Assignment SQL & OOPS Ticket Booking System

Name: Shardul Satish Kulkarni

SupersetId: 5270707

Task 1: Database Design

1. Create a Database Named 'TicketBookingSystem'

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| hexatraining       |
| information_schema |
| mysql              |
| performance_schema |
| sakila             |
| sys                |
| world              |
+--------------------+
7 rows in set (0.00 sec)

mysql> create database TicketBookingSystem;
Query OK, 1 row affected (0.02 sec)

mysql> use TicketBookingSystem;
Database changed
mysql>
```

2. Write SQL scripts to create the mentioned tables with appropriate data types, constraints, and relationships.
    a. Venu
    b. Event
    c. Customers
    d. Booking

1. Venue:

```sql
create table Venue (
    venue_id int(10) AUTO_INCREMENT PRIMARY KEY,
    venue_name varchar(25),
    address varchar(100)
);
```

2. Event:

```
create table Event (
    event_id int AUTO_INCREMENT PRIMARY KEY,
    event_name varchar(255),
    event_date DATE,
    event_time TIME,
    venue_id int(10),
    total_seats int(100),
    available_seats int(100),
    ticket_price DECIMAL,
    event_type ENUM('Movie', 'Sports', 'Concert') NOT NULL,
    booking_id int(10),
    FOREIGN KEY (venue_id) REFERENCES Venue (venue_id)
);
```

3. Customer:

```
create table customer (
    customer_id int PRIMARY KEY AUTO_INCREMENT,
    email VARCHAR(20) UNIQUE NOT NULL,
    phone_number varchar(100) UNIQUE NOT NULL
);
```

4. Booking:

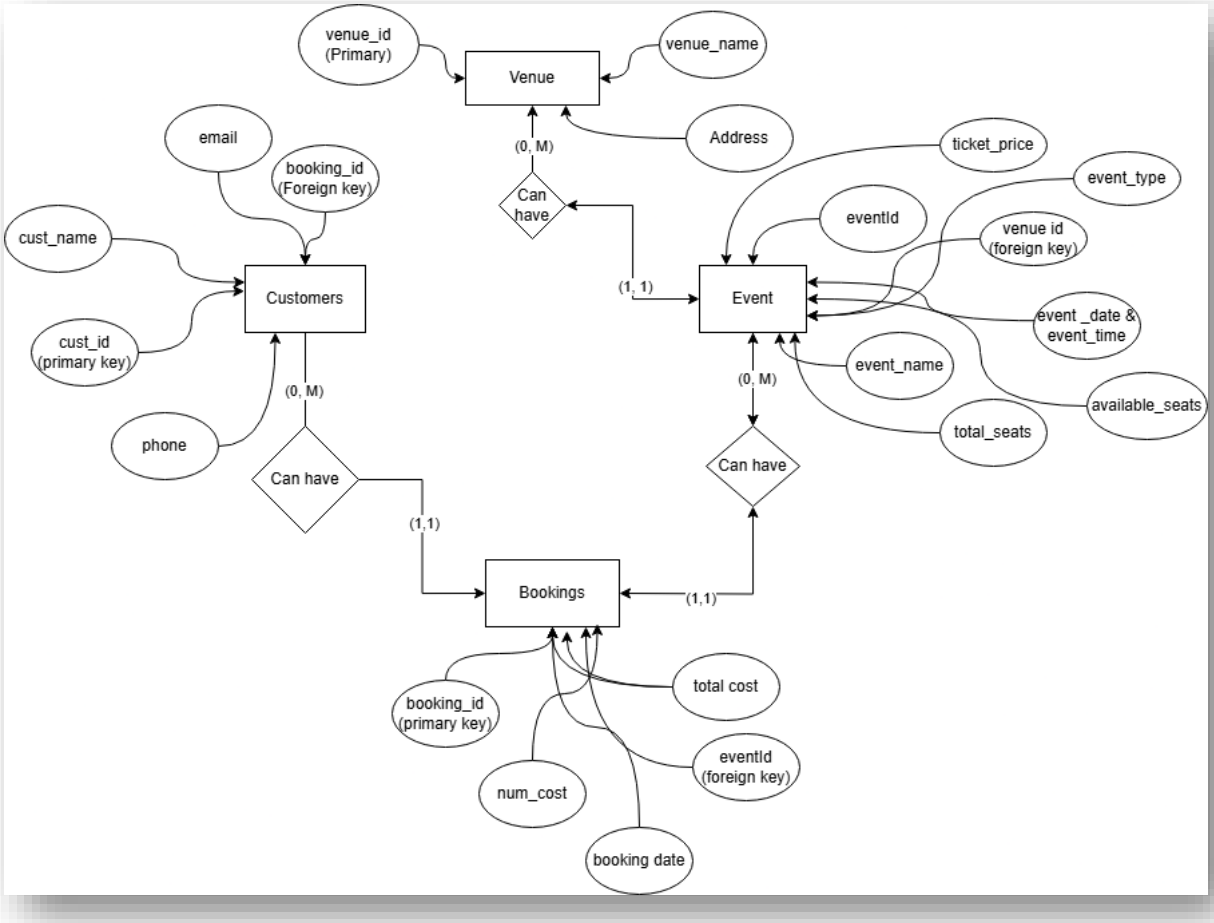
```
create table booking (
    booking_id int PRIMARY KEY AUTO_INCREMENT,
    customer_id int,
    event_id int,
    num_tickets int(50),
    total_cost int(50),
    booking_date DATE,
    FOREIGN KEY (customer_id) REFERENCES customer (customer_id),
    FOREIGN KEY (event_id) REFERENCES Event (event_id)
);
```

3. Creating **ER Diagram** to understand relationships between tables:

1. **Active elements of the entities of TicketBooking System**
    a. Venue
    b. Event
    c. Customers
    d. Booking

2. **Relationships:**
    a. **Venue ↔ Event:**
        i. (0,M) on Venue means a venue can have zero or many events.
        ii. (1,1) on Event means an event must have exactly one venue.
    b. **Event ↔ Booking:**
        i. (0,M) on Event means an event can have zero or many bookings.
        ii. (1,1) on Booking means a booking is for exactly one event.
    c. **Customer ↔ Booking:**
        i. (0,M) on Customer means a customer can have zero or many bookings.
        ii. (1,1) on Booking means each booking belongs to exactly one customer.

3. **Finding Primary Key:**
    a. Vid (VenueId)
    b. eventide
    c. customerId
    d. BookingId
4. **Other Attributes**
    a. Venue
        i. Vid (primary key)
        ii. venue_name,
        iii. address
    b. Event
        i. Eventid(primary key)
        ii. event_name,
        iii. event_date DATE,
        iv. event_time TIME,
        v. venue_id (Foreign Key),
        vi. total_seats,
        vii. available_seats,
        viii. ticket_price DECIMAL,
        ix. event_type ('Movie', 'Sports', 'Concert')
        x. booking_id (Foreign Key)
    c. Customer Table
        i. customer_id (Primary key)
        ii. customer_name,
        iii. email,
        iv. phone_number,
        v. booking_id (Foreign Key)
            a.
    d. Booking Table
        i. booking_id (Primary Key),
        ii. customer_id (Foreign Key),
        iii. event_id (Foreign Key),
        iv. num_tickets,
        v. total_cost,
        vi. booking_date,



ER Diagram TicketBookingSystem

4. Create appropriate Primary Key and Foreign Key constraints for referential integrity

