Software Requirement Specification

Movie Ticket Booking System

Purpose:

The purpose of this document is to present a detailed description of the Movie Ticket Booking System. It will explain the purpose and features of the system, the interfaces of the system, what the system will do, the constraints under which it must operate and how the system will react to external stimuli.

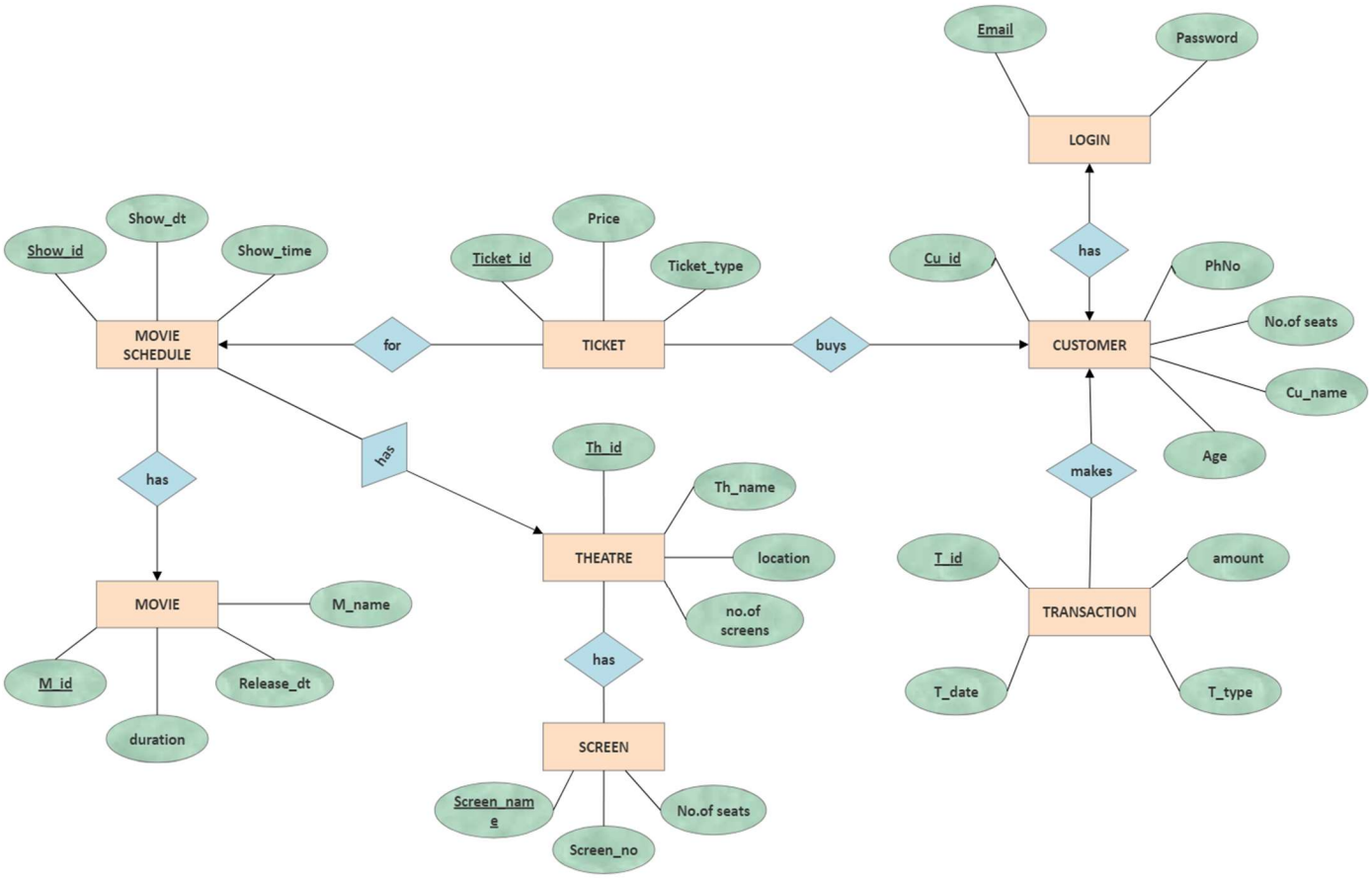
Scope of Project :

This software system will be a Movie Ticket Booking System. This system will be designed to maximize the ease of the customers to book tickets, which would otherwise have to be booked by standing in long queues at the theaters. More specifically, this system is designed to allow a customer to book and cancel his tickets at any time of the day. This software will inform the customer by sending an E-mail, confirming his booking/canceling of his tickets. Preformatted reply forms are used in every stage of the tickets' booked/cancelled progress through the system to provide a uniform review process. The system also contains a relational database containing a list of Movies, Timings, Seats, Prices and Bookings.

