**SVKM’s NMIMS**

**School of Technology Management & Engineering, Chandigarh**

A.Y. 2023 - 24

**Course: Database Management Systems**

**Project Report**

|  |  |  |
| --- | --- | --- |
| Program | MBA tech Computer Engineering | |
| Semester | 4 | |
| Name of the Project: | OTT Management System | |
|  | | |
| Details of Project Members |  |  |
| Batch 1 | Roll No. A181/A173/A174 | Name- Aryan Gaikwad, Archit Wadekar,  Utkarsh Wadekar |
| B2 |  |  |
| Date of Submission: 04-04-2024 | | |

**Contribution of each project Members:**

|  |  |  |
| --- | --- | --- |
| Roll No. | Name: | Contribution |
| A181 | Aryan Gaikwad |  |
| A174 | Utkarsh Agarwal |  |
| A173 | Archit Wadekar |  |

**Github link of your project:**

**Note:**

1. Create a readme file if you have multiple files
2. All files must be properly named (Example:R004\_DBMSProject)
3. Submit all relevant files of your work ( Report, all SQL files, Any other files)
4. **Plagiarism is highly discouraged (Your report will be checked for plagiarism)**

**Rubrics for the Project evaluation:**

|  |  |
| --- | --- |
| First phase of evaluation:  Innovative Ideas (5 Marks)  Design and Partial implementation (5 Marks) | 10 marks |
| Final phase of evaluation  Implementation, presentation and viva, Self-Learning and Learning Beyond classroom | 10 marks |

**Table of Contents**

|  |  |  |
| --- | --- | --- |
| **Sr no.** | **Topic** | **Page no.** |
| **1** | Storyline |  |
| **2** | Components of Database Design |  |
| **3** | Entity Relationship Diagram |  |
| **4** | Relational Model |  |
| **5** | Normalization |  |
| **6** | SQL Queries |  |
| **7** | Learning from the Project |  |
| **8** | Project Demonstration |  |
| **9** | Self-learning beyond classroom |  |
| **10** | Learning from the project |  |
| **8** | Challenges faced |  |
| **9** | Conclusion |  |

1. **Storyline:**

**Problem Statement:**

In today's fast-paced world, the demand for digital entertainment is soaring. With the proliferation of OTT platforms offering a plethora of content choices, users often find themselves overwhelmed by fragmented experiences, inconsistent content quality, and subscription fatigue. Content creators, on the other hand, struggle to effectively distribute their work, reach their target audience, and monetize their creations.

**Synopsis:**

Amidst this backdrop, Streamify emerges as a revolutionary OTT management system aimed at addressing the pain points faced by users, content creators, and administrators alike. Streamify leverages state-of-the-art technology and innovative features to streamline the delivery, discovery, and management of digital content, transforming the entertainment landscape for the better.

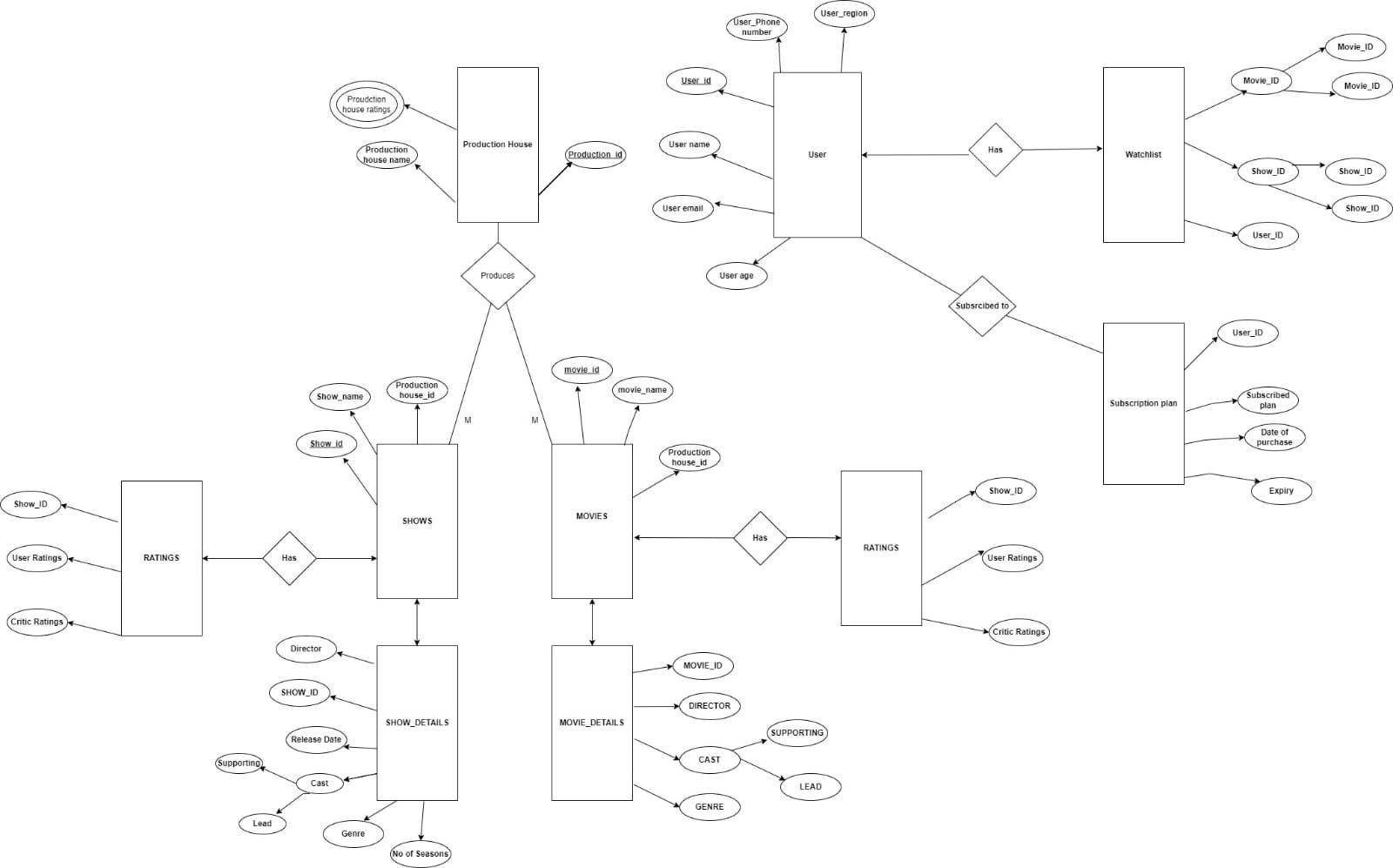
**II. Components of Database Design**

**(entities, attributes, relationships and cardinality)**

**Components:**

* Shows (show\_id, show name, production\_house\_id)
* Movies (movie\_id, movie name, production\_house\_id)
* Show details (sdirector, show\_id, Release Date, s\_supporting cast, s\_lead cast, s\_genre, No of Seasons)
* MOVIE\_DETAILS (movie\_id, m\_director, m\_supporting cast, m\_lead cast, m\_genre)
* Show Ratings (Show\_ID, sUser Ratings (multivalued), sCritic Ratings (multivalued))
* Movie Ratings (Movie\_id, mUser Ratings (multivalued), mCritic Ratings (multivalued))
* Production House (Production house ratings (multivalued), Production\_id, Production house name)
* User (User\_id, User\_Phone number, User\_region, User name, User email, User age)
* User\_Showlist (User\_ID, Show\_ID)
* User\_Movielist (User\_ID, Movie\_ID)
* Subscription plan (User\_ID, Subscribed plan, Date of purchase, Expiry)

