USE imdb;

/\* Now that you have imported the data sets, letâ€™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.\*/

-- Segment 1:

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

-- Type your code below:

-- Number of rows = 3867

SELECT COUNT(\*) FROM DIRECTOR\_MAPPING;

-- Number of rows = 14662

SELECT COUNT(\*) FROM GENRE ;

-- Number of rows = 7997

SELECT COUNT(\*) FROM MOVIE;

-- Number of rows = 25735

SELECT COUNT(\*) FROM NAMES;

-- Number of rows = 7997

SELECT COUNT(\*) FROM RATINGS;

-- Number of rows = 15615

SELECT COUNT(\*) FROM ROLE\_MAPPING;

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

-- Type your code below:

-- Query to count the number of nulls in each column using case statements

SELECT Sum(CASE

WHEN id IS NULL THEN 1

ELSE 0

END) AS ID\_NULL\_COUNT,

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;

-- Country, worlwide\_gross\_income, languages and production\_company columns have NULL values

-- Now as you can see four columns of the movie table has null values. Let's look at the at the movies released each year.

-- 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 | number\_of\_movies|

+-------------------+----------------

| 2017 | 2134 |

| 2018 | . |

| 2019 | . |

+---------------+-------------------+

Output format for the second part of the question:

+---------------+-------------------+

| month\_num | number\_of\_movies|

+---------------+----------------

| 1 | 134 |

| 2 | 231 |

| . | . |

+---------------+-------------------+ \*/

-- Type your code below:

-- Number of movies released each year

SELECT year,

Count(title) AS NUMBER\_OF\_MOVIES

FROM movie

GROUP BY year;

-- Number of movies released each month

SELECT Month(date\_published) AS MONTH\_NUM,

Count(\*) AS NUMBER\_OF\_MOVIES

FROM movie

GROUP BY month\_num

ORDER BY month\_num;

/\*The highest number of movies is produced in the month of March.

So, now that you have understood the month-wise trend of movies, letâ€™s take a look at the other details in the movies table.

We know USA and India produces huge number of movies each year. Lets find the number of movies produced by USA or India for the last year.\*/

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

-- Type your code below:

-- Pattern matching using LIKE operator for country column

SELECT Count(DISTINCT id) AS number\_of\_movies, year

FROM movie

WHERE ( country LIKE '%INDIA%'

OR country LIKE '%USA%' )

AND year = 2019;

-- 1059 movies were produced in the USA or India in the 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!!

Letâ€™s find out the different genres in the dataset.\*/

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

-- Type your code below:

-- Finding unique genres using DISTINCT keyword

SELECT DISTINCT genre

FROM genre;

-- Movies belong to 13 genres in the dataset.

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

Now, wouldnâ€™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. \*/

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

Now, wouldnâ€™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?

-- Type your code below:

-- Using LIMIT clause to display only the genre with highest number of movies produced

SELECT genre,

Count(m.id) AS number\_of\_movies

FROM movie AS m

INNER JOIN genre AS g

where g.movie\_id = m.id

GROUP BY genre

ORDER BY number\_of\_movies DESC limit 1 ;

-- 4265 Drama movies were produced in total and are the highest among all genres.

/\* So, based on the insight that you just drew, RSVP Movies should focus on the â€˜Dramaâ€™ genre.

But wait, it is too early to decide. A movie can belong to two or more genres.

So, letâ€™s find out the count of movies that belong to only one genre.\*/

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

-- Type your code below:

-- Using genre table to find movies which belong to only one genre

-- Grouping rows based on movie id and finding the distinct number of genre each movie belongs to

-- Using the result of CTE, we find the count of movies which belong to only one genre

WITH movies\_with\_one\_genre

AS (SELECT movie\_id

FROM genre

GROUP BY movie\_id

HAVING Count(DISTINCT genre) = 1)

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

FROM movies\_with\_one\_genre;

-- 3289 movies belong to only one genre

/\* There are more than three thousand movies which has only one genre associated with them.

So, this figure appears significant.

