

**Movie Booking System**

**Submitted By**

Shaurya Jain 102203594

Abhishek Mittal 102203693

Arshdeep Aswal 102203747

Vishal 102203987

**Submitted To**

Dr. Shashank Singh

Contents

Introduction 3

1.1 Overview 3

1.2 Objective 3

1.3 Importance of Movie Booking System 4

1.4 Requirement Analysis 5

System Design

2.1 Database Schema 6-7

2.2 Entity-Relationship (ER) diagram 8-9

2.3 Entity-Relationship (ER) diagram to Table 10

2.4 Normalization 11

SQL & PL/SQL

3.1 SQL statements for table creation 12-13

3.2 SQL statements for data insertion 14

3.3 PL/SQL statements 15-20

Conclusion 22

**INTRODUCTION**

In the era of digitalization, the movie industry has embraced technological advancements to enhance the customer experience and streamline operations. One such innovation is the implementation of online movie booking systems, which have revolutionized the way audiences plan and purchase their movie tickets.

For moviegoers, the system offers a seamless online booking experience, allowing them to browse movie listings, view showtimes, select seats, and purchase tickets conveniently. Additionally, it provides features for managing personal accounts, viewing booking history, and receiving notifications about new releases or promotional offers.

On the other hand, the system empowers theater administrators with a powerful management interface. They can effortlessly maintain movie schedules, update theater and auditorium details, monitor ticket sales, and generate insightful

* 1. **Overview**

The movie booking and management system is a comprehensive solution designed to cater to the needs of both moviegoers and theater administrators. It leverages the power of SQL and PL/SQL to provide a robust and user-friendly platform that enhances the movie-going experience while enabling efficient theater operations.

* 1. **Objective**

Our goal in developing the Movie Booking System is to create a seamless platform for booking movie tickets easily. We aim to provide customers a user-friendly interface to find movies, choose showtimes, and select seats confidently. Simultaneously, the system offers theater administrators robust tools to manage schedules, monitor bookings, and oversee ticketing. With comprehensive data and reporting, administrators can optimize revenue and enhance the overall theater experience.

* 1. **Importance of Movie Booking System**

The movie booking and management system holds significant importance in today's digital age, offering numerous benefits to both customers and theater administrators. It enhances the overall movie-going experience for customers by providing a convenient and user-friendly platform for browsing movies, selecting showtimes, and booking tickets online or through mobile devices. Additionally, the system streamlines theater operations by automating and centralizing movie scheduling, seat allocation, pricing strategies, and sales monitoring, reducing manual effort and increasing efficiency.

Furthermore, the system offers real-time data and insights into customer preferences, sales trends, and theater performance through comprehensive reporting and analytics capabilities, enabling data-driven decision-making for theater administrators. It also facilitates effective inventory management by tracking seat availability and occupancy rates across different auditoriums and movies, optimizing resource utilization and maximizing revenue potential.

Notably, the system reduces operational costs by minimizing manual processes, eliminating errors, and optimizing resource allocation, leading to increased profitability for theater businesses. Finally, it provides a competitive edge in the market by offering a modern and technologically advanced platform that meets the evolving expectations of customers and adapts to changing industry trends.

* 1. **Requirement Analysis**

**Functional Requirements**

**For Moviegoers**

1. **User Registration and Authentication:**  
   Allow users to create a new account with personal details. Enable secure login and logout functionality.
2. **Movie Browsing and Search:**  
   Allow searching for movies by title, genre, language, etc. Display movie details such as synopsis, cast, ratings, and reviews.
3. **Showtime and Seat Selection:**  
   Display available showtimes for selected movies. Enable selection of preferred date, time, and theater location.Indicate seat availability and pricing.
4. **Ticket Booking:**

Users should be able to view available shows and book tickets for them and should be able to select the number of seats they want to book for a particular show.Once booked, the system should update the available seats for the respective show.

1. **Feedback Management:**

Users should be able to leave feedback for movies they have watched. The system should store feedback provided by users along with the associated movie and user details.

**System Design**

**2.1 Database Schema**

The schema aims to efficiently manage various aspects of movie bookings, including customer information, movie details, theater locations, show timings, feedback, and actor profiles.

**1. tbl\_customer:**

* Attributes: customer\_id, customer\_name, phone\_number, dob, email, username, password
* Purpose: This table stores essential customer information necessary for booking tickets and managing accounts. Each customer is uniquely identified by customer\_id, ensuring data integrity and security.