**III. Entity Relationship Diagram**



**IV. Relational Model**

Shows (show\_id PK, show\_name, production\_house\_id FK)

Movies (movie\_id PK, movie\_name, production\_house\_id FK)

Show\_details (show\_id PK, sdirector, Release\_Date, s\_supporting\_cast, s\_lead\_cast, s\_genre, No\_of\_Seasons, FOREIGN KEY (show\_id) REFERENCES Shows(show\_id))

MOVIE\_DETAILS (movie\_id PK, m\_director, m\_supporting\_cast, m\_lead\_cast, m\_genre, FOREIGN KEY (movie\_id) REFERENCES Movies(movie\_id))

Production\_House (Production\_id PK, Production\_house\_name, Production\_house\_ratings)

Show\_Ratings (Show\_ID PK, sUser\_Ratings, sCritic\_Ratings)

Movie\_Ratings (Movie\_id PK, mUser\_Ratings, mCritic\_Ratings)

User (User\_id PK, User\_Phone\_number, User\_region, User\_name, User\_email, User\_age)

Watchlist (Movie\_ID PK, User\_ID PK, FOREIGN KEY (Movie\_ID) REFERENCES Movies(movie\_id), FOREIGN KEY (User\_ID) REFERENCES User(User\_id))

Subscription\_plan (User\_ID PK, Subscribed\_plan, Date\_of\_purchase, Expiry, FOREIGN KEY (User\_ID) REFERENCES User(User\_id))

User\_Plan (User\_ID PK, Subscribed\_plan, Date\_of\_purchase, Expiry, FOREIGN KEY (User\_ID) REFERENCES User(User\_id))

User\_Showlist (User\_ID PK, Show\_ID PK, FOREIGN KEY (User\_ID) REFERENCES User(User\_id), FOREIGN KEY (Show\_ID) REFERENCES Shows(show\_id))

User\_Movielist (User\_ID PK, Movie\_ID PK, FOREIGN KEY (User\_ID) REFERENCES User(User\_id), FOREIGN KEY (Movie\_ID) REFERENCES Movies(movie\_id))

**V. Normalization**

Perform normalization (1NF, 2NF, 3NF, BCNF) as applicable for the entire database.

**Original tables:-**

Shows- (show\_id(primary key), show name, production house\_id)

Movies- (movie\_id(primary key),movie name,production house\_id)

show details- (sdirector, SHOW\_ID, Release Date,S\_supporting cast, s\_lead cast,s\_Genre,No of Seasons)

MOVIE\_DETAILS- (MOVIE\_ID,MDIRECTOR,m\_SUPPORTING CAST, m\_LEAD CAST,m\_GENRE)

 Show Ratings- (Show\_ID, sUser Ratings(multivalued),sCritic Ratings (multivalued))

Movie Ratings- (Movie\_id, mUser Ratings (multivalued),mCritic Ratings (multivalued))

Production House- (Proudction house ratings (multivalued) , Production\_id(primary key),Production house name)

User- (User\_id(primary key), User\_Phone number,User\_region,User name,User email,User age)

Watchlist-( Movie\_ID(primary key),Show\_ID,User\_ID)

Subscription plan- (User\_ID,Subscribed plan,Date of purchase,Expiry)

**Normalized version:**

To normalize the given set of tables, we need to ensure that they adhere to the principles of database normalization, specifically aiming for at least the third normal form (3NF). Here's a normalized version of the tables:

1. Shows

- show\_id (primary key)

- show\_name

- production\_house\_id (foreign key referencing Production\_House)

2. Movies

- movie\_id (primary key)

- movie\_name

- production\_house\_id (foreign key referencing Production\_House)

3. Show\_Details

- show\_id (primary key, foreign key referencing Shows)

- s\_director

- release\_date

- s\_supporting\_cast

- s\_lead\_cast

- s\_genre

- no\_of\_seasons

4. Movie\_Details

- movie\_id (primary key, foreign key referencing Movies)

- m\_director

- m\_supporting\_cast

- m\_lead\_cast

- m\_genre

5. Show\_Ratings

- show\_id (primary key, foreign key referencing Shows)

- user\_id (primary key, foreign key referencing User)

- user\_rating (multivalued)

- critic\_rating (multivalued)

6. Movie\_Ratings

- movie\_id (primary key, foreign key referencing Movies)

- user\_id (primary key, foreign key referencing User)

- user\_rating (multivalued)

- critic\_rating (multivalued)

7. Production\_House

- production\_id (primary key)

- production\_house\_name

- production\_house\_ratings (multivalued)

8. User

- user\_id (primary key)

- user\_phone\_number

- user\_region

- user\_name

- user\_email

- user\_age

9. User\_Showlist

-user\_id(primary key)

-show\_id(foreign key referencing Shows)

10.User\_Movielist

-user\_id(primary key)

- movie\_id(foreign key referencing Movies)

11. Subscription\_Plan

- user\_id (primary key, foreign key referencing User)

