```
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:
SHOW TABLES;
SELECT COUNT(*)  as 'Total number of rows in director_mapping table' FROM
director_mapping;
SELECT COUNT(*)  as 'Total number of rows in genre table' FROM genre;
SELECT COUNT(*)  as 'Total number of rows in movie table' FROM movie;
SELECT COUNT(*)  as 'Total number of rows in names table' FROM names;
SELECT COUNT(*)  as 'Total number of rows in ratings table' FROM ratings;
SELECT COUNT(*)  as 'Total number of rows in role_mapping table' FROM
role_mapping;

-- Q2. Which columns in the movie table have null values?
-- Type your code below:

DESCRIBE movie;

SELECT * FROM movie;

SELECT count(*) FROM movie
WHERE id IS NULL;

SELECT count(*) FROM movie
WHERE title IS NULL;

SELECT count(*) FROM movie
WHERE year IS NULL;

SELECT count(*) FROM movie
WHERE date_published IS NULL;

SELECT count(*) FROM movie
WHERE country IS NULL;

SELECT count(*) FROM movie
WHERE worlwide_gross_income IS NULL;

SELECT count(*) FROM movie
```

```
WHERE languages IS NULL;

SELECT count(*) FROM movie
WHERE production_company IS NULL;



 /*country worlwide_gross_income, languages, production_company have null
values*/
-- Now as you can see four columns of the movie table has null values.
Let's look at the at the movies released each year.
-- Q3. Find the total number of movies released each year? How does the
trend look month wise? (Output expected)

/* Output format for the first part:

+---------------+-------------------+
| Year                |     number_of_movies|
+-------------------+----------------
|     2017       |     2134             |
|     2018       |           .                |
|     2019       |           .                |
+---------------+-------------------+


Output format for the second part of the question:
+---------------+-------------------+
|     month_num   |     number_of_movies|
+---------------+----------------
|     1                |      134             |
|     2                |      231             |
|     .                |           .                  |
+---------------+-------------------+ */
-- Type your code below:
SELECT year AS Year,
COUNT(*) AS number_of_movies
FROM
movie
GROUP BY year;

/*The highest number of movies is produced in the YEAR 2017.*/

SELECT MONTH(date_published) AS month_num,
COUNT(*) AS number_of_movies
FROM
movie
GROUP BY MONTH(date_published)
ORDER BY number_of_movies DESC;

/*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(*) AS 'Number of movies released in USA or India in the year
2019'
FROM
movie
WHERE year=2019 AND (LOWER(country) LIKE '%usa%' OR LOWER(country) LIKE
'%india%');


/* USA and India produced more than a thousand movies 1059 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 AS 'Genre', count(*) AS 'Number of movies'

FROM movie as m
INNER JOIN
genre as g
ON m.id = g.movie_id

WHERE year = '2019'

GROUP BY genre
ORDER BY COUNT(*) 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 single_genre_movie as (
select movie_id
from genre
GROUP BY movie_id
having count(*) = '1'
)
select count(*) as 'Number of movies belonging to only one genre' from
single_genre_movie;


/* 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(m.duration)) AS avg_duration
FROM movie AS m
INNER JOIN
genre AS g
ON m.id = g.movie_id
GROUP BY genre;



/* 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:

SELECT g.genre,
count(g.movie_id) as movie_count,
rank() over(order by count(g.movie_id) desc) as genre_rank
FROM movie AS m
INNER JOIN
genre AS g
ON m.id = g.movie_id

GROUP BY 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 round(min(avg_rating)) as min_avg_rating,
 round(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 movie.title,
ratings.avg_rating,
row_number() OVER(order by ratings.avg_rating desc) as movie_rank
FROM
movie
inner join
ratings
on movie.id = ratings.movie_id
order by ratings.avg_rating desc 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(*) 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:

WITH HIT_MOVIE_SUMMARY AS
(
SELECT movie.production_company,
count(movie.id) AS movie_count,