**2. tbl\_theatre:**

* Attributes: theatre\_id, theatre\_name, address
* Purpose: Contains details about theaters where movies are screened. The theatre\_id serves as a primary key, facilitating efficient retrieval of theater information for scheduling and ticket booking purposes.

**3. tbl\_movie:**

* Attributes: movie\_id, movie\_name, rating, release\_date, director, description, duration, genre
* Purpose: Stores comprehensive information about movies available for screening. The table captures vital details such as movie name, release date, director, and genre, facilitating informed decision-making for customers and administrators.

**4. tbl\_feedback:**

* Attributes: feedback\_id, feedback, movie\_id, customer\_id
* Purpose: Facilitates the collection of customer feedback for movies they have watched. The table establishes relationships between customers, movies, and their respective feedback, enabling insights into audience preferences and sentiments.

**5. tbl\_show:**

* Attributes: show\_id, theatre\_id, movie\_id, show\_time, show\_date, seats\_left, language
* Purpose: Represents individual movie showings at theaters, capturing crucial details such as show timings, available seats, and language preferences. This table enables efficient scheduling and management of movie screenings across different theaters.

**6. tbl\_tickets:**

* Attributes: ticket\_id, customer\_id, show\_id, noOfSeats
* Purpose: Tracks booked tickets, associating customers with specific showings and the number of seats reserved. This table facilitates seamless ticketing processes and ensures accurate recording of customer bookings.

