Assignment 5

Create Database

```
mysql> show databases;
+--------------------------+
| Database                 |
+--------------------------+
| bookingsystem            |
| couriermanagementsystem  |
| hmbank                   |
| information_schema       |
| mysql                    |
| performance_schema       |
| petpals                  |
| sakila                   |
| sisdb                    |
| subquery                 |
| sys                      |
| techshop                 |
| ticketbookingsystem      |
| world                    |
+--------------------------+
14 rows in set (0.00 sec)

mysql> use BookingSystem;
Database changed
mysql>
```

Create Tables

- Venue
- Event
- Customer
- Booking

```
mysql> CREATE TABLE Venue (
    ->     venue_id INT PRIMARY KEY,
    ->     venue_name VARCHAR(100),
    ->     address VARCHAR(255)
    -> );
Query OK, 0 rows affected (0.04 sec)

mysql> desc Venue;
+------------+--------------+------+-----+---------+-------+
| Field      | Type         | Null | Key | Default | Extra |
+------------+--------------+------+-----+---------+-------+
| venue_id   | int          | NO   | PRI | NULL    |       |
| venue_name | varchar(100) | YES  |     | NULL    |       |
| address    | varchar(255) | YES  |     | NULL    |       |
+------------+--------------+------+-----+---------+-------+
3 rows in set (0.00 sec)

mysql>
```

```
mysql> CREATE TABLE Event (
    ->     event_id INT PRIMARY KEY,
    ->     event_name VARCHAR(100),
    ->     event_date DATE,
    ->     event_time TIME,
    ->     venue_id INT,
    ->     total_seats INT,
    ->     available_seats INT,
    ->     ticket_price DECIMAL(10, 2),
    ->     event_type ENUM('Movie', 'Sports', 'Concert'),
    ->     booking_id INT,
    ->     FOREIGN KEY (venue_id) REFERENCES Venue(venue_id)
    -> );
Query OK, 0 rows affected (0.05 sec)

mysql> desc Event;
+----------------+---------------------------------+------+-----+---------+-------+
| Field          | Type                            | Null | Key | Default | Extra |
+----------------+---------------------------------+------+-----+---------+-------+
| event_id       | int                             | NO   | PRI | NULL    |       |
| event_name     | varchar(100)                    | YES  |     | NULL    |       |
| event_date     | date                            | YES  |     | NULL    |       |
| event_time     | time                            | YES  |     | NULL    |       |
| venue_id       | int                             | YES  | MUL | NULL    |       |
| total_seats    | int                             | YES  |     | NULL    |       |
| available_seats | int                            | YES  |     | NULL    |       |
| ticket_price   | decimal(10,2)                   | YES  |     | NULL    |       |
| event_type     | enum('Movie','Sports','Concert') | YES  |     | NULL    |       |
| booking_id     | int                             | YES  |     | NULL    |       |
+----------------+---------------------------------+------+-----+---------+-------+
10 rows in set (0.00 sec)

mysql>
```

```
mysql> CREATE TABLE Customer (
    ->     customer_id INT PRIMARY KEY,
    ->     customer_name VARCHAR(100),
    ->     email VARCHAR(100),
    ->     phone_number VARCHAR(15),
    ->     booking_id INT);
Query OK, 0 rows affected (0.03 sec)

mysql> desc Customer;
+---------------+--------------+------+-----+---------+-------+
| Field         | Type         | Null | Key | Default | Extra |
+---------------+--------------+------+-----+---------+-------+
| customer_id   | int          | NO   | PRI | NULL    |       |
| customer_name | varchar(100) | YES  |     | NULL    |       |
| email         | varchar(100) | YES  |     | NULL    |       |
| phone_number  | varchar(15)  | YES  |     | NULL    |       |
| booking_id    | int          | YES  |     | NULL    |       |
+---------------+--------------+------+-----+---------+-------+
5 rows in set (0.00 sec)

mysql>
```

```
mysql> CREATE TABLE Booking (
    ->     booking_id INT PRIMARY KEY,
    ->     customer_id INT,
    ->     event_id INT,
    ->     num_tickets INT,
    ->     total_cost DECIMAL(10, 2),
    ->     booking_date DATETIME,
    ->     FOREIGN KEY (customer_id) REFERENCES Customer(customer_id),
    ->     FOREIGN KEY (event_id) REFERENCES Event(event_id)
    -> );
Query OK, 0 rows affected (0.05 sec)

mysql> desc Booking;
+--------------+---------------+------+-----+---------+-------+
| Field        | Type          | Null | Key | Default | Extra |
+--------------+---------------+------+-----+---------+-------+
| booking_id   | int           | NO   | PRI | NULL    |       |
| customer_id  | int           | YES  | MUL | NULL    |       |
| event_id     | int           | YES  | MUL | NULL    |       |
| num_tickets  | int           | YES  |     | NULL    |       |
| total_cost   | decimal(10,2) | YES  |     | NULL    |       |
| booking_date | datetime      | YES  |     | NULL    |       |
+--------------+---------------+------+-----+---------+-------+
6 rows in set (0.00 sec)

mysql>
```