row_number() over (order by count(*) desc) as prod_company_rank
FROM
movie
INNER JOIN
ratings
on movie.id = ratings.movie_id
WHERE avg_rating > 8 and movie.production_company is not null
GROUP BY movie.production_company
)
SELECT * FROM HIT_MOVIE_SUMMARY
WHERE prod_company_rank in ('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.genre,
count(*) AS movie_count
FROM
movie
INNER JOIN
genre
ON movie.id = genre.movie_id
INNER JOIN
ratings
USING(movie_id)
WHERE year = '2017' AND MONTH(date_published) = '3' AND movie.country =
'USA' AND ratings.total_votes > 1000
GROUP BY genre.genre;


-- 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
INNER JOIN
ratings
ON movie.id = ratings.movie_id
INNER JOIN
genre
ON movie.id = genre.movie_id
WHERE movie.title REGEXP '^The' AND avg_rating > 8
GROUP BY genre;

-- 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:

with btw_date as (
SELECT id, title as Title, date_published
FROM
movie
WHERE date_published BETWEEN '2018-04-01' and '2019-04-01')
select count(*) as 'Number of movies given 8 median rating which released
between 1 April 2018 and 1 April 2019' from btw_date
INNER JOIN
ratings
on btw_date.id = ratings.movie_id
where 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 sum(ratings.total_votes) as Total_votes,
languages
FROM
movie
INNER JOIN
ratings
on movie.id = ratings.movie_id
group by languages, country
having languages in ('German','Italian') and country in ('germany',
'Italy') ;

-- 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 count(*) as name_nulls
FROM
names
where name is null;

select count(*) as height_nulls
FROM
names
where height is null;

select count(*) as date_of_birth_nulls
FROM
names
where date_of_birth is null;

select count(*) as known_for_movies_nulls
FROM
names
where known_for_movies is null;



/* 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:

SELECT
```

```
name as director_name,
count(movie.id) as movie_count
FROM
director_mapping
INNER JOIN
names
on director_mapping.name_id = names.id
INNER JOIN
ratings
using(movie_id)
INNER JOIN
genre
using(movie_id)
INNER JOIN
movie
on ratings.movie_id = movie.id
where avg_rating > 8
group by genre, director_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
name as actor_name,
count(movie.id) as movie_count
FROM
role_mapping
INNER JOIN
names
on role_mapping.name_id = names.id
INNER JOIN
ratings
using(movie_id)
INNER JOIN
movie
on movie.id = ratings.movie_id
where median_rating >= 8
group by actor_name
order by movie_count desc limit 3;
```

```
/* 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(ratings.total_votes) AS 'vote_count',
RANK() OVER(ORDER BY sum(ratings.total_votes) desc) AS 'prod_comp_rank'

FROM
movie
INNER JOIN
ratings
on movie.id = ratings.movie_id
GROUP BY 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 INDIAN_ACTOR_SUMMARY AS
(
SELECT name AS actor_name,
sum(ratings.total_votes) as total_votes,
count(*) as movie_count,
round((sum(avg_rating*total_votes)/sum(total_votes)),2) as
actor_avg_rating

FROM
movie
INNER JOIN
role_mapping
on movie.id = role_mapping.movie_id
INNER JOIN
names
on  role_mapping.name_id = names.id
INNER JOIN
ratings
using(movie_id)
WHERE movie.country = 'India' AND role_mapping.category = 'actor'
GROUP BY name
)
SELECT *,
RANK() OVER(ORDER BY actor_avg_rating DESC, total_votes DESC ) AS
'actor_rank'
FROM
INDIAN_ACTOR_SUMMARY
WHERE movie_count >= 5 LIMIT 1;



-- 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 INDIAN_ACTRESS_SUMMARY AS
(
SELECT name AS actress_name,
sum(ratings.total_votes) as total_votes,
count(*) as movie_count,
round((sum(avg_rating*total_votes)/sum(total_votes)),2) as
actress_avg_rating

FROM
movie
INNER JOIN
role_mapping
on movie.id = role_mapping.movie_id
INNER JOIN
names
on  role_mapping.name_id = names.id
INNER JOIN
ratings
using(movie_id)
WHERE movie.country = 'India' AND role_mapping.category = 'actress' AND
movie.languages = 'Hindi'
GROUP BY actress_name
)
SELECT *,
RANK() OVER(ORDER BY ACTRESS_avg_rating DESC, total_votes DESC ) AS
'actor_rank'
FROM
INDIAN_ACTRESS_SUMMARY
WHERE movie_count >= 3  limit 1;

/* Taapsee Pannu tops with average rating 7.03.