```
Alter Table Event add CONSTRAINT FOREIGN KEY(booking_id) REFERENCES
booking(booking_id);
Alter table Customer add column customer_name varchar(255);
Alter table Customer add column booking_id int;
Alter table Customer add CONSTRAINT FOREIGN KEY(booking_id)
REFERENCES booking(booking_id);
```

Using alter command Added NOT NULL Constraint just not to have null values

```
ALTER TABLE Venue
MODIFY venue_name VARCHAR(25) NOT NULL,
MODIFY address VARCHAR(100) NOT NULL;

ALTER TABLE Event
MODIFY event_date DATE NOT NULL,
MODIFY event_time TIME NOT NULL,
MODIFY venue_id INT(10) NOT NULL,
```

```
MODIFY total_seats INT(100) NOT NULL,
MODIFY available_seats INT(100) NOT NULL,
MODIFY ticket_price DECIMAL NOT NULL,
MODIFY event_type ENUM('Movie', 'Sports', 'Concert') NOT NULL,
MODIFY booking_id INT(10) NOT NULL;

ALTER TABLE customer
MODIFY email VARCHAR(20) UNIQUE NOT NULL,
MODIFY phone_number INT(10) UNIQUE NOT NULL;

ALTER TABLE booking
MODIFY customer_id INT NOT NULL,
MODIFY event_id INT NOT NULL,
MODIFY num_tickets INT(50) NOT NULL,
MODIFY total_cost INT(50) NOT NULL,
MODIFY booking_date DATE NOT NULL;
```

----------Task 1 Completed----------

Task 2: : Select, Where, Between, AND, LIKE:

1. Insert 10 records in each table
    a. Venue