- subscribed\_plan

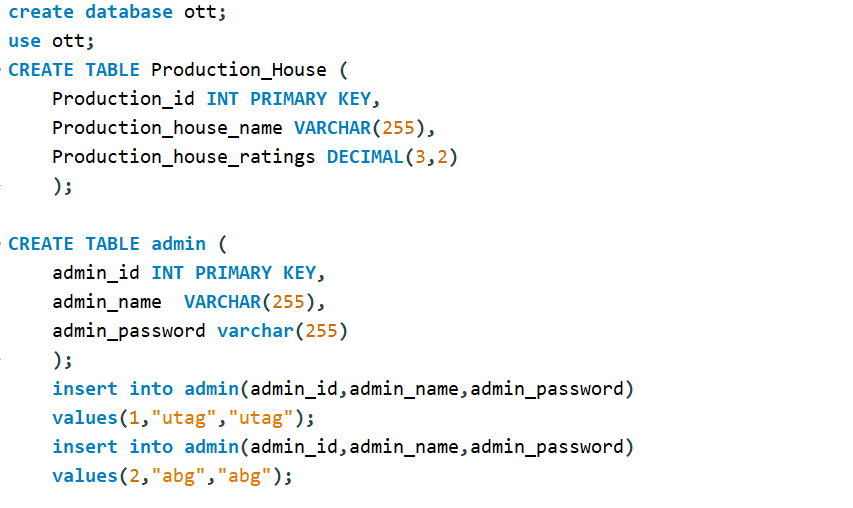
- date\_of\_purchase

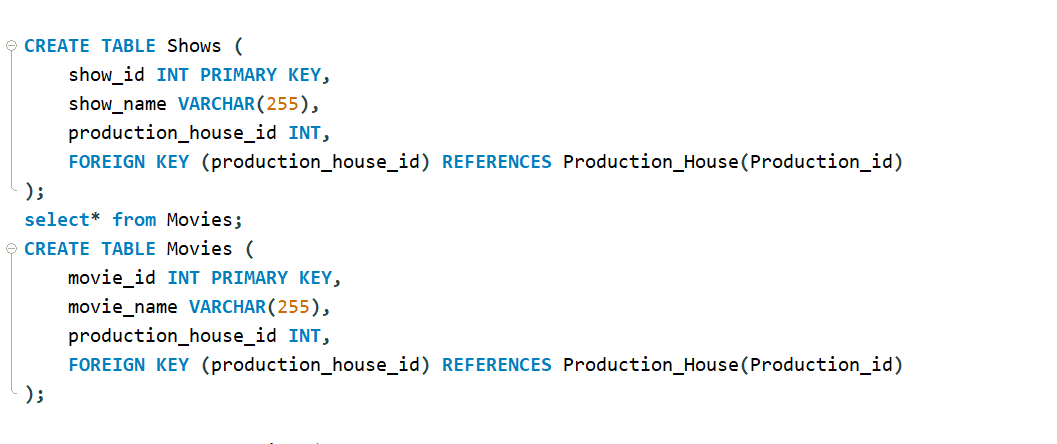
- expiry\_date

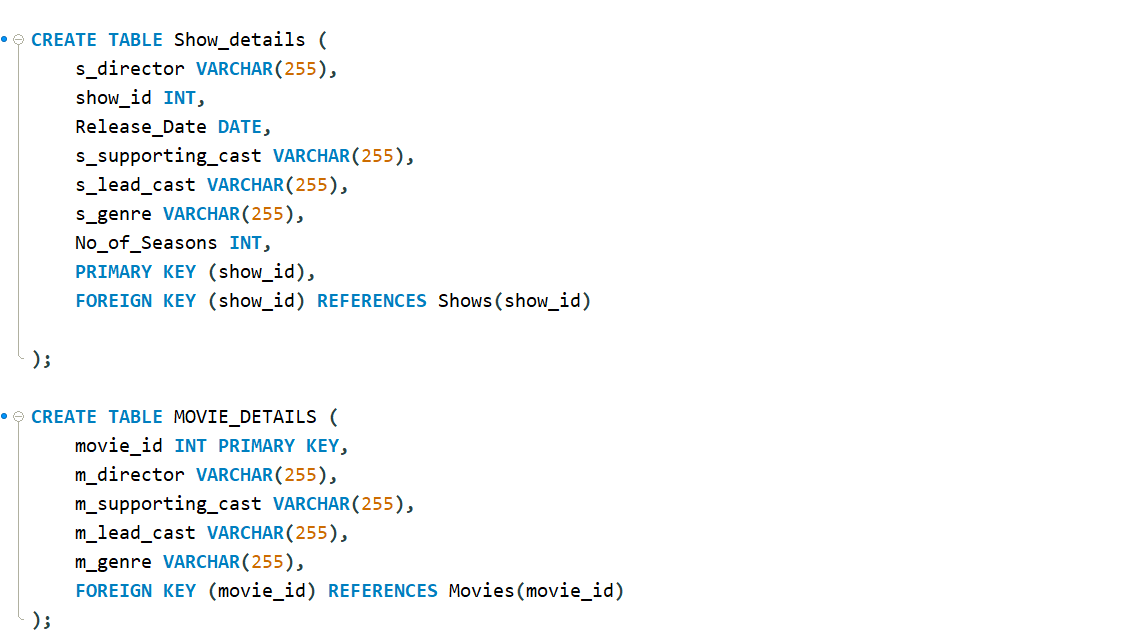
This normalization splits the tables into smaller, more manageable units, reducing redundancy and improving data integrity. Each table serves a specific purpose and relates to others through foreign key constraints. Additionally, multivalued attributes have been separated into separate tables to adhere to 3NF.

