PAT Task-8

Create Database:

```
mysql> CREATE DATABASE IMDB;
Query OK, 1 row affected (0.01 sec)

mysql> USE IMDB;
Database changed
```

Create Tables

1. **Movies Table** - Contains details about movies.

```
mysql> CREATE TABLE Movies(movie_id INT AUTO_INCREMENT PRIMARY KEY, movie_name Varchar(255));
Query OK, 0 rows affected (0.04 sec)

mysql> INSERT INTO Movies (Movie_Name) VALUES ("Avatar The Last Airbender"),
    -> ("Naruto") , ("Harry Potter") , ("Iron Man"), ("Pokemon") , ("Thor Ragnarok");
Query OK, 6 rows affected (0.01 sec)
Records: 6  Duplicates: 0  Warnings: 0

mysql> ^C
mysql> SELECT * from Movies;
+----------+---------------------------+
| movie_id | movie_name                |
+----------+---------------------------+
|        1 | Avatar The Last Airbender |
|        2 | Naruto                    |
|        3 | Harry Potter              |
|        4 | Iron Man                  |
|        5 | Pokemon                   |
|        6 | Thor Ragnarok             |
+----------+---------------------------+
```

2. **Media Table** - Multiple media like video or images(comic).

```
mysql> CREATE TABLE Media(media_ID INT AUTO_INCREMENT PRIMARY KEY,movie_ID INT,media_Type VARCHAR(255),FOREIGN KEY (movie_ID) REFEREN
CES Movies(movie_ID));
Query OK, 0 rows affected (0.06 sec)

mysql> INSERT INTO Media (movie_ID,media_type) VALUES (1,'Video'), (2,'image'),(3,'image'),(4,'video'),(5,'video'),(6,'video');
Query OK, 6 rows affected (0.01 sec)
Records: 6  Duplicates: 0  Warnings: 0

mysql> SELECT * from Media;
+----------+----------+------------+
| media_ID | movie_ID | media_Type |
+----------+----------+------------+
|        1 |        1 | Video      |
|        2 |        2 | image      |
|        3 |        3 | image      |
|        4 |        4 | video      |
|        5 |        5 | video      |
|        6 |        6 | video      |
+----------+----------+------------+
```

3. **Genre Table** - Various Genres.

```
mysql> CREATE TABLE Genre(
    ->     genre_ID INT AUTO_INCREMENT PRIMARY KEY,
    ->     genre_name VARCHAR(255));
Query OK, 0 rows affected (0.04 sec)

mysql> INSERT INTO Genre (genre_name) VALUES ('Action'),('Adventure'), ('Fantasy'),('comic'),('Fantasy Literature'),
    ->                                       ('Superhero fiction'),('Animation'),('Comedy');
Query OK, 8 rows affected (0.01 sec)
Records: 8  Duplicates: 0  Warnings: 0

mysql> SELECT * from Genre;
+----------+--------------------+
| genre_ID | genre_name         |
+----------+--------------------+
|        1 | Action             |
|        2 | Adventure          |
|        3 | Fantasy            |
|        4 | comic              |
|        5 | Fantasy Literature |
|        6 | Superhero fiction  |
|        7 | Animation          |
|        8 | Comedy             |
+----------+--------------------+
```

4. **MovieGenres Table** - Movies have multiple genres.

```
mysql> Create Table MovieGenre(
    ->     movie_ID INT,
    ->     genre_ID INT,
    ->     FOREIGN KEY (movie_ID) REFERENCES Movies(movie_ID),
    ->     FOREIGN KEY (genre_ID) REFERENCES Genre (genre_ID));
Query OK, 0 rows affected (0.07 sec)

mysql> INSERT INTO MovieGenre (movie_ID, genre_ID) VALUES (1,1),(1,2),(1,3),
    -> (2,3),(2,4),(3,5),(4,6),(5,2),(5,7),(6,6);
Query OK, 10 rows affected (0.01 sec)
Records: 10  Duplicates: 0  Warnings: 0

mysql> SELECT * from MovieGenre;
+----------+----------+
| movie_ID | genre_ID |
+----------+----------+
|        1 |        1 |
|        1 |        2 |
|        1 |        3 |
|        2 |        3 |
|        2 |        4 |
|        3 |        5 |
|        4 |        6 |
|        5 |        2 |
|        5 |        7 |
|        6 |        6 |
+----------+----------+
10 rows in set (0.00 sec)
```