Now, let's find out the possible duration of RSVP Moviesâ€™ next project.\*/

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

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

/\* Output format:

+---------------+-------------------+

| genre | avg\_duration |

+-------------------+----------------

| thriller | 105 |

| . | . |

| . | . |

+---------------+-------------------+ \*/

-- Type your code below:

-- Finding the average duration of movies by grouping the genres that movies belong to

SELECT genre,

Round(Avg(duration),2) AS avg\_duration

FROM movie AS m

INNER JOIN genre AS g

ON g.movie\_id = m.id

GROUP BY genre

ORDER BY avg\_duration DESC;

-- Action genre has the highest duration of 112.88 seconds followed by romance and crime genres.

/\* 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€˜thrillerâ€™ genre of movies among all the genres in terms of number of movies produced?

-- (Hint: Use the Rank function)

/\* Output format:

+---------------+-------------------+---------------------+

| genre | movie\_count | genre\_rank |

+---------------+-------------------+---------------------+

|drama | 2312 | 2 |

+---------------+-------------------+---------------------+\*/

-- Type your code below:

-- CTE : Finds the rank of each genre based on the number of movies in each genre

-- Select query displays the genre rank and the number of movies belonging to Thriller genre

WITH genre\_summary AS

(

SELECT genre,

Count(movie\_id) AS movie\_count ,

Rank() OVER(ORDER BY Count(movie\_id) DESC) AS genre\_rank

FROM genre

GROUP BY genre )

SELECT \*

FROM genre\_summary

WHERE genre = "THRILLER" ;

-- Thriller has rank=3 and movie count of 1484

/\*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|

+---------------+-------------------+---------------------+----------------------+-----------------+-----------------+

| 0 | 5 | 177 | 2000 | 0 | 8 |

+---------------+-------------------+---------------------+----------------------+-----------------+-----------------+\*/

-- Type your code below:

-- Using MIN and MAX functions for the query

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;

/\* 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.

Now, letâ€™s find out the top 10 movies based on average rating.\*/

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

/\* Output format:

+---------------+-------------------+---------------------+

| title | avg\_rating | movie\_rank |

+---------------+-------------------+---------------------+

| Fan | 9.6 | 5 |

| . | . | . |

| . | . | . |

| . | . | . |

+---------------+-------------------+---------------------+\*/

-- Type your code below:

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

-- Finding the rank of each movie based on it's average rating

-- Displaying the top 10 movies using LIMIT clause

SELECT title,

avg\_rating,

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

FROM ratings AS r

INNER JOIN movie AS m

ON m.id = r.movie\_id limit 10;

-- top 10 movies can also be displayed using WHERE caluse with CTE

WITH MOVIE\_RANK AS

(

SELECT title,

avg\_rating,

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

FROM ratings AS r

INNER JOIN movie AS m

ON m.id = r.movie\_id

)

SELECT \* FROM MOVIE\_RANK

WHERE movie\_rank<=10;

-- Top 3 movies have average rating >= 9.8

/\* 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:

+---------------+-------------------+

| median\_rating | movie\_count |

+-------------------+----------------

| 1 | 105 |

| . | . |

| . | . |

+---------------+-------------------+ \*/

-- Type your code below:

-- Order by is good to have

-- Order by is good to have

-- Finding the number of movies vased on median rating and sorting based on movie count.

SELECT median\_rating,

Count(movie\_id) AS movie\_count

FROM ratings

GROUP BY median\_rating

ORDER BY movie\_count DESC;

/\* 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 |

+------------------+-------------------+---------------------+\*/

-- Type your code below:

-- CTE: Finding the rank of production company based on movie count with average rating > 8 using RANK function.

