELECT title, avg\_rating, genre
FROM genre g
INNER JOIN movie m
ON g.movie\_id = m.id
INNER JOIN ratings r
ON m.id = r.movie\_id
WHERE title LIKE 'The%'
AND avg\_rating > 8
ORDER BY avg\_rating DESC;

- -- The Brighton Miracle has the highest average rating among all the movies starting with 'The'.
- -- You should also try your hand at median rating and check whether the â€~median rating' column gives any significant insights.
- -- Q16. Of the movies released between 1 April 2018 and 1 April 2019, how many were given a median rating of 8?

```
SELECT Count(movie_id) AS movie_count

FROM ratings r

INNER JOIN movie m

ON r.movie_id = m.id

WHERE date_published BETWEEN '2018-04-01' AND '2019-04-01'

AND median rating = 8;
```

-- A total of 361 movies with median rating 8 were published between dates, '2018-04-01' and '2019-04-01'.

### -- Q17. Do German movies get more votes than Italian movies?

-- Hint: Here you have to find the total number of votes for both German and Italian movies. code below:

-- Comparing total votes with respect to country of origin, between Germany and Italy

SELECT country,

Sum(total\_votes) AS Total\_votes

FROM ratings r

INNER JOIN movie m

ON r.movie id = m.id

WHERE country IN ('Germany', 'Italy')

GROUP BY country;

-- Comparing total votes with respect to language the movies are available in, between German and Italian

WITH german\_movies

AS (SELECT languages, SUM(total\_votes) AS Total\_votes

FROM ratings r

inner join movie m

ON r.movie id = m.id

WHERE languages LIKE '%German%'

**GROUP BY languages)** 

SELECT 'German' AS LANGUAGE,

SUM(total\_votes) AS Total\_votes

FROM german\_movies

#### UNION

(WITH italian\_movies

AS (SELECT languages, SUM(total votes) AS Total votes

FROM ratings r

inner join movie m

ON r.movie\_id = m.id

WHERE languages LIKE '%Italian%'

**GROUP BY languages)** 

SELECT 'Italian' AS LANGUAGE,

SUM(total\_votes) AS Total\_votes

FROM italian\_movies);

- -- In both of these cases German movies get more votes than Italian movies.
- -- Answer is Yes

/\* Now that you have analysed the movies, genres and ratings tables, let us now analyse another table, the names table. Let's begin by searching for null values in the tables.\*/ -- Segment 3: -- Q18. Which columns in the names table have null values?? /\*Hint: You can find null values for individual columns or follow below output format +-----+ | name\_nulls | height\_nulls | date\_of\_birth\_nulls | known\_for\_movies\_nulls | - 1 0 123 - 1 1234 12345 | +-----+\*/ -- Type your code below: 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

/\* There are no Null value in the column 'name'.

END) AS known for movies nulls

FROM names;

The director is the most important person in a movie crew.

WHEN known for movies IS NULL THEN 1

Let's find out the top three directors in the top three genres who can be hired by RSVP Movies.\*/

- -- Q19. Who are the top three directors in the top three genres whose movies have an average rating > 8?
- -- (Hint: The top three genres would have the most number of movies with an average rating > 8.) /\* Output format:

```
movie count
director name
James Mangold
-- Type your code below:
-- using cte with genre ranking
WITH top_genres AS
     SELECT genre
     FROM
              genre
     INNER JOIN ratings
     using (movie_id)
     WHERE avg_rating > 8
     GROUP BY genre
     ORDER BY Count(movie_id) DESC limit 3),
-- top 3 directors based on total number of movies in top 3 genres
top_directors AS
(
     SELECT n.NAME
                                         AS director_name,
           Count(g.movie id)
                                          AS movie_count,
           Rank() OVER(ORDER BY Count(g.movie_id) DESC) AS director_rank
     FROM
              genre g
     INNER JOIN director_mapping dm
     ON
             g.movie id = dm.movie id
     INNER JOIN names n
             dm.name_id = n.id
     INNER JOIN ratings r
     ON
             g.movie_id = r.movie_id,
           top genres
     WHERE avg_rating > 8
     AND
            g.genre IN (top_genres.genre)
     GROUP BY n.NAME)
SELECT director name,
   movie_count
FROM top_directors
WHERE director_rank <=3;
/* Did not use LIMIT clause as it will only give top 'n' records as output irrespective of movie_count.
Here Soubin Shahir, Joe Russo and Anthony Russo have same movie_count and therefore it's important
that we display
them all intead of just first 3.*/
```

/\* James Mangold can be hired as the director for RSVP's next project. Do you remeber his movies, 'Logan' and 'The Wolverine'.