Insert at least 10 values in each of the table

```
mysql> INSERT INTO Venue (venue_id, venue_name, address)
    -> VALUES
    ->     (1, 'Example Venue 1', '123 Main Street, City1, Country1'),
    ->     (2, 'Example Venue 2', '456 Park Avenue, City2, Country2'),
    ->     (3, 'Example Venue 3', '789 Oak Lane, City3, Country3'),
    ->     (4, 'Example Venue 4', '101 Pine Road, City4, Country4'),
    ->     (5, 'Example Venue 5', '202 Cedar Street, City5, Country5'),
    ->     (6, 'Example Venue 6', '303 Elm Avenue, City6, Country6'),
    ->     (7, 'Example Venue 7', '404 Birch Lane, City7, Country7'),
    ->     (8, 'Example Venue 8', '505 Maple Road, City8, Country8'),
    ->     (9, 'Example Venue 9', '606 Walnut Street, City9, Country9'),
    ->     (10, 'Example Venue 10', '707 Spruce Avenue, City10, Country10');
Query OK, 10 rows affected (0.01 sec)
Records: 10  Duplicates: 0  Warnings: 0

mysql> select * from Venue;
+----------+------------------+--------------------------------------+
| venue_id | venue_name       | address                              |
+----------+------------------+--------------------------------------+
|        1 | Example Venue 1  | 123 Main Street, City1, Country1     |
|        2 | Example Venue 2  | 456 Park Avenue, City2, Country2     |
|        3 | Example Venue 3  | 789 Oak Lane, City3, Country3        |
|        4 | Example Venue 4  | 101 Pine Road, City4, Country4       |
|        5 | Example Venue 5  | 202 Cedar Street, City5, Country5    |
|        6 | Example Venue 6  | 303 Elm Avenue, City6, Country6      |
|        7 | Example Venue 7  | 404 Birch Lane, City7, Country7      |
|        8 | Example Venue 8  | 505 Maple Road, City8, Country8      |
|        9 | Example Venue 9  | 606 Walnut Street, City9, Country9   |
|       10 | Example Venue 10 | 707 Spruce Avenue, City10, Country10 |
+----------+------------------+--------------------------------------+
10 rows in set (0.00 sec)

mysql>
```

```
mysql> INSERT INTO Event (event_id, event_name, event_date, event_time, venue_id, total_seats, available_seats, ticket_price, event_type, booking_id)
    -> VALUES
    ->     (1, 'Concert A', '2023-12-15', '19:00:00', 1, 500, 500, 50.00, 'Concert', 1),
    ->     (2, 'Movie Night', '2023-12-18', '20:30:00', 2, 300, 300, 10.00, 'Movie', 2),
    ->     (3, 'Sports Tournament', '2023-12-20', '14:00:00', 3, 1000, 1000, 20.00, 'Sports', 3),
    ->     (4, 'Concert B', '2023-12-22', '18:30:00', 4, 700, 700, 40.00, 'Concert', 4),
    ->     (5, 'Conference', '2023-12-25', '09:00:00', 5, 200, 200, 30.00, 'Movie', 5),
    ->     (6, 'Movie Premiere', '2023-12-28', '19:45:00', 6, 400, 400, 15.00, 'Movie', 6),
    ->     (7, 'Concert Z', '2023-12-30', '21:00:00', 7, 600, 600, 25.00, 'Concert', 7),
    ->     (8, 'Sports Game', '2024-01-02', '15:30:00', 8, 800, 800, 35.00, 'Sports', 8),
    ->     (9, 'Concert C', '2024-01-05', '20:00:00', 9, 900, 900, 45.00, 'Concert', 9),
    ->     (10, 'Sports Show', '2024-01-08', '12:00:00', 10, 150, 150, 5.00, 'Sports', 10);
Query OK, 10 rows affected (0.01 sec)
Records: 10  Duplicates: 0  Warnings: 0

mysql> select * from Event;
+----------+------------------+------------+------------+----------+-------------+-----------------+--------------+------------+------------+
| event_id | event_name       | event_date | event_time | venue_id | total_seats | available_seats | ticket_price | event_type | booking_id |
+----------+------------------+------------+------------+----------+-------------+-----------------+--------------+------------+------------+
|        1 | Concert A        | 2023-12-15 | 19:00:00   |        1 |         500 |             500 |        50.00 | Concert    |          1 |
|        2 | Movie Night      | 2023-12-18 | 20:30:00   |        2 |         300 |             300 |        10.00 | Movie      |          2 |
|        3 | Sports Tournament | 2023-12-20 | 14:00:00   |        3 |        1000 |            1000 |        20.00 | Sports     |          3 |
|        4 | Concert B        | 2023-12-22 | 18:30:00   |        4 |         700 |             700 |        40.00 | Concert    |          4 |
|        5 | Conference       | 2023-12-25 | 09:00:00   |        5 |         200 |             200 |        30.00 | Movie      |          5 |
|        6 | Movie Premiere   | 2023-12-28 | 19:45:00   |        6 |         400 |             400 |        15.00 | Movie      |          6 |
|        7 | Concert Z        | 2023-12-30 | 21:00:00   |        7 |         600 |             600 |        25.00 | Concert    |          7 |
|        8 | Sports Game      | 2024-01-02 | 15:30:00   |        8 |         800 |             800 |        35.00 | Sports     |          8 |
|        9 | Concert C        | 2024-01-05 | 20:00:00   |        9 |         900 |             900 |        45.00 | Concert    |          9 |