-- Querying the CTE to find the production company with rank=1

WITH production\_company\_hit\_movie\_summary

AS (SELECT production\_company,

Count(movie\_id) AS MOVIE\_COUNT,

Rank()

OVER(

ORDER BY Count(movie\_id) DESC ) AS PROD\_COMPANY\_RANK

FROM ratings AS R

INNER JOIN movie AS M

ON M.id = R.movie\_id

WHERE avg\_rating > 8

AND production\_company IS NOT NULL

GROUP BY production\_company)

SELECT \*

FROM production\_company\_hit\_movie\_summary

WHERE prod\_company\_rank = 1;

-- Dream Warrior Pictures and National Theatre Live production houses has produced the most number of hit movies (average rating > 8)

-- They have rank=1 and movie count =3

-- 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:

+---------------+-------------------+

| genre | movie\_count |

+-------------------+----------------

| thriller | 105 |

| . | . |

| . | . |

+---------------+-------------------+ \*/

-- Type your code below:

-- Query to find

-- 1. Number of movies released in each genre

-- 2. During March 2017

-- 3. In the USA (LIKE operator is used for pattern matching)

-- 4. Movies had more than 1,000 votes

SELECT genre,

Count(M.id) AS MOVIE\_COUNT

FROM movie AS M

INNER JOIN genre AS G

ON G.movie\_id = M.id

INNER JOIN ratings AS R

ON R.movie\_id = M.id

WHERE year = 2017

AND Month(date\_published) = 3

AND country LIKE '%USA%'

AND total\_votes > 1000

GROUP BY genre

ORDER BY movie\_count DESC;

-- Lets try to analyse with a unique problem statement.

-- Q15. Find movies of each genre that start with the word â€˜Theâ€™ and which have an average rating > 8?

/\* Output format:

+---------------+-------------------+---------------------+

| title | avg\_rating | genre |

+---------------+-------------------+---------------------+

| Theeran | 8.3 | Thriller |

| . | . | . |

| . | . | . |

| . | . | . |

+---------------+-------------------+---------------------+\*/

-- Type your code below:

-- Query to find:

-- 1. Number of movies of each genre that start with the word â€˜Theâ€™ (LIKE operator is used for pattern matching)

-- 2. Which have an average rating > 8?

-- Grouping by title to fetch distinct movie titles as movie belog to more than one genre

SELECT title,

avg\_rating,

genre

FROM movie AS M

INNER JOIN genre AS G

ON G.movie\_id = M.id

INNER JOIN ratings AS R

ON R.movie\_id = M.id

WHERE avg\_rating > 8

AND title LIKE 'THE%'

GROUP BY title

ORDER BY avg\_rating DESC;

-- There are 8 movies which begin with "The" in their title.

-- The Brighton Miracle has highest average rating of 9.5.

-- All the movies belong to the top 3 genres.

-- You should also try your hand at median rating and check whether the â€˜median ratingâ€™ 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?

-- Type your code below:

-- BETWEEN operator is used to find the movies released between 1 April 2018 and 1 April 2019

SELECT median\_rating, Count(\*) AS movie\_count

FROM movie AS M

INNER JOIN ratings AS R

ON R.movie\_id = M.id

WHERE median\_rating = 8

AND date\_published BETWEEN '2018-04-01' AND '2019-04-01'

GROUP BY median\_rating;

-- 361 movies have released between 1 April 2018 and 1 April 2019 with a median rating of 8

-- Once again, try to solve the problem given below.

-- 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.

-- Type your code below:

-- Two approaches - one is search based on language and the other by country column

-- Approach 1: By language column

-- Compute the total number of votes for German and Italian movies.

SELECT languages,

Sum(total\_votes) AS VOTES

FROM movie AS M

INNER JOIN ratings AS R

ON R.movie\_id = M.id

WHERE languages LIKE '%Italian%'

UNION

SELECT languages,

Sum(total\_votes) AS VOTES

FROM movie AS M

INNER JOIN ratings AS R

ON R.movie\_id = M.id

WHERE languages LIKE '%GERMAN%'

ORDER BY votes DESC;

-- Query to check if German votes > Italian votes using SELECT IF statement

-- Answer is YES if German votes > Italian votes

-- Answer is NO if German votes <= Italian votes

WITH VOTES\_SUMMARY AS

(

SELECT languages, SUM(total\_votes) AS VOTES

FROM MOVIE AS M

INNER JOIN RATINGS AS R

ON R.MOVIE\_ID = M.ID

WHERE languages like '%Italian%'