```
insert into Venue (venue_name, address) VALUES
("Raj Auditoriam", "123 MG Road Pune"),
("Vasudev Kala Mandir", "456 Sangli Road, Miraj"),
("Rangmanch", "Sector 12 Mumbai"),
("DY Patil Cricket Stadium", "Mumbai Pune Highway Pune"),
("Sangeet Bhavan", "88 Opera Street, Mumbai"),
("Lok Kala Kendra", "15 MG Road Kolhapur"),
("Shahu Event Place", "Shahupuri, Kolhapur"),
("Bhalchandra Talkies", "Chandni chouk, Kolhapur"),
("Shilp Sadan", "9 East end, Mumbai"),
("Nritya Kala Mandir", "25 Kothrud, Pune");
```

    b. Event

```
insert into Event (event_name, event_date, event_time, venue_id,
total_seats, available_seats, ticket_price, event_type) VALUES
('Bollywood Drama', '2025-05-01', '19:00:00', 1, 100, 80, 500.00,
'Concert'),
('Cricket Championship Cup', '2025-05-10', '14:30:00', 4, 200, 150,
2000.00, 'Sports'),
('Tollywood Blockbuster Premiere', '2025-06-05', '18:45:00', 3, 150, 100,
600.00, 'Movie'),
('Indian Classical Music Night', '2025-06-15', '20:00:00', 4, 120, 90,
1000.00, 'Concert'),
```

```
('Kabaddi League Opener', '2025-07-01', '16:00:00', 5, 180, 130, 650.00,
'Sports'),
('South Indian Film Festival', '2025-07-10', '15:30:00', 6, 160, 140,
1000.00, 'Movie'),
('Rock Concert India', '2025-08-01', '21:00:00', 7, 250, 200, 800.00,
'Concert'),
('Indian Premier League Cup', '2025-08-15', '17:00:00', 4, 300, 250,
1500.00, 'Sports'),
('Hindi Film Soiree', '2025-09-05', '19:30:00', 9, 140, 110, 700.00,
'Movie'),
('Fusion Music Fest', '2025-09-20', '20:30:00', 10, 220, 190, 850.00,
'Concert');
```

c. Customer

```
insert into customer(email, phone_number, customer_name, booking_id) VALUES
('arjun.patil@example.com', 9876543210, 'Arjun Patil', 0),
('shardul@example.com', 9876543211, 'Shardul', 0),
('ravi.kumar@example.com', 9876543212, 'Ravi Kumar', 0),
('neha.singh@example.com', 9876543213, 'Neha Singh', 0),
('nitishreddy@example.com', 9876543214, 'Nitish Reddy', 0),
('Rohit sharma', 9876543215, 'Rohit sharma', 0),
('rahuldravid@example.com', 9876543216, 'Rahul Dravid', 0),
('pooja.reddy@example.com', 9876543217, 'Pooja Reddy', 0),
('suresh.menon@example.com', 9876543218, 'Suresh Menon', 0),
('anjali.gupta@example.com', 9876543219, 'Anjali Gupta', 0);
```

(Initially keeping the booking_id 0 as initially customers will have 0 bookings, later adding booking_id of their first booking)

d. Booking

```
insert into booking (customer_id, event_id, num_tickets, total_cost,
booking_date)
VALUES
  (2, 8, 2, 3000, '2025-04-01'),
  (2, 2, 1, 2000, '2025-04-02'),
  (3, 3, 3, 1800, '2025-04-03'),
  (4, 4, 2, 2000, '2025-04-04'),
  (5, 5, 4, 2600, '2025-04-05'),
  (6, 6, 2, 2000, '2025-04-06'),
  (7, 7, 3, 2400, '2025-04-07'),
  (8, 8, 1, 1500, '2025-04-08'),
  (9, 9, 2, 1400, '2025-04-09'),
  (10,10, 4, 3400, '2025-04-10');
```

Note: While inserting the data in the tables, in the customer table there is foreign_key booking_id, even though a customer can have many bookings here while storing the booking_id I stored the first ever booking made by that customer in that section instead of adding all the bookings cause it will

make the data redundant and the schema provided requires a single booking_id reference in the Customer and Event tables

2. List all events from event table