|       10 | Sports Show      | 2024-01-08 | 12:00:00   |       10 |         150 |             150 |         5.00 | Sports     |         10 |
+----------+------------------+------------+------------+----------+-------------+-----------------+--------------+------------+------------+
10 rows in set (0.00 sec)

mysql>
```

```
mysql> INSERT INTO Customer (customer_id, customer_name, email, phone_number, booking_id)
    -> VALUES
    ->     (1, 'John Doe', 'john.doe@email.com', '123-456-7890', 1),
    ->     (2, 'Alice Smith', 'alice.smith@email.com', '234-567-8901', 2),
    ->     (3, 'Bob Johnson', 'bob.johnson@email.com', '345-678-9012', 3),
    ->     (4, 'Emily Davis', 'emily.davis@email.com', '456-789-0123', 4),
    ->     (5, 'Michael Wilson', 'michael.wilson@email.com', '567-890-1234', 5),
    ->     (6, 'Sophia Brown', 'sophia.brown@email.com', '678-901-2345', 6),
    ->     (7, 'Daniel Lee', 'daniel.lee@email.com', '789-012-3456', 7),
    ->     (8, 'Olivia Turner', 'olivia.turner@email.com', '890-123-4567', 8),
    ->     (9, 'Liam Miller', 'liam.miller@email.com', '901-234-5678', 9),
    ->     (10, 'Emma Harris', 'emma.harris@email.com', '012-345-6789', 10);
Query OK, 10 rows affected (0.01 sec)
Records: 10  Duplicates: 0  Warnings: 0

mysql> select * from Customer;
+-------------+---------------+--------------------------+--------------+------------+
| customer_id | customer_name | email                    | phone_number | booking_id |
+-------------+---------------+--------------------------+--------------+------------+
|           1 | John Doe      | john.doe@email.com       | 123-456-7890 |          1 |
|           2 | Alice Smith   | alice.smith@email.com    | 234-567-8901 |          2 |
|           3 | Bob Johnson   | bob.johnson@email.com    | 345-678-9012 |          3 |
|           4 | Emily Davis   | emily.davis@email.com    | 456-789-0123 |          4 |
|           5 | Michael Wilson | michael.wilson@email.com | 567-890-1234 |          5 |
|           6 | Sophia Brown  | sophia.brown@email.com   | 678-901-2345 |          6 |
|           7 | Daniel Lee    | daniel.lee@email.com     | 789-012-3456 |          7 |
|           8 | Olivia Turner | olivia.turner@email.com  | 890-123-4567 |          8 |
|           9 | Liam Miller   | liam.miller@email.com    | 901-234-5678 |          9 |
|          10 | Emma Harris   | emma.harris@email.com    | 012-345-6789 |         10 |
+-------------+---------------+--------------------------+--------------+------------+
10 rows in set (0.00 sec)

mysql>
```

```
mysql> INSERT INTO Booking (booking_id, customer_id, event_id, num_tickets, total_cost, booking_date)
    -> VALUES
    ->     (1, 1, 1, 2, 100.00, '2023-12-10 15:30:00'),
    ->     (2, 2, 2, 3, 30.00, '2023-12-11 18:45:00'),
    ->     (3, 3, 3, 5, 100.00, '2023-12-12 12:00:00'),
    ->     (4, 4, 4, 1, 40.00, '2023-12-13 20:15:00'),
    ->     (5, 5, 5, 4, 120.00, '2023-12-14 14:30:00'),
    ->     (6, 6, 6, 2, 30.00, '2023-12-15 17:45:00'),
    ->     (7, 7, 7, 3, 75.00, '2023-12-16 19:00:00'),
    ->     (8, 8, 8, 6, 210.00, '2023-12-17 21:15:00'),
    ->     (9, 9, 9, 1, 45.00, '2023-12-18 11:30:00'),
    ->     (10, 10, 10, 2, 10.00, '2023-12-19 13:45:00');
Query OK, 10 rows affected (0.01 sec)
Records: 10  Duplicates: 0  Warnings: 0

mysql> select * from Booking;
+------------+-------------+----------+-------------+------------+---------------------+
| booking_id | customer_id | event_id | num_tickets | total_cost | booking_date        |
+------------+-------------+----------+-------------+------------+---------------------+
|          1 |           1 |        1 |           2 |     100.00 | 2023-12-10 15:30:00 |
|          2 |           2 |        2 |           3 |      30.00 | 2023-12-11 18:45:00 |
|          3 |           3 |        3 |           5 |     100.00 | 2023-12-12 12:00:00 |
|          4 |           4 |        4 |           1 |      40.00 | 2023-12-13 20:15:00 |
|          5 |           5 |        5 |           4 |     120.00 | 2023-12-14 14:30:00 |
|          6 |           6 |        6 |           2 |      30.00 | 2023-12-15 17:45:00 |
|          7 |           7 |        7 |           3 |      75.00 | 2023-12-16 19:00:00 |
|          8 |           8 |        8 |           6 |     210.00 | 2023-12-17 21:15:00 |
|          9 |           9 |        9 |           1 |      45.00 | 2023-12-18 11:30:00 |
|         10 |          10 |       10 |           2 |      10.00 | 2023-12-19 13:45:00 |
+------------+-------------+----------+-------------+------------+---------------------+
10 rows in set (0.00 sec)

mysql>
```

```
mysql> ALTER TABLE Event
    -> ADD FOREIGN KEY (booking_id) REFERENCES Booking(booking_id);
Query OK, 0 rows affected (0.07 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> ALTER TABLE Customer
    -> ADD FOREIGN KEY (booking_id) REFERENCES Booking(booking_id);
Query OK, 0 rows affected (0.06 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql>
```