```

```
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 movie.title as 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 'Movie_verdict'
from
movie
INNER JOIN
ratings
INNER JOIN
genre
using(movie_id)
on movie.id = ratings.movie_id
WHERE genre.genre = 'Thriller'
GROUP BY Title
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:


WITH genre_summary_AVG as (
WITH genre_summary AS
(
SELECT genre,
avg(duration) AS 'avg_duration'
FROM
movie
INNER JOIN
genre
on movie.id = genre.movie_id
GROUP BY genre
)
SELECT *,

sum(avg_duration) over w1 as running_total_duration,
avg(avg_duration) over w2 as moving_avg_duration
from genre_summary
window w1 as (order by avg_duration rows UNBOUNDED preceding),
w2 as (order by avg_duration rows UNBOUNDED preceding))
select genre, round(avg_duration) as avg_duration,
round(running_total_duration,1) AS running_total_duration,
round(moving_avg_duration,2) AS moving_avg_duration  FROM
genre_summary_AVG;


-- 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 worldwide_gross as (
with top_3_genre_movies as (
with master_movie as (
with dollar_To_inr as (
with formatedtostring as (
select *, replace(replace(CONVERT(worlwide_gross_income, char),'$ ', ''),
'INR ', '') AS values_without_currency_symbol
from movie
)
select *, convert(values_without_currency_symbol, float) as
back_numeric_state
from formatedtostring
)
select *,
case
when worlwide_gross_income regexp '^INR' then back_numeric_state/78.94
when worlwide_gross_income regexp '^[$]' then back_numeric_state/1
end as worldwide_gross_income_dummy
from dollar_To_inr )
select genre, year, title as movie_name, worlwide_gross_income,
worldwide_gross_income_dummy
from master_movie
INNER JOIN
genre
on master_movie.id = genre.movie_id
where genre in(with top_3_genre as
(
SELECT genre
from
movie
INNER JOIN
genre
on movie.id = genre.movie_id
group by genre
order by count(*) desc LIMIT 3)
SELECT *
from top_3_genre)
) SELECT *,
```

```
rank() over(partition by year order by worldwide_gross_income_dummy desc)
AS movie_rank
from top_3_genre_movies
group by movie_name)

select genre, year, movie_name, worlwide_gross_income AS
worldwide_gross_income, movie_rank from worldwide_gross

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:
with Production_comp_summary as (
with master_movie as (
select * from movie
where POSITION(',' IN languages)>0)
select production_company, count(*) as movie_count,
rank() over(order by count(*) desc) as prod_comp_rank
from master_movie
INNER JOIN
ratings
on master_movie.id = ratings.movie_id
where median_rating >= 8 and production_company is not null
