```
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:

-- COUNT() function and UNION to Display total Number of rows in each table of the schema.

SELECT 'movie' AS `Table Name`, COUNT(*) AS `Num of Rows` FROM movie
UNION
SELECT 'genre' AS `Table Name`, COUNT(*) AS `Num of Rows` FROM genre
UNION
SELECT 'director_mapping' AS `Table Name`, COUNT(*) AS `Num of Rows` FROM
director_mapping
UNION
SELECT 'role_mapping' AS `Table Name`, COUNT(*) AS `Num of Rows` FROM role_mapping
UNION
SELECT 'names' AS `Table Name`, COUNT(*) AS `Num of Rows` FROM names
UNION
SELECT 'ratings' AS `Table Name`, COUNT(*) AS `Num of Rows` FROM ratings;



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

SELECT column_name
FROM information_schema.columns
WHERE table_name = 'movie' AND  is_nullable = 'YES';
```

-- column marked as is_nullable='yes' means that it allows null values, which indicates that the column can have missing or unknown 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:
-- Total number of movies released each year

```
SELECT year, COUNT(*) AS number_of_movies
FROM movie
WHERE year IS NOT NULL
GROUP BY year;
```

-- it looks like there were a total of 3052 movies released in 2017, 2944 movies released in 2018, and 2001 movies released in 2019.

-- Trend month-wise

```
SELECT MONTH(date_published) AS month_num, COUNT(*) AS number_of_movies
FROM movie
WHERE date_published IS NOT NULL
```

```
GROUP BY month_num;
-- The second part of the query shows the number of movies released each month
```

/*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
FROM movie
WHERE (country = 'USA' OR country = 'India') AND year = 2019;

--  887 movies were produced in either the USA or India in the year 2019.
```

/* USA and India produced more than a thousand movies(you know the exact number!) in the year 2019.
Exploring table Genre would be fun!!
Let’s find out the different genres in the dataset.*/

```
-- Q5. Find the unique list of the genres present in the data set?
-- Type your code below:

SELECT DISTINCT genre FROM genre;
```

/* So, RSVP Movies plans to make a movie of one of these genres.
Now, wouldn't you want to know which genre had the highest number of movies produced in the last year?
Combining both the movie and genres table can give more interesting insights. */

-- Q6.Which genre had the highest number of movies produced overall?
-- Type your code below:

```
SELECT genre, COUNT(*) AS number_of_movies
FROM genre
JOIN movie ON genre.movie_id = movie.id
GROUP BY genre
ORDER BY number_of_movies DESC
LIMIT 1;
```

-- Drama genre had the highest number of movies produced overall, 4285 movies.
-- Drama is a popular genre
-- This information could be useful for movie production companies, as they may want to consider producing more movies in this genre to appeal to a wider audience.
-- It could also be helpful for movie-goers who are looking for movies to watch in this genre.

/* 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:

```
-- Find the count of movies that belong to only one genre
SELECT COUNT(*) AS movies_with_single_genre
FROM (
  SELECT COUNT(movie_id) AS cnt
  FROM genre
  GROUP BY movie_id
  HAVING cnt = 1
```

) AS single_genre_movies;

-- 3289 movies belong to only one genre.

/* There are more than three thousand movies which has only one genre associated with them.
So, this figure appears significant.
Now, let's find out the possible duration of RSVP 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.genre, AVG(movie.duration) AS avg_duration
FROM movie
JOIN genre ON movie.id = genre.movie_id
GROUP BY genre.genre;

-- the average duration of drama movies being the highest at 106.7746 minutes, followed by romance and crime movies.
-- horror and sci-fi movies have the lowest average duration, with horror movies being the shortest at 92.7243 minutes.

/* 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 the genre and count the number of movies in each genre
SELECT genre, COUNT(*) AS movie_count,
RANK() OVER (ORDER BY COUNT(*) DESC) AS genre_rank
FROM genre
JOIN movie ON genre.movie_id = movie.id
WHERE genre = 'thriller'
GROUP BY genre;

-- 'thriller' genre has a count of 1484.
-- The query uses the RANK function to assign the rank of 1 to the 'thriller' genre, indicating that it is the genre with the most movies produced among all the genres in the database.
-- This result suggests that the 'thriller' genre is a popular genre for movie production.

/*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;
```

-- The minimum average rating is 1.0 and the maximum average rating is 10.0.
-- The minimum total votes for a movie is 100 and the maximum is 725138.
-- The minimum median rating is 1 and the maximum is 10.