UNION

SELECT languages, SUM(total\_votes) AS VOTES

FROM MOVIE AS M

INNER JOIN RATINGS AS R

ON R.MOVIE\_ID = M.ID

WHERE languages like '%GERMAN%'

),

LANGUAGE\_VOTE AS

(

SELECT languages FROM VOTES\_SUMMARY

ORDER BY VOTES DESC

LIMIT 1)

SELECT IF (languages LIKE 'GERMAN' , 'YES', 'NO') AS ANSWER

FROM LANGUAGE\_VOTE ;

-- Approach 2: By country column

SELECT country, sum(total\_votes) as total\_votes

FROM movie AS m

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

WHERE country = 'Germany' or country = 'Italy'

GROUP BY country;

-- By observation, German movies received the highest number of votes when queried against language and country columns.

-- Answer is Yes

/\* Now that you have analysed the movies, genres and ratings tables, let us now analyse another table, the names table.

Letâ€™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|

+---------------+-------------------+---------------------+----------------------+

| 0 | 123 | 1234 | 12345 |

+---------------+-------------------+---------------------+----------------------+\*/

-- Type your code below:

-- NULL counts for individual columns of names table

SELECT Count(\*) AS name\_nulls

FROM names

WHERE NAME IS NULL;

SELECT Count(\*) AS height\_nulls

FROM names

WHERE height IS NULL;

SELECT Count(\*) AS date\_of\_birth\_nulls

FROM names

WHERE date\_of\_birth IS NULL;

SELECT Count(\*) AS known\_for\_movies\_nulls

FROM names

WHERE known\_for\_movies IS NULL;

-- NULL counts for columns of names table using CASE statements

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;

-- Height, date\_of\_birth, known\_for\_movies columns contain NULLS

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

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

Letâ€™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:

+---------------+-------------------+

| director\_name | movie\_count |

+---------------+-------------------|

|James Mangold | 4 |

| . | . |

| . | . |

+---------------+-------------------+ \*/

-- Type your code below:

-- CTE: Computes the top 3 genres using average rating > 8 condition and highest movie counts

-- Using the top genres derived from the CTE, the directors are found whose movies have an average rating > 8 and are sorted based on number of movies made.

WITH top\_3\_genres AS

(

SELECT genre,

Count(m.id) AS movie\_count ,

Rank() OVER(ORDER BY Count(m.id) DESC) AS genre\_rank

FROM movie AS m

INNER JOIN genre AS g

ON g.movie\_id = m.id

INNER JOIN ratings AS r

ON r.movie\_id = m.id

WHERE avg\_rating > 8

GROUP BY genre limit 3 )

SELECT n.NAME AS director\_name ,

Count(d.movie\_id) AS movie\_count

FROM director\_mapping AS d

INNER JOIN genre G

using (movie\_id)

INNER JOIN names AS n

ON n.id = d.name\_id

INNER JOIN top\_3\_genres

using (genre)

INNER JOIN ratings

using (movie\_id)

WHERE avg\_rating > 8

GROUP BY NAME

ORDER BY movie\_count DESC limit 3 ;

-- James Mangold , Joe Russo and Anthony Russo are top three directors in the top three genres whose movies have an average rating > 8

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

Now, letâ€™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 | 10 |

| . | . |

+---------------+-------------------+ \*/

-- Type your code below:

SELECT N.name AS actor\_name,

Count(movie\_id) AS movie\_count

FROM role\_mapping AS RM

INNER JOIN movie AS M

ON M.id = RM.movie\_id

INNER JOIN ratings AS R USING(movie\_id)

INNER JOIN names AS N

ON N.id = RM.name\_id

WHERE R.median\_rating >= 8

AND category = 'ACTOR'

GROUP BY actor\_name

ORDER BY movie\_count DESC

LIMIT 2;

-- Top 2 actors are Mammootty and 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.