Now, let's find out the top two actors.\*/

## -- Q20. Who are the top two actors whose movies have a median rating >= 8?

```
/* Output format:
+----+
actor_name movie_count
+-----
|Christain Bale |
                        1
code below:
SELECT
 n.name AS actor_name,
 COUNT(r.movie_id) AS movie_count
FROM
 names n
      INNER JOIN role_mapping rm ON n.id = rm.name_id
                              ON rm.movie id = r.movie id
      INNER JOIN ratings r
WHERE
 median_rating >= 8
GROUP BY n.name
ORDER BY movie_count DESC
LIMIT 2;
```

/\*Used limit clause here as the movie\_count corresponding to 3rd actor is less than that of second actor.

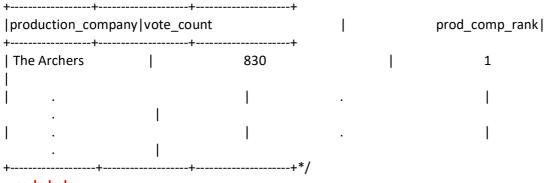
It can be easily verified by setting LIMIT as 3.\*/

-- Mammootty is the top actor based on total number of movies with median rating greater than 8. He is followed by Mohanlal.

/\* Have you find your favourite actor 'Mohanlal' in the list. If no, please check your code again. RSVP Movies plans to partner with other global production houses. Let's find out the top three production houses in the world.\*/

# -- Q21. Which are the top three production houses based on the number of votes received by their movies?

### /\* Output format:



#### -- code below:

-- Top production houses based on number of votes

WITH prod\_info

AS (SELECT production\_company,

Sum(total\_votes) AS vote\_count,

Rank() OVER(ORDER BY Sum(total\_votes) DESC) AS prod\_comp\_rank

FROM ratings r

INNER JOIN movie m

ON r.movie id = m.id

GROUP BY production\_company)

SELECT \*

FROM prod\_info

WHERE prod\_comp\_rank <= 3;

-- Marvel studios tops the total vote count, followed by Twentieth century fox and Warner Bros.

/\*Yes Marvel Studios rules the movie world.

So, these are the top three production houses based on the number of votes received by the movies they have produced.

Since RSVP Movies is based out of Mumbai, India also wants to woo its local audience. RSVP Movies also wants to hire a few Indian actors for its upcoming project to give a regional feel.

# -- Q22. Rank actors with movies released in India based on their average ratings. Which actor is at the top of the list?

- -- Note: The actor should have acted in at least five Indian movies.
- -- (Hint: You should use the weighted average based on votes. If the ratings clash, then the total number of votes should act as the tie breaker.)

# /\* Output format: | actor\_name | total\_votes movie count actor\_avg\_rating |actor\_rank 1 Yogi Babu 3455 11 8.42 1 code below: /\*Sorting and Ranking of indian actors with at least 5 movies on weighted average of movie ratings, with weights being 'total\_votes', and 2nd level sorting with total number of votes.\*/ WITH ind\_actors AS (SELECT n.NAME, Sum(total votes) AS total\_votes, Count(r.movie\_id) AS movie\_count,

Rank() OVER(ORDER BY Round(Sum(total\_votes \* avg\_rating)/Sum(total\_votes), 2) DESC,

AS

Sum(total\_votes) DESC) actor rank

FROM names n

actor avg rating,

INNER JOIN role\_mapping rm

Round(Sum(total\_votes \* avg\_rating) / Sum(total\_votes), 2) AS

ON n.id = rm.name\_id

INNER JOIN movie m

ON rm.movie\_id = m.id

INNER JOIN ratings r

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

WHERE country = 'India'

GROUP BY n.NAME

HAVING movie\_count >= 5)

**SELECT** \*

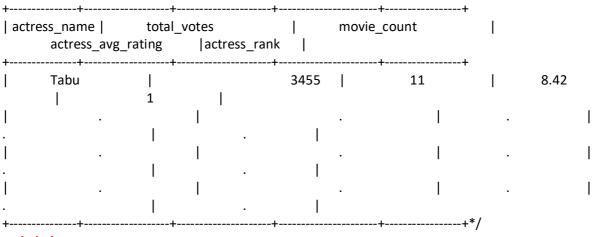
FROM ind\_actors;

<sup>--</sup> Top actor is Vijay Sethupathi

# **\_Q23.Find out the top five actresses in Hindi movies released in India** based on their average ratings?

- -- Note: The actresses should have acted in at least three Indian movies.
- -- (Hint: You should use the weighted average based on votes. If the ratings clash, then the total number of votes should act as the tie breaker.)