group by production_company)
SELECT * from Production_comp_summary 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      |
```

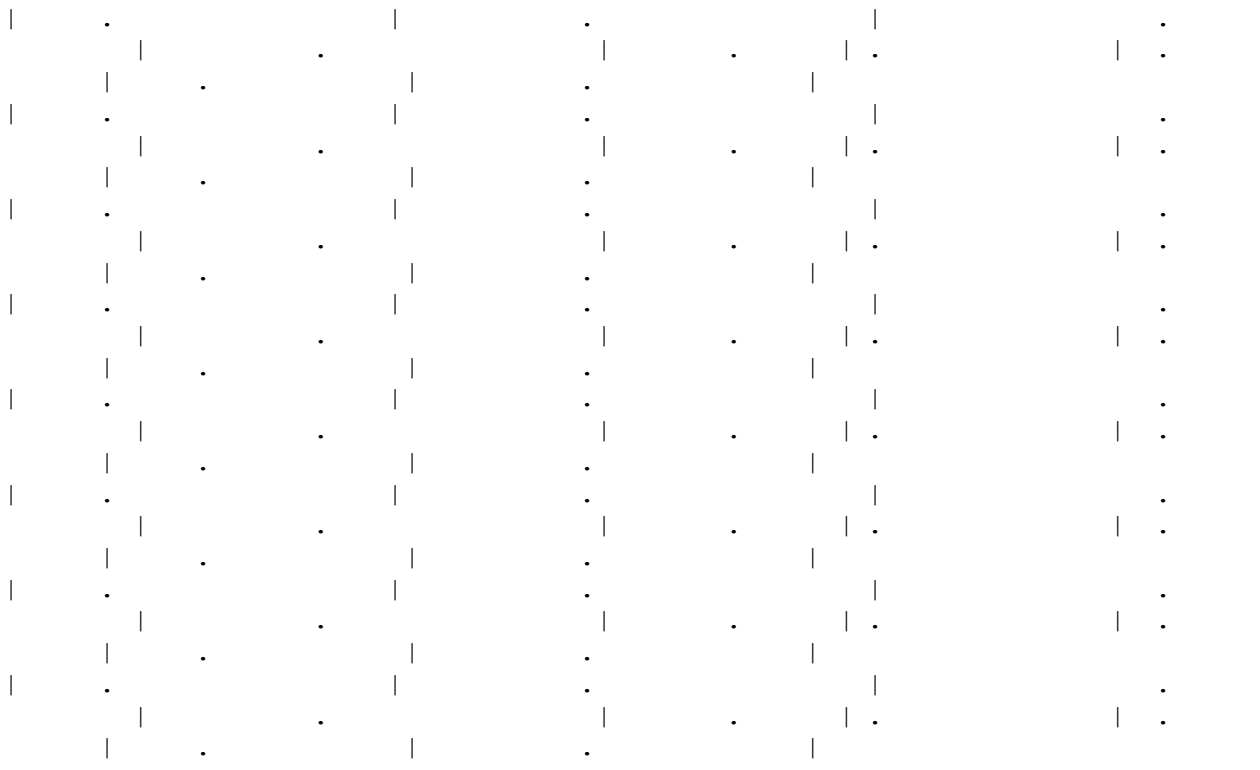
```
+---------------+-------------------+---------------------+---------------
-------+-----------------+
|     Laura Dern  |                1016  |            1            |
9.60                  |           1           |
|          .          |                  .          |            .
|        .             |          .          |
|           .          |                  .          |            .
|        .             |          .          |
+---------------+-------------------+---------------------+---------------
-------+-----------------+*/
-- Type your code below:
with hit_movies_actress as (
with hit_movies as (
with drama_movie as (
SELECT *
from movie
INNER JOIN
genre
on movie.id = genre.movie_id
where genre = 'drama')
select * from drama_movie
inner join
ratings
using(movie_id)
where avg_rating > 8)
SELECT name, total_votes, avg_rating FROM hit_movies
INNER JOIN
role_mapping
on hit_movies.id = role_mapping.movie_id
INNER JOIN
names
on role_mapping.name_id = names.id
where names.id in (SELECT names.id as actress_name
FROM
role_mapping
INNER JOIN
names
on role_mapping.name_id = names.id
where role_mapping.category = 'actress'))
select name as 'actress_name', sum(total_votes) as total_votes, count(*)
as movie_count, round(avg(avg_rating),2) as actress_avg_rating,
row_number() over( order by count(*) desc, avg(avg_rating) desc,
sum(total_votes) desc) as actress_rank from hit_movies_actress
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 director_summary_days_interval as (
with director_summary as (
with id_duration as (
select id as movies_id, duration , date_published

FROM movie)
```

```
SELECT *,  Lead(id_duration.date_published, 1) OVER(PARTITION BY
director_mapping.name_id ORDER BY id_duration.date_published, movies_id)
as lead_dates
FROM
id_duration
INNER JOIN
director_mapping
on id_duration.movies_id = director_mapping.movie_id
INNER JOIN
names
on director_mapping.name_id = names.id
INNER JOIN
ratings
using(movie_id))
select name_id AS director_id, name as director_name, count(*) AS
number_of_movies,
round(avg(datediff(lead_dates,date_published))) as 'Average inter movie
duration in days',
 round((sum(avg_rating*total_votes)/sum(total_votes)),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 director_summary
group by name_id
order by count(*) desc, avg_rating desc, total_votes desc limit 10)
select * from director_summary_days_interval;
```