SQL Worksheet 5 Answers

Refer the following ERD and answer all the questions in this worksheet. You have to write the queries using MySQL for the required Operation.

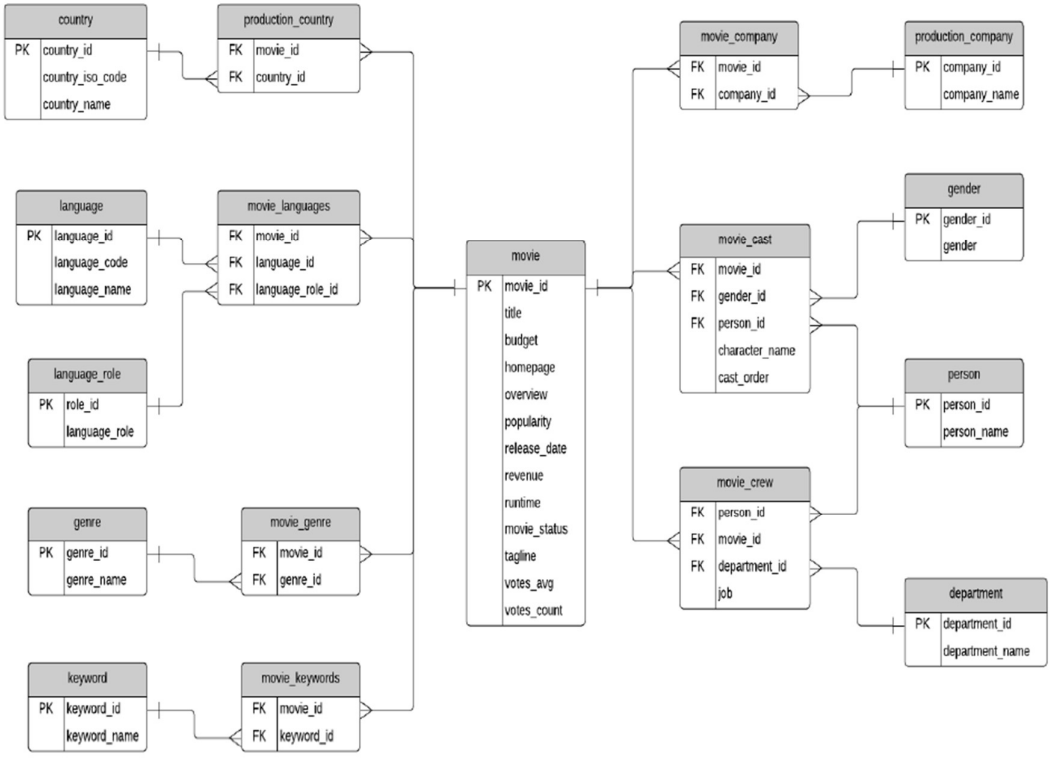

Table Explanations:

- The **movie** table contains information about each movie. There are text descriptions such as title and overview. Some fields are more obvious than others: revenue (the amount of money the movie made), budget (the amount spent on creating the movie). Other fields are calculated based on data used to create the data source: popularity, votes_avg, and votes_count. The status indicates if the movie is Released, Rumoured, or in Post-Production.
- The **country** list contains a list of different countries, and the **movie_country** table contains a record of which countries a movie was filmed in (because some movies are filmed in multiple countries). This is a standard many-to-many table, and you'll find these in a lot of databases.
- The same concept applies to the **production_company** table. There is a list of production companies and a many-to-many relationship with movies which is captured in the **movie_company** table.
- The **languages** table has a list of languages, and the **movie_languages** capture a list of languages in a movie. The difference with this structure is the addition of a **language_role** table.
- This **language_role** table contains two records: Original and Spoken. A movie can have an original language (e.g. English), but many Spoken languages. This is captured in the **movie_languages** table along with a role.