System Environment:

The Movie Ticket Booking System has two active actors, customer and admin. The Customer accesses the Movie Ticket Booking System through the Internet. Any customer communication with the system is through email. The Admin accesses the entire system directly.

ERD:



Functional Requirements Specification:

This section outlines the use cases for each of the active Customers separately. The customer is the main actor in this system.

Customer Use Case

- Customer Registration:
  The Customer registers in the Movie Ticket Booking System.

Initial Step-By-Step Description :

1. The Customer registers with his/her details in the Movie Ticket Booking System.
2. The System generates and sends an email acknowledgement.

Customer

- Login :
  The Customer accesses the system to login to his/her account.

Initial Step-By-Step Description :

1. The Customer is logged into his/her account in the Movie Ticket Booking System.
2. The Customer then proceeds to book a movie ticket.

- The Customer accesses the Movie Ticket Booking System, searches for a movie.

Initial Step-By-Step Description:

The system shall enable customers to view a list of movies showcased at each venue by date and time.

1. The Customer chooses to search by movie name, timing or date and location.
2. The system displays the choices to the Customer.
3. The Customer selects the movie desired.
4. The system presents the abstract of the movie to the Customer.
5. The Customer chooses to book the movie.

- The Customer selects a movie, time, seats.

Initial Step-By-Step Description :

1. The Customer chooses the Book ticket button.
2. Then Customer selects location, time, no of seats,seats option (silver , gold , platinum).

- The Customer proceeds to payment for his/her booking.

Initial Step-By-Step Description:

1. The Customer chooses the Make Payment button.
2. The Customer then is allowed to make his payment via various methods (Net banking, Credit Card, Debit Card).
4. The System sends an acknowledgement of booking on the mobile number of Customers.

The customer can possibly cancel his/her booking 1 hour before the movie timing, which has not been shown in the above diagram.

Admin Use Cases:

- The Admin can upload the list of movies.

Initial Step-By-Step Description:

Before this use case can be initiated, the Admin has already accessed the main page of the Movie Ticket Booking System.

1. The Admin selects to Upload Movies.
2. The System presents a form for uploading movies.
3. The Admin fills in the information and submits the form.

- The Admin can update the seats, add screens in the theaters for the respective movies.

Initial Step-By-Step Description:

1. The Admin selects a choice of adding or updating.
2. The Admin can update seats or add screens for the respective movie.

- The Admin views the payments made by the customer.

Initial Step-By-Step Description :

1. The Admin selects View Payments.
2. The system presents a list of customers.
3. The system presents the payment to the chosen customer.

Admin View Payments

Software requirements:

Operating System :Windows 7/8/10
Front-end:Jframe
Back-end :MYSQL
Database connectivity :JDBC
IDE :Eclipse

User Characteristics :

The Customer is expected to be Internet literate and be able to use a search engine. The main screen of the Movie Ticket Booking System will have the search function. The Admin is expected to be Windows literate and to be able to use buttons, pulldown menus, and similar tools.

Non-Functional Requirements :

The Movie Ticket Booking System will be on a server with high speed Internet capability. The software developed here assumes the use of a tool such as Tomcat for connection between the Web pages and the database. The speed of the Customer's connection will depend on the hardware used rather than characteristics of this system.

Normalization:

The tables are in 1 NF because each attribute has atomic values.
The tables are in 2NF because it satisfies both the following conditions hold:
Table is in 1NF (First normal form)
No non-prime attribute is dependent on the proper subset of any candidate key of table.
The tables are in 3NF because it is in 2NF and for each functional dependency X-> Y at least one of the following conditions hold:
X is a super key of table
Y is a prime attribute of table