/* 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

/* To find the top 10 movies based on average rating, we need to join the 'movies' and 'ratings' tables
and calculate the average rating for each movie. Then, we can rank the movies based on the average rating. */

SELECT m.title, r.avg_rating, RANK() OVER (ORDER BY r.avg_rating DESC) AS movie_rank
FROM movie m
JOIN ratings r ON m.id = r.movie_id
ORDER BY r.avg_rating DESC
LIMIT 10;

-- It is interesting to see that two movies, "Kirket" and "Love in Kilnerry," have received a perfect rating of 10.0 and are ranked first.
-- This indicates that these movies have received extremely positive reviews from viewers.
-- The third-ranked movie, "Gini Helida Kathe," has a high average rating of 9.8, indicating that it is also very popular among viewers.
-- The fourth and fifth-ranked movies, "Runam" and "Fan," respectively, have an average rating of 9.7 and 9.6. These movies are also highly rated and seem to be popular among viewers.
-- It is interesting to see that two movies, "Android Kunjappan Version 5.25" and "Fan," have the same average rating of 9.6 and are ranked jointly at the fifth position.
-- Similarly, three movies, "Yeh Suhaagraat Impossible," "Safe," and "The Brighton Miracle," have the same average rating of 9.5 and are ranked jointly at the seventh position.
-- This indicates that these movies have received similar positive reviews from viewers.

-- Overall, the output shows that the top-ranked movies have received extremely positive reviews from viewers and are highly recommended to watch.

/* 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 median_rating;
```

-- The most common median rating value appears to be 6, with 1975 movies having this rating.
-- the least common median rating values are 1 and 2, with only 94 and 119 movies having those ratings, respectively.

/* 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:

-- First, we need to join the movie and ratings tables to get the average rating for each movie

```
SELECT m.production_company, COUNT(*) AS movie_count, AVG(r.avg_rating) AS
avg_movie_rating
FROM movie m
JOIN ratings r ON m.id = r.movie_id
GROUP BY m.production_company
HAVING AVG(r.avg_rating) > 8
ORDER BY movie_count DESC
LIMIT 1;
```

-- "National Theatre Live" has produced 3 movies that have an average rating greater than 8.
-- The average rating of the movies produced by this company is 8.5.

-- 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:

-- Count the number of movies in each genre released in March 2017 in the USA with more than 1000 votes

```
SELECT g.genre, COUNT(*) AS movie_count
FROM movie m
JOIN genre g ON m.id = g.movie_id
JOIN ratings r ON m.id = r.movie_id
WHERE m.country = 'USA'
```

AND MONTH(m.date_published) = 3
AND YEAR(m.date_published) = 2017
AND r.total_votes > 1000
GROUP BY g.genre;

-- there were 16 drama movies released in March 2017 in the USA that had more than 1000 votes.
-- Similarly, there were 5 crime movies, 4 sci-fi movies, and 5 horror movies that met the conditions.

-- This information can be useful for filmmakers, production companies, and movie studios to understand which genres are popular among the audience and what kind of movies they should focus on producing to attract more viewers and increase their revenue.

-- 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:

-- Find movies of each genre that start with the word 'The' and which have an average rating > 8
-- select the required columns from movie and genre table
SELECT m.title, r.avg_rating, g.genre
FROM movie m
JOIN genre g ON m.id = g.movie_id
JOIN ratings r ON m.id = r.movie_id
WHERE g.genre LIKE 'The%' AND r.avg_rating > 8
ORDER BY g.genre, m.title;

-- No output for the above query
-- There are no movies that start with 'The' in each genre and have a average rating > 8

-- 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(*) AS movie_count
FROM movie m
JOIN ratings r ON m.id = r.movie_id
WHERE date_published BETWEEN '2018-04-01' AND '2019-04-01' AND r.median_rating = 8;
```

-- there were 361 movies released between 1 April 2018 and 1 April 2019 that were given a median rating of 8.