- **Genres** define which category a movie fits into, such as Comedy or Horror. A movie can have multiple genres, which is why the **movie_genres** table exists.
- The same concept applies to keywords, but there are a lot more keywords than genres. I'm not sure what qualifies as a keyword, but you can explore the data and take a look. Some examples as "paris", "gunslinger", or "saving the world".
- The cast and crew section of the database is a little more complicated. Actors, actresses, and crew members are all people, playing different roles in a movie. Rather than have separate lists of names for crew and cast, this database contains a table called person, which has each person's name.
- The **movie_cast** table contains records of each person in a movie as a cast member. It has their character name, along with the cast_order, which I believe indicates that lower numbers appear higher on the cast list.
- The **movie_cast** table also links to the gender table, to indicate the gender of each character. The gender is linked to the movie_cast table rather than the person table to cater for characters which may be a different gender than the person, or characters of unknown gender. This means that there is no gender table linked to the person table, but that's because of the sample data.
- **The movie_crew** table follows a similar concept and stores all crew members for all movies. Each crew member has a job, which is part of a department (e.g. Camera).

1. Write SQL query to show all the data in the Movie table.

```
SELECT * FROM movie;
```

2. Write SQL query to show the title of the longest runtime movie.

```
SELECT title FROM movie ORDER BY runtime DESC LIMIT 1;
```

3. Write SQL query to show the highest revenue generating movie title.

```
SELECT title FROM movie ORDER BY revenue DESC LIMIT 1;
```

4. Write SQL query to show the movie title with maximum value of revenue/budget.

```
SELECT title FROM movie ORDER BY budget DESC LIMIT 1;
```

5. Write a SQL query to show the movie title and its cast details like name of the person, gender, character name, cast order.

```
SELECT title, gender, character_name, cast_order, person_name FROM movie AS a INNER JOIN movie_cast AS b ON a.movie_id=b.movie_id INNER JOIN gender AS c ON c.gender_id=b.gender_id INNER JOIN person AS d ON d.person_id= b.person_id;
```

6. Write a SQL query to show the country name where maximum number of movies has been produced, along with the number of movies produced.

SELECT country_name, COUNT(country_name) AS count FROM country AS a INNER JOIN production_country AS b ON b.country_id=a.country_id GROUP BY country_name ORDER BY count DESC limit 1;

7. Write a SQL query to show all the genre_id in one column and genre_name in second column.

Select * from genre;

8. Write a SQL query to show name of all the languages in one column and number of movies in that particular column in another column.

SELECT language_name, movie_id, COUNT(language_name) FROM movie_languages AS a JOIN language AS b ON a.language_id=b.language_id GROUP BY language_name ORDER BY count(language_name) DESC;

9. Write a SQL query to show movie name in first column, no. of crew members in second column and number of cast members in third column.

SELECT m.title AS movie_name, COUNT(cr.person_id) AS no_of_crews, COUNT(ca.person_id) AS no_of_cast FROM movie AS m INNER JOIN movie_crew AS cr ON cr.movie_id=m.movie_id INNER JOIN movie_cast AS ca ON ca.person_id=cr_person_id;

10. Write a SQL query to list top 10 movies title according to popularity column in decreasing order.

SELECT title FROM movie ORDER BY popularity DESC LIMIT 10;

11. Write a SQL query to show the name of the 3rd most revenue generating movie and its revenue.

SELECT title FROM movie ORDER BY revenue DESC OFFSET 3 LIMIT 1;

12. Write a SQL query to show the names of all the movies which have “rumoured” movie status.

SELECT title FROM movie WHERE movie_status LIKE ‘rumored’;

13. Write a SQL query to show the name of the "United States of America" produced movie which generated maximum revenue.

SELECT title, revenue FROM movie AS a INNER JOIN production_country AS b ON b.movie_id = a.movie_id INNER JOIN country AS c ON c.country_id = b. country_id WHERE country_name= 'United State of America';

14. Write a SQL query to print the movie_id in one column and name of the production company in the second column for all the movies.

SELECT m.movie_id, pc.company_name FROM movie AS m INNER JOIN movie_company AS mc ON mc.movie_id = m.movie_id INNER JOIN production_company AS pc ON pc.company_id =mc.company_id;

15. Write a SQL query to show the title of top 20 movies arranged in decreasing order of their budget.

SELECT title FROM movie ORDER BY budget DESC LIMIT 20;