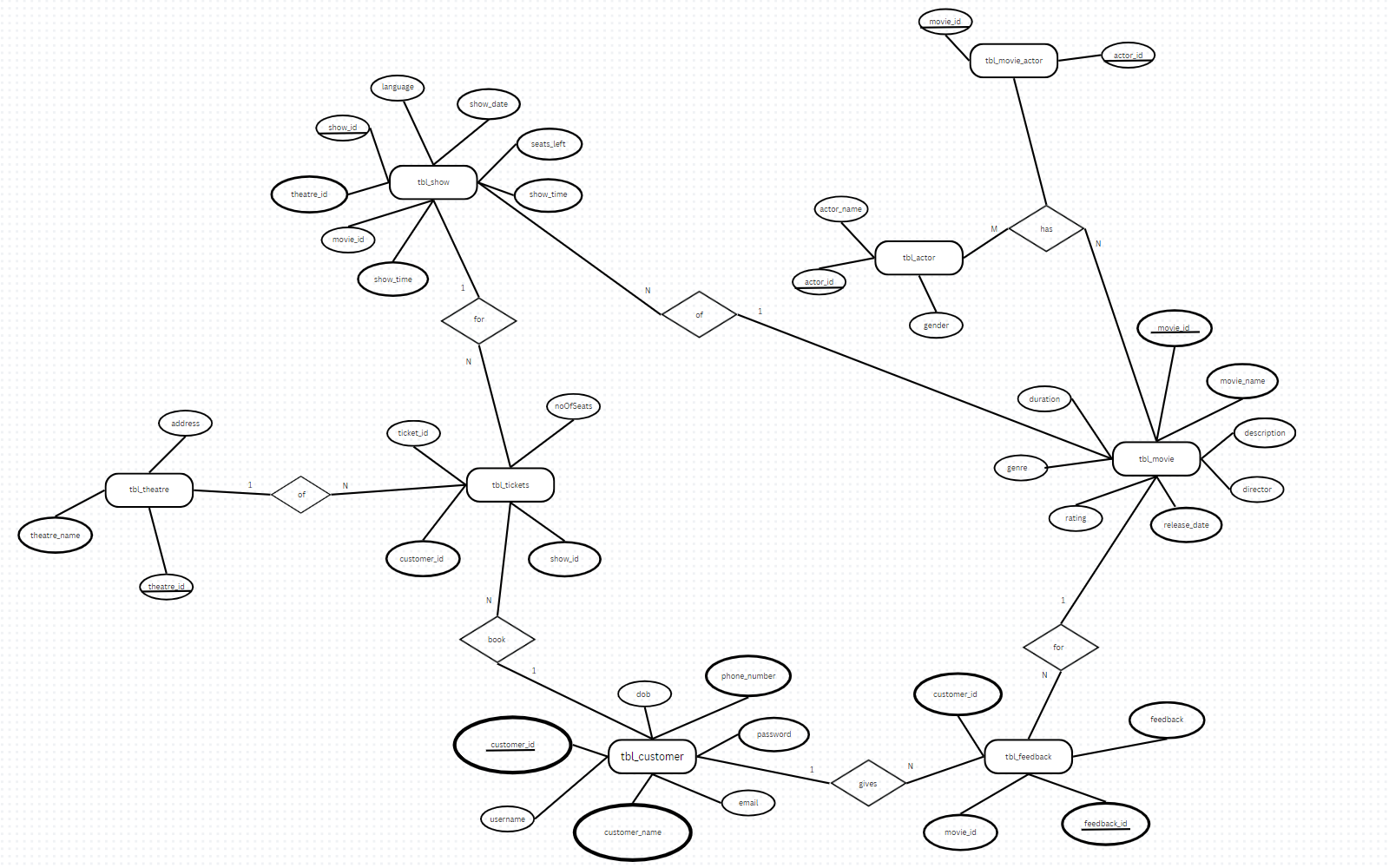
**7. tbl\_actor:**

* Attributes: actor\_id, actor\_name, gender
* Purpose: Stores information about actors featured in movies, including their names and genders. This table serves as a reference for associating actors with the movies they appear in, enriching the movie database with comprehensive cast details.

**8. tbl\_movie\_actor:**

* Attributes: movie\_id, actor\_id
* Purpose: Establishes the many-to-many relationship between movies and actors, linking each movie with its respective cast members. This table enables efficient retrieval of actor information for movies and supports detailed analyses of cast compositions across different films.

**2.2 Entity-Relationship (ER) diagram**



### 1. One-to-One Relationships:

* **tbl\_customer** <-> **tbl\_feedback**: One customer can leave multiple feedbacks, but each feedback is associated with only one customer.

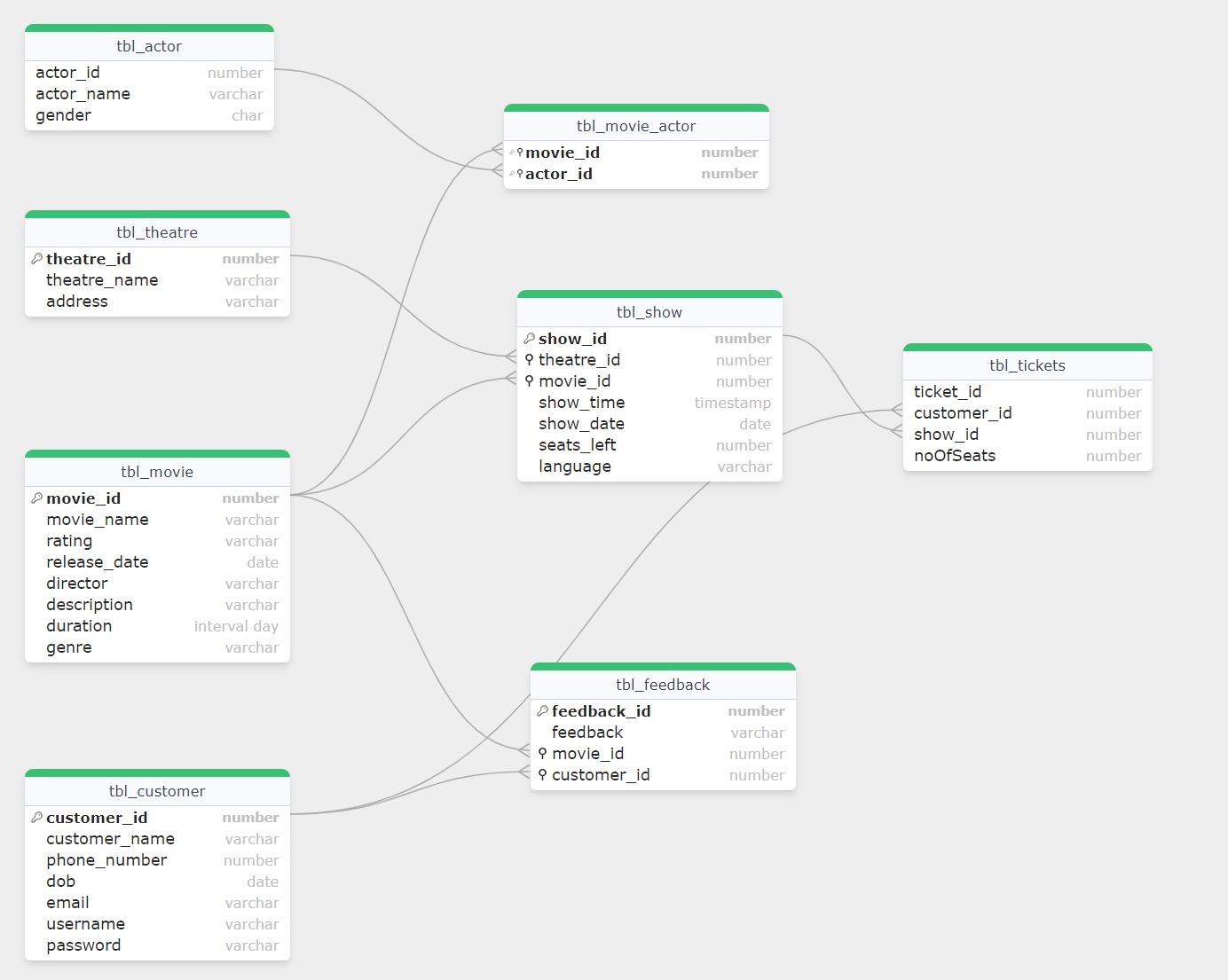
### 2. One-to-Many Relationships:

* **tbl\_theatre** -> **tbl\_show**: One theatre can have multiple movie shows, but each show belongs to only one theatre.
* **tbl\_movie** -> **tbl\_feedback**: One movie can receive multiple feedbacks, but each feedback is associated with only one movie.
* **tbl\_movie** -> **tbl\_movie\_actor**: One movie can have multiple actors, but each actor can be associated with multiple movies.

### 3. Many-to-Many Relationships:

* **tbl\_movie** <-> **tbl\_actor**: Many movies can have many actors, forming a many-to-many relationship.
* **tbl\_movie** <-> **tbl\_show**: Many movies can have multiple showings across different theatres, and each show can feature multiple movies.
* **tbl\_customer** <-> **tbl\_tickets**: Many customers can book multiple tickets, but each ticket is booked by only one customer.
* **tbl\_show** <-> **tbl\_tickets**: Many shows can have multiple tickets booked, but each ticket is associated with only one show.

**2.3 Entity-Relationship (ER) diagram To Table**



**2.4 Normalization**

Normalization is a crucial process in database design aimed at reducing redundancy and ensuring data integrity by organizing data into logical, interconnected tables. The database schema for the movie ticket booking system has undergone normalization to adhere to standard normalization forms, thereby optimizing data storage and enhancing system efficiency.

**1. First Normal Form (1NF):** The schema satisfies 1NF by ensuring that each attribute in every table holds atomic values. For instance:

* In tbl\_customer, attributes such as customer\_name and phone\_number hold single values.
* In tbl\_movie, attributes like movie\_name and rating contain individual, non-divisible data elements.

**2. Second Normal Form (2NF):** The schema conforms to 2NF by eliminating partial dependencies, ensuring that each non-key attribute is fully dependent on the primary key. For example:

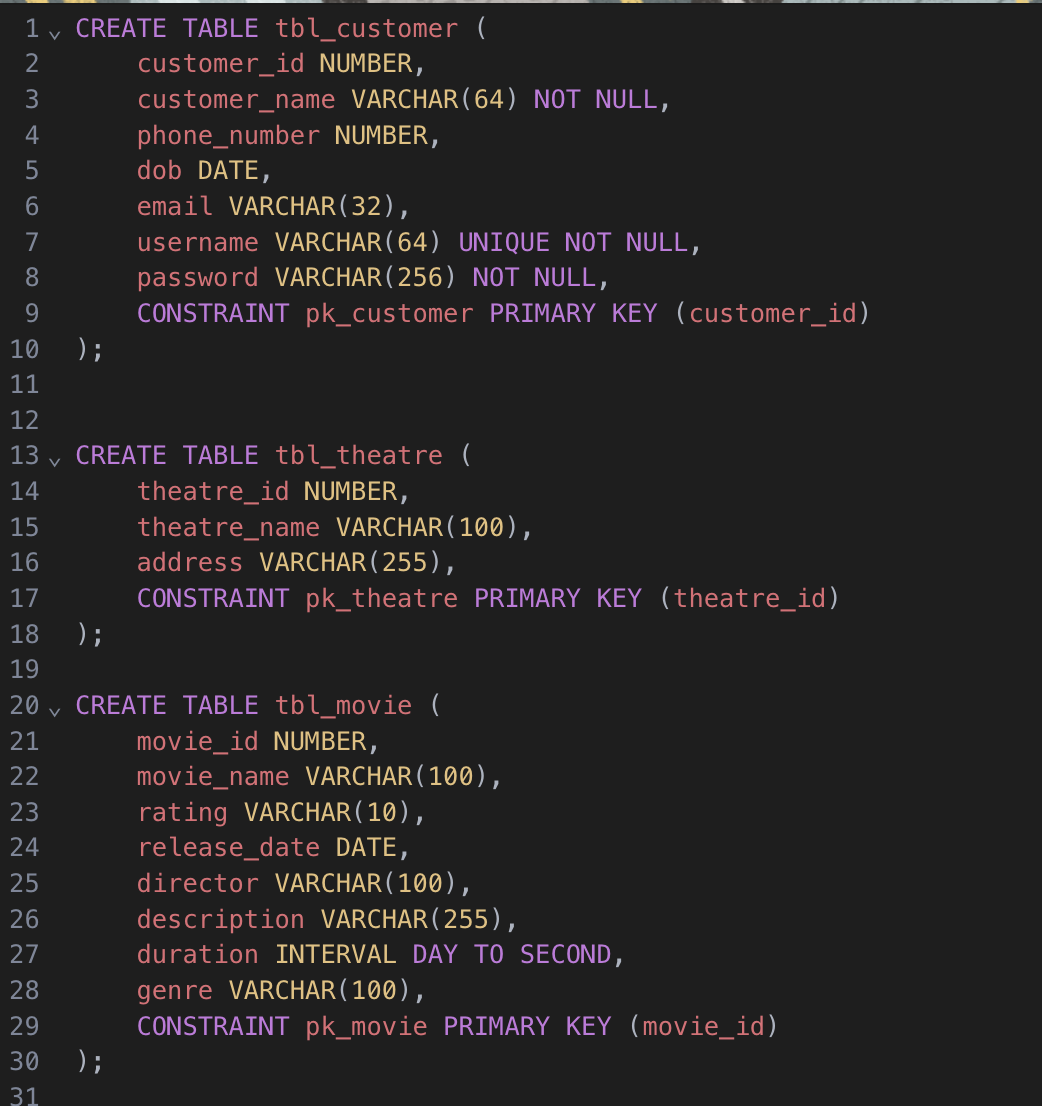
* In tbl\_show, attributes like show\_time and show\_date are functionally dependent on the primary key show\_id.
* In tbl\_feedback, the attribute feedback depends solely on the feedback\_id, ensuring no partial dependencies exist.

**3. Third Normal Form (3NF):** The schema adheres to 3NF by removing transitive dependencies, ensuring that no non-key attribute depends on another non-key attribute. For instance:

* In tbl\_movie, attributes such as director and description are independent of each other, ensuring no transitive dependencies exist.
* Similarly, in tbl\_theatre, attributes like theatre\_name and address are independent, avoiding transitive dependency issues.