-- Once again, try to solve the problem given below.
-- Q17. Do German movies get more votes than Italian movies?
-- Hint: Here you have to find the total number of votes for both German and Italian movies.
-- Type your code below:

```
SELECT m.country, SUM(r.total_votes) AS total_votes
FROM movie m
JOIN ratings r ON m.id = r.movie_id
WHERE m.country IN ('Germany', 'Italy')
GROUP BY m.country;
```

-- It seems that movies produced in Germany received more votes than movies produced in Italy.

-- it's important to note that this analysis only considers the number of votes and does not take into account other factors that may impact the popularity or quality of the movies, such as their genre.

-- 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(CASE WHEN name IS NULL THEN 1 END) AS name_nulls,
  COUNT(CASE WHEN height IS NULL THEN 1 END) AS height_nulls,
  COUNT(CASE WHEN date_of_birth IS NULL THEN 1 END) AS date_of_birth_nulls,
  COUNT(CASE WHEN known_for_movies IS NULL THEN 1 END) AS known_for_movies_nulls
FROM names;
```

-- The output will show the number of null values for each column in separate columns.

/* 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 n.name AS director_name, COUNT(DISTINCT dm.movie_id) AS movie_count
FROM director_mapping dm
JOIN names n ON dm.name_id = n.id
JOIN genre g ON dm.movie_id = g.movie_id
JOIN ratings r ON g.movie_id = r.movie_id
WHERE g.genre IN (
  SELECT genre
  FROM (
    SELECT genre, COUNT(*) AS movie_count
    FROM genre
    JOIN ratings ON genre.movie_id = ratings.movie_id
    WHERE ratings.avg_rating > 8
    GROUP BY genre
    ORDER BY movie_count DESC
  ) top_genres
)
GROUP BY dm.name_id
HAVING AVG(r.avg_rating) > 8
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(DISTINCT rm.movie_id) AS movie_count
FROM names n
JOIN role_mapping rm ON n.id = rm.name_id
JOIN ratings r ON rm.movie_id = r.movie_id
WHERE rm.category = 'actor' AND r.median_rating >= 8
GROUP BY n.name
ORDER BY movie_count DESC
LIMIT 2;
```

-- It indicates that Mammootty has acted in 8 movies that have a median rating greater than or equal to 8,
-- while Mohanlal has acted in 5 such movies.

/* 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 AS 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
JOIN ratings r ON m.id = r.movie_id
WHERE m.production_company IS NOT NULL
GROUP BY m.production_company
ORDER BY vote_count DESC
LIMIT 3;

-- It indicates that Marvel Studios is the production company with the highest number of votes
received by their movies,
-- with a total of 2,656,967 votes.
-- Twentieth Century Fox is ranked second with 2,411,163 votes, followed by Warner Bros. with
2,396,057 votes.

/*Yes Marvel Studios rules the movie world.
So, these are the top three production houses based on the number of votes received by the
movies they have produced.

Since RSVP Movies is based out of Mumbai, India also wants to woo its local audience.
RSVP Movies also wants to hire a few Indian actors for its upcoming project to give a regional
feel.
Let's find who these actors could be.*/

-- Q22. Rank actors with movies released in India based on their average ratings. Which actor is
at the top of the list?
-- Note: The actor should have acted in at least five Indian movies.
-- (Hint: You should use the weighted average based on votes. If the ratings clash, then the total
number of votes should act as the tie breaker.)

/* Output format:
+---------------+-------------------+---------------------+----------------------+-----------------+
| actor_name   |          total_votes                |          movie_count                |
actor_avg_rating         |actor_rank       |
+---------------+-------------------+---------------------+----------------------+-----------------+
```

```
|       Yogi Babu       |                       3455    |               11              |           8.42
        |               1               |
|               .               |                       .               |           .           |
.                       |               .               |
|               .               |                       .               |           .           |
.                       |               .               |
|               .               |                       .               |           .           |
.                       |               .               |
+---------------+-------------------+---------------------+----------------------+-----------------+*/
-- Type your code below:


SELECT n.name AS actor_name,
       SUM(r.total_votes) AS total_votes,
       COUNT(DISTINCT m.id) AS movie_count,
       ROUND(SUM(r.avg_rating * r.total_votes) / SUM(r.total_votes), 2) AS actor_avg_rating,
       DENSE_RANK() OVER (ORDER BY ROUND(SUM(r.avg_rating * r.total_votes) /
SUM(r.total_votes), 2) DESC, SUM(r.total_votes) DESC) AS actor_rank
FROM names n
JOIN role_mapping rm ON n.id = rm.name_id
JOIN movie m ON rm.movie_id = m.id
JOIN ratings r ON m.id = r.movie_id
WHERE m.country = 'India'
GROUP BY n.id
HAVING COUNT(DISTINCT CASE WHEN m.country = 'India' THEN m.id END) >= 5
ORDER BY actor_rank ASC;
```

-- The query shows the ranking of actors with movies released in India based on their average ratings.
-- Vijay Sethupathi is at the top of the list with an average rating of 8.42 based on 5 movies,
-- followed by Fahadh Faasil with an average rating of 7.99 based on 5 movies and Yogi Babu with an average rating of 7.83 based on 11 movies.
-- It seems that actors with fewer movies tend to have higher average ratings, but this may not necessarily be true in all cases.

-- 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:

-- select actresses who have acted in at least three Hindi movies

```
SELECT n.name AS actress_name,
       SUM(r.total_votes) AS total_votes,
       COUNT(DISTINCT r.movie_id) AS movie_count,
       ROUND(SUM(r.avg_rating * r.total_votes) / SUM(r.total_votes), 2) AS actress_avg_rating,
       ROW_NUMBER() OVER (ORDER BY ROUND(SUM(r.avg_rating * r.total_votes) /
SUM(r.total_votes), 2) DESC, SUM(r.total_votes) DESC) AS actress_rank
FROM names n
JOIN role_mapping rm ON n.id = rm.name_id
JOIN movie m ON rm.movie_id = m.id
JOIN ratings r ON m.id = r.movie_id
WHERE rm.category = 'actress'
  AND m.country = 'India'
  AND m.languages LIKE '%Hindi%'
GROUP BY n.id
HAVING COUNT(DISTINCT r.movie_id) >= 3
ORDER BY actress_rank
LIMIT 5;
```

/* Taapsee Pannu tops with average rating 7.74.
Now let us divide all the thriller movies in the following categories and find out their numbers.*/

/* Q24. Select thriller movies as per avg rating and classify them in the following category:

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

-- Type your code below:

```
SELECT m.title,
      r.avg_rating,
      CASE
         WHEN r.avg_rating > 8 THEN 'Superhit movies'
         WHEN r.avg_rating BETWEEN 7 AND 8 THEN 'Hit movies'
         WHEN r.avg_rating BETWEEN 5 AND 7 THEN 'One-time-watch movies'
         WHEN r.avg_rating < 5 THEN 'Flop movies'
      END AS category
FROM movie m
JOIN ratings r ON m.id = r.movie_id
JOIN genre g ON m.id = g.movie_id
WHERE g.genre = 'Thriller'
ORDER BY r.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,
  SUM(avg_duration) OVER (PARTITION BY genre ORDER BY genre) AS
running_total_duration,
  AVG(avg_duration) OVER (PARTITION BY genre ORDER BY genre ROWS BETWEEN
UNBOUNDED PRECEDING AND CURRENT ROW) AS moving_avg_duration
FROM
  (SELECT
    g.genre,
    AVG(m.duration) AS avg_duration
  FROM
    genre g
    JOIN movie m ON g.movie_id = m.id
  GROUP BY
    g.genre) subquery;
```

-- 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

```sql
WITH top_genres AS (
   SELECT genre, COUNT(*) AS movie_count
   FROM genre
   GROUP BY genre
   ORDER BY movie_count DESC
   LIMIT 3
),
top_movies AS (
   SELECT m.year, g.genre, m.title AS movie_name, m.worlwide_gross_income,
        ROW_NUMBER() OVER (PARTITION BY m.year, g.genre ORDER BY
m.worlwide_gross_income DESC) AS movie_rank
   FROM movie m
   JOIN genre g ON m.id = g.movie_id
   JOIN top_genres tg ON g.genre = tg.genre
)
SELECT genre, year, movie_name, worlwide_gross_income, movie_rank
FROM top_movies
WHERE movie_rank <= 5
ORDER BY genre, year, movie_rank;
```

-- This query utilizes Common Table Expressions (CTEs) to first determine the top three genres with the most number of movies.
-- Then, it selects the movies from those genres and assigns a rank to each movie based on its worldwide gross income within each genre and year combination.
-- Finally, the query filters for movies with ranks up to 5 and orders the results by genre, year, and movie rank.

-- 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 hit_movies AS (
   SELECT movie_id
   FROM ratings
   WHERE median_rating >= 8
),
multilingual_hits AS (
   SELECT movie_id
   FROM hit_movies
   WHERE movie_id IN (
      SELECT movie_id
      FROM movie
      WHERE languages LIKE '%,%'
   )
),
production_house_counts AS (
   SELECT production_company, COUNT(*) AS movie_count,
        ROW_NUMBER() OVER (ORDER BY COUNT(*) DESC) AS prod_comp_rank
   FROM movie
   WHERE id IN (
      SELECT movie_id
      FROM multilingual_hits
   )
   GROUP BY production_company
)
SELECT production_company, movie_count, prod_comp_rank
FROM production_house_counts
WHERE prod_comp_rank <= 2;
```