==========

```
mysql> create database movieticketbooking;
Query OK, 1 row affected (0.01 sec)

mysql> use movieticketbooking;
Database changed
mysql> create table Customer(
    ->      C_id int primary key,
    ->      CName varchar(30),
    ->      PhNo dec(10),
    ->      Age int,
    ->      NoOfSeats int);
Query OK, 0 rows affected (0.04 sec)

mysql> create table Login(
    ->      Email varchar(50) primary key,
    ->      Password varchar(15) unique);
Query OK, 0 rows affected (0.10 sec)

mysql> create table has_login(
    ->       C_id int primary key,
    ->      Email varchar(50),
    ->      foreign key(C_id) references Customer(C_id) on delete cascade on update cascade,
    ->      foreign key (Email) references Login(Email) on delete cascade on update cascade);
Query OK, 0 rows affected (0.08 sec)

mysql> create table Ticket(
    ->         Tkt_id int primary key,
    ->         Tkt_type varchar(8),
    ->         price int,
    ->        C_id int,
    ->       Show_id int);
Query OK, 0 rows affected (0.06 sec)
```

```
mysql> create table Transaction(
    ->         T_id int primary key,
    ->         T_date date,
    ->         T_type varchar(15),
    ->         Amount int,
    ->         C_id int);
Query OK, 0 rows affected (0.03 sec)

mysql> create table Movie(
    ->          M_id int primary key,
    ->          MName varchar(30),
    ->          duration_in_mins int,
    ->          Release_dt date);
Query OK, 0 rows affected (0.05 sec)

mysql> create table Theater(
    ->          Th_id int primary key,
    ->          ThName varchar(30),
    ->          location varchar(50),
    ->          no_of_screens int);
Query OK, 0 rows affected (0.06 sec)

mysql> create table Screen(
    ->         ScreenName varchar(20) primary key,
    ->         Screen_no int,
    ->         No_of_seats int);
Query OK, 0 rows affected (0.04 sec)

mysql> create table TheaterScreen(
    -> Th_id int,ScreenName varchar(20),
    -> primary key(Th_id,ScreenName),
    -> foreign key(Th_id) references Theater(Th_id),
    -> foreign key(ScreenName) references Screen(ScreenName));
Query OK, 0 rows affected (0.12 sec)

mysql> create table MovieSchedule(
    ->          Show_id int primary key,
    ->         Show_date date,
    ->         Show_time time,
    ->          M_id int,
    ->          Th_id int,
    ->         foreign key(M_id) references Movie(M_id) on delete cascade on update cascade,
    ->         foreign key(Th_id) references Theater(Th_id) on delete cascade on update cascade);
```

```
Query OK, 0 rows affected (0.12 sec)

mysql> alter table Transaction add foreign key(C_id) references Customer(C_id);
Query OK, 0 rows affected (0.08 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> alter table Ticket add foreign key(Show_id) references MovieSchedule(Show_id), add
foreign key(C_id) references Customer(C_id);
Query OK, 0 rows affected (0.16 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> alter table MovieSchedule add foreign key(M_id) references Movie(M_id), add foreign
key(Th_id) references Theater(Th_id);
Query OK, 0 rows affected (0.15 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> desc Login;
+----------+-------------+------+-----+---------+-------+
| Field    | Type        | Null | Key | Default | Extra |
+----------+-------------+------+-----+---------+-------+
| Email    | varchar(50) | NO   | PRI | NULL    |       |
| Password | varchar(15) | YES  | UNI | NULL    |       |
+----------+-------------+------+-----+---------+-------+
2 rows in set (0.00 sec)

mysql> desc has_login;
+-------+-------------+------+-----+---------+-------+
| Field | Type        | Null | Key | Default | Extra |
+-------+-------------+------+-----+---------+-------+
| C_id  | int         | NO   | PRI | NULL    |       |
| Email | varchar(50) | YES  | MUL | NULL    |       |
+-------+-------------+------+-----+---------+-------+
2 rows in set (0.00 sec)

mysql> desc Customer;
+-----------+---------------+------+-----+---------+-------+
| Field     | Type          | Null | Key | Default | Extra |
+-----------+---------------+------+-----+---------+-------+
| C_id      | int           | NO   | PRI | NULL    |       |
| CName     | varchar(30)   | YES  |     | NULL    |       |
| PhNo      | decimal(10,0) | YES  |     | NULL    |       |
| Age       | int           | YES  |     | NULL    |       |
| NoOfSeats | int           | YES  |     | NULL    |       |
+-----------+---------------+------+-----+---------+-------+
```

5 rows in set (0.00 sec)

mysql> desc Transaction;

Field	Type	Null	Key	Default	Extra
T_id	int	NO	PRI	NULL	
T_date	date	YES		NULL	
T_type	varchar(15)	YES		NULL	
Amount	int	YES		NULL	
C_id	int	YES	MUL	NULL	

5 rows in set (0.00 sec)

mysql> desc Ticket;

Field	Type	Null	Key	Default	Extra
Tkt_id	int	NO	PRI	NULL	
Tkt_type	varchar(8)	YES		NULL	
price	int	YES		NULL	
C_id	int	YES	MUL	NULL	
Show_id	int	YES	MUL	NULL	

5 rows in set (0.00 sec)

mysql> desc MovieSchedule;

Field	Type	Null	Key	Default	Extra
Show_id	int	NO	PRI	NULL	
Show_date	date	YES		NULL	
Show_time	time	YES		NULL	
M_id	int	YES	MUL	NULL	
Th_id	int	YES	MUL	NULL	

5 rows in set (0.00 sec)

mysql> desc Movie;