#### /\* Output format:



#### code below:

/\*Sorting and Ranking of indian actresses with at least 3 movies on weighted average of movie ratings, with weights being 'total\_votes', and 2nd level sorting with total number of votes.\*/

WITH ind actress

AS (SELECT n.NAME AS

actress\_name,

Sum(total votes) AS

total votes,

Count(r.movie id) AS

movie count,

Round(Sum(total\_votes \* avg\_rating) / Sum(total\_votes), 2) AS

actor\_avg\_rating,

Rank() OVER(ORDER BY Round(Sum(total\_votes \* avg\_rating)/Sum(total\_votes), 2) DESC,

Sum(total votes) DESC) AS

actress\_rank

FROM names n

INNER JOIN role\_mapping rm

ON n.id = rm.name\_id

INNER JOIN movie m

ON rm.movie\_id = m.id

INNER JOIN ratings r

ON rm.movie id = r.movie id

WHERE country = 'India'

AND category = 'actress'

AND languages = 'Hindi'

GROUP BY n.NAME
HAVING movie\_count >= 3)
SELECT \*
FROM ind\_actress
WHERE actress\_rank <= 5;

/\* Taapsee Pannu tops with average rating 7.74.

Now let us divide all the thriller movies in the following categories and find out their numbers.\*/

# /\* Q24. Select thriller movies as per avg rating and classify them in the following category:

Rating > 8: Superhit movies
Rating between 7 and 8: Hit movies
Rating between 5 and 7: One-time-watch movies
Rating < 5: Flop movies

#### code below:

-- Classifying movies of Thriller genre
-- Using case statement to create classes of avg\_rating as variable, 'Movie\_type'
SELECT title AS movie\_title,
avg\_rating,
CASE
WHEN avg\_rating > 8 THEN 'Superhit movie'
WHEN avg\_rating BETWEEN 7 AND 8 THEN 'Hit movie'
WHEN avg\_rating BETWEEN 5 AND 7 THEN 'One-time-watch movie'
ELSE 'Flop movie'
END Movie\_type
FROM genre g

INNER JOIN movie m

ON g.movie\_id = m.id

INNER JOIN ratings r

ON m.id = r.movie\_id

WHERE g.genre = 'Thriller'

ORDER BY movie\_title;

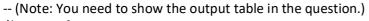
-- ordered by title

/\* Until now, you have analysed various tables of the data set.

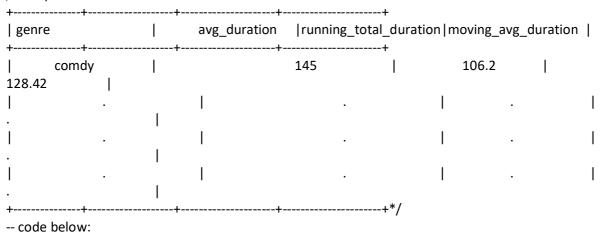
Now, you will perform some tasks that will give you a broader understanding of the data in this segment.\*/

-- Segment 4:

# -- Q25. What is the genre-wise running total and moving average of the average movie duration?



/\* Output format:

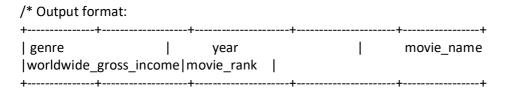


```
SELECT genre,
Round(Avg(duration)) AS avg_duration,
round(sum(Avg(duration)) OVER w1, 1) AS running_total_duration,
round(avg(avg(duration)) OVER w2, 2) AS moving_avg_duration
FROM genre g
INNER JOIN movie m
ON g.movie_id = m.id
GROUP BY genre
WINDOW w1 AS (ORDER BY genre rows UNBOUNDED PRECEDING),
w2 AS (ORDER BY genre rows BETWEEN 2 PRECEDING AND 2 following);
```

- -- Round is good to have and not a must have; Same thing applies to sorting
- -- Let us find top 5 movies of each year with top 3 genres.