Letâ€™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:

+------------------+--------------------+---------------------+

|production\_company|vote\_count | prod\_comp\_rank|

+------------------+--------------------+---------------------+

| The Archers | 830 | 1 |

| . | . | . |

| . | . | . |

+-------------------+-------------------+---------------------+\*/

-- Type your code below:

-- Approach 1: Using select statement

SELECT production\_company,

Sum(total\_votes) AS vote\_count,

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

FROM movie AS m

INNER JOIN ratings AS r

ON r.movie\_id = m.id

GROUP BY production\_company limit 3;

-- Approach 2: using CTEs

WITH ranking AS(

SELECT production\_company, sum(total\_votes) AS vote\_count,

RANK() OVER(ORDER BY SUM(total\_votes) DESC) AS prod\_comp\_rank

FROM movie AS m

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

GROUP BY production\_company)

SELECT production\_company, vote\_count, prod\_comp\_rank

FROM ranking

WHERE prod\_comp\_rank<4;

-- Top three production houses based on the number of votes received by their movies are Marvel Studios, 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.

Letâ€™s find who these actors could be.\*/

-- 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 |

+---------------+-------------------+---------------------+----------------------+-----------------+

| Yogi Babu | 3455 | 11 | 8.42 | 1 |

| . | . | . | . | . |

| . | . | . | . | . |

| . | . | . | . | . |

+---------------+-------------------+---------------------+----------------------+-----------------+\*/

-- Type your code below:

WITH actor\_summary

AS (SELECT N.NAME AS

actor\_name

,

total\_votes,

Count(R.movie\_id) AS

movie\_count,

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

actor\_avg\_rating

FROM movie AS M

INNER JOIN ratings AS R

ON M.id = R.movie\_id

INNER JOIN role\_mapping AS RM

ON M.id = RM.movie\_id

INNER JOIN names AS N

ON RM.name\_id = N.id

WHERE category = 'ACTOR'

AND country = "india"

GROUP BY NAME

HAVING movie\_count >= 5)

SELECT \*,

Rank()

OVER(

ORDER BY actor\_avg\_rating DESC) AS actor\_rank

FROM actor\_summary;

-- Top actor is Vijay Sethupathi followed by Fahadh Faasil and Yogi Babu.

-- 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:

+---------------+-------------------+---------------------+----------------------+-----------------+

| actress\_name | total\_votes | movie\_count | actress\_avg\_rating |actress\_rank |

+---------------+-------------------+---------------------+----------------------+-----------------+

| Tabu | 3455 | 11 | 8.42 | 1 |

| . | . | . | . | . |

| . | . | . | . | . |

| . | . | . | . | . |

+---------------+-------------------+---------------------+----------------------+-----------------+\*/

-- Type your code below:

WITH actress\_summary AS

(

SELECT n.NAME AS actress\_name,

total\_votes,

Count(r.movie\_id) AS movie\_count,

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

FROM movie AS m

INNER JOIN ratings AS r

ON m.id=r.movie\_id

INNER JOIN role\_mapping AS rm

ON m.id = rm.movie\_id

INNER JOIN names AS n

ON rm.name\_id = n.id

WHERE category = 'ACTRESS'

AND country = "INDIA"

AND languages LIKE '%HINDI%'

GROUP BY NAME

HAVING movie\_count>=3 )

SELECT \*,

Rank() OVER(ORDER BY actress\_avg\_rating DESC) AS actress\_rank

FROM actress\_summary LIMIT 5;

-- Top five actresses in Hindi movies released in India based on their average ratings are Taapsee Pannu, Kriti Sanon, Divya Dutta, Shraddha Kapoor, Kriti Kharbanda

/\* 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

--------------------------------------------------------------------------------------------\*/

-- Type your code below:

-- Using CASE statements to classify thriller movies as per avg rating

WITH thriller\_movies

AS (SELECT DISTINCT title,

avg\_rating

FROM movie AS M

INNER JOIN ratings AS R

ON R.movie\_id = M.id

INNER JOIN genre AS G using(movie\_id)

WHERE genre LIKE 'THRILLER')

SELECT \*,

CASE

WHEN avg\_rating > 8 THEN 'Superhit movies'

WHEN avg\_rating BETWEEN 7 AND 8 THEN 'Hit movies'

WHEN avg\_rating BETWEEN 5 AND 7 THEN 'One-time-watch movies'

ELSE 'Flop movies'

END AS avg\_rating\_category

FROM thriller\_movies;

/\* 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?