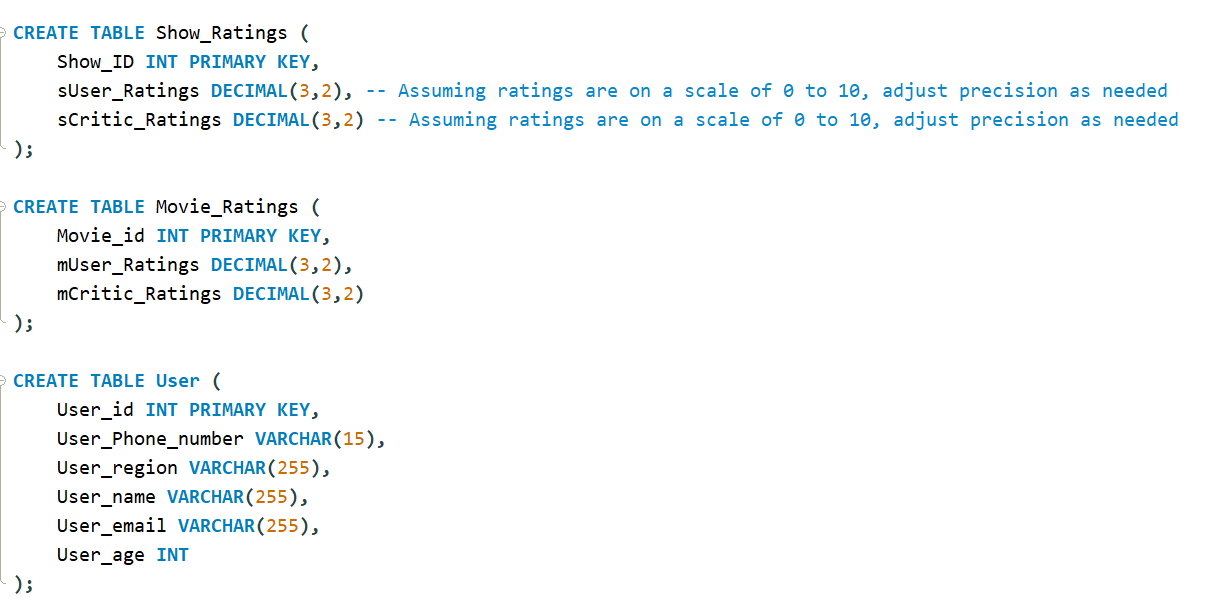
**VI. SQL Queries**

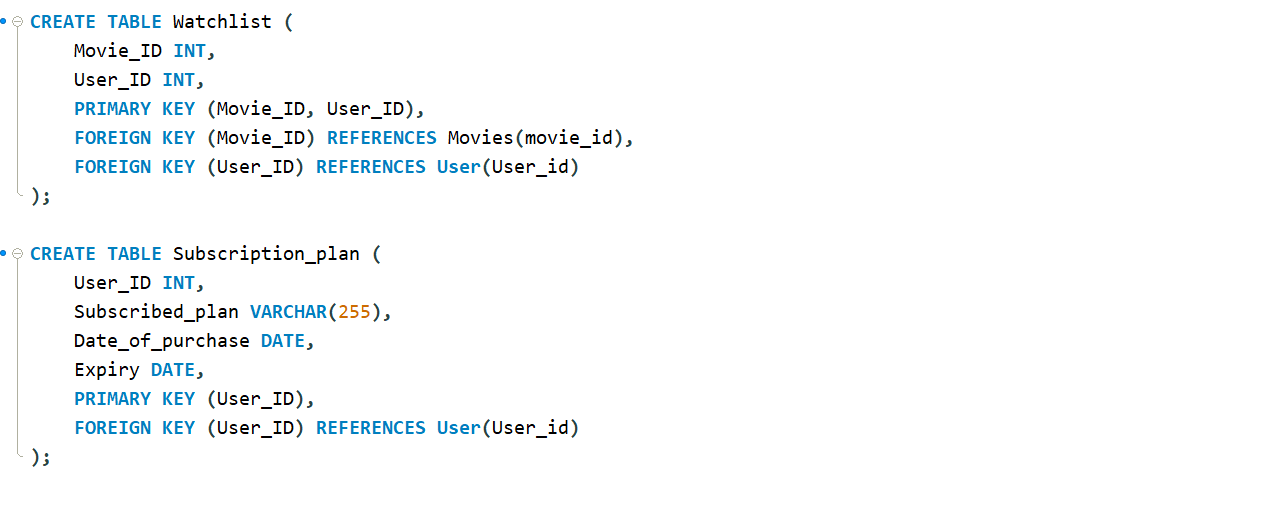
**Creating the tables and Populating the tables: sql queries to create tables and insert values**