SQL & PL/SQL

3.1 SQL Statements for table creation



A computer screen with text on it

Description automatically generated

A computer screen with text

Description automatically generated

A screen shot of a black background

Description automatically generated

3.2 SQL Statements for data insertion

A screen shot of a computer program

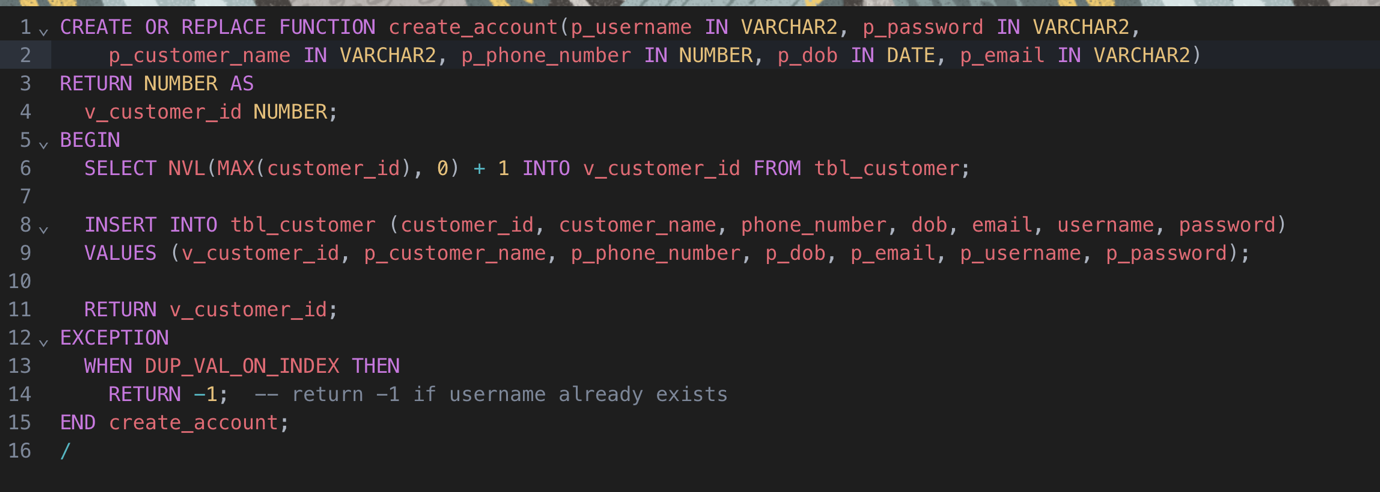
Description automatically generated

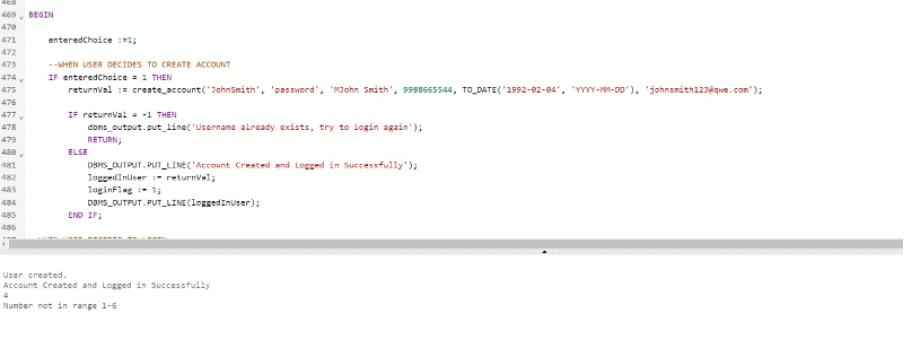
A screenshot of a computer program

Description automatically generated

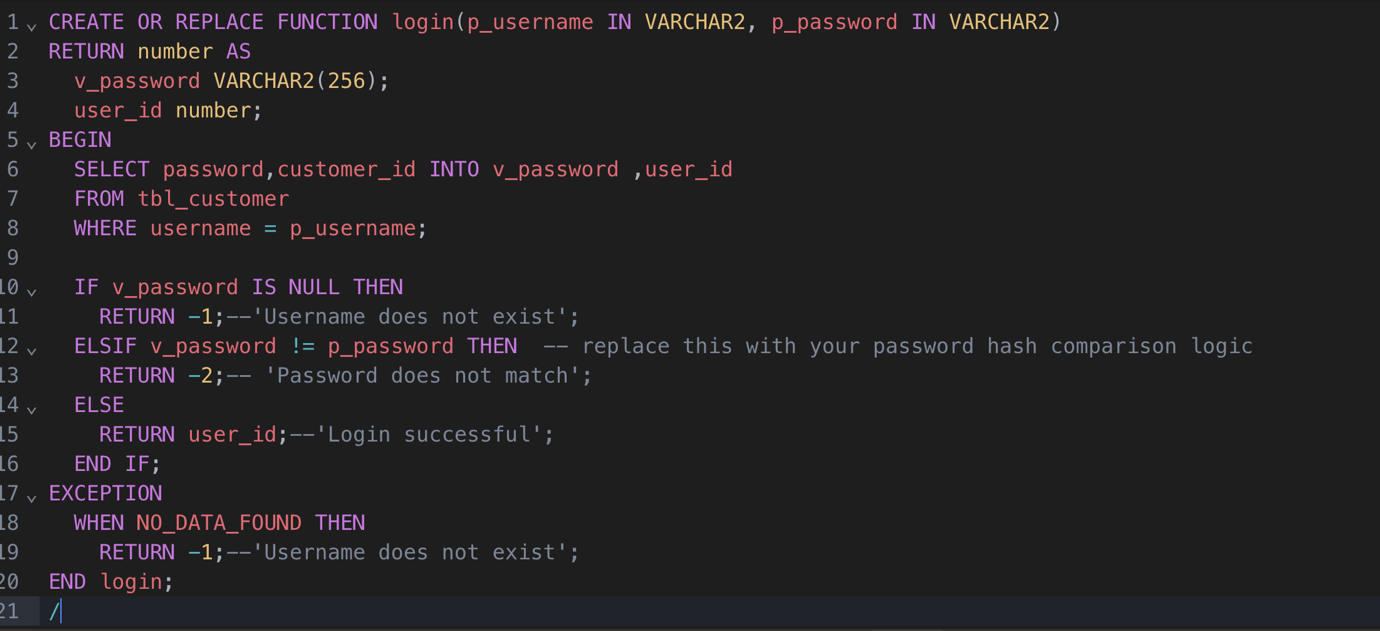
3.3 PL/SQL Statements

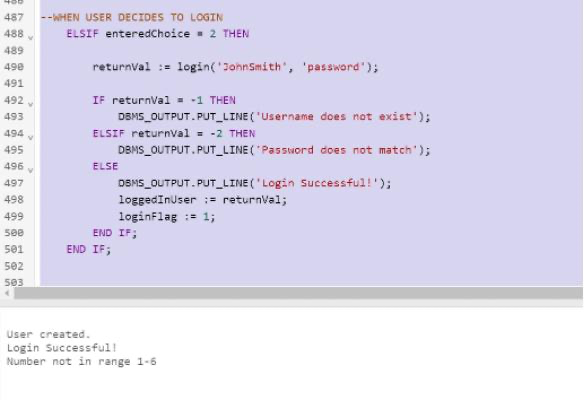
Create Account Function





Login Function



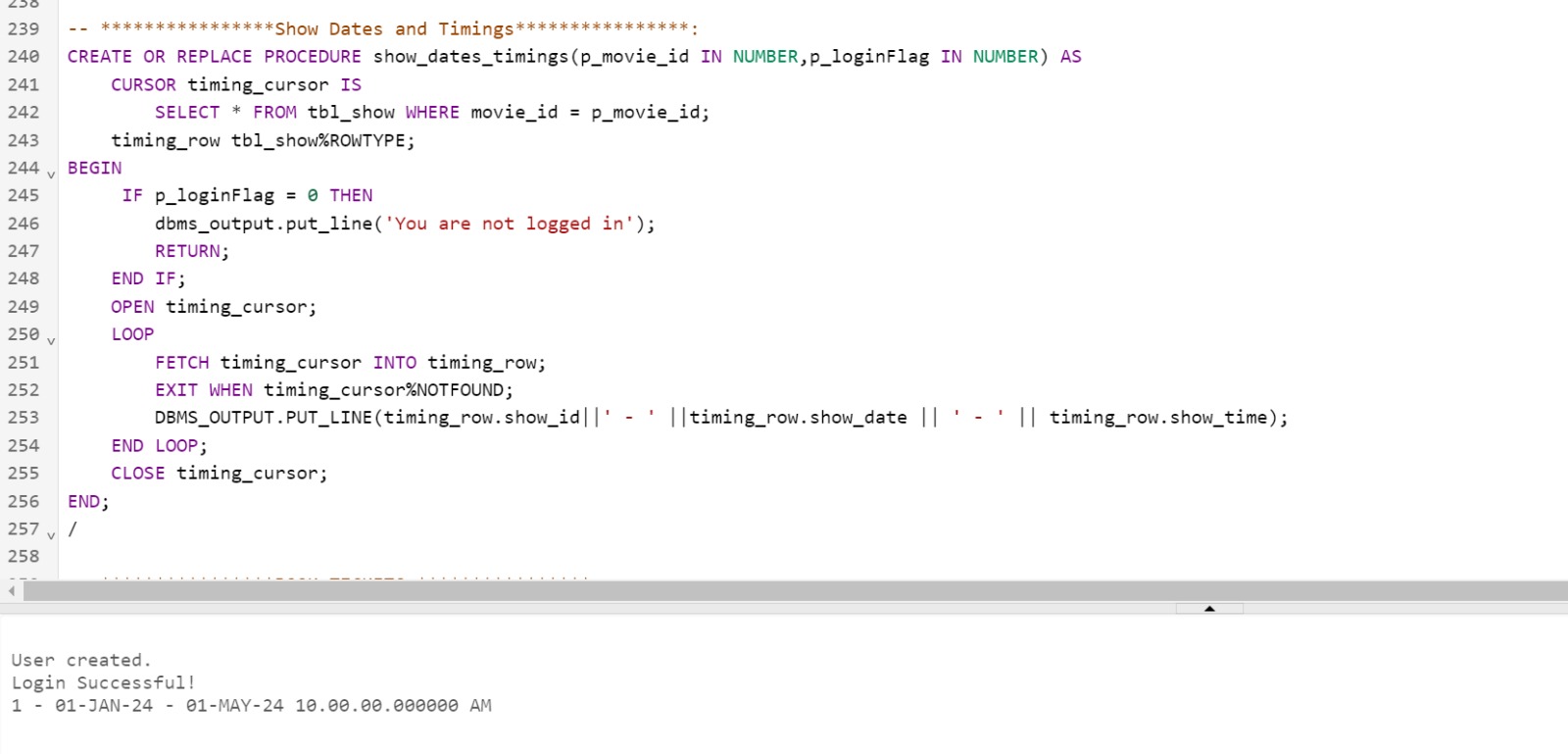


Show Movies ProcedureA screenshot of a computer

Description automatically generated

