```
-- Q1. Find the total number of rows in each table of the schema?
-- Type your code below:

SELECT count(*) from director_mapping;
SELECT count(*) from genre;
SELECT count(*) from movie;
SELECT count(*) from names;
SELECT count(*) from ratings;
SELECT count(*) 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
date_published_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
worlwide_gross_income_NULL,
      SUM(CASE WHEN languages IS NULL THEN 1 ELSE 0 END) AS
languages_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 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:

-- FIRST PART
select year, count(distinct id) as number_of_movies
from movie
group by year;


-- SECOND PART
select month(date_published) as month_num, count(distinct id) as
number_of_movies
from movie
group 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, 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)
from movie
where year = '2019' and (country like '%India%' OR country like '%USA%');
```

/* 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)
from genre g
inner join movie m on m.id = g.movie_id
group by genre
order by count(m.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 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;

/* 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 genre g
inner join movie m on g.movie_id = m.id
group by genre
order by avg_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 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


/*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:
-- Keep in mind that multiple movies can be at the same rank. You only
have to find out the top 10 movies (if there are more than one movies at
the 10th place, consider them all.)

select m.title, r.avg_rating, rank() over(order by avg_rating desc) as
movie_rank
from movie m
inner join ratings 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 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:

select m.production_company, count(m.id), rank() over(order by count(m.id)
desc)
from movie m
inner join ratings r on m.id = r.movie_id
where  r.avg_rating>8 and production_company IS NOT NULL
group by 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 g.movie_id = m.id
inner join ratings r on m.id = r.movie_id
where m.country like'%USA%' and m.year='2017' and month(date_published) =
3 and r.total_votes>1000
group by g.genre
order by movie_count desc;

-- 24 Drama movies were released during March 2017 in the USA and had more
than 1,000 votes
-- Top 3 genres are drama, comedy and action during March 2017 in the USA
and had more than 1,000 votes



-- 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 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%'
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 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;
```

```
-- 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 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;

-- YES, German movies have 106710 votes, whereas Italian movies have only
77965 votes
-- 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_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 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 n.name AS actor_name,
       COUNT(r.movie_id) AS movie_count
FROM names n
INNER JOIN role_mapping rm ON n.id = rm.name_id
INNER JOIN movie m ON rm.movie_id = m.id
INNER JOIN ratings r ON r.movie_id = m.id
WHERE r.median_rating >= 8
GROUP BY n.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.
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 m.production_company, sum(r.total_votes) as vote_count, rank() over
(order by sum(r.total_votes) desc) as prod_comp_rank
from movie m
inner join ratings r on m.id = r.movie_id
group by m.production_company
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 actor_stats AS (
    SELECT
        n.name AS actor_name,
        SUM(r.total_votes) AS total_votes,
        COUNT(r.movie_id) AS movie_count,
        ROUND(SUM(r.avg_rating * r.total_votes) / SUM(r.total_votes), 2)
AS avg_rating
    FROM names n
    INNER JOIN role_mapping rm ON rm.name_id = n.id
    INNER JOIN movie m ON rm.movie_id = m.id
    INNER JOIN ratings r ON m.id = r.movie_id
    WHERE rm.category = 'actor' AND m.country = 'India'
    GROUP BY n.name
    HAVING COUNT(r.movie_id) >= 5
)
SELECT
    actor_name,
    total_votes,
    movie_count,
    avg_rating,
    RANK() OVER (ORDER BY avg_rating desc) AS actor_rank
FROM actor_stats;




-- 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_details as
      (
    select name as actress_name, sum(total_votes) as Total_votes,
count(m.id) as
movie_count,Round(Sum(avg_rating*total_votes)/Sum(total_votes),2) as
actress_avg_rating
      from names n
      INNER JOIN role_mapping rm on rm.name_id = n.id
      INNER JOIN movie m on m.id= rm.movie_id
      INNER JOIN ratings r on r.movie_id = m.id
      where category = 'actress' and country = 'India' and languages like
'%Hindi%'
      group by actress_name
      having movie_count>=3
      )
SELECT  *,
         Rank() OVER(ORDER BY actress_avg_rating DESC) AS actress_rank
FROM     actress_details 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. Consider thriller movies having at least 25,000 votes. Classify
them according to their average ratings in
   the following categories:

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

    Note: Sort the output by average ratings (desc).
-------------------------------------------------------------------------
------------------*/
/* Output format:
+---------------+-------------------+
| movie_name     |     movie_category   |
+---------------+-------------------+
|     Get Out          |                Hit         |
|          .           |                .           |
```

```
|          .          |                .          |
+---------------+------------------+*/

-- Type your code below:

select title,
CASE
      WHEN avg_rating>8 THEN 'Superhit'
    WHEN avg_rating between 7 and 8 then 'Hit'
    WHEN avg_rating between 5 and 7 then 'One time watch'
    WHEN avg_rating < 5 then 'Flop'
    END as movie_category
from movie m
INNER JOIN ratings r on m.id = r.movie_id
INNER JOIN genre g on g.movie_id = m.id
where genre like '%Thriller%' and total_votes>=25000
order by avg_rating desc;




/* 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,
```

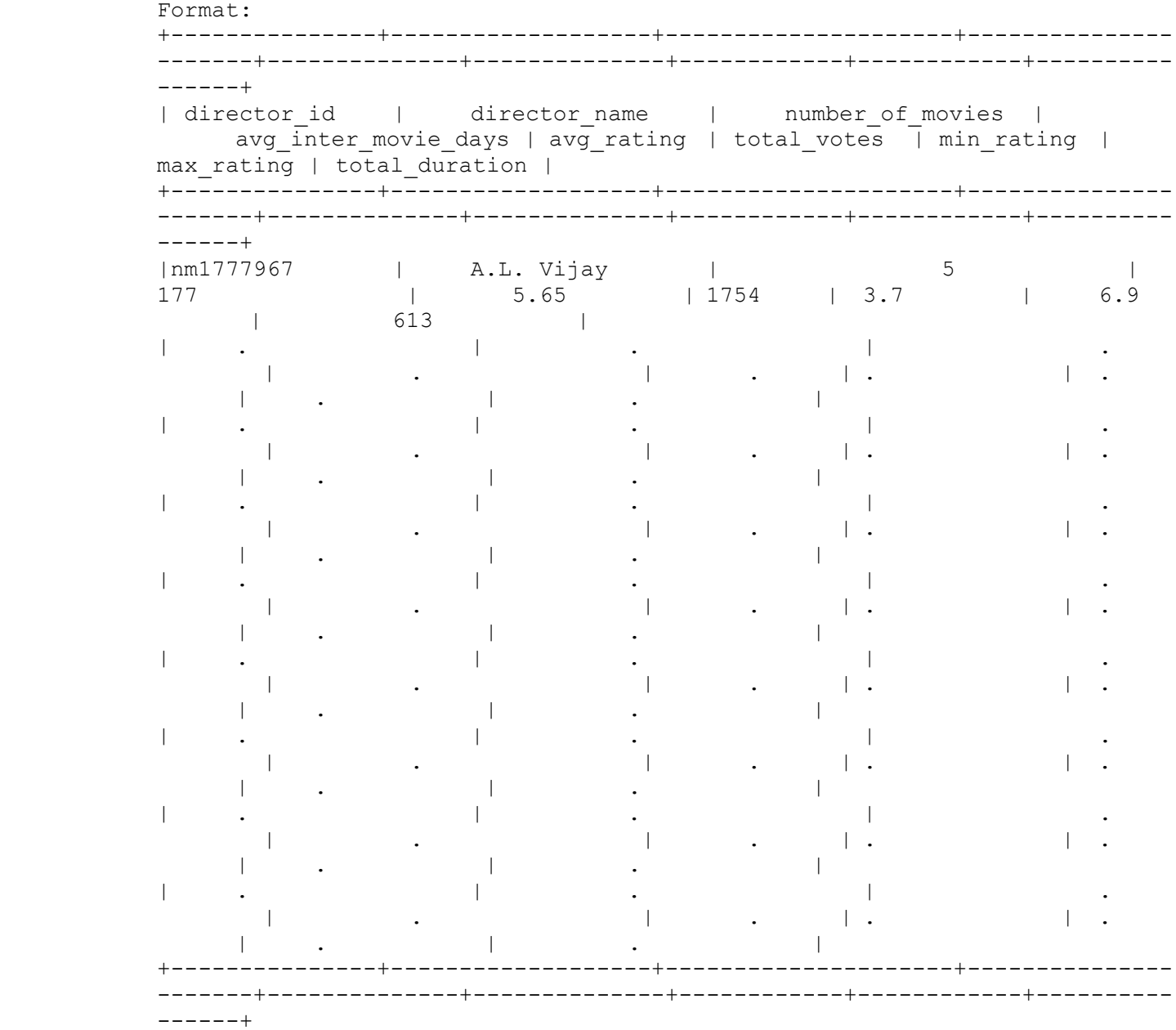
```
    AVG(duration) AS avg_duration,
    SUM(AVG(duration)) OVER (ORDER BY genre ROWS UNBOUNDED PRECEDING) AS
running_total_duration,
    AVG(AVG(duration)) OVER (ORDER BY genre ROWS UNBOUNDED PRECEDING) AS
moving_avg_duration
FROM movie AS m
INNER JOIN genre AS g
ON m.id = g.movie_id
GROUP BY genre
ORDER BY genre asc;