```
select event_name from event;
```

```
mysql> select event_name, event_date from event;
+--------------------------------+------------+
| event_name                     | event_date |
+--------------------------------+------------+
| Bollywood Drama                | 2025-05-01 |
| Cricket Championship Cup       | 2025-05-10 |
| Tollywood Blockbuster Premiere | 2025-06-05 |
| Indian Classical Music Night   | 2025-06-15 |
| Kabaddi League Opener          | 2025-07-01 |
| South Indian Film Festival     | 2025-07-10 |
| Rock Concert India             | 2025-08-01 |
| Indian Premier League Cup      | 2025-08-15 |
| Hindi Film Soiree              | 2025-09-05 |
| Fusion Music Fest              | 2025-09-20 |
+--------------------------------+------------+
10 rows in set (0.00 sec)
```

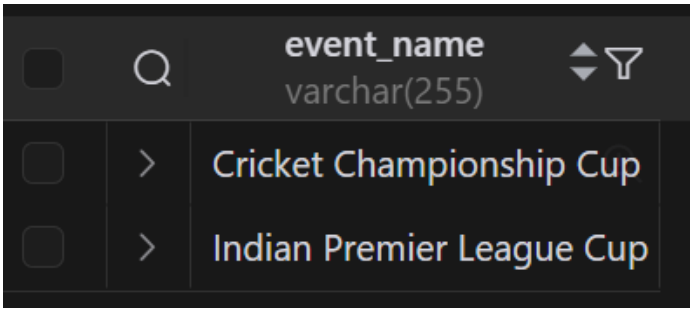
3. Write a SQL query to select events with available tickets

```
select event_name, event_date, event_time, available_seats from event
WHERE available_seats > 0;
```

event_name varchar(255)	* available_seats int
Bollywood Drama	80
Tollywood Blockbuster Prem	100
Indian Classical Music Nigh	90
Kabaddi League Opener	130
South Indian Film Festival	140
Rock Concert India	200
Hindi Film Soiree	110
Fusion Music Fest	190

4. Write a SQL query to select events name partial match with ‘cup’

```
select event_name from event WHERE event_name LIKE '%cup%';
```

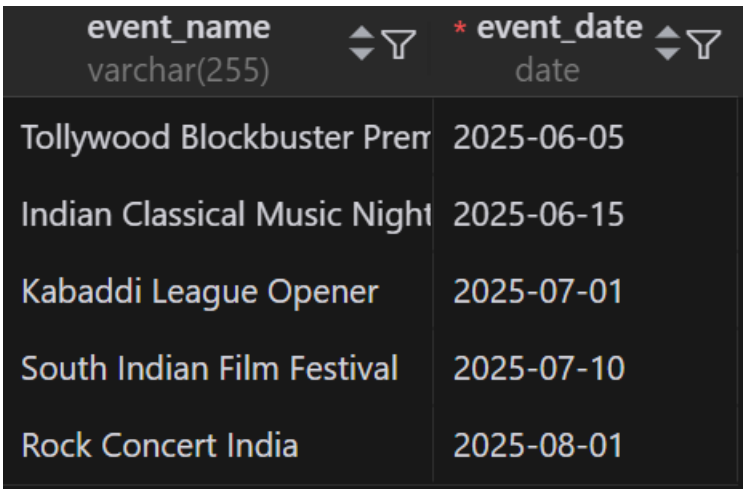


5. Events between the ticket price 1000 and 2500

```
select event_name, ticket_price from event
WHERE ticket_price BETWEEN 1000 AND 2500;
```

6. SQL query to retrieve events with dates falling within a specific range

```
select event_name, event_date from event WHERE event_date BETWEEN '2025-06-
01' AND '2025-08-01';
```



7. SQL query to retrieve events with available tickets that also have "Concert" in their name

event_name varchar(255)	* available_seats int	* event_type enum('Movie','Sports',
Bollywood Drama	80	Concert
Indian Classical Music Night	90	Concert
Rock Concert India	200	Concert
Fusion Music Fest	190	Concert

8. SQL query to retrieve users in batches of 5, starting from the 6th user

```
SELECT *
FROM customer
LIMIT 5 OFFSET 5;
```

* customer_id int	email varchar(200)	* phone_number varchar(100)	customer_name varchar(255)	booking_id int
6	Rohit sharma	9876543215	Rohit sharma	6
7	rahuldravid@example.com	9876543216	Rahul Dravid	7
8	pooja.reddy@example.com	9876543217	Pooja Reddy	8
9	suresh.menon@example.co	9876543218	Suresh Menon	9
10	anjali.gupta@example.com	9876543219	Anjali Gupta	10

9. SQL query to retrieve bookings details contains booked no of ticket more than 4.

Only Booking details:

```
1. select * from booking WHERE num_tickets > 4;
```

```
mysql> select * from booking where num_tickets>4;
+------------+-------------+----------+-------------+------------+--------------+
| booking_id | customer_id | event_id | num_tickets | total_cost | booking_date |
+------------+-------------+----------+-------------+------------+--------------+
|         21 |           2 |        8 |           6 |       3000 | 2025-03-02   |
|         22 |           1 |        2 |           5 |       3000 | 2025-03-02   |
+------------+-------------+----------+-------------+------------+--------------+
```

Booking details with customer name:

```
1. select booking.booking_id, customer.customer_name from booking JOIN customer ON
booking.customer_id = customer.customer_id where booking.num_tickets >= 4;
```