-- (Note: You need to show the output table in the question.)

/\* Output format:

+---------------+-------------------+---------------------+----------------------+

| genre | avg\_duration |running\_total\_duration|moving\_avg\_duration |

+---------------+-------------------+---------------------+----------------------+

| comdy | 145 | 106.2 | 128.42 |

| . | . | . | . |

| . | . | . | . |

| . | . | . | . |

+---------------+-------------------+---------------------+----------------------+\*/

-- Type your code below:

SELECT genre,

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

SUM(ROUND(AVG(duration),2)) OVER(ORDER BY genre ROWS UNBOUNDED PRECEDING) AS running\_total\_duration,

AVG(ROUND(AVG(duration),2)) OVER(ORDER BY genre ROWS 10 PRECEDING) AS moving\_avg\_duration

FROM movie AS m

INNER JOIN genre AS g

ON m.id= g.movie\_id

GROUP BY genre

ORDER BY genre;

-- 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.)

/\* Output format:

+---------------+-------------------+---------------------+----------------------+-----------------+

| genre | year | movie\_name |worldwide\_gross\_income|movie\_rank |

+---------------+-------------------+---------------------+----------------------+-----------------+

| comedy | 2017 | indian | $103244842 | 1 |

| . | . | . | . | . |

| . | . | . | . | . |

| . | . | . | . | . |

+---------------+-------------------+---------------------+----------------------+-----------------+\*/

-- Type your code below:

-- Top 3 Genres based on most number of movies

-- Top 3 Genres based on most number of movies

WITH top\_genres AS

(

SELECT genre,

Count(m.id) AS movie\_count ,

Rank() OVER(ORDER BY Count(m.id) DESC) AS genre\_rank

FROM movie AS m

INNER JOIN genre AS g

ON g.movie\_id = m.id

INNER JOIN ratings AS r

ON r.movie\_id = m.id

WHERE avg\_rating > 8

GROUP BY genre limit 3 ), movie\_summary AS

(

SELECT genre,

year,

title AS movie\_name,

CAST(replace(replace(ifnull(worlwide\_gross\_income,0),'INR',''),'$','') AS decimal(10)) AS worlwide\_gross\_income ,

DENSE\_RANK() OVER(partition BY year ORDER BY CAST(replace(replace(ifnull(worlwide\_gross\_income,0),'INR',''),'$','') AS decimal(10)) DESC ) AS movie\_rank

FROM movie AS m

INNER JOIN genre AS g

ON m.id = g.movie\_id

WHERE genre IN

(

SELECT genre

FROM top\_genres)

GROUP BY movie\_name

)

SELECT \*

FROM movie\_summary

WHERE movie\_rank<=5

ORDER BY YEAR;

-- Finally, letâ€™s find out the names of the top two production houses that have produced the highest number of hits among multilingual movies.

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

/\* Output format:

+-------------------+-------------------+---------------------+

|production\_company |movie\_count | prod\_comp\_rank|

+-------------------+-------------------+---------------------+

| The Archers | 830 | 1 |

| . | . | . |

| . | . | . |

+-------------------+-------------------+---------------------+\*/

-- Type your code below:

WITH production\_company\_summary

AS (SELECT production\_company,

Count(\*) AS movie\_count

FROM movie AS m

inner join ratings AS r

ON r.movie\_id = m.id

WHERE median\_rating >= 8

AND production\_company IS NOT NULL

AND Position(',' IN languages) > 0

GROUP BY production\_company

ORDER BY movie\_count DESC)

SELECT \*,

Rank()

over(

ORDER BY movie\_count DESC) AS prod\_comp\_rank

FROM production\_company\_summary

LIMIT 2;

-- Star Cinema and Twentieth Century Fox are the top two production houses that have produced the highest number of hits among 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?