Field	Type	Null	Key	Default	Extra
M_id	int	NO	PRI	NULL	
MName	varchar(30)	YES		NULL	

```
| duration_in_mins | int         | YES  |     | NULL    |       |
| Release_dt       | date        | YES  |     | NULL    |       |
+------------------+-------------+------+-----+---------+-------+
4 rows in set (0.00 sec)

mysql> desc Theater;
+---------------+-------------+------+-----+---------+-------+
| Field         | Type        | Null | Key | Default | Extra |
+---------------+-------------+------+-----+---------+-------+
| Th_id         | int         | NO   | PRI | NULL    |       |
| ThName        | varchar(30) | YES  |     | NULL    |       |
| location      | varchar(50) | YES  |     | NULL    |       |
| no_of_screens | int         | YES  |     | NULL    |       |
+---------------+-------------+------+-----+---------+-------+
4 rows in set (0.00 sec)

mysql> desc Screen;
+-------------+-------------+------+-----+---------+-------+
| Field       | Type        | Null | Key | Default | Extra |
+-------------+-------------+------+-----+---------+-------+
| ScreenName  | varchar(20) | NO   | PRI | NULL    |       |
| Screen_no   | int         | YES  |     | NULL    |       |
| No_of_seats | int         | YES  |     | NULL    |       |
+-------------+-------------+------+-----+---------+-------+
3 rows in set (0.00 sec)

mysql> desc TheaterScreen;
+------------+-------------+------+-----+---------+-------+
| Field      | Type        | Null | Key | Default | Extra |
+------------+-------------+------+-----+---------+-------+
| Th_id      | int         | NO   | PRI | NULL    |       |
| ScreenName | varchar(20) | NO   | PRI | NULL    |       |
+------------+-------------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```

QUERIES

1.Display Number of screens in Pavillion Mall

```
mysql> select no_of_screens from theater where ThName ="Pavillion Mall";
+---------------+
| no_of_screens |
+---------------+
|             5 |
+---------------+
1 row in set (0.00 sec)
```

2.Count the no. of movies released in 2021?

```
mysql> select count(*) from movie where Release_dt >= "2021-01-01" and
Release_dt<= "2021-12-31";
+----------+
| count(*) |
+----------+
|        8 |
+----------+
1 row in set (0.00 sec)
```

3. Display all movies whose duration is greater than 2 hours.

```
mysql> select MName from movie where duration_in_mins >120 ;
+-------------------------+
| MName                   |
+-------------------------+
| Ghostbusters: afterlife  |
| Spider-Man: No Way Home  |
| No time to die          |
| Dune                    |
| Operation Mincemeat     |
| Secrets of Dumbledore    |
| Eternals                |
| The Matrix Resurrections |
| Dracula                 |
+-------------------------+
9 rows in set (0.00 sec)
```

4.Display list of all movies whose names contain “rr”.

```
mysql> select * from movie  WHERE MName LIKE "%rr%";
+------+-------------------------+-----------------+------------+
| M_id | MName                   | duration_in_mins | Release_dt |
+------+-------------------------+-----------------+------------+
| 1011 | The Matrix Resurrections |             148 | 2021-12-22 |
+------+-------------------------+-----------------+------------+
1 row in set (0.00 sec)
```

5.Display list of movies in ascending order of release_dt.

```
mysql> select * from movie order by Release_dt;
+------+-------------------------+-----------------+------------+
| M_id | MName                   | duration_in_mins | Release_dt |
+------+-------------------------+-----------------+------------+
| 1009 | Old                     |              108 | 2021-07-19 |
| 1003 | No time to die          |              163 | 2021-09-30 |
| 1010 | Pleasure                |              100 | 2021-10-08 |
| 1004 | Dune                    |              155 | 2021-10-22 |
| 1001 | Ghostbusters: afterlife |              124 | 2021-11-19 |
| 1002 | Spider-Man: No Way Home |              148 | 2021-12-16 |
| 1011 | The Matrix Resurrections |              148 | 2021-12-22 |
| 1005 | Sing 2                  |              110 | 2021-12-31 |
| 1008 | Eternals                |              156 | 2022-01-12 |
| 1012 | The Lost City           |              120 | 2022-03-25 |
| 1007 | Secrets of Dumbledore   |              142 | 2022-04-08 |
| 1006 | Operation Mincemeat     |              128 | 2022-04-15 |
| 1013 | Dracula                 |              180 | 2022-04-25 |
+------+-------------------------+-----------------+------------+
```

13 rows in set (0.00 sec)

6. Display the Th_id, ThName, location of the Theatre with maximum no.of screens.

mysql> select Th_id, ThName, location from theater where no_of_screens=(select max(no_of_screens) from theater);

```
+-------+---------------+--------------+
| Th_id | ThName        | location     |
+-------+---------------+--------------+
|  4810 | Pavillion Mall | SB road, Pune |
+-------+---------------+--------------+
```

1 row in set (0.03 sec)

7. Display the count of each type of transaction.

mysql> select T_type,count(*) from transaction group by T_type;

```
+--------+----------+
| T_type | count(*) |
+--------+----------+
| online |        2 |
| cash   |        2 |
+--------+----------+
```

2 rows in set (0.01 sec)

8.Display the movie name along with the show_date,show_time and Th_id (Join by Cartesian product).

