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.\*/

-- Segment 1:

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

-- Type your code below:

SELECT

table\_name, table\_rows

FROM

information\_schema.tables

WHERE

table\_schema = 'imdb';

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

-- Type your code below:

SELECT

SUM(CASE

WHEN id IS NULL THEN 1

ELSE 0

END) AS ID\_Null\_values,

SUM(CASE

WHEN title IS NULL THEN 1

ELSE 0

END) AS title\_Null\_values,

SUM(CASE

WHEN year IS NULL THEN 1

ELSE 0

END) AS year\_Null\_values,

SUM(CASE

WHEN date\_published IS NULL THEN 1

ELSE 0

END) AS date\_published\_Null\_values,

SUM(CASE

WHEN duration IS NULL THEN 1

ELSE 0

END) AS duration\_Null\_values,

SUM(CASE

WHEN country IS NULL THEN 1

ELSE 0

END) AS country\_Null\_values,

SUM(CASE

WHEN worlwide\_gross\_income IS NULL THEN 1

ELSE 0

END) AS worlwide\_gross\_income\_Null\_values,

SUM(CASE

WHEN languages IS NULL THEN 1

ELSE 0

END) AS languages\_Null\_values,

SUM(CASE

WHEN production\_company IS NULL THEN 1

ELSE 0

END) AS production\_company\_Null\_values

FROM

movie;

-- 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:

-- Total number of movies released each year

SELECT

year AS Year, COUNT(id) AS number\_of\_movies

FROM

movie

GROUP BY year

ORDER BY year;

-- Month-wise Trend

SELECT

MONTH(date\_published) AS month\_num,

COUNT(id) AS number\_of\_movies

FROM

movie

GROUP BY MONTH(date\_published)

ORDER BY MONTH(date\_published);

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

So, now that you have understood the month-wise trend of movies, let’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:

SELECT

COUNT(id) AS number\_of\_movies\_produced\_in\_2019

FROM

movie

WHERE

country LIKE '%USA%' or country LIKE '%India%' AND year=2019

ORDER BY

COUNT(id) DESC;

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

Exploring table Genre would be fun!!

Let’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:

SELECT DISTINCT

genre

FROM

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?

-- Type your code below:

SELECT

genre, COUNT(movie\_id) AS number\_of\_movies\_produced

FROM

genre AS g

INNER JOIN

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

/\*WHERE

year = 2019

(The comment above the problem statement instructs to find the highest number of movies produced in last year i.e. 2019

whereas the problem statement says produced overall. In case of last year, WHERE statement has to be added to the code.)

\*/

GROUP BY genre

ORDER BY COUNT(movie\_id) DESC

LIMIT 1;

/\* So, based on the insight that you just drew, RSVP Movies should focus on the ‘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?

-- Type your code below:

WITH count\_of\_movie\_genre AS

(

SELECT

movie\_id,

COUNT(genre) AS number\_of\_genre\_of\_movie

FROM

genre

GROUP BY

movie\_id

HAVING

number\_of\_genre\_of\_movie=1

)

SELECT

COUNT(movie\_id) AS number\_of\_movies\_with\_one\_genre

FROM

count\_of\_movie\_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 Movies’ 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:

SELECT

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

FROM

genre AS g

INNER JOIN

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

GROUP BY GENRE

ORDER BY avg\_duration DESC;

-- Took 2 Decimal places because next comment hint mentioned average duration of Drama genre movies as 106.77

/\* 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 ‘thriller’ 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:

WITH rank\_genre AS

(

SELECT

genre,

COUNT(movie\_id) as movie\_count,

RANK() OVER (ORDER BY COUNT(movie\_id) DESC) AS genre\_rank

FROM

genre AS g

INNER JOIN

movie AS m

ON

g.movie\_id = m.id

GROUP BY

GENRE

)

SELECT \*

FROM

rank\_genre

WHERE

genre LIKE 'Thriller';

-- Thriller genre is ranked 3 amongst all genres with respect to movie count in each genre

/\*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:

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, let’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

SELECT

title,

avg\_rating,

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

FROM

movie AS m

INNER JOIN

ratings AS r

ON

m.id=r.movie\_id

limit 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:

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

| median\_rating | movie\_count |

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

| 1 | 105 |

| . | . |

| . | . |

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

-- Type your 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;

/\* 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:

WITH hit\_movies AS

(

SELECT

production\_company,

COUNT(id) AS movie\_count,

RANK() OVER(ORDER BY COUNT(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

ORDER BY

movie\_count DESC

)

SELECT \*

FROM

hit\_movies

WHERE

prod\_company\_rank=1;

-- 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:

SELECT

genre,

COUNT(g.movie\_id) AS movie\_count

FROM

genre as g

INNER JOIN

movie as m

ON

g.movie\_id=m.id

INNER JOIN

ratings as r

ON

g.movie\_id=r.movie\_id

WHERE

country like '%USA%' AND MONTH(date\_published)=3 AND year=2017 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 ‘The’ and which have an average rating > 8?

/\* Output format:

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

| title | avg\_rating | genre |

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

| Theeran | 8.3 | Thriller |

| . | . | . |

| . | . | . |

| . | . | . |

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

-- Type your code below:

SELECT

title, avg\_rating, genre

FROM

movie AS m

INNER JOIN

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

INNER JOIN

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

WHERE

title LIKE 'The%' AND avg\_rating > 8

GROUP BY title

ORDER BY avg\_rating DESC;

-- You should also try your hand at median rating and check whether the ‘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?

-- Type your code below:

SELECT

COUNT(id) as movies\_between\_01Apr18\_till\_01Apr19\_median\_rating\_8

FROM

movie as m

INNER JOIN

ratings as r

ON

m.id=r.movie\_id

WHERE

date\_published BETWEEN '2018-04-01' AND '2019-04-01' AND median\_rating=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:

SELECT

languages, total\_votes

FROM

movie AS m

INNER JOIN

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

GROUP BY languages

HAVING languages IN ('German' , 'Italian')

ORDER BY total\_votes DESC;

-- 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|

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

| 0 | 123 | 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

WHEN known\_for\_movies IS NULL THEN 1

ELSE 0

END) AS known\_for\_movies\_nulls

FROM

names;

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

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

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:

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

| director\_name | movie\_count |

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

|James Mangold | 4 |

| . | . |

| . | . |

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

-- Type your code below:

WITH top\_3\_genre AS

(

SELECT

genre,

RANK() OVER(ORDER BY COUNT(movie\_id) DESC) as genre\_rank

FROM

genre

INNER JOIN

ratings

USING(movie\_id)

WHERE

avg\_rating>8

GROUP BY

genre

LIMIT 3

),

director\_top\_3 AS

(

SELECT

name AS director\_name,

COUNT(g.movie\_id) AS movie\_count,

top\_3\_genre.genre,

RANK() OVER(ORDER BY COUNT(g.movie\_id) DESC) AS director\_rank

FROM

names AS n

INNER JOIN

director\_mapping AS dm

ON

n.id=dm.name\_id

INNER JOIN

genre AS g

ON

dm.movie\_id = g.movie\_id

INNER JOIN

ratings AS r

ON

g.movie\_id=r.movie\_id,

top\_3\_genre

WHERE

g.genre IN (top\_3\_genre.genre) AND avg\_rating>8

GROUP BY

director\_name

ORDER BY

movie\_count DESC

)

SELECT \*

FROM

director\_top\_3

WHERE

director\_rank<=3

LIMIT 4;

/\* The rank of the directors is as follows:

director\_name director\_rank

1. James Mangold 1

2. Soubin Shahir 2

3. Joe Russo 2

4. Anthony Russo 2

There are 3 directors having director\_rank '2'.

However, while limiting the results only director\_rank '1' and 2 directors out of 3 having director\_rank '2' are being displayed.