```
+------------+---------------+
| booking_id | customer_name |
+------------+---------------+
|          5 | Nitish Reddy  |
|         10 | Anjali Gupta  |
|         16 | Shardul       |
|         20 | Neha Singh    |
|         21 | Shardul       |
|         22 | Arjun Patil   |
+------------+---------------+
```

10. Write a SQL query to retrieve customer information whose phone number end with ‘000’

```
1. select customer_id, customer_name, phone_number, email from customer WHERE phone_number LIKE
"%000";
```

customer_id int	customer_name varchar(255)	phone_numbe varchar(100)	email varchar(200)
1	Arjun Patil	98765616000	arjun.patil@example.com
2	Shardul	9825334000	shardul@example.com
10	Anjali Gupta	8928982000	anjali.gupta@example.com

11. Write a SQL query to retrieve the events in order whose seat capacity more than 15000.

```
1.select * from event WHERE total_seats > 15000;
```

```
mysql> select * from event WHERE total_seats > 15000;
+----------+---------------------------+------------+------------+----------+-------------+-----------------+
| event_id | event_name                | event_date | event_time | venue_id | total_seats | available_seats |
+----------+---------------------------+------------+------------+----------+-------------+-----------------+
|        2 | Cricket Championship Cup  | 2025-05-10 | 14:30:00   |        4 |     2000000 |               0 |
|        8 | Indian Premier League Cup | 2025-08-15 | 17:00:00   |        4 |     3000000 |               0 |
+----------+---------------------------+------------+------------+----------+-------------+-----------------+
2 rows in set (0.01 sec)
```

12. Write a SQL query to select events name not start with ‘x’, ‘y’, ‘z’

```
select * from event WHERE event_name NOT LIKE "x%"
AND event_name NOT LIKE "y%"
AND event_name NOT LIKE "z%";
```

```
mysql> select * from event WHERE event_name NOT LIKE "x%"
    -> AND event_name NOT LIKE "y%"
    -> AND event_name NOT LIKE "z%";
+----------+-------------------------------+------------+------------+----------+-------------+-----------------+--------------+------------+
| event_id | event_name                    | event_date | event_time | venue_id | total_seats | available_seats | ticket_price | event_type |
+----------+-------------------------------+------------+------------+----------+-------------+-----------------+--------------+------------+
|        1 | Bollywood Drama               | 2025-05-01 | 19:00:00   |        1 |         100 |              80 |          500 | Concert    |
|        2 | Cricket Championship Cup      | 2025-05-10 | 14:30:00   |        4 |     2000000 |               0 |         2000 | Sports     |
|        3 | Tollywood Blockbuster Premiere | 2025-06-05 | 18:45:00   |        3 |         150 |             100 |          600 | Movie      |
|        4 | Indian Classical Music Night  | 2025-06-15 | 20:00:00   |        4 |         120 |              90 |         1000 | Concert    |
|        5 | Kabaddi League Opener         | 2025-07-01 | 16:00:00   |        5 |         180 |             130 |          650 | Sports     |
|        6 | South Indian Film Festival    | 2025-07-10 | 15:30:00   |        6 |         160 |             140 |         1000 | Movie      |
|        7 | Rock Concert India            | 2025-08-01 | 21:00:00   |        7 |         250 |             200 |          800 | Concert    |
|        8 | Indian Premier League Cup     | 2025-08-15 | 17:00:00   |        4 |     3000000 |               0 |         1500 | Sports     |
|        9 | Hindi Film Soiree             | 2025-09-05 | 19:30:00   |        9 |         140 |             110 |          700 | Movie      |
|       10 | Fusion Music Fest             | 2025-09-20 | 20:30:00   |       10 |         220 |             190 |          850 | Concert    |
+----------+-------------------------------+------------+------------+----------+-------------+-----------------+--------------+------------+
10 rows in set (0.01 sec)
```

----------Task 2 Completed----------

Task 3: Aggregate functions, Having, Order By, GroupBy and Joins:

1. Write a SQL query to List Events and Their Average Ticket Prices

```
select event_name, avg(ticket_price) from event
GROUP BY event_name;
```

event_name varchar(255)	avg(ticket_price)
Bollywood Drama	500.0000
Cricket Championship Cup	2000.0000
Tollywood Blockbuster Prem	600.0000
Indian Classical Music Night	1000.0000
Kabaddi League Opener	650.0000
South Indian Film Festival	1000.0000
Rock Concert India	800.0000
Indian Premier League Cup	1500.0000
Hindi Film Soiree	700.0000
Fusion Music Fest	850.0000

2. Write a SQL query to Calculate the Total Revenue Generated by Events

```
select event.event_name, SUM(booking.total_cost) as Total_Revenue from
booking JOIN event ON event.event_id = booking.event_id GROUP BY
event.event_name;
```

event_name varchar	Total_Revenue decimal
Bollywood Drama	1000
Cricket Championship Cup	7000
Tollywood Blockbuster Prem	3600
Indian Classical Music Night	4000
Kabaddi League Opener	5200
South Indian Film Festival	4000
Rock Concert India	2400
Indian Premier League Cup	10500
Hindi Film Soiree	2800
Fusion Music Fest	6800

3. Write a SQL query to find the event with the highest ticket sales.