```
mysql> select MName, Show_date, Show_time, Th_id From MovieSchedule
CROSS JOIN Movie ON MovieSchedule.M_id = Movie.M_id;
+------------------------+-----------+-----------+-------+
| MName                  | Show_date  | Show_time | Th_id |
+------------------------+-----------+-----------+-------+
| The Matrix Resurrections | 2022-04-11 | 07:00:00  |  4801 |
| Old                    | 2022-03-17 | 10:00:00  |  4810 |
| Sing 2                 | 2022-04-27 | 07:00:00  |  4801 |
| Eternals               | 2022-05-04 | 10:00:00  |  4801 |
| Dune                   | 2022-04-12 | 07:00:00  |  4801 |
| Eternals               | 2022-05-10 | 09:00:00  |  4810 |
| No time to die         | 2022-04-20 | 09:00:00  |  4810 |
| The Lost City          | 2022-04-26 | 09:00:00  |  4810 |
| Ghostbusters: afterlife  | 2022-04-30 | 09:00:00  |  4801 |
| Dracula                | 2022-05-01 | 07:00:00  |  4801 |
| Spider-Man: No Way Home  | 2022-05-03 | 09:00:00  |  4801 |
| Spider-Man: No Way Home  | 2022-05-03 | 09:00:00  |  4810 |
+------------------------+-----------+-----------+-------+
12 rows in set (0.00 sec)
```

9.Display the theater name, location, Show id, Show date and Show time (Inner Join).

```
mysql> select ThName, location, Show_id, Show_date, Show_time From
movieschedule INNER JOIN Theater ON movieschedule.Th_id=theater.Th_id;
+---------------+---------------+---------+-----------+-----------+
| ThName        | location      | Show_id | Show_date  | Show_time |
+---------------+---------------+---------+-----------+-----------+
| Pacific Mall   | Swargate, Pune |     111 | 2022-04-11 | 07:00:00  |
| Pacific Mall   | Swargate, Pune |     113 | 2022-04-27 | 07:00:00  |
| Pacific Mall   | Swargate, Pune |     114 | 2022-05-04 | 10:00:00  |
| Pacific Mall   | Swargate, Pune |     115 | 2022-04-12 | 07:00:00  |
| Pacific Mall   | Swargate, Pune |     119 | 2022-04-30 | 09:00:00  |
| Pacific Mall   | Swargate, Pune |     120 | 2022-05-01 | 07:00:00  |
| Pacific Mall   | Swargate, Pune |     121 | 2022-05-03 | 09:00:00  |
| Pavillion Mall | SB road, Pune  |     112 | 2022-03-17 | 10:00:00  |
| Pavillion Mall | SB road, Pune  |     116 | 2022-05-10 | 09:00:00  |
| Pavillion Mall | SB road, Pune  |     117 | 2022-04-20 | 09:00:00  |
| Pavillion Mall | SB road, Pune  |     118 | 2022-04-26 | 09:00:00  |
| Pavillion Mall | SB road, Pune  |     122 | 2022-05-03 | 09:00:00  |
+---------------+---------------+---------+-----------+-----------+
12 rows in set (0.00 sec)
```

10.Demonstrate the use of Left outer join.

```
mysql> select MName,Show_date,Show_time,Th_id from movie  LEFT JOIN
movieschedule ON movie.M_id = movieschedule.M_id;