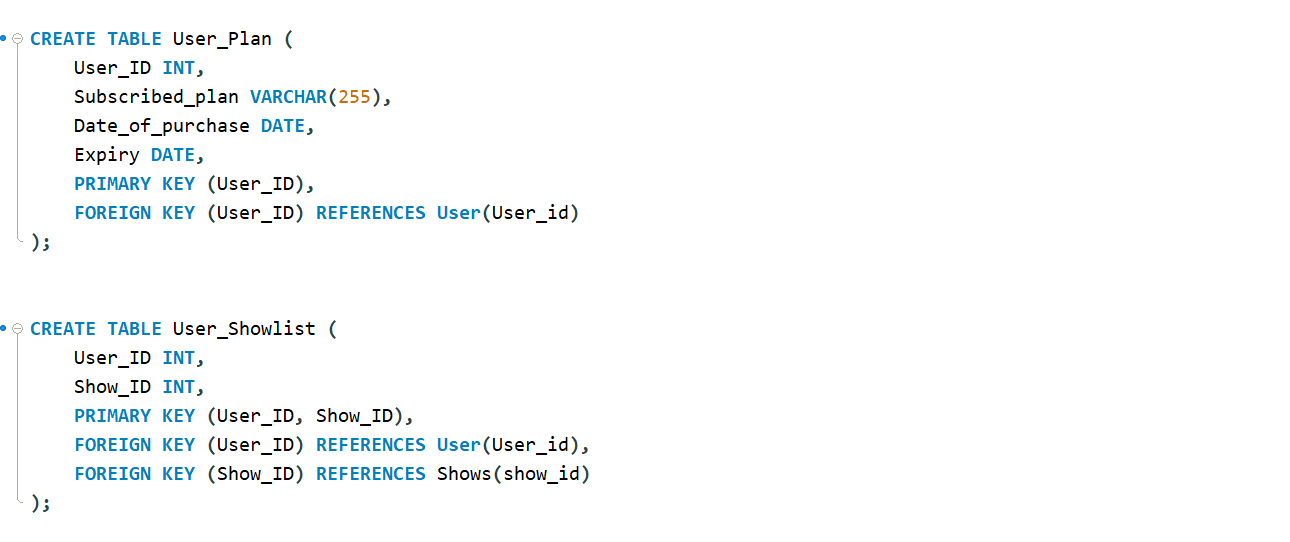


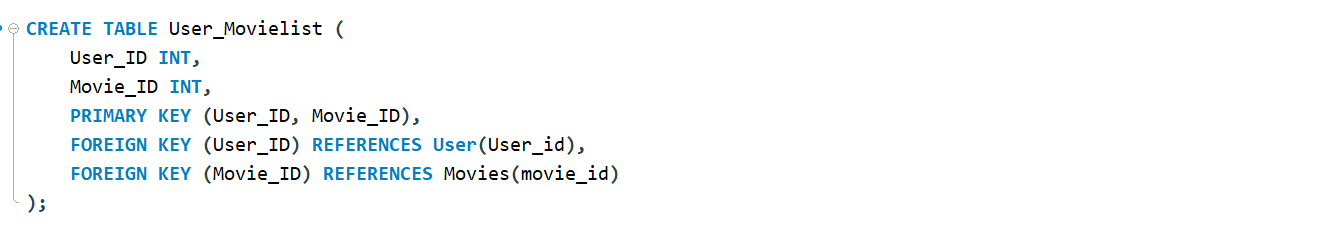


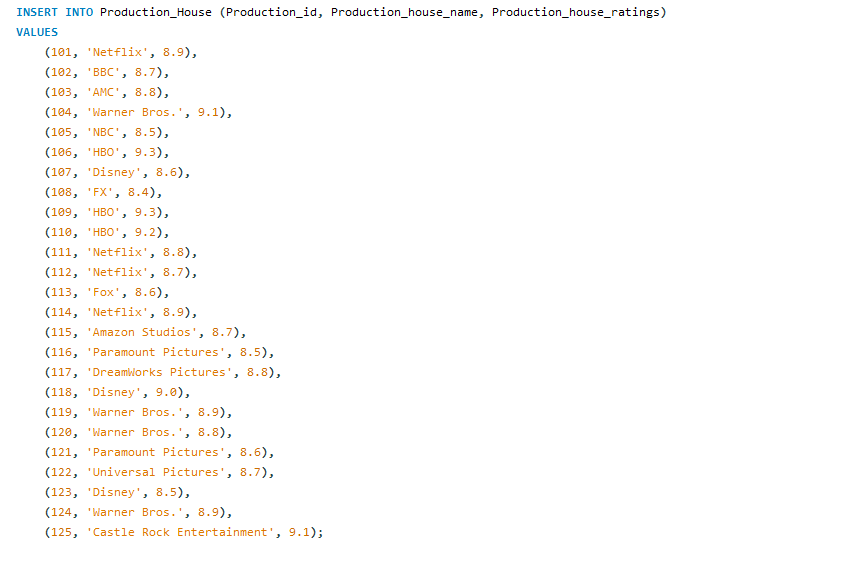


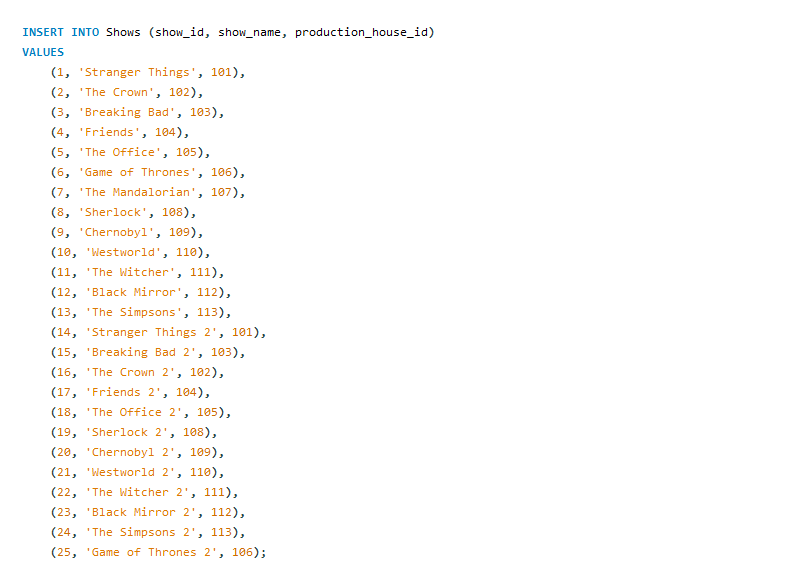


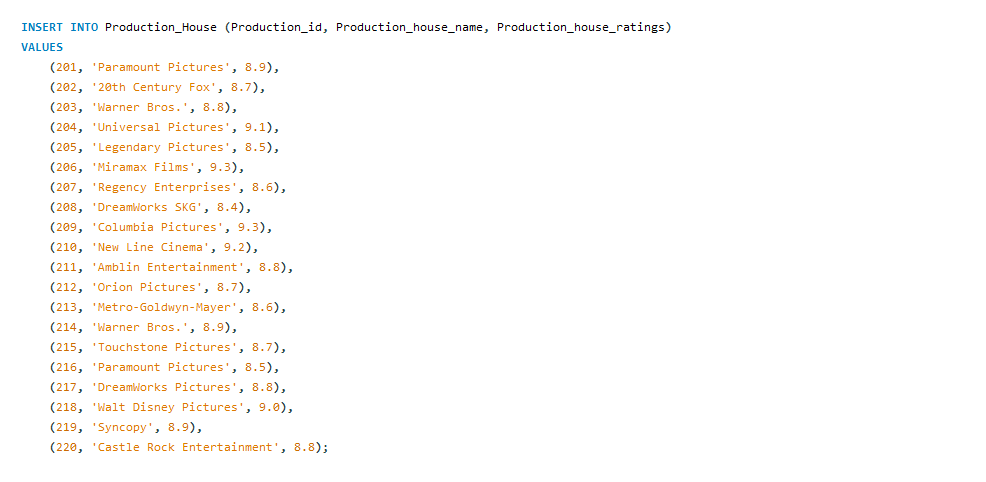