-- 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 top3genre AS (
    SELECT g.genre
    FROM genre g
    INNER JOIN movie m ON m.id = g.movie_id
    GROUP BY g.genre
    ORDER BY COUNT(g.movie_id) DESC
    LIMIT 3
),
top5movie AS (
```

```
    SELECT distinct
                g.genre,
        m.year,
        m.title,
        m.worlwide_gross_income,
        RANK() OVER (PARTITION BY m.year ORDER BY m.worlwide_gross_income
DESC) AS movie_rank
    FROM movie m
    INNER JOIN genre g ON m.id = g.movie_id
    WHERE g.genre IN (SELECT genre FROM top3genre)
    )

SELECT *
FROM top5movie
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 m
inner join ratings r on r.movie_id = m.id
where median_rating>=8 and production_company IS NOT NULL and  languages
like '%,%'
group by production_company
limit 2;

-- OR

select production_company, count(id) as movie_count, rank() over(order by
count(id) desc) as prod_comp_rank
from movie m
inner join ratings 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
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 the number of Super Hit
movies (Superhit movie: average rating of movie > 8) in 'drama' genre?

-- Note: Consider only superhit movies to calculate the actress average
ratings.
-- (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. If number of votes are same, sort
alphabetically by actress name.)

/* Output format:
+---------------+-------------------+---------------------+---------------
-------+-----------------+
| actress_name   |     total_votes      |     movie_count        |
actress_avg_rating |actress_rank     |
+---------------+-------------------+---------------------+---------------
-------+-----------------+
|     Laura Dern |                1016  |            1           |
9.6000                |           1           |
|          .          |                .          |           .
|        .            |          .           |
|           .          |                .          |           .
|        .             |          .           |
+---------------+-------------------+---------------------+---------------
-------+-----------------+*/

-- Type your code below:

select name, sum(total_votes), count(m.id), avg(avg_rating) as
actress_avg_rating, rank() over(order by avg(avg_rating) desc ) as
actress_rank
from names n
INNER JOIN role_mapping rm on n.id = rm.name_id
INNER JOIN movie m on m.id = rm.movie_id
INNER JOIN genre g on g.movie_id = m.id
INNER JOIN ratings r on r.movie_id = m.id
where category = 'actress' and avg_rating > 8 and 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 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;
```