5. **Users Table** - Contains user information.

```
mysql> Insert into User (user_name) VALUES ('Sam'),('Ash'),('Serena'),('Harry'),('Shree');
Query OK, 5 rows affected (0.03 sec)
Records: 5  Duplicates: 0  Warnings: 0

mysql> SELECT * from User;
+---------+-----------+
| user_ID | user_name |
+---------+-----------+
|       1 | Sam       |
|       2 | Ash       |
|       3 | Serena    |
|       4 | Harry     |
|       5 | Shree     |
+---------+-----------+
5 rows in set (0.00 sec)
```

6. **MovieReview Table** - Stores reviews for movies.

```
mysql> Create TABLE MovieReview(
    ->     movie_ID INT,
    ->     user_ID INT,
    ->     Review VARCHAR (255),
    ->     FOREIGN KEY (movie_ID) REFERENCES Movies (movie_ID),
    ->     FOREIGN KEY (user_ID) REFERENCES User (user_ID));
Query OK, 0 rows affected (0.05 sec)

mysql> INSERT INTO MovieReview (movie_ID, user_ID, review) VALUES (4,1,'Very Good'), (6,1,'Good'), (5,2,'Excellent'),
    -> (5,3,'Very Good'), (3,4,'Average'), (2,5,'Good'), (1,5,'Excellent');
Query OK, 7 rows affected (0.01 sec)
Records: 7  Duplicates: 0  Warnings: 0

mysql> SELECT * from MovieReview;
+----------+---------+-----------+
| movie_ID | user_ID | Review    |
+----------+---------+-----------+
|        4 |       1 | Very Good |
|        6 |       1 | Good      |
|        5 |       2 | Excellent |
|        5 |       3 | Very Good |
|        3 |       4 | Average   |
|        2 |       5 | Good      |
|        1 |       5 | Excellent |
+----------+---------+-----------+
```

7. **Artists Table** - Contains information about artists.

```
mysql> CREATE TABLE Artists(
    -> artist_ID INT AUTO_INCREMENT PRIMARY KEY,
    -> artist_name VARCHAR (255));
Query OK, 0 rows affected (0.04 sec)

mysql> INSERT INTO Artists (artist_name) VALUES ('Michael DiMartino'),('Masashi Kishimoto'),
    -> ('Daniel Radcliffe'),('Robert Downey Jr.'),('Satoshi Tajiri'),('Chris Hemsworth');
Query OK, 6 rows affected (0.01 sec)
Records: 6  Duplicates: 0  Warnings: 0

mysql> SELECT * from Artists;
+-----------+-------------------+
| artist_ID | artist_name       |
+-----------+-------------------+
|         1 | Michael DiMartino |
|         2 | Masashi Kishimoto |
|         3 | Daniel Radcliffe  |
|         4 | Robert Downey Jr. |
|         5 | Satoshi Tajiri    |
|         6 | Chris Hemsworth   |
+-----------+-------------------+
6 rows in set (0.00 sec)
```

8. **Skills Table** - Contains different skills an artist can have.

```
mysql> CREATE TABLE Skills(skill_ID INT AUTO_INCREMENT PRIMARY KEY,
    -> skill_name VARCHAR (255));
Query OK, 0 rows affected (0.03 sec)

mysql> INSERT INTO Skills (skill_name) VALUES ('Actor'),('Author'),('Director'),('Creator');
Query OK, 4 rows affected (0.01 sec)
Records: 4  Duplicates: 0  Warnings: 0

mysql> SELECT * from Skills;
+----------+------------+
| skill_ID | skill_name |
+----------+------------+
|        1 | Actor      |
|        2 | Author     |
|        3 | Director   |
|        4 | Creator    |
+----------+------------+
4 rows in set (0.00 sec)
```