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

Write a SQL query to list all Events.

```
mysql> SELECT * FROM Event;
+----------+------------------+------------+------------+----------+-------------+-----------------+--------------+------------+------------+
| event_id | event_name       | event_date | event_time | venue_id | total_seats | available_seats | ticket_price | event_type | booking_id |
+----------+------------------+------------+------------+----------+-------------+-----------------+--------------+------------+------------+
|        1 | Concert A        | 2023-12-15 | 19:00:00   |        1 |         500 |             500 |        50.00 | Concert    |          1 |
|        2 | Movie Night      | 2023-12-18 | 20:30:00   |        2 |         300 |             300 |        10.00 | Movie      |          2 |
|        3 | Sports Tournament | 2023-12-20 | 14:00:00   |        3 |        1000 |            1000 |        20.00 | Sports     |          3 |
|        4 | Concert B        | 2023-12-22 | 18:30:00   |        4 |         700 |             700 |        40.00 | Concert    |          4 |
|        5 | Conference       | 2023-12-25 | 09:00:00   |        5 |         200 |             200 |        30.00 | Movie      |          5 |
|        6 | Movie Premiere   | 2023-12-28 | 19:45:00   |        6 |         400 |             400 |        15.00 | Movie      |          6 |
|        7 | Concert Z        | 2023-12-30 | 21:00:00   |        7 |         600 |             600 |        25.00 | Concert    |          7 |
|        8 | Sports Game      | 2024-01-02 | 15:30:00   |        8 |         800 |             800 |        35.00 | Sports     |          8 |
|        9 | Concert C        | 2024-01-05 | 20:00:00   |        9 |         900 |             900 |        45.00 | Concert    |          9 |
|       10 | Sports Show      | 2024-01-08 | 12:00:00   |       10 |         150 |             150 |         5.00 | Sports     |         10 |
+----------+------------------+------------+------------+----------+-------------+-----------------+--------------+------------+------------+
10 rows in set (0.00 sec)

mysql>
```

