WORKSHEET 5

SQL



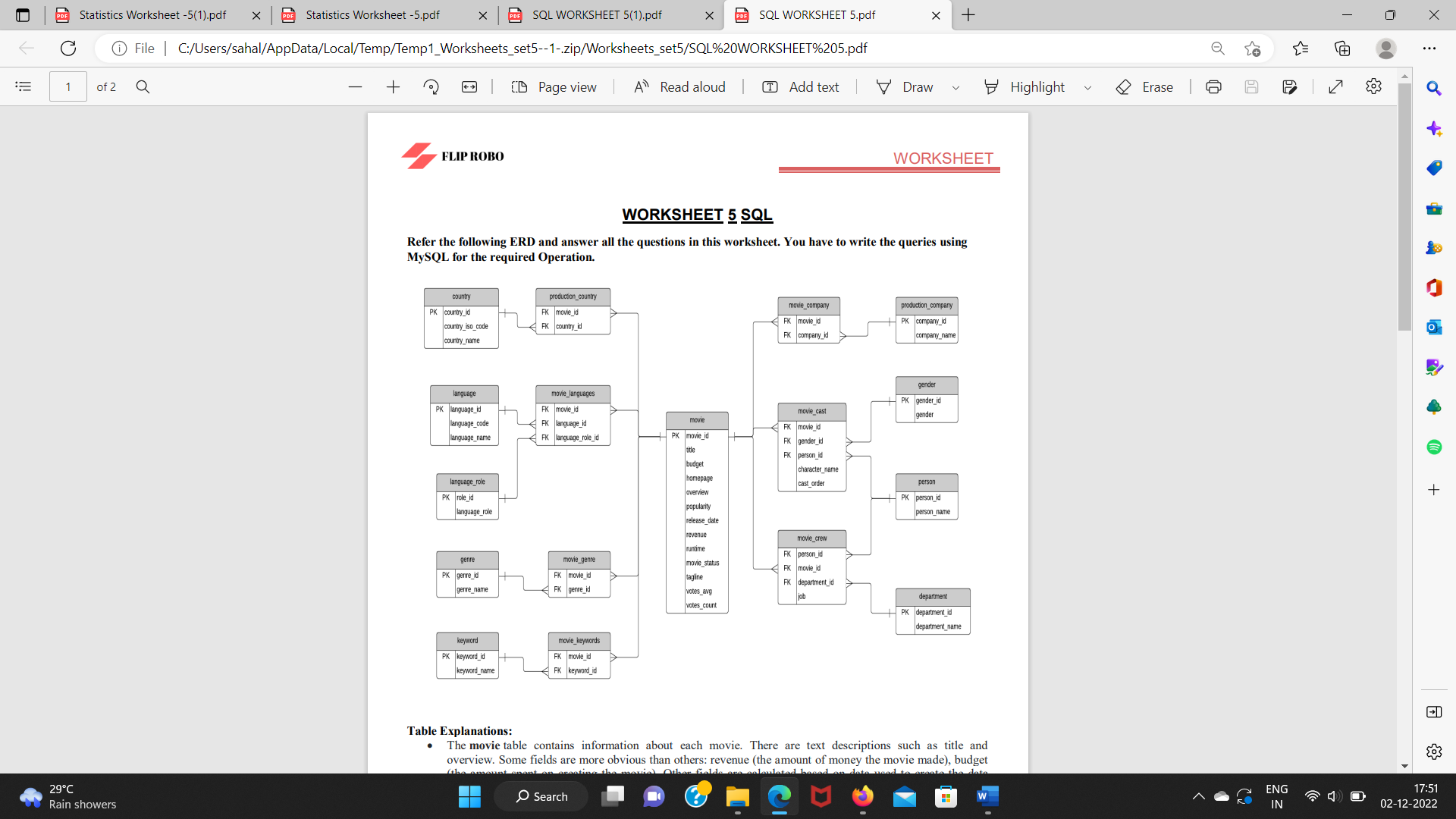
INERNSHIP 32

FLIP ROBO TECHNOLOGIES

Submitted By : Shahla M

SME : KHUSHBOO GARG

**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 captures 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. WORKSHEET

• 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).

QUESTIONS:

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

Answer:

SELECT \* FROM movie;

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

Answer:

SELECT title FROM movie WHERE runtime= (SELECT MAX (runtime) FROM movie;

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

. Answer:

SELECT title FROM movie WHERE revenue= (SELECT MAX (revenue) FROM movie;

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

Answer:

SELECT title FROM movie WHERE revenue= (SELECT MAX (revenue) FROM movie OR

budget= (SELECT MAX (budget) FROM movie;

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

Answer:

SELECT movie.title, person.person\_name, gender.gender, movie\_cast.cast\_order

FROM movie\_cast

INNER JOIN movie

ON movie\_cast.movie\_id= movie. movie\_id

INNER JOIN person

ON movie\_cast. person\_id= person. person \_id

INNER JOIN gender

ON movie\_cast.gender\_id= gender. gender\_id;

1. **Write a SQL query to show the country name where maximum number of movies has been produced, alongwith the number of movies produced.**

Answer:

SELECT country\_name, count(country\_name) AS no\_mov\_prd

FROM country

INNER JOIN production\_country ON country. country\_id= production. country \_id GROUP BY country\_name

ORDER BY count(country.country\_name) desc limit 1;

1. Write a SQL query to show all the genre\_id in one column and genre\_name in second column.

Answer:

SELECT \* FROM genre;.

1. **Write a SQL query to show name of all the languages in one column and number of movies in thatparticular column in another column.**

Answer:

SELECT language\_name, count (language\_name) AS no\_of\_movie

FROM language

INNER JOIN movie\_ language

ON movie\_ language. language\_ id= language.language \_ id

INNER JOIN movie

ON movie\_ language.movie\_ id=movie. movie\_ id

GROUP BY language\_name;

1. **Write a SQL query to show movie name in first column, no. of crew members in second column andnumber of cast members in third column.**

Answer:

SELECT movie.title, count (movie\_crew.job), count(movie\_cast.character\_name)

FROM movie\_crew

INNER JOIN movie

ON movie\_crew.movie\_id= movie.movie\_id

INNER JOIN movie\_cast

ON movie\_crew.movie\_id=movie\_cast.movie\_id

GROUP BY movie.title;

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

Answer:

SELECT title, popularity FROM movie ORDER BY popularity DESC LIMIT 10;

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

Answer:

SELECT title, revenue FROM movie

ORDER BY revenue DESC

LIMIT 2,1;

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

Answer:

SELECT tiltle FROM movie WHERE movie\_status= “rumoured”;

1. **Write a SQL query to show the name of the “United States of America” produced movie which generated maximum revenue**

Answer:

SELECT movie.title, production\_company.company\_name, max(movie.revenue)

FROM movie\_company

INNER JOIN movie

ON movie\_company.movie\_id= movie.movie\_id

INNER JOIN production\_company

ON movie\_company.company\_id=production\_company.company\_id

WHERE production\_company.company\_name= “United States of America”

ORDER BY revenue DESC;

1. **Write a SQL query to print the movie\_id in one column and name of the production company in the secondcolumn for all the movies.**

Answer:

SELECT movie.movie\_id, production\_company.company\_name

FROM movie\_company.movie\_id=movie.movie\_id

INNER JOIN production\_company

ON movie\_company. company\_id= production\_company. company\_id;

1. **Write a SQL query to show the title of top 20 movies arranged in decree**

**sing order of their budget.**

Answer:

SELECT title from movie ORDER BY budget DESC LIMIT 10 :