+-------------------------+------------+-----------+-------+

| MName                   | Show_date  | Show_time | Th_id |

+-------------------------+------------+-----------+-------+

| Ghostbusters: afterlife  | 2022-04-30 | 09:00:00  |  4801 |

| Spider-Man: No Way Home  | 2022-05-03 | 09:00:00  |  4801 |

| Spider-Man: No Way Home  | 2022-05-03 | 09:00:00  |  4810 |

| No time to die          | 2022-04-20 | 09:00:00  |  4810 |

| Dune                    | 2022-04-12 | 07:00:00  |  4801 |

| Sing 2                  | 2022-04-27 | 07:00:00  |  4801 |

| Operation Mincemeat     | NULL       | NULL      |  NULL |

| Secrets of Dumbledore   | NULL       | NULL      |  NULL |

| Eternals                | 2022-05-04 | 10:00:00  |  4801 |

| Eternals                | 2022-05-10 | 09:00:00  |  4810 |
```

| Old | 2022-03-17 | 10:00:00 | 4810 |

| Pleasure | NULL | NULL | NULL |

| The Matrix Resurrections | 2022-04-11 | 07:00:00 | 4801 |

| The Lost City | 2022-04-26 | 09:00:00 | 4810 |

| Dracula | 2022-05-01 | 07:00:00 | 4801 |

+-------------------------+------------+-----------+-------+

15 rows in set (0.00 sec)

11.Demonstrate the use of Right outer join.

mysql> select MName,Show_date,Show_time,Th_id from movie RIGHT JOIN moviesschedule ON movie.M_id = moviesschedule.M_id;

+-------------------------+------------+-----------+-------+

| MName | Show_date | Show_time | Th_id |

+-------------------------+------------+-----------+-------+

| The Matrix Resurrections | 2022-04-11 | 07:00:00 | 4801 |

| Old | 2022-03-17 | 10:00:00 | 4810 |

| Sing 2 | 2022-04-27 | 07:00:00 | 4801 |

| Eternals | 2022-05-04 | 10:00:00 | 4801 |

| Dune | 2022-04-12 | 07:00:00 | 4801 |

| Eternals | 2022-05-10 | 09:00:00 | 4810 |

| No time to die | 2022-04-20 | 09:00:00 | 4810 |

| The Lost City | 2022-04-26 | 09:00:00 | 4810 |

| Ghostbusters: afterlife | 2022-04-30 | 09:00:00 | 4801 |

| Dracula | 2022-05-01 | 07:00:00 | 4801 |

| Spider-Man: No Way Home | 2022-05-03 | 09:00:00 | 4801 |

| Spider-Man: No Way Home | 2022-05-03 | 09:00:00 | 4810 |

+-------------------------+------------+-----------+-------+

12 rows in set (0.00 sec)

12.Display the C_id and T_type of the Transaction with an amount higher than average.

mysql> select C_id, T_type, Amount from transaction where Amount>(select avg(Amount) from transaction);

+------+--------+--------+

| C_id | T_type | Amount |

+------+--------+--------+

| 311 | online | 350 |

| 316 | cash | 400 |

+------+--------+--------+

2 rows in set (0.00 sec)

mysql> select avg(Amount) from transaction;

+-------------+

| avg(Amount) |

+-------------+

| 332.5000 |

+-------------+

1 row in set (0.00 sec)

13.Display the Customer with maximum No of seats.

mysql> select CName, NoOfSeats from customer where NoOfSeats=(select max(NoOfSeats) from customer);

+---------------+-----------+

| CName | NoOfSeats |

```
+--------------+----------+

| Yogita Sharma |        5 |

+--------------+----------+

1 row in set (0.00 sec)
```

14. Create a view containing : CName, PhNo, Age and check if it is updatable.

```
mysql> CREATE VIEW V1 AS select CName, PhNo, Age from customer;
Query OK, 0 rows affected (0.02 sec)

mysql> select * from V1;
+----------------+------------+------+
| CName          | PhNo       | Age  |
+----------------+------------+------+
| Sharat Chandran | 8368526639 |   21 |
| Vikram Singh    | 9368269267 |   34 |
| Vinit Katariya  | 7854637893 |   53 |
| Yash Mittal     | 7374528452 |   28 |
| Supriya Sen     | 8358468262 |   45 |
| Yogita Sharma   | 9369046387 |   37 |
| Gareema Trivedi | 9086372796 |   48 |
| Satish Kumar    | 8469268369 |   25 |
| Shree Roy       | 8093688570 |   24 |
```

| Sakshi Tanwar | 7685377890 | 18 |

+----------------+------------+------+

10 rows in set (0.03 sec)

To check if it is updatable:

Insert:

mysql> insert into V1 values("Radha Sharma", 9574658374, 19);

ERROR 1423 (HY000): Field of view 'movieticketbooking.v1' underlying table doesn't have a default value

Cannot insert as primary key is not part of the view.

Update:

mysql> update V1 set Age=19 where CName="Sakshi Tanwar";

Query OK, 1 row affected (0.03 sec)

Rows matched: 1 Changed: 1 Warnings: 0

mysql> select * from V1;

+----------------+------------+------+

| CName | PhNo | Age |

+----------------+------------+------+

| Sharat Chandran | 8368526639 | 21 |

| Vikram Singh | 9368269267 | 34 |

| Vinit Katariya | 7854637893 | 53 |

| Yash Mittal | 7374528452 | 28 |

| Supriya Sen | 8358468262 | 45 |

| Yogita Sharma | 9369046387 | 37 |

| Gareema Trivedi | 9086372796 | 48 |

| Satish Kumar | 8469268369 | 25 |

| Shree Roy | 8093688570 | 24 |

| Sakshi Tanwar | 7685377890 | 19 |

+----------------+------------+------+

10 rows in set (0.00 sec)

Delete:

mysql> delete from V1 where CName="Sakshi Tanwar";

Query OK, 1 row affected (0.02 sec)

mysql> select * from V1;

+----------------+------------+------+

| CName | PhNo | Age |

+----------------+------------+------+

| Sharat Chandran | 8368526639 | 21 |

| Vikram Singh | 9368269267 | 34 |

| Vinit Katariya | 7854637893 | 53 |

| Yash Mittal | 7374528452 | 28 |

| Supriya Sen | 8358468262 | 45 |

| Yogita Sharma | 9369046387 | 37 |

| Gareema Trivedi | 9086372796 | 48 |

| Satish Kumar | 8469268369 | 25 |

| Shree Roy | 8093688570 | 24 |

+----------------+------------+------+

9 rows in set (0.00 sec)

15.Create a view containing a count of tickets of each type and check if it is updatable.

mysql> create view V2 as select Tkt_type, count(Tkt_type)from ticket group by Tkt_type;

mysql> select * from V2;