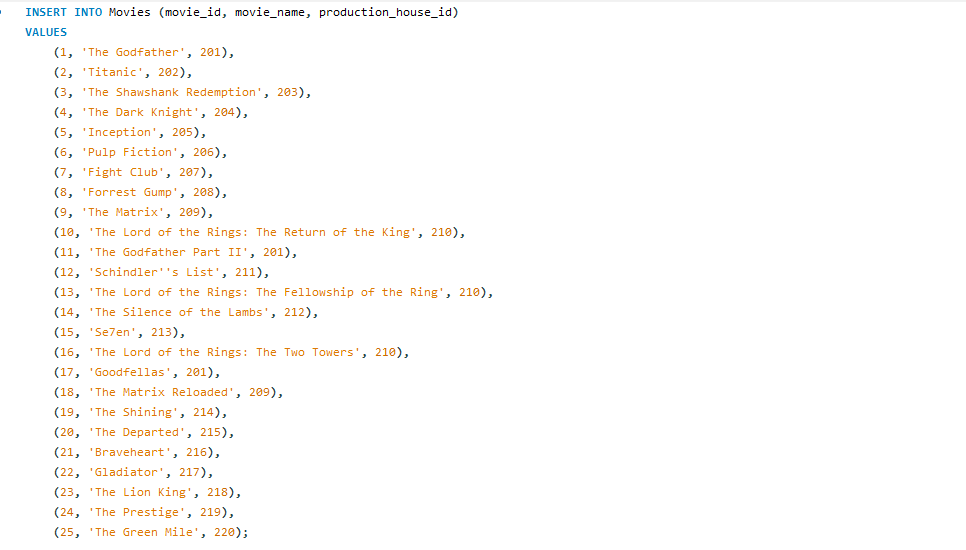




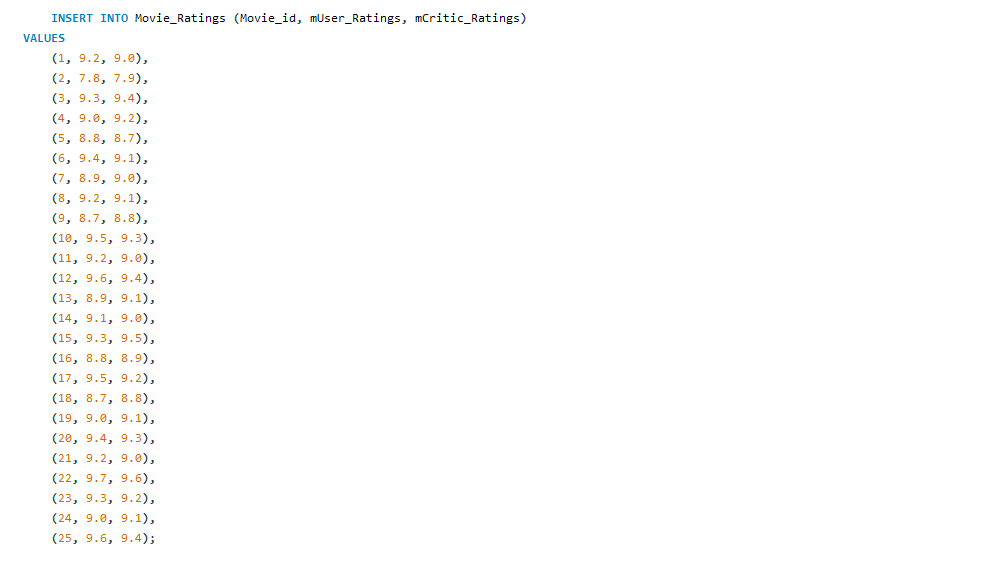


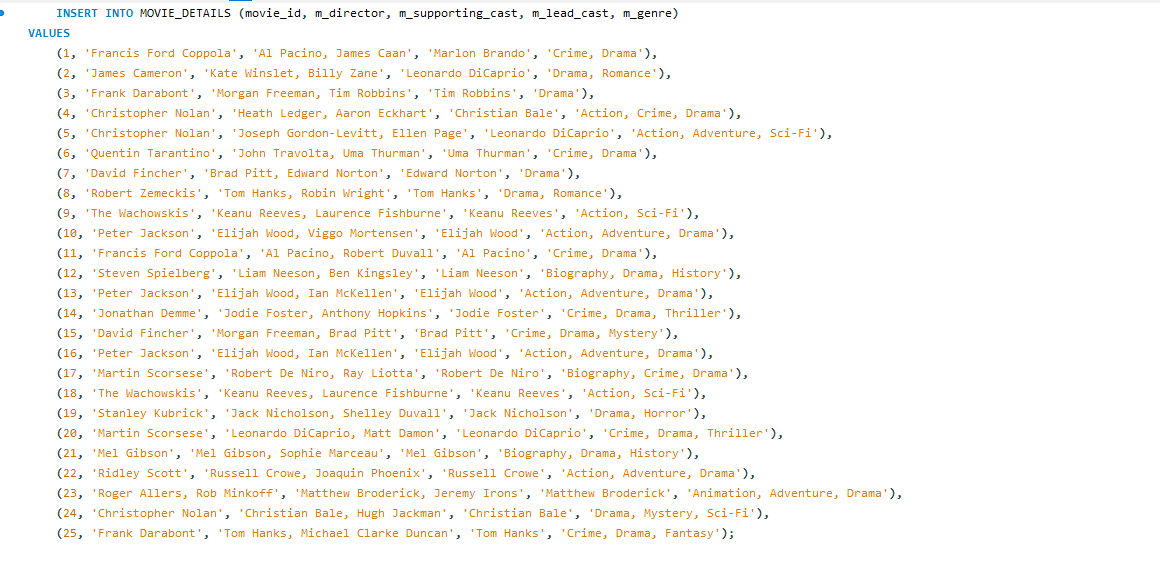


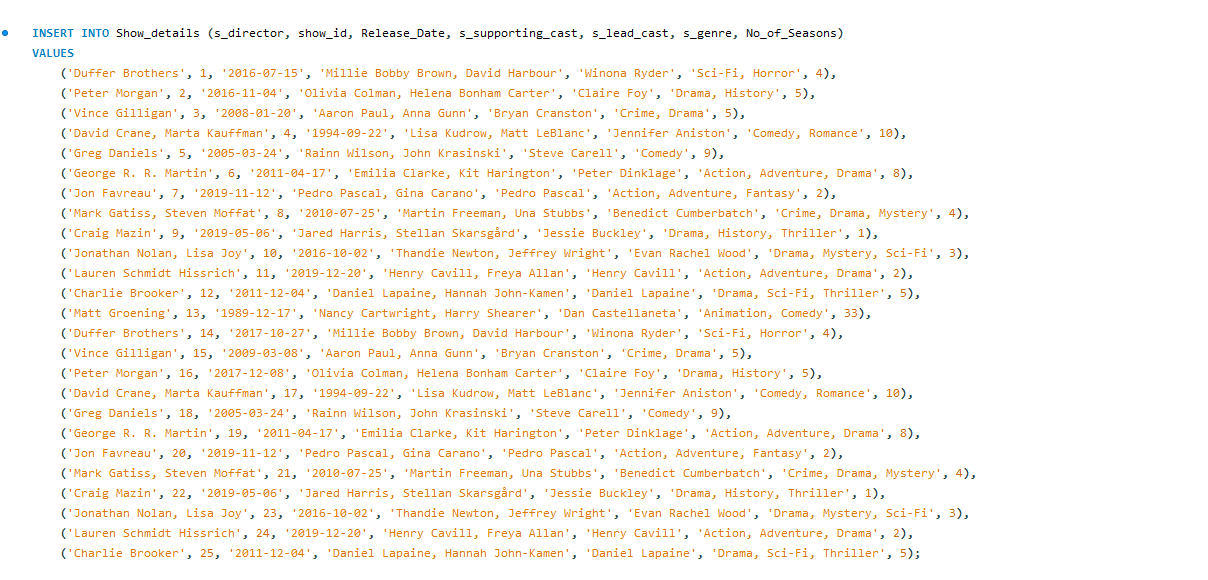


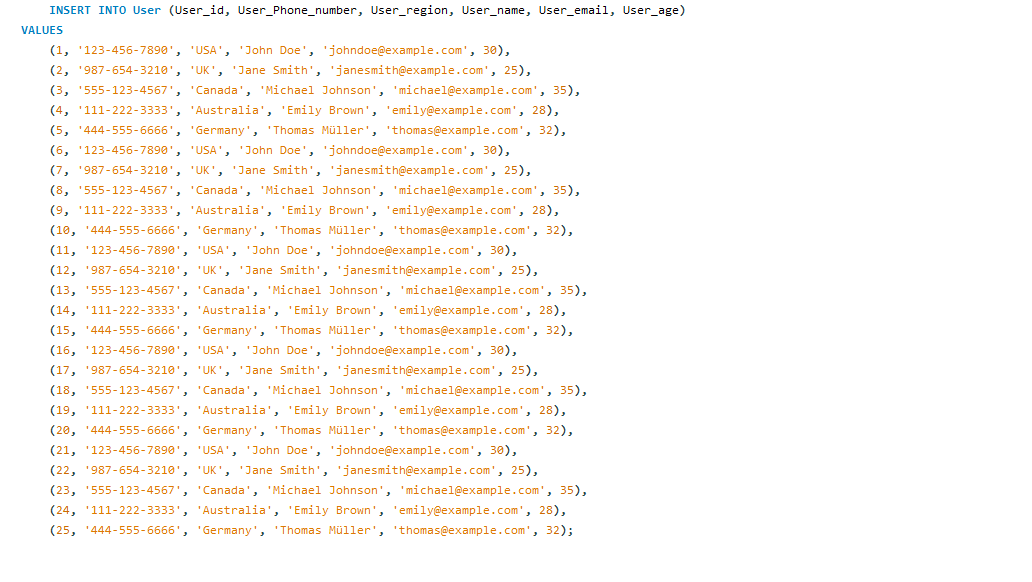