# - Q26. Which are the five highest-grossing movies of each year that belong to the top three genres?

-- (Note: The top 3 genres would have the most number of movies.)



```
comedy
                                             2017
                                                               indian
                                                                                    $103244842
               1
                                                                                           1
                       ١
                       1
-- code below:
-- checking data-type of worldwide gross income column
SELECT column name,
data type
FROM information_schema.columns
WHERE table schema = 'imdb'
AND table name = 'movie'
AND column_name = 'worlwide_gross_income';
-- 'varchar'.
-- Top 3 Genres based on most number of movies(most number of movies with >8 avg_rating)
WITH top_genres AS
       SELECT genre
FROM genre g
INNER JOIN ratings r
ON g.movie id = r.movie id
WHERE avg_rating > 8
GROUP BY genre
ORDER BY Count(r.movie_id) DESC
LIMIT 3
),
-- worldwide gross income of movies of each year from top 3 genre
-- Converting worldwide gross income datatype from 'varchar' to decimal
-- Converting values in INR to to dollars after the data-type is corrected using conversion equation, 1
USD = 75 INR
movie_income AS
(
       SELECT g.genre, year, title AS movie_name,
CASE
WHEN worlwide_gross_income LIKE 'INR%'
THEN Cast(Replace(worlwide_gross_income, 'INR', ") AS DECIMAL(12)) / 75
WHEN worlwide_gross_income LIKE '$%'
THEN Cast(Replace(worlwide_gross_income, '$', '') AS DECIMAL(12))
ELSE Cast(worlwide_gross_income AS DECIMAL(12))
END worldwide_gross_income
FROM genre g
INNER JOIN movie m
```

```
ON g.movie id = m.id,
top_genres
WHERE g.genre IN (top genres.genre)
-- group by for distinct movie titles. To avoid repetitions, since one movie can belong to many genres
GROUP BY movie_name
ORDER BY year
),
-- five highest-grossing movies of each year from top 3 genre
top movies AS
(SELECT *,
Dense_rank() OVER(partition BY year ORDER BY worldwide_gross_income DESC) AS movie_rank
FROM movie income
)
SELECT *
FROM top_movies
WHERE movie_rank <= 5;
/*'Star Wars: Episode VIII - The Last Jedi' is on top for year 2017,
'Avengers: Infinity War' is ranked one for year 2018,
'Avengers: Endgame' is ranked one for the year 2019*/
```

# -- Q27. Which are the top two production houses that have produced the highest number of hits (median rating >= 8) among multilingual movies?

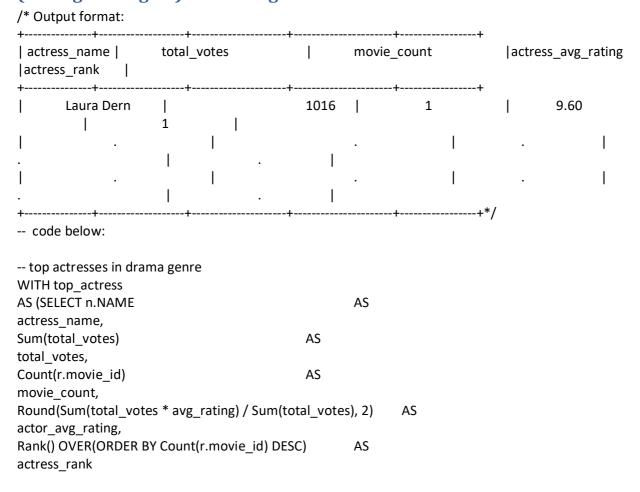
```
+-----+
|production_company |movie_count
                                               prod_comp_rank|
+----+
| The Archers
                 1
                             830
                                                           1
-- code below:
-- Ranking production houses based on number of hit multilingual movies
WITH prod_comp_info
AS (SELECT production_company,
Count(movie id)
                                   AS movie count,
Rank() over(ORDER BY Count(movie_id) DESC) AS prod_comp_rank
FROM ratings r
INNER JOIN movie m
ON r.movie id = m.id
```

/\* Output format:

```
WHERE production_company IS NOT NULL
AND median_rating >= 8
AND Position(',' IN languages) > 0
GROUP BY production_company)
SELECT *
FROM prod_comp_info
WHERE prod_comp_rank <= 2;
```