-- This query uses Common Table Expressions (CTEs) to first identify the movie IDs of hits (median rating >= 8).
-- Then, it selects the movie IDs that belong to multilingual movies by checking if the movie's languages field contains a comma (',').
-- Next, it counts the movies produced by each production company for the multilingual hits and assigns a rank based on the count.

-- 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 super_hit_movies AS (
   SELECT movie_id
   FROM ratings
   WHERE avg_rating > 8
),
drama_super_hit_movies AS (
   SELECT movie_id
   FROM super_hit_movies
   WHERE movie_id IN (
      SELECT movie_id
      FROM genre
      WHERE genre = 'drama'
   )
),
actress_counts AS (
   SELECT nm.name AS actress_name, COUNT(*) AS movie_count, SUM(r.total_votes) AS total_votes,
         AVG(r.avg_rating) AS actress_avg_rating,
```

```
        ROW_NUMBER() OVER (ORDER BY COUNT(*) DESC) AS actress_rank
    FROM role_mapping rm
    JOIN names nm ON rm.name_id = nm.id
    JOIN drama_super_hit_movies dshm ON rm.movie_id = dshm.movie_id
    JOIN ratings r ON rm.movie_id = r.movie_id
    WHERE rm.category = 'actress'
    GROUP BY nm.name
)
SELECT actress_name, total_votes, movie_count, actress_avg_rating, actress_rank
FROM actress_counts
WHERE actress_rank <= 3;
```

-- This query uses Common Table Expressions (CTEs) to first identify the movie IDs of Super Hit movies (average rating > 8) and select those that belong to the drama genre.
-- Then, it counts the number of movies, total votes, and calculates the average rating for each actress who played a role in the drama Super Hit movies.
-- The actresses are ranked based on the count of movies.

-- Parvathy Thiruvothu has established a strong presence in the drama genre, as evidenced by the high average rating and significant total votes.
-- Susan Brown and Amanda Lawrence have also achieved success in the drama genre with their performances in Super Hit movies,
-- although their total number of votes is relatively lower compared to Parvathy Thiruvothu.

/* 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_details AS (
   SELECT dm.name_id AS director_id, nm.name AS director_name, COUNT(*) AS
number_of_movies,
        AVG(m.duration) AS avg_inter_movie_days, AVG(r.avg_rating) AS avg_rating,
        SUM(r.total_votes) AS total_votes, MIN(r.avg_rating) AS min_rating, MAX(r.avg_rating)
AS max_rating,
        SUM(m.duration) AS total_duration,
        ROW_NUMBER() OVER (ORDER BY COUNT(*) DESC) AS director_rank
   FROM director_mapping dm
   JOIN names nm ON dm.name_id = nm.id
```

```
    JOIN movie m ON dm.movie_id = m.id
    JOIN ratings r ON dm.movie_id = r.movie_id
    GROUP BY dm.name_id, nm.name
)
SELECT director_id, director_name, number_of_movies, avg_inter_movie_days, avg_rating,
       total_votes, min_rating, max_rating, total_duration
FROM director_details
WHERE director_rank <= 9;

-- This query uses a Common Table Expression (CTE) named director_details to calculate the
required details for each director.
-- It retrieves information from the director_mapping, names, movie, and ratings tables, including
the director ID, name,
-- number of movies, average inter-movie duration in days, average movie ratings, total votes,
minimum rating, maximum rating,
-- and total movie durations.
-- The directors are ranked based on the count of movies.
```