```sql
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 COUNT(movie_id) as Total_Rows FROM director_mapping;
SELECT COUNT(movie_id) as Total_Rows FROM genre;
SELECT COUNT(id) as Total_Rows FROM movie;
SELECT COUNT(id) as Total_Rows FROM names;
SELECT COUNT(movie_id) as Total_Rows FROM ratings;
SELECT COUNT(movie_id) as Total_Rows FROM role_mapping;

-- 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,
SUM(CASE WHEN title IS NULL THEN 1 ELSE 0 END) AS Title_NULL,
SUM(CASE WHEN year IS NULL THEN 1 ELSE 0 END) AS Year_NULL,

SUM(CASE WHEN date_published IS NULL THEN 1 ELSE 0 END) AS DatePublished_NULL,

SUM(CASE WHEN duration IS NULL THEN 1 ELSE 0 END) AS Duration_NULL,

SUM(CASE WHEN country IS NULL THEN 1 ELSE 0 END) AS Country_NULL,

SUM(CASE WHEN worlwide_gross_income IS NULL THEN 1 ELSE 0 END) AS WorldWide_NULL,

SUM(CASE WHEN languages IS NULL THEN 1 ELSE 0 END) AS Language_NULL,

SUM(CASE WHEN production_company IS NULL THEN 1 ELSE 0 END) AS production_company_NULL

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:

SELECT
year, COUNT(id) AS number_of_movies
FROM movie
GROUP BY year;

-- month wise

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(DISTINCT id) AS number_of_movies, year
FROM movie
WHERE ( country LIKE '%INDIA%'
OR country LIKE '%USA%' )
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!!
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
g.genre, COUNT(m.id) AS num_of_movie
FROM genre g
INNER JOIN movie m
```

```
ON g.movie_id = m.id
GROUP BY genre
ORDER BY COUNT(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 ct_genre AS
(
SELECT
movie_id, COUNT(genre) AS number_of_movies
FROM genre
GROUP BY movie_id
HAVING number_of_movies=1
)
SELECT COUNT(movie_id) AS number_of_movies
FROM ct_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
g.genre, ROUND(AVG(m.duration),2) AS Avg_Duration
FROM movie m
INNER JOIN genre g
ON m.id = g.movie_id
GROUP BY genre
ORDER BY AVG(m.duration) DESC;
```

/* 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 ct_genre_rank
AS
(
SELECT
genre, COUNT(movie_id),
RANK() OVER(ORDER BY COUNT(movie_id) DESC) genre_rank
FROM genre
GROUP BY genre
)
SELECT * FROM ct_genre_rank
WHERE genre="Thriller";


/*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


WITH ct_avg_rating_rank

AS

(

SELECT

m.title AS title, r.avg_rating as avg_rating,

DENSE_RANK() OVER(ORDER BY r.avg_rating DESC) movie_rank

FROM movie m

INNER JOIN ratings r

ON m.id = r.movie_id

)
```

```
SELECT * FROM ct_avg_rating_rank 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:

```
+---------------+-------------------+
| 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 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:

```sql
SELECT
production_company, COUNT(id) AS movie_count,
DENSE_RANK() OVER(ORDER BY COUNT(id) DESC) prod_company_rank
FROM movie m
INNER JOIN ratings r
ON m.id = r.movie_id
WHERE r.avg_rating > 8 AND m.production_company IS NOT NULL
GROUP BY m.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:

```
+---------------+-------------------+
| genre          |     movie_count       |
+-------------------+----------------
|           thriller  |                      105                             |
|           .                               |                     .                               |
|           .                               |                     .                               |
+---------------+-------------------+ */
-- Type your code below:


SELECT
g.genre, COUNT(m.id) AS movie_count
FROM genre g
INNER JOIN movie m
ON m.id = g.movie_id
INNER JOIN ratings r
ON m.id = r.movie_id
WHERE m.year = 2017 AND MONTH(m.date_published) = 3 AND m.country LIKE'%USA%' AND
r.total_votes>1000
GROUP BY genre
ORDER BY COUNT(m.id) 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 m
INNER JOIN ratings r
ON m.id = r.movie_id
INNER JOIN genre g
ON g.movie_id = m.id
WHERE title LIKE'The%' AND avg_rating>8
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 movie_released,
median_rating
FROM movie m
INNER JOIN ratings r
ON m.id = r.movie_id
WHERE median_rating = 8 AND date_published BETWEEN '2018-04-01' AND '2019-04-01'
GROUP BY median_rating;
```

-- 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 country, sum(total_votes) AS votes_count

FROM movie as m

INNER JOIN ratings as r

ON r.movie_id=m.id

WHERE country = 'germany' OR country = 'italy'