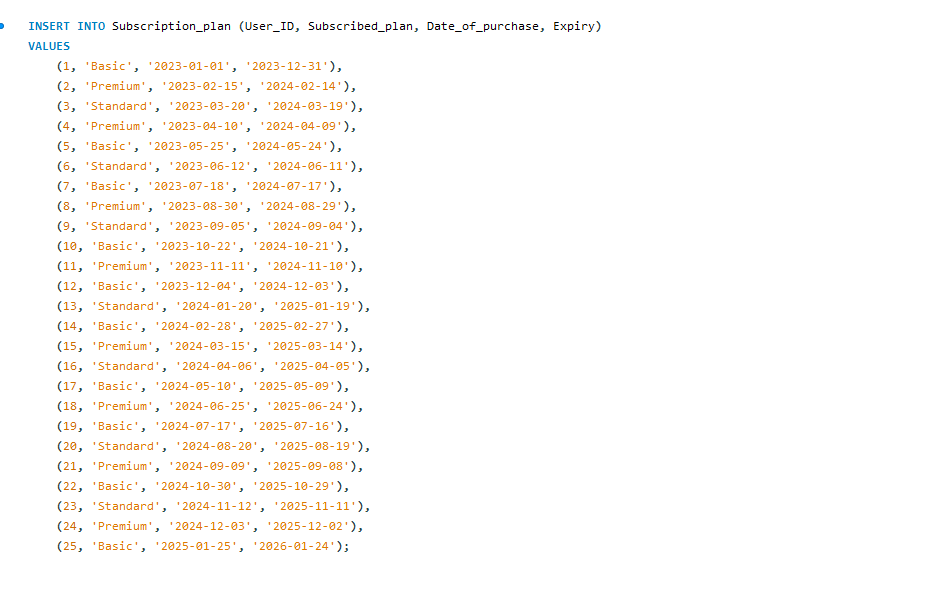






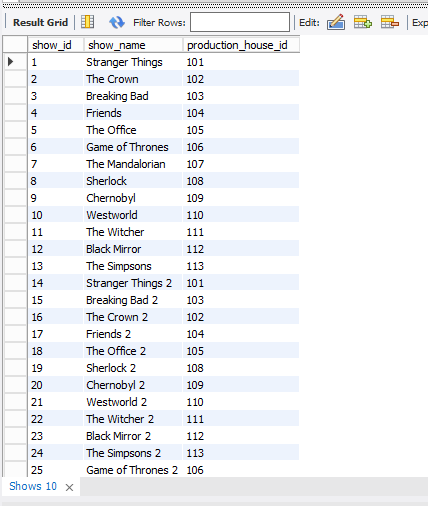
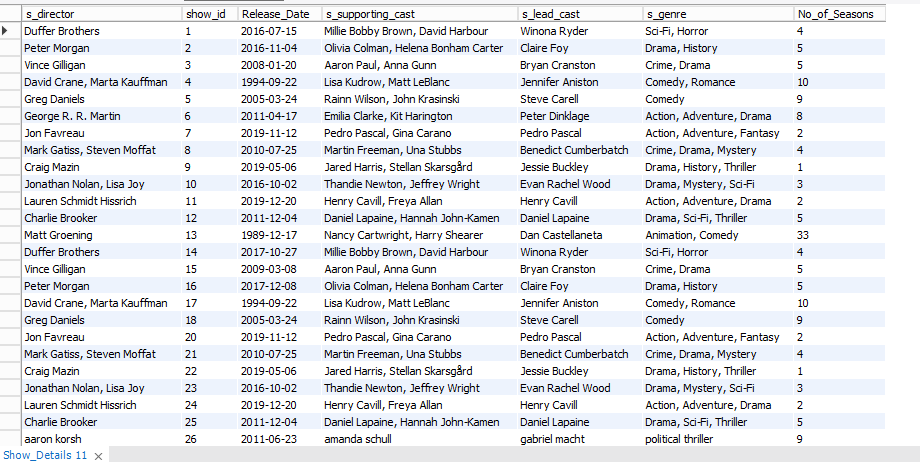




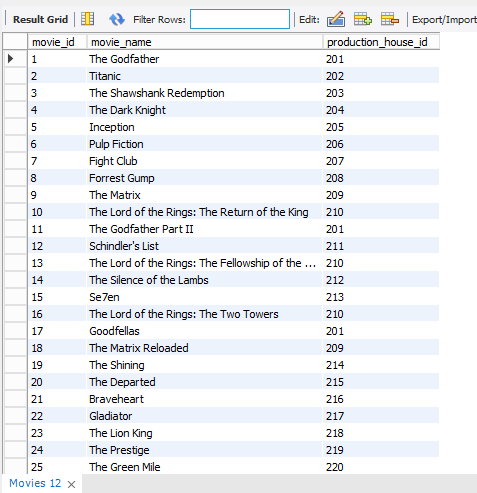
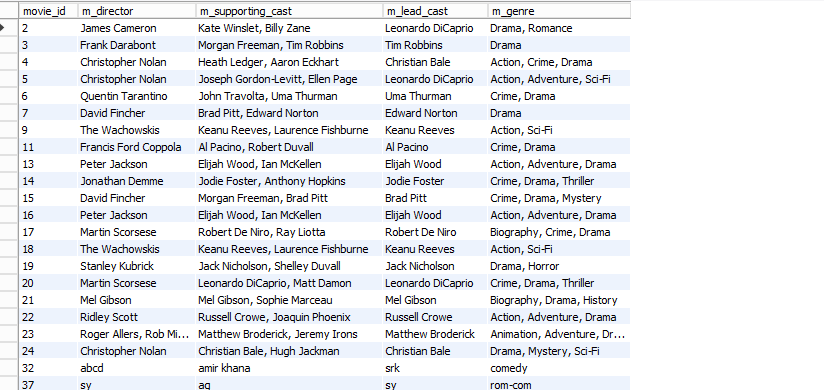