```
select event.event_name, SUM(booking.num_tickets) as TotalTicketSales from
booking JOIN event ON booking.event_id = event.event_id GROUP BY
event.event_name ORDER BY TotalTicketSales DESC LIMIT 1;
```

event_name varchar	TotalTicketSales decimal
Indian Premier League Cup	11

4. Write a SQL query to Calculate the Total Number of Tickets Sold for Each Event.

```
select event.event_name, SUM(booking.num_tickets) as TotalTicketSales from
booking JOIN event ON booking.event_id = event.event_id GROUP BY
event.event_name;
```

event_name varchar	TotalTicketSales decimal
Bollywood Drama	2
Cricket Championship Cup	7
Tollywood Blockbuster Prem	6
Indian Classical Music Night	4
Kabaddi League Opener	8
South Indian Film Festival	4
Rock Concert India	3
Indian Premier League Cup	11
Hindi Film Soiree	4
Fusion Music Fest	8

5. Write a SQL query to Find Events with No Ticket Sales.

```
select event_name as eventWithNoSales FROM event where booking_id IS NULL
OR booking_id=0;
```

eventWithNoSales
PSL

6. Write a SQL query to Find the User Who Has Booked the Most Tickets

```
select customer.customer_name, SUM(booking.num_tickets) as
UserWithMostTickets from customer JOIN booking ON customer.customer_id =
booking.customer_id GROUP BY customer_name ORDER BY UserWithMostTickets
DESC LIMIT 1;
```

customer_name varchar	UserWithMostTickets decimal
Shardul	18

7. Write a SQL query to List Events and the total number of tickets sold for each month.

```
select event.event_name, DATE_FORMAT(booking.booking_date, '%Y-%m') as
month, SUM(booking.num_tickets) as total_tickets_sold from booking JOIN
event ON booking.event_id=event.event_id GROUP BY event.event_name,
DATE_FORMAT(booking.booking_date, '%Y-%m');
```

event_name varchar	month varchar	total_tickets_sold decimal
Bollywood Drama	2025-04	2
Cricket Championship Cup	2025-04	2
Cricket Championship Cup	2025-03	5
Tollywood Blockbuster Prem	2025-04	6
Indian Classical Music Night	2025-04	4
Kabaddi League Opener	2025-04	8
South Indian Film Festival	2025-04	4
Rock Concert India	2025-04	3
Indian Premier League Cup	2025-04	5
Indian Premier League Cup	2025-03	6
Hindi Film Soiree	2025-04	4
Fusion Music Fest	2025-04	8

8. Write a SQL query to calculate the average Ticket Price for Events in Each Venue

```
select venue.venue_name, AVG(event.ticket_price) from event JOIN venue
ON event.venue_id = venue.venue_id GROUP BY venue.venue_name,
event.event_name;
```

venue_name varchar	AVG(event.ticket_pric decimal
Raj Auditoriam	500.0000
DY Patil Cricket Stadium	2000.0000
Rangmanch	600.0000
DY Patil Cricket Stadium	1000.0000
Sangeet Bhavan	650.0000
Lok Kala Kendra	1000.0000
Shahu Event Place	800.0000
DY Patil Cricket Stadium	1500.0000
Shilp Sadan	700.0000
Nritya Kala Mandir	850.0000
Bhalchandra Talkies	500.0000

9. Write a SQL query to calculate the total Number of Tickets Sold for Each Event Type.

```
select event.event_type, SUM(booking.num_tickets) as TicketsSold from
booking JOIN event ON
booking.event_id = event.event_id GROUP BY event.event_type;
```

event_type string	TicketsSold decimal
Sports	26
Movie	14
Concert	17

10. Write a SQL query to calculate the total Revenue Generated by Events in Each Year.