```
+----------+-----------------+
| Tkt_type | count(Tkt_type) |
+----------+-----------------+
| Silver   |               2 |
| Gold     |               1 |
+----------+-----------------+
```

2 rows in set (0.01 sec)

To check if it is updatable:

Insert:

mysql> insert into V2 (Tkt_type, count) values ("Platinum", 3);

ERROR 1471 (HY000): The target table V2 of the INSERT is not insertable-into

Cannot insert in the view as primary key is not part of the view and aggregate function is part of the view.

Update:

mysql> update V2 set Tkt_type="Silver" where count(Tkt_type)=1;

ERROR 1288 (HY000): The target table V2 of the UPDATE is not updatable

Cannot update the view as aggregate function is part of the view.

Delete:

mysql> delete from V2 where Tkt_type ="Silver";

ERROR 1288 (HY000): The target table V2 of the DELETE is not updatable

Cannot delete from the view as aggregate function is part of the view.

16.Create an index on the CName field and see the improvement in the performance using profiling.

mysql> set profiling = 1;

Query OK, 0 rows affected, 1 warning (0.00 sec)

mysql> select * from customer where CName="Sakshi Tanwar";

```
+------+--------------+------------+------+-----------+
| C_id | CName        | PhNo       | Age  | NoOfSeats |
+------+--------------+------------+------+-----------+
|  320 | Sakshi Tanwar | 7685377890 |   18 |         3 |
+------+--------------+------------+------+-----------+
1 row in set (0.00 sec)

mysql> create index I1 on customer(CName);
Query OK, 0 rows affected (0.03 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> select * from customer where CName="Sakshi Tanwar";
+------+--------------+------------+------+-----------+
| C_id | CName        | PhNo       | Age  | NoOfSeats |
+------+--------------+------------+------+-----------+
|  320 | Sakshi Tanwar | 7685377890 |   18 |         3 |
+------+--------------+------------+------+-----------+
1 row in set (0.00 sec)

mysql> show profiles;
+----------+------------+--------------------------------------------------+
| Query_ID | Duration   | Query                                            |
```

```
+----------+------------+-------------------------------------------------+
|        1 | 0.00047900 | select * from customer where CName="Sakshi Tanwar" |
|        2 | 0.02984375 | create index I1 on customer(CName)              |
|        3 | 0.00034500 | select * from customer where CName="Sakshi Tanwar" |
+----------+------------+-------------------------------------------------+
3 rows in set, 1 warning (0.00 sec)
```

17.Create table:

Ex_booking(Cu_id, T_id, T_date, T_type, Amount);

Write a trigger which will add the cancel booking record to the Ex_booking table when it

is deleted from the Transaction table.

```
mysql>  create table Ex_booking( Cu_id int, T_id int, T_date date, T_type
varchar(10), Amount int);
Query OK, 0 rows affected (0.04 sec)

mysql> delimiter //
mysql> create trigger t3 after delete on Transaction for each row  begin insert into
Ex_Booking values(old.C_id,old.T_id,old.T_date,old.T_type,old.Amount);
   -> end//
Query OK, 0 rows affected (0.01 sec)