9. **ArtistSkills Table** - Associates artists with their skills

```
mysql> CREATE TABLE ArtistSkills ( artist_ID INT, skill_ID INT,
    -> FOREIGN KEY (artist_ID) REFERENCES Artists(artist_ID),
    -> FOREIGN KEY (skill_ID) REFERENCES Skills (skill_ID));
Query OK, 0 rows affected (0.06 sec)

mysql> INSERT INTO ArtistSkills (artist_ID,skill_ID) VALUES (1,2),(2,4),(3,1),(4,1),(5,2),(6,1);
Query OK, 6 rows affected (0.01 sec)
Records: 6  Duplicates: 0  Warnings: 0

mysql> SELECT * from ArtistSkills;
+-----------+----------+
| artist_ID | skill_ID |
+-----------+----------+
|         1 |        2 |
|         2 |        4 |
|         3 |        1 |
|         4 |        1 |
|         5 |        2 |
|         6 |        1 |
+-----------+----------+
6 rows in set (0.00 sec)
```

10. **Roles Table** - Contains the roles in movies.

```
mysql> CREATE TABLE Roles (
    -> role_id INT AUTO_INCREMENT PRIMARY KEY,
    -> role_name VARCHAR(100));
Query OK, 0 rows affected (0.04 sec)

mysql> INSERT INTO Roles (role_name) VALUES ('Lead Role'),('Voice Actor'),('Superhero);
    '> ^C
mysql> INSERT INTO Roles (role_name) VALUES ('Lead Role'),('Voice Actor'),('Superhero');
Query OK, 3 rows affected (0.01 sec)
Records: 3  Duplicates: 0  Warnings: 0

mysql> SELECT * from Roles;
+---------+-------------+
| role_id | role_name   |
+---------+-------------+
|       1 | Lead Role   |
|       2 | Voice Actor |
|       3 | Superhero   |
+---------+-------------+
3 rows in set (0.00 sec)
```

11. **ArtistRoles Table** - Artist performs multiple roles in movies.

```
mysql> CREATE TABLE ArtistRoles (artist_id INT, movie_id INT, role_id INT,
    -> FOREIGN KEY (artist_id) REFERENCES Artists(artist_id),
    -> FOREIGN KEY (movie_id) REFERENCES Movies(movie_id),
    -> FOREIGN KEY (role_id) REFERENCES Roles(role_id));
Query OK, 0 rows affected (0.07 sec)

mysql> INSERT INTO ArtistRoles (artist_id, movie_id, role_id) VALUES
    -> (1, 1, 2),(2, 2, 2),(3, 3, 1),(4, 4, 1),(4, 4, 3),(5, 5, 2),(6, 6, 1),(6, 6, 3);
Query OK, 8 rows affected (0.01 sec)
Records: 8  Duplicates: 0  Warnings: 0

mysql> SELECT * from ArtistRoles;
+-----------+----------+---------+
| artist_id | movie_id | role_id |
+-----------+----------+---------+
|         1 |        1 |       2 |
|         2 |        2 |       2 |
|         3 |        3 |       1 |
|         4 |        4 |       1 |
|         4 |        4 |       3 |
|         5 |        5 |       2 |
|         6 |        6 |       1 |
|         6 |        6 |       3 |
+-----------+----------+---------+
8 rows in set (0.00 sec)
```

PROBLEM 1: Movie should have multiple media(Video or Image)

```
mysql> Select Movies.movie_ID, Movies.movie_name, Media.media_type
    -> FROM Movies
    -> INNER JOIN Media ON Media.movie_ID = Movies.movie_ID;
+----------+---------------------------+------------+
| movie_ID | movie_name                | media_type |
+----------+---------------------------+------------+
|        1 | Avatar The Last Airbender | Video      |
|        2 | Naruto                    | image      |
|        3 | Harry Potter              | image      |
|        4 | Iron Man                  | video      |
|        5 | Pokemon                   | video      |
|        6 | Thor Ragnarok             | video      |
+----------+---------------------------+------------+
6 rows in set (0.00 sec)
```