```
select DATE_FORMAT(booking.booking_date, "%Y"), SUM(booking.total_cost) as
totalRevenue from booking JOIN event ON
booking.event_id = event.event_id GROUP BY
event.event_name,  DATE_FORMAT(booking.booking_date, "%Y");
```

event_name varchar	DATE_FORMAT(booki varchar	totalRevenue decimal
Bollywood Drama	2025	1000
Cricket Championship Cup	2025	7000
Tollywood Blockbuster Prem	2025	3600
Indian Classical Music Night	2025	4000
Kabaddi League Opener	2025	5200
South Indian Film Festival	2025	4000
Rock Concert India	2025	2400
Indian Premier League Cup	2025	10500
Hindi Film Soiree	2025	2800
Fusion Music Fest	2025	6800

11. Write a SQL query to list users who have booked tickets for multiple events.

```
select customer.customer_name, booking.customer_id from booking JOIN
customer ON booking.customer_id = customer.customer_id
GROUP BY customer.customer_name, booking.customer_id HAVING COUNT(DISTINCT
booking.event_id) > 1;
```

customer_name varchar	customer_id int
Arjun Patil	1
Neha Singh	4
Ravi Kumar	3
Shardul	2

12. Write a SQL query to calculate the Total Revenue Generated by Events for Each User.

```
select customer.customer_name, SUM(booking.total_cost) from booking
JOIN customer ON booking.customer_id = customer.customer_id GROUP BY
customer.customer_name;
```

customer_name varchar	SUM(booking.total_cc decimal
Arjun Patil	9300
Shardul	16100
Ravi Kumar	3200
Neha Singh	5400
Nitish Reddy	2600
Rohit sharma	2000
Rahul Dravid	2400
Pooja Reddy	1500
Suresh Menon	1400
Anjali Gupta	3400

13. Write a SQL query to calculate the Average Ticket Price for Events in Each Category and Venue.

```
select event.event_type, venue.venue_name, AVG(event.ticket_price) from
event JOIN venue ON event.venue_id = venue.venue_id GROUP BY
event.event_type, venue.venue_name;
```

event_type string	venue_name varchar	AVG(event.ticket_pric decimal
Concert	Raj Auditoriam	500.0000
Sports	DY Patil Cricket Stadium	1750.0000
Movie	Rangmanch	600.0000
Concert	DY Patil Cricket Stadium	1000.0000
Sports	Sangeet Bhavan	650.0000
Movie	Lok Kala Kendra	1000.0000
Concert	Shahu Event Place	800.0000
Movie	Shilp Sadan	700.0000
Concert	Nritya Kala Mandir	850.0000
Sports	Bhalchandra Talkies	500.0000

14. Write a SQL query to list Users and the Total Number of Tickets They've Purchased in the Last 30 Days.

```
select customer.customer_name, SUM(booking.num_tickets) as totalTickets
from booking JOIN customer ON booking.customer_id = customer.customer_id
WHERE
booking.booking_date >= ADDDATE(CURDATE(), INTERVAL -30 DAY) GROUP BY
customer.customer_name;
```

customer_name (varchar)	totalTickets (decimal)
Shardul	18
Ravi Kumar	5
Neha Singh	6
Nitish Reddy	4
Rohit sharma	2
Rahul Dravid	3
Pooja Reddy	1
Suresh Menon	2
Anjali Gupta	4
Arjun Patil	12

----------Task 3 completed----------

TASK 4: Subquery and its types

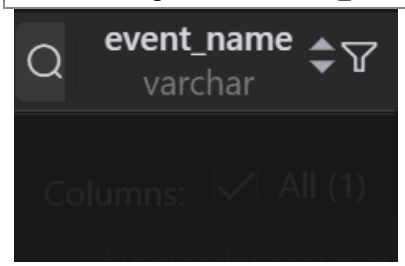
1. Calculate the Average Ticket Price for Events in Each Venue Using a Subquery.

```
select venue.venue_name, (select AVG(event.ticket_price) from event where
event.venue_id = venue.venue_id) as TotalAvg from venue;
```

venue_name varchar	TotalAvg decimal
Raj Auditoriam	500.0000
Vasudev Kala Mandir	(NULL)
Rangmanch	600.0000
DY Patil Cricket Stadium	1500.0000
Sangeet Bhavan	650.0000
Lok Kala Kendra	1000.0000
Shahu Event Place	800.0000
Bhalchandra Talkies	500.0000
Shilp Sadan	700.0000
Nritya Kala Mandir	850.0000

2. Find Events with More Than 50% of Tickets Sold using subquery.

```
select e.event_name from event e WHERE (select SUM(b.num_tickets)
from booking WHERE b.event_id = e.event_id) > (0.5 * e.total_seats);
```



3. Calculate the Total Number of Tickets Sold for Each Event.