Write a SQL query to select events with available tickets

```
mysql> SELECT * FROM Event
    -> WHERE available_seats > 0;
+----------+------------------+------------+------------+----------+-------------+-----------------+--------------+------------+------------+
| event_id | event_name       | event_date | event_time | venue_id | total_seats | available_seats | ticket_price | event_type | booking_id |
+----------+------------------+------------+------------+----------+-------------+-----------------+--------------+------------+------------+
|        1 | Concert A        | 2023-12-15 | 19:00:00   |        1 |         500 |             500 |        50.00 | Concert    |          1 |
|        2 | Movie Night      | 2023-12-18 | 20:30:00   |        2 |         300 |             300 |        10.00 | Movie      |          2 |
|        3 | Sports Tournament | 2023-12-20 | 14:00:00   |        3 |        1000 |            1000 |        20.00 | Sports     |          3 |
|        4 | Concert B        | 2023-12-22 | 18:30:00   |        4 |         700 |             700 |        40.00 | Concert    |          4 |
|        5 | Conference       | 2023-12-25 | 09:00:00   |        5 |         200 |             200 |        30.00 | Movie      |          5 |
|        6 | Movie Premiere   | 2023-12-28 | 19:45:00   |        6 |         400 |             400 |        15.00 | Movie      |          6 |
|        7 | Concert Z        | 2023-12-30 | 21:00:00   |        7 |         600 |             600 |        25.00 | Concert    |          7 |
|        8 | Sports Game      | 2024-01-02 | 15:30:00   |        8 |         800 |             800 |        35.00 | Sports     |          8 |
|        9 | Concert C        | 2024-01-05 | 20:00:00   |        9 |         900 |             900 |        45.00 | Concert    |          9 |
|       10 | Sports Show      | 2024-01-08 | 12:00:00   |       10 |         150 |             150 |         5.00 | Sports     |         10 |
+----------+------------------+------------+------------+----------+-------------+-----------------+--------------+------------+------------+
10 rows in set (0.00 sec)

mysql>
```

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

```
mysql> SELECT * FROM Event
    -> WHERE event_name LIKE '%Sports%';
+----------+------------------+------------+------------+----------+-------------+-----------------+--------------+------------+------------+
| event_id | event_name       | event_date | event_time | venue_id | total_seats | available_seats | ticket_price | event_type | booking_id |
+----------+------------------+------------+------------+----------+-------------+-----------------+--------------+------------+------------+
|        3 | Sports Tournament | 2023-12-20 | 14:00:00   |        3 |        1000 |            1000 |        20.00 | Sports     |          3 |
|        8 | Sports Game      | 2024-01-02 | 15:30:00   |        8 |         800 |             800 |        35.00 | Sports     |          8 |
|       10 | Sports Show      | 2024-01-08 | 12:00:00   |       10 |         150 |             150 |         5.00 | Sports     |         10 |
+----------+------------------+------------+------------+----------+-------------+-----------------+--------------+------------+------------+
3 rows in set (0.00 sec)

mysql>
```