PROBLEM 2: Movie can belongs to multiple Genre

```
mysql> SELECT Movies.movie_ID, Movies.movie_name, Genre.genre_name
    -> FROM Movies
    -> INNER JOIN MovieGenre ON Movies.movie_ID = MovieGenre.movie_ID
    -> INNER JOIN Genre ON MovieGenre.genre_ID = Genre.genre_ID;
+----------+---------------------------+-------------------+
| movie_ID | movie_name                | genre_name        |
+----------+---------------------------+-------------------+
|        1 | Avatar The Last Airbender | Action            |
|        1 | Avatar The Last Airbender | Adventure         |
|        1 | Avatar The Last Airbender | Fantasy           |
|        2 | Naruto                    | Fantasy           |
|        2 | Naruto                    | comic             |
|        3 | Harry Potter              | Fantasy Literature |
|        4 | Iron Man                  | Superhero fiction |
|        5 | Pokemon                   | Adventure         |
|        5 | Pokemon                   | Animation         |
|        6 | Thor Ragnarok             | Superhero fiction |
+----------+---------------------------+-------------------+
10 rows in set (0.00 sec)
```

PROBLEM 3: Movie can have multiple reviews and Review can belongs to a user

```
mysql> SELECT User.user_name, Movies.movie_ID, Movies.movie_name, MovieReview.review
    -> FROM Movies
    -> INNER JOIN MovieReview ON MovieReview.movie_ID = Movies.movie_ID
    -> INNER JOIN User ON User.user_ID = MovieReview.user_ID;
+-----------+----------+---------------------------+-----------+
| user_name | movie_ID | movie_name                | review    |
+-----------+----------+---------------------------+-----------+
| Sam       |        4 | Iron Man                  | Very Good |
| Sam       |        6 | Thor Ragnarok             | Good      |
| Ash       |        5 | Pokemon                   | Excellent |
| Serena    |        5 | Pokemon                   | Very Good |
| Harry     |        3 | Harry Potter              | Average   |
| Shree     |        2 | Naruto                    | Good      |
| Shree     |        1 | Avatar The Last Airbender | Excellent |
+-----------+----------+---------------------------+-----------+
7 rows in set (0.00 sec)
```

PROBLEM 4: Artist can have multiple skills

```
mysql> Select Artists.artist_name, ArtistSkills.skill_ID, Skills.skill_name
    -> FROM Artists
    -> INNER JOIN ArtistSkills ON ArtistSkills.artist_ID = Artists.artist_ID
    -> INNER JOIN Skills ON Skills.skill_ID = ArtistSkills.skill_ID;
+------------------+----------+------------+
| artist_name      | skill_ID | skill_name |
+------------------+----------+------------+
| Daniel Radcliffe |        1 | Actor      |
| Robert Downey Jr. |        1 | Actor      |
| Chris Hemsworth  |        1 | Actor      |
| Michael DiMartino |        2 | Author     |
| Satoshi Tajiri   |        2 | Author     |
| Masashi Kishimoto |        4 | Creator    |
+------------------+----------+------------+
6 rows in set (0.01 sec)
```

PROBLEM 5: Artist can perform multiple role in a single film

```
mysql> SELECT Artists.artist_name, Movies.movie_name, Roles.role_name
    -> FROM ArtistRoles
    -> INNER JOIN Artists ON ArtistRoles.artist_ID = Artists.artist_ID
    -> INNER JOIN Roles ON ArtistRoles.role_ID = Roles.role_ID
    -> INNER JOIN Movies ON ArtistRoles.movie_ID = Movies.movie_ID;
+------------------+---------------------------+-------------+
| artist_name      | movie_name                | role_name   |
+------------------+---------------------------+-------------+
| Daniel Radcliffe | Harry Potter              | Lead Role   |
| Robert Downey Jr. | Iron Man                  | Lead Role   |
| Chris Hemsworth  | Thor Ragnarok             | Lead Role   |
| Michael DiMartino | Avatar The Last Airbender | Voice Actor |
| Masashi Kishimoto | Naruto                    | Voice Actor |
| Satoshi Tajiri   | Pokemon                   | Voice Actor |
| Robert Downey Jr. | Iron Man                  | Superhero   |
| Chris Hemsworth  | Thor Ragnarok             | Superhero   |
+------------------+---------------------------+-------------+
8 rows in set (0.00 sec)
```