GROUP BY country;
```

-- 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_genre

AS

(

SELECT

g.genre,

COUNT(g.movie_id) as movie_count

FROM genre g

INNER JOIN ratings r

ON g.movie_id = r.movie_id

WHERE avg_rating>8

GROUP BY genre

ORDER BY movie_count DESC

LIMIT 3

),

top_director

AS

(

SELECT

n.name as director_name,

COUNT(d.movie_id) as movie_count,

RANK() OVER(ORDER BY COUNT(d.movie_id) DESC) director_rank

FROM names n

INNER JOIN director_mapping d

ON n.id = d.name_id

INNER JOIN ratings r

ON r.movie_id = d.movie_id

INNER JOIN genre g

ON g.movie_id = d.movie_id,

top_genre
```

```sql
WHERE r.avg_rating > 8 AND g.genre IN (top_genre.genre)

GROUP BY n.name

ORDER BY movie_count DESC

)

SELECT director_name,

movie_count

FROM top_director

WHERE director_rank <= 3

LIMIT 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  |                       10                              |

|           .                                  |                     .                                 |

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

-- Type your code below:

SELECT * from role_mapping;

SELECT

n.name AS actor_name,

COUNT(rm.movie_id) AS movie_count
```

```
FROM role_mapping rm
INNER JOIN names n
ON n.id = rm.name_id
INNER JOIN ratings r
ON r.movie_id = rm.movie_id
WHERE category="actor"
AND r.median_rating >= 8
GROUP BY n.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,
DENSE_RANK() OVER(ORDER BY SUM(total_votes) DESC) AS prod_comp_rank
FROM movie m
INNER JOIN ratings 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:

WITH top_actor

    AS (SELECT b.NAME

             AS

             actor_name,

             Sum(c.total_votes)

             AS

               total_votes,

             Count(DISTINCT a.movie_id)

             AS

               movie_count,

             Round(Sum(c.avg_rating * c.total_votes) / Sum(c.total_votes), 2)

             AS

             actor_avg_rating

         FROM   role_mapping a

             INNER JOIN names b

                  ON a.name_id = b.id

             INNER JOIN ratings c

                  ON a.movie_id = c.movie_id

             INNER JOIN movie d

                  ON a.movie_id = d.id
```

```
     WHERE  a.category = 'actor'
         AND d.country LIKE '%India%'
     GROUP  BY a.name_id,
           b.NAME
     HAVING Count(DISTINCT a.movie_id) >= 5)
SELECT *,
    Rank()
     OVER (
      ORDER BY actor_avg_rating DESC) AS actor_rank
FROM   top_actor;


-- 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 top_actor
    AS (SELECT b.NAME
               AS
               actor_name,
               Sum(c.total_votes)
               AS
                 total_votes,
               Count(DISTINCT a.movie_id)
               AS
                 movie_count,
               Round(Sum(c.avg_rating * c.total_votes) / Sum(c.total_votes), 2)
               AS
               actor_avg_rating
        FROM   role_mapping a
               INNER JOIN names b
                      ON a.name_id = b.id
               INNER JOIN ratings c
                      ON a.movie_id = c.movie_id
               INNER JOIN movie d
                      ON a.movie_id = d.id
        WHERE  a.category = 'actress'
               AND d.country LIKE '%India%'
        GROUP  BY a.name_id,
                 b.NAME
        HAVING Count(DISTINCT a.movie_id) >=5)
SELECT *,
      Rank()
```

```sql
        OVER (
          ORDER BY actor_avg_rating DESC) AS actor_rank
FROM   top_actor;

/* 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,
avg_rating,
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 movie m
INNER JOIN genre g
ON m.id = g.movie_id
```

```
INNER JOIN ratings r

ON r.movie_id = m.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(AVG(duration)) OVER(ORDER BY genre) AS running_total_duration,
AVG(AVG(duration)) OVER(ORDER BY genre) AS moving_avg_duration

FROM movie m

INNER JOIN genre g

ON m.id = g.movie_id

GROUP 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 top3_genre
AS
(
SELECT
genre,
COUNT(movie_id) as movie_count
FROM genre
GROUP BY genre
ORDER BY movie_count DESC
LIMIT 3
),
top5_movie
AS
(
SELECT
genre,
YEAR,
title as movie_name,
worlwide_gross_income,
DENSE_RANK() OVER(PARTITION BY year ORDER BY worlwide_gross_income DESC) AS movie_rank
FROM movie m
INNER JOIN genre g
ON m.id = g.movie_id
WHERE genre IN(SELECT genre FROM top3_genre)
)
SELECT *
FROM top5_movie
WHERE movie_rank<=5;
```

-- 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,

ROW_NUMBER() OVER(ORDER BY COUNT(id) DESC) AS prod_comp_rank

FROM movie m

INNER JOIN ratings r

ON m.id = r.movie_id

WHERE median_rating>=8

AND production_company IS NOT NULL

AND POSITION(',' IN languages)>0

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:

```
SELECT name as actress_name, SUM(total_votes) AS total_votes, COUNT(rm.movie_id) as movie_id,
Round(Sum(avg_rating)/Sum(total_votes),2) AS actress_avg_rating,

RANK() OVER(ORDER BY COUNT(rm.movie_id) DESC) AS actress_rank

FROM names n

INNER JOIN role_mapping rm

ON n.id = rm.name_id

INNER JOIN ratings r

ON r.movie_id = rm.movie_id

INNER JOIN genre g

ON g.movie_id = r.movie_id

WHERE category="actress" AND avg_rating>8 AND g.genre="Drama"
```

```
GROUP BY name

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 ctf_date_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 ctf_date_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;
```