/\* Output format:

+---------------+-------------------+---------------------+----------------------+-----------------+

| actress\_name | total\_votes | movie\_count |actress\_avg\_rating |actress\_rank |

+---------------+-------------------+---------------------+----------------------+-----------------+

| Laura Dern | 1016 | 1 | 9.60 | 1 |

| . | . | . | . | . |

| . | . | . | . | . |

+---------------+-------------------+---------------------+----------------------+-----------------+\*/

-- Type your code below:

-- Top 3 actresses based on number of Super Hit movies

WITH actress\_summary AS

(

SELECT n.NAME AS actress\_name,

SUM(total\_votes) AS total\_votes,

Count(r.movie\_id) AS movie\_count,

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

FROM movie AS m

INNER JOIN ratings AS r

ON m.id=r.movie\_id

INNER JOIN role\_mapping AS rm

ON m.id = rm.movie\_id

INNER JOIN names AS n

ON rm.name\_id = n.id

INNER JOIN GENRE AS g

ON g.movie\_id = m.id

WHERE category = 'ACTRESS'

AND avg\_rating>8

AND genre = "Drama"

GROUP BY NAME )

SELECT \*,

Rank() OVER(ORDER BY movie\_count DESC) AS actress\_rank

FROM actress\_summary LIMIT 3;

/\* 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

Format:

+---------------+-------------------+---------------------+----------------------+--------------+--------------+------------+------------+----------------+

| director\_id | director\_name | number\_of\_movies | avg\_inter\_movie\_days | avg\_rating | total\_votes | min\_rating | max\_rating | total\_duration |

+---------------+-------------------+---------------------+----------------------+--------------+--------------+------------+------------+----------------+

|nm1777967 | A.L. Vijay | 5 | 177 | 5.65 | 1754 | 3.7 | 6.9 | 613 |

| . | . | . | . | . | . | . | . | . |

| . | . | . | . | . | . | . | . | . |

| . | . | . | . | . | . | . | . | . |

| . | . | . | . | . | . | . | . | . |

| . | . | . | . | . | . | . | . | . |

| . | . | . | . | . | . | . | . | . |

| . | . | . | . | . | . | . | . | . |

| . | . | . | . | . | . | . | . | . |

+---------------+-------------------+---------------------+----------------------+--------------+--------------+------------+------------+----------------+

--------------------------------------------------------------------------------------------\*/

-- Type you code below:

WITH next\_date\_published\_summary AS

(

SELECT d.name\_id,

NAME,

d.movie\_id,

duration,

r.avg\_rating,

total\_votes,

m.date\_published,

Lead(date\_published,1) OVER(partition BY d.name\_id ORDER BY date\_published,movie\_id ) AS next\_date\_published

FROM director\_mapping AS d

INNER JOIN names AS n

ON n.id = d.name\_id

INNER JOIN movie AS m

ON m.id = d.movie\_id

INNER JOIN ratings AS r

ON r.movie\_id = m.id ), top\_director\_summary AS

(

SELECT \*,

Datediff(next\_date\_published, date\_published) AS date\_difference

FROM next\_date\_published\_summary )

SELECT name\_id AS director\_id,

NAME AS director\_name,

Count(movie\_id) AS number\_of\_movies,

Round(Avg(date\_difference),2) AS avg\_inter\_movie\_days,

Round(Avg(avg\_rating),2) AS avg\_rating,

Sum(total\_votes) AS total\_votes,

Min(avg\_rating) AS min\_rating,

Max(avg\_rating) AS max\_rating,

Sum(duration) AS total\_duration

FROM top\_director\_summary

GROUP BY director\_id

ORDER BY Count(movie\_id) DESC limit 9;