\*/

/\* 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 | 10 |

| . | . |

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

-- Type your code below:

SELECT

name AS actor\_name, COUNT(r.movie\_id) AS movie\_count

FROM

names AS n

INNER JOIN

role\_mapping AS rm ON n.id = rm.name\_id

INNER JOIN

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

WHERE

median\_rating >= 8

AND category LIKE 'actor'

GROUP BY name

ORDER BY movie\_count DESC

LIMIT 2;

/\* 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:

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

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

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

| The Archers | 830 | 1 |

| . | . | . |

| . | . | . |

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

-- Type your code below:

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

m.id=r.movie\_id

GROUP BY

production\_company

ORDER BY

vote\_count DESC

LIMIT 3;

/\*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.

Let’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:

SELECT

n.name AS actor\_name,

r.total\_votes,

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

ROUND(SUM(r.avg\_rating\*r.total\_votes)/SUM(r.total\_votes),2) AS actor\_avg\_rating,

RANK() OVER (ORDER BY r.avg\_rating DESC) AS actor\_rank

FROM

names AS n

INNER JOIN

role\_mapping AS rm

ON

n.id=rm.name\_id

INNER JOIN

ratings AS r

ON

rm.movie\_id=r.movie\_id

INNER JOIN

movie AS m

ON

m.id= rm.movie\_id

WHERE

m.country LIKE 'India' AND rm.category LIKE 'actor'

GROUP BY

actor\_name

HAVING

movie\_count>=5;

-- 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 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 m

INNER JOIN ratings r

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 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;

/\* 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:

SELECT

title,

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'

WHEN avg\_rating < 5 THEN 'Flop movies'

END AS avg\_rating\_category

FROM

movie m

INNER JOIN

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

INNER JOIN

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

WHERE

genre = 'thriller';

/\* 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 5 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

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 m

INNER JOIN genre g

ON g.movie\_id = m.id

INNER JOIN ratings 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 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, let’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:

SELECT

production\_company,

COUNT(id) AS movie\_count,

RANK() OVER (ORDER BY COUNT(id) DESC) AS prod\_comp\_rank

FROM

movie AS m

INNER JOIN

ratings AS r

ON

m.id=r.movie\_id

WHERE

median\_rating>=8 AND languages LIKE '%,%' AND production\_company IS NOT NULL

GROUP BY

production\_company

LIMIT 2;

-- 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:

WITH actress\_summary AS

(

SELECT 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 m

INNER JOIN ratings 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 movie\_date\_info AS

(

SELECT d.name\_id, name, d.movie\_id,

m.date\_published,

LEAD(date\_published, 1) OVER(PARTITION BY d.name\_id ORDER BY date\_published, d.movie\_id) AS next\_movie\_date

FROM director\_mapping d

JOIN names AS n

ON d.name\_id=n.id

JOIN movie AS m

ON d.movie\_id=m.id

),

date\_difference AS

(

SELECT \*, DATEDIFF(next\_movie\_date, date\_published) AS diff

FROM movie\_date\_info

),

avg\_inter\_days AS

(

SELECT name\_id, AVG(diff) AS avg\_inter\_movie\_days

FROM date\_difference

GROUP BY name\_id

),

final\_result AS

(

SELECT d.name\_id AS director\_id,

name AS director\_name,

COUNT(d.movie\_id) AS number\_of\_movies,

ROUND(avg\_inter\_movie\_days) 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,

ROW\_NUMBER() OVER(ORDER BY COUNT(d.movie\_id) DESC) AS director\_row\_rank

FROM

names AS n

JOIN director\_mapping AS d

ON n.id=d.name\_id

JOIN ratings AS r

ON d.movie\_id=r.movie\_id

JOIN movie AS m

ON m.id=r.movie\_id

JOIN avg\_inter\_days AS a

ON a.name\_id=d.name\_id

GROUP BY director\_id

)

SELECT

director\_id,

director\_name,

number\_of\_movies,

avg\_inter\_movie\_days,

avg\_rating,

total\_votes,

min\_rating,

max\_rating,

total\_duration

FROM final\_result

LIMIT 9;