Search Movies ProcedureA screenshot of a computer

Description automatically generated

Show timings Procedure

Show Feedback ProcedureA computer screen shot of a program

Description automatically generated

Book Tickets Procedure and Trigger to update seat count

A screenshot of a computer code

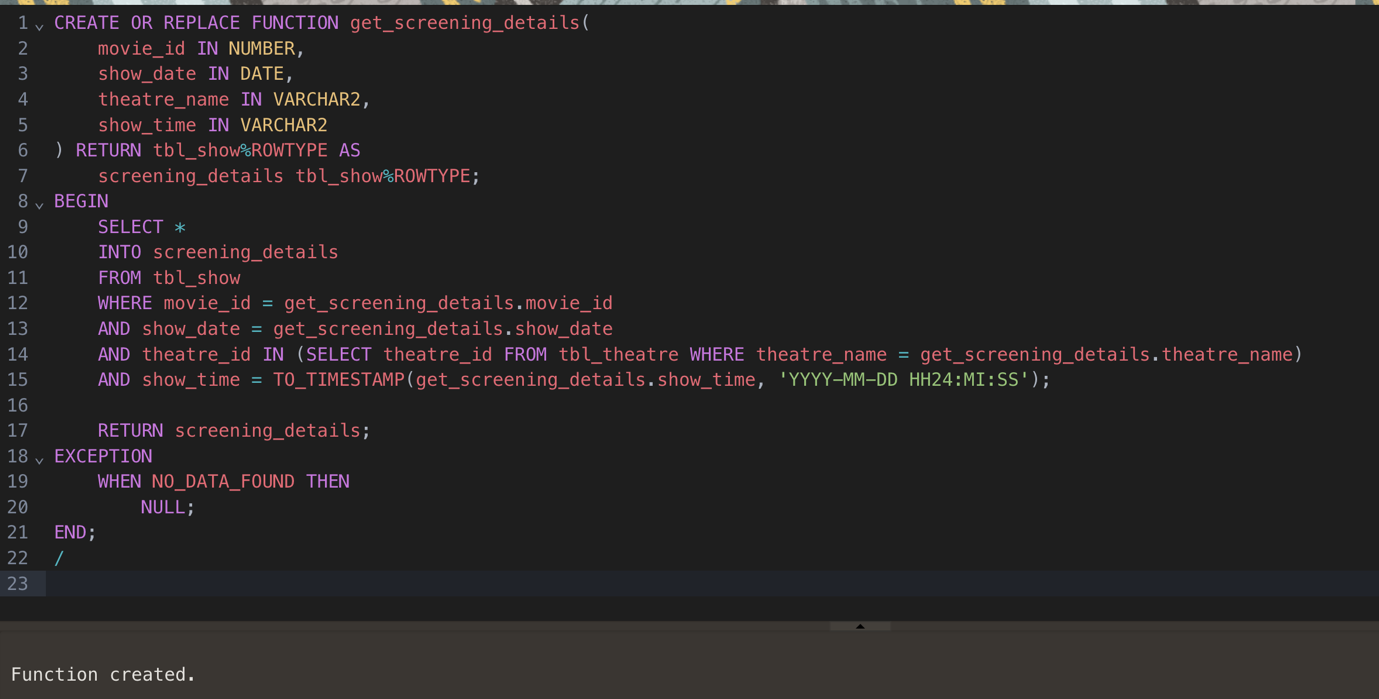
Description automatically generated

Cancel Tickets Procedure and Trigger to update seat count

A screenshot of a computer program

Description automatically generated

Get Screening Details function

****

A screen shot of a computer

Description automatically generated

CONCLUSION

In the realm of modern entertainment, the significance of efficient and user-friendly platforms for movie booking cannot be overstated. Through our SQL and PL/SQL mini-project on a movie booking site, we have endeavored to address this need by developing a comprehensive database schema and accompanying functionalities tailored to enhance the movie-going experience for both customers and administrators.

Throughout this project, we have designed and implemented a normalized database schema, normalization forms (1NF, 2NF, and 3NF). This approach ensures data integrity, minimizes redundancy, and optimizes storage.

Our SQL queries and PL/SQL procedures offer a range of features and capabilities essential for seamless movie booking operations. Customers can easily browse movies, view show timings, book tickets, and provide feedback, all within a user-friendly interface designed for optimal convenience.

Thank you for the opportunity to work on this project, and we look forward to further innovations and enhancements in the realm of movie booking systems