- -- Star Cinema, Twentieth Century Fox are the top two production houses in terms of number of hit multilingual movies.
- -- Multilingual is the important piece in the above question. It was created using POSITION(',' IN languages)>0 logic
- -- If there is a comma, that means the movie is of more than one language

# -- Q28. Who are the top 3 actresses based on number of Super Hit movies (average rating >8) in drama genre?

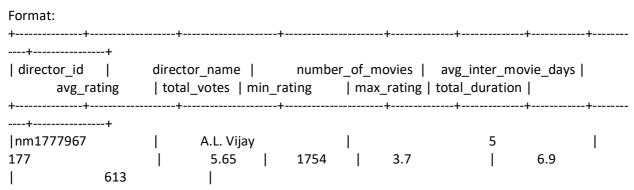


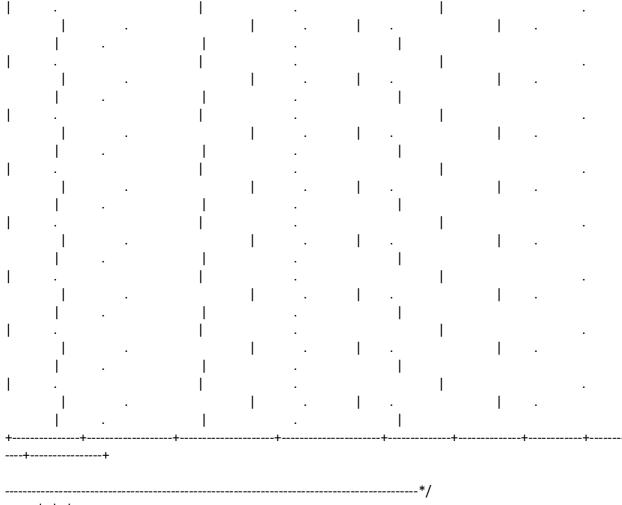
```
FROM names n
INNER JOIN role mapping rm
ON n.id = rm.name id
INNER JOIN genre g
ON rm.movie_id = g.movie_id
INNER JOIN ratings r
ON rm.movie_id = r.movie_id
WHERE category = 'actress'
AND genre = 'Drama'
AND avg rating > 8
GROUP BY n.NAME
SELECT *
FROM top actress
WHERE actress_rank <= 3;
/* Did not use LIMIT clause as it will only give top 'n' records as output irrespective of movie count.
Here first four actresses in the output have same movie_count and therefore it's important that we
display
```

# /\* Q29. Get the following details for top 9 directors (based on number of movies)

Director id
Name
Number of movies
Average inter movie duration in days
Average movie ratings
Total votes
Min rating
Max rating
total movie durations

them all intead of just first 3.\*/





<sup>--</sup> code below:

WITH director info

AS (SELECT dm.name\_id,

n.name,

dm.movie\_id,

r.avg\_rating,

r.total\_votes,

m.duration,

date\_published,

Lag(date\_published, 1) OVER(PARTITION BY dm.name\_id

ORDER BY date\_published) AS previous\_date\_published

FROM names n

INNER JOIN director\_mapping dm

ON n.id = dm.name\_id

INNER JOIN movie m

<sup>--</sup> creating a new variable/column in output to display date of last publication for every record, 'Previous\_date\_published'

<sup>--</sup> Summary of directors with a new variabe, 'Previous\_date\_published'

```
ON dm.movie id = m.id
INNER JOIN ratings r
ON m.id = r.movie id),
-- renaming columns and ranking directors on number_of_movies
top_directors
AS (SELECT name_id AS
director_id,
NAME AS
director_name,
Count(movie id) AS
number_of_movies,
Round(Avg(Datediff(date published, previous date published))) AS
avg_inter_movie_days,
Round(sum(avg_rating*total_votes)/sum(total_votes), 2) AS
avg rating,
Sum(total_votes) AS
total votes,
Round(Min(avg_rating), 1) AS
min_rating,
Round(Max(avg_rating), 1) AS
max rating,
Sum(duration)
                AS
total duration,
Rank() OVER(ORDER BY Count(movie_id) DESC)
                                                      AS
director_rank
FROM director info
GROUP BY director_id)
-- top 9 directors' details
SELECT director_id,
director name,
number_of_movies,
avg_inter_movie_days,
avg rating,
total_votes,
min_rating,
max_rating,
total duration
FROM top_directors
WHERE director_rank <= 9;
```