mysql> delimiter ;
```

```
mysql> select * from transaction;
+------+------------+--------+--------+------+
| T_id | T_date     | T_type | Amount | C_id |
+------+------------+--------+--------+------+
|  901 | 2022-04-12 | online |    280 |  312 |
|  916 | 2022-04-26 | cash   |    300 |  319 |
|  921 | 2022-04-12 | online |    350 |  311 |
|  936 | 2022-04-26 | cash   |    400 |  316 |
+------+------------+--------+--------+------+
4 rows in set (0.01 sec)
mysql> delete from transaction where C_id=319;
Query OK, 1 row affected (0.03 sec)
mysql> select * from transaction;
+------+------------+--------+--------+------+
| T_id | T_date     | T_type | Amount | C_id |
+------+------------+--------+--------+------+
|  901 | 2022-04-12 | online |    280 |  312 |
|  921 | 2022-04-12 | online |    350 |  311 |
|  936 | 2022-04-26 | cash   |    400 |  316 |
+------+------------+--------+--------+------+
3 rows in set (0.00 sec)

mysql> select * from Ex_booking;
+-------+------+------------+--------+--------+
| Cu_id | T_id | T_date     | T_type | Amount |
```

```
+-------+------+------------+--------+--------+
|   319 |  916 | 2022-04-26 | cash   |    300 |
+-------+------+------------+--------+--------+
1 row in set (0.00 sec)
```

18. Write a function which takes C_id as input parameter and returns count of tickets purchased by that customer.

```
mysql> Delimiter //
mysql> Create function Count_tickets(Customers varchar(30)) returns int
    -> Begin
    -> Declare cnt int;
    -> select count(*) into cnt from Ticket where C_id=Customers;
    -> return cnt;
    -> End//
Query OK, 0 rows affected (0.03 sec)

mysql> Delimiter ;
mysql> select * from ticket;
+--------+----------+-------+------+---------+
| Tkt_id | Tkt_type | price | C_id | Show_id |
+--------+----------+-------+------+---------+
|    881 | Silver   |   180 |  319 |     113 |
|    885 | Gold     |   280 |  318 |     114 |
|    887 | Silver   |   200 |  317 |     112 |
```

```
+--------+----------+-------+------+---------+
3 rows in set (0.00 sec)

mysql> Select Count_tickets(319);
+-------------------+
| Count_tickets(319) |
+-------------------+
|                 1 |
+-------------------+
1 row in set (0.01 sec)
```

19. Write a trigger which will convert the M_name to upper-case before inserting a new record in Movie table.

```
mysql> Delimiter //
mysql> create trigger t2 before insert on Movie for each row
   -> begin
   -> set new.MName=upper(new.MName);
   -> end//
Query OK, 0 rows affected (0.01 sec)

mysql> delimiter ;
mysql> Insert into movie values(1020, "The fight club",148, "2020-12-16");
Query OK, 1 row affected (0.01 sec)

mysql> select * from movie;
```

```
+------+-------------------------+-----------------+------------+
| M_id | MName                   | duration_in_mins | Release_dt |
+------+-------------------------+-----------------+------------+
| 1001 | Ghostbusters: afterlife  |             124 | 2021-11-19 |
| 1002 | Spider-Man: No Way Home  |             148 | 2021-12-16 |
| 1003 | No time to die           |             163 | 2021-09-30 |
| 1004 | Dune                     |             155 | 2021-10-22 |
| 1005 | Sing 2                   |             110 | 2021-12-31 |
| 1006 | Operation Mincemeat      |             128 | 2022-04-15 |
| 1007 | Secrets of Dumbledore    |             142 | 2022-04-08 |
| 1008 | Eternals                 |             156 | 2022-01-12 |
| 1009 | Old                      |             108 | 2021-07-19 |
| 1010 | Pleasure                 |             100 | 2021-10-08 |
| 1011 | The Matrix Resurrections |             148 | 2021-12-22 |
| 1012 | The Lost City            |             120 | 2022-03-25 |
| 1013 | Dracula                  |             180 | 2022-04-25 |
| 1020 | THE FIGHT CLUB           |             148 | 2020-12-16 |
+------+-------------------------+-----------------+------------+
14 rows in set (0.00 sec)
```

20. Write a procedure to update the location of the given Theater (ThName and Newlocation are input parameters)

```
mysql> Delimiter //
mysql> Create Procedure update_location(IN temp_ThName varchar(30),IN
Newlocation varchar(30))
```

```
   -> Begin
   -> UPDATE theater set location = Newlocation where ThName = temp_ThName;
   -> end //
Query OK, 0 rows affected (0.01 sec)

mysql> delimiter ;
mysql> call update_location("Pacific Mall", "Shankar Seth Road, Pune");
Query OK, 1 row affected (0.01 sec)

mysql> select * from theater;
+-------+---------------+------------------------+---------------+
| Th_id | ThName        | location               | no_of_screens |
+-------+---------------+------------------------+---------------+
|  4801 | Pacific Mall  | Shankar Seth Road, Pune |             3 |
|  4810 | Pavillion Mall | SB road, Pune          |             5 |
+-------+---------------+------------------------+---------------+
2 rows in set (0.00 sec)
```