**Output of Tables(Screenshot of table outputs)**

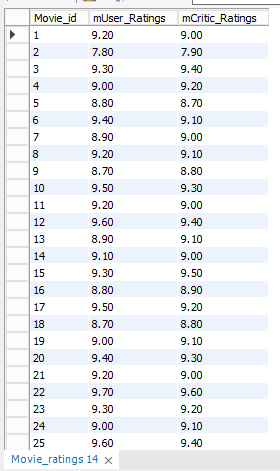
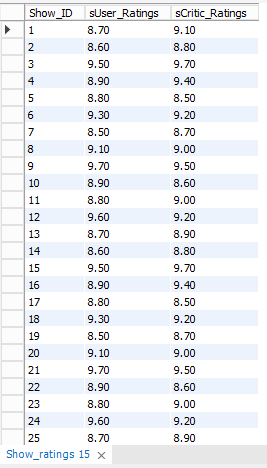
**Shows Show details**

**** ****

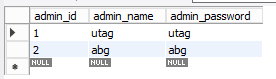
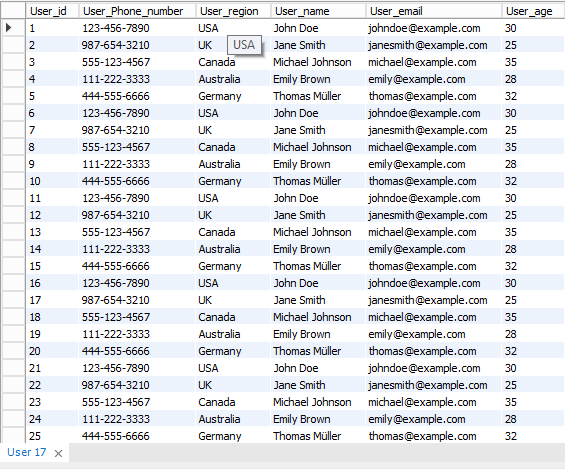
**Movies Movie\_details**

**** ****

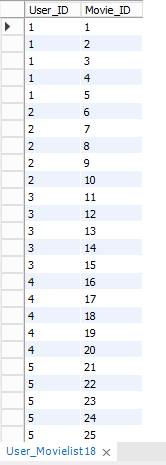
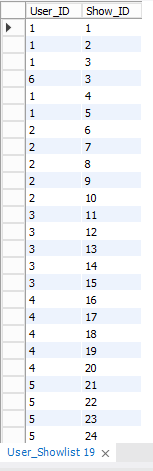
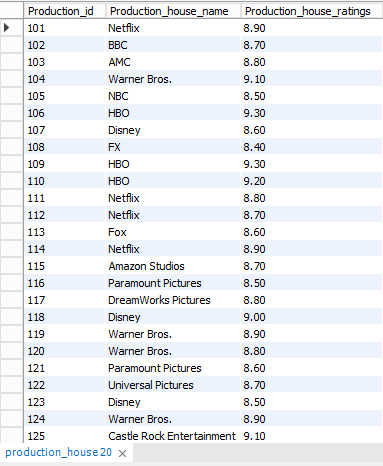
**Movie ratings Show ratings**

**** ****

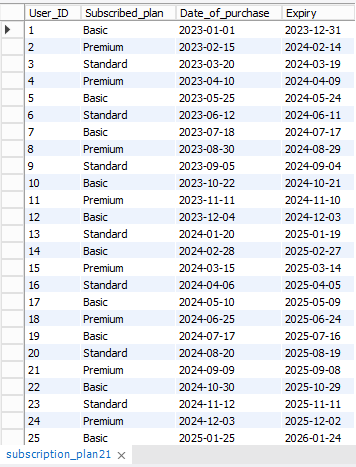
Admin Users

**** ****

User movie\_list user showlist production house

**** **** 

Subscription plan



**SQL Queries:minimum 20 sql queries and outputs:**

**VII. Project demonstration**

* I have used MySQL for this Project.

**VIII. Self -Learning beyond classroom**

By working on this project, we had the opportunity to delve deeper into the intricacies of database management, query optimization, and system architecture. Additionally, exploring industry best practices, staying updated on emerging technologies, and participating in online forums or communities enable individuals to broaden their understanding and stay abreast of the latest trends in OTT management and database technologies. This self-directed learning journey fosters a culture of continuous improvement, innovation, and adaptability, equipping individuals with the expertise and resilience needed to thrive in the dynamic and competitive landscape of OTT management and digital entertainment.

**IX. Learning from the Project**

The OTT management project using an SQL database offers several valuable learnings:

Database Design and Management: Through the creation of SQL tables and the establishment of relationships between them, learners gain a deep understanding of database design principles. They learn to organize data efficiently, ensuring data integrity and optimizing database performance.

Query Optimization: Crafting SQL queries to retrieve, manipulate, and analyze data teaches learners the importance of query optimization. They discover techniques to improve query efficiency, such as using indexes, reducing unnecessary joins, and optimizing database schema.

Data Modeling: Designing tables and defining relationships between them helps learners develop skills in data modeling. They understand how to represent real-world entities and their relationships in a relational database, facilitating effective data management and analysis.

**X. Challenges Faced**

Complexity of Queries: Crafting complex SQL queries to retrieve specific data or perform advanced operations can be challenging, especially for beginners. Understanding the syntax and logic of SQL commands, such as joins, subqueries, and aggregate functions, requires thorough practice and familiarity with the language.

Data Integrity: Ensuring data integrity within the database is crucial but can be challenging, particularly when dealing with large datasets or frequent data modifications. Maintaining consistency, accuracy, and completeness of data across tables while performing transactions and updates is essential but requires careful planning and validation mechanisms.

Error Handling and Debugging: Debugging errors or troubleshooting issues encountered during query execution or database operations is very time-consuming and challenging.

**XI. Conclusion**

The OTT management project using an SQL database provides a comprehensive learning experience that combines theoretical knowledge with practical application, fostering skills essential for database management and professional growth. By engaging in database design, query optimization, and collaborative problem-solving, we developed proficiency in SQL database management and cultivated a mindset of continuous improvement and innovation. Overall, the project served as a springboard for us to navigate the dynamic landscape of database management and pursue opportunities for lifelong learning and career advancement.