```
Select event.event_name, (Select SUM(booking.num_tickets) from booking WHERE booking.event_id =
event.event_id) as TotalTicketSales from event
```

event_name varchar	TotalTicketSales decimal
Bollywood Drama	2
Cricket Championship Cup	7
Tollywood Blockbuster Prem	6
Indian Classical Music Night	4
Kabaddi League Opener	8
South Indian Film Festival	4
Rock Concert India	3
Indian Premier League Cup	11
Hindi Film Soiree	4
Fusion Music Fest	8
PSL	(NULL)

4. Find Users Who Have Not Booked Any Tickets Using a NOT EXISTS Subquery.

```
Select customer.customer_name from customer WHERE NOT EXISTS(Select booking.booking_id from
booking where booking.customer_id = customer.customer_id);
```

customer_name varchar
Ravi Ashwin

5. List Events with No Ticket Sales Using a NOT IN Subquery.

```
select event.event_id, event.event_name from event WHERE event.event_id NOT IN(select
booking.event_id from booking);
```

event_id int	event_name varchar
11	PSL

Columns: All (2)

6. Calculate the Total Number of Tickets Sold for Each Event Type Using a Subquery in the FROM Clause.

```
Select t.event_type, SUM(t.tickets_sold) AS total_tickets_sold FROM (
    Select event.event_type, SUM(booking.num_tickets) AS tickets_sold from event LEFT JOIN
booking ON event.event_id = booking.event_id GROUP BY event.event_id, event.event_type
) t GROUP BY t.event_type;
```

event_name varchar	ticket_price decimal
Cricket Championship Cup	2000
Indian Classical Music Night	1000
South Indian Film Festival	1000
Indian Premier League Cup	1500

7. Find Events with Ticket Prices Higher Than the Average Ticket Price Using a Subquery in the WHERE Clause.

```
Select event.event_name, event.ticket_price from event WHERE ticket_price > (select
AVG(event.ticket_price) from event);
```

event_name varchar	ticket_price decimal
Cricket Championship Cup	2000
Indian Classical Music Night	1000
South Indian Film Festival	1000
Indian Premier League Cup	1500

8. Calculate the Total Revenue Generated by Events for Each User Using a Correlated Subquery.

```
Select customer.customer_name, (Select SUM(booking.total_cost) FROM booking where
booking.customer_id = customer.customer_id) as total_Revenue from Customer;
```

customer_name varchar	total_Revenue decimal
Arjun Patil	9300
Shardul	16100
Ravi Kumar	3200
Neha Singh	5400
Nitish Reddy	2600
Rohit sharma	2000
Rahul Dravid	2400
Pooja Reddy	1500
Suresh Menon	1400
Anjali Gupta	3400
Ravi Ashwin	(NULL)

9. List Users Who Have Booked Tickets for Events in a Given Venue Using a Subquery in the WHERE Clause.

```
Select customer.customer_name from customer WHERE customer.customer_id IN (select
booking.customer_id from booking JOIN event ON booking.event_id = event.event_id WHERE
event.venue_id=3);
```

customer_name varchar
Ravi Kumar
Arjun Patil

10. Calculate the Total Number of Tickets Sold for Each Event Category Using a Subquery with GROUP BY.

```
1. Select event.event_type, SUM(booking.num_tickets) as totalTicektsSold from booking JOIN event
ON booking.event_id = event.event_id GROUP BY event.event_type;
```

event_type string	totalTicektsSold decimal
Sports	26
Movie	14
Concert	17

11. Find Users Who Have Booked Tickets for Events in each Month Using a Subquery with DATE_FORMAT.

```
1. Select customer.customer_name, DATE_FORMAT(booking.booking_date, "%Y - %M") as Month,
SUM(booking.num_tickets) as total_tickets  from booking JOIN customer ON booking.customer_id =
customer.customer_id GROUP BY customer.customer_id, Month
```

customer_name varchar	Month varchar	total_tickets decimal
Shardul	2025 - April	12
Shardul	2025 - March	6
Ravi Kumar	2025 - April	5
Neha Singh	2025 - April	6
Nitish Reddy	2025 - April	4
Rohit sharma	2025 - April	2
Rahul Dravid	2025 - April	3
Pooja Reddy	2025 - April	1
Suresh Menon	2025 - April	2
Anjali Gupta	2025 - April	4

12. Calculate the Average Ticket Price for Events in Each Venue Using a Subquery

```
Select venue.venue_name, (Select AVG(event.ticket_price) from event WHERE event.venue_id =
venue.venue_id) from venue;
```

venue_name varchar	(Select AVG(event.tick decimal
Raj Auditoriam	500.0000
Vasudev Kala Mandir	(NULL)
Rangmanch	600.0000
DY Patil Cricket Stadium	1500.0000
Sangeet Bhavan	650.0000
Lok Kala Kendra	1000.0000
Shahu Event Place	800.0000
Bhalchandra Talkies	500.0000
Shilp Sadan	700.0000
Nritya Kala Mandir	850.0000