e a SQL query to select events with ticket price range is between 20 to 40

```
mysql> SELECT * FROM Event
    -> WHERE ticket_price BETWEEN 20 AND 40;
+----------+------------------+------------+------------+----------+-------------+-----------------+--------------+------------+------------+
| event_id | event_name       | event_date | event_time | venue_id | total_seats | available_seats | ticket_price | event_type | booking_id |
+----------+------------------+------------+------------+----------+-------------+-----------------+--------------+------------+------------+
|        3 | Sports Tournament | 2023-12-20 | 14:00:00   |        3 |        1000 |            1000 |        20.00 | Sports     |          3 |
|        4 | Concert B        | 2023-12-22 | 18:30:00   |        4 |         700 |             700 |        40.00 | Concert    |          4 |
|        5 | Conference       | 2023-12-25 | 09:00:00   |        5 |         200 |             200 |        30.00 | Movie      |          5 |
|        7 | Concert Z        | 2023-12-30 | 21:00:00   |        7 |         600 |             600 |        25.00 | Concert    |          7 |
|        8 | Sports Game      | 2024-01-02 | 15:30:00   |        8 |         800 |             800 |        35.00 | Sports     |          8 |
+----------+------------------+------------+------------+----------+-------------+-----------------+--------------+------------+------------+
5 rows in set (0.00 sec)

mysql>
```

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

```
mysql> SELECT * FROM Event
    -> WHERE event_date BETWEEN '2023-01-01' AND '2023-12-31';
+----------+------------------+------------+------------+----------+-------------+----------------+--------------+------------+------------+
| event_id | event_name       | event_date | event_time | venue_id | total_seats | available_seats | ticket_price | event_type | booking_id |
+----------+------------------+------------+------------+----------+-------------+----------------+--------------+------------+------------+
|        1 | Concert A        | 2023-12-15 | 19:00:00   |        1 |         500 |             500 |        50.00 | Concert    |          1 |
|        2 | Movie Night      | 2023-12-18 | 20:30:00   |        2 |         300 |             300 |        10.00 | Movie      |          2 |
|        3 | Sports Tournament | 2023-12-20 | 14:00:00   |        3 |        1000 |            1000 |        20.00 | Sports     |          3 |
|        4 | Concert B        | 2023-12-22 | 18:30:00   |        4 |         700 |             700 |        40.00 | Concert    |          4 |
|        5 | Conference       | 2023-12-25 | 09:00:00   |        5 |         200 |             200 |        30.00 | Movie      |          5 |
|        6 | Movie Premiere   | 2023-12-28 | 19:45:00   |        6 |         400 |             400 |        15.00 | Movie      |          6 |
|        7 | Concert Z        | 2023-12-30 | 21:00:00   |        7 |         600 |             600 |        25.00 | Concert    |          7 |
+----------+------------------+------------+------------+----------+-------------+----------------+--------------+------------+------------+
7 rows in set (0.00 sec)

mysql>
```

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

```
mysql> SELECT * FROM Event
    -> WHERE available_seats > 0 AND event_name LIKE '%Concert%';
+----------+------------+------------+------------+----------+-------------+----------------+--------------+------------+------------+
| event_id | event_name | event_date | event_time | venue_id | total_seats | available_seats | ticket_price | event_type | booking_id |
+----------+------------+------------+------------+----------+-------------+----------------+--------------+------------+------------+
|        1 | Concert A  | 2023-12-15 | 19:00:00   |        1 |         500 |             500 |        50.00 | Concert    |          1 |
|        4 | Concert B  | 2023-12-22 | 18:30:00   |        4 |         700 |             700 |        40.00 | Concert    |          4 |
|        7 | Concert Z  | 2023-12-30 | 21:00:00   |        7 |         600 |             600 |        25.00 | Concert    |          7 |
|        9 | Concert C  | 2024-01-05 | 20:00:00   |        9 |         900 |             900 |        45.00 | Concert    |          9 |
+----------+------------+------------+------------+----------+-------------+----------------+--------------+------------+------------+
4 rows in set (0.00 sec)

mysql>
```

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

```
mysql> SELECT * FROM Booking
    -> WHERE num_tickets > 4;
+------------+-------------+-----------+-------------+-------------+---------------------+
| booking_id | customer_id | event_id | num_tickets | total_cost | booking_date        |
+------------+-------------+-----------+-------------+-------------+---------------------+
|          3 |           3 |         3 |           5 |      100.00 | 2023-12-12 12:00:00 |
|          8 |           8 |         8 |           6 |      210.00 | 2023-12-17 21:15:00 |
+------------+-------------+-----------+-------------+-------------+---------------------+
2 rows in set (0.00 sec)

mysql>
```

Write a SQL query to retrieve customer information whose phone number end with ‘1234’

```
mysql> SELECT * FROM Customer
    -> WHERE phone_number LIKE '%1234';
+-------------+---------------+-------------------------+--------------+------------+
| customer_id | customer_name  | email                   | phone_number | booking_id |
+-------------+---------------+-------------------------+--------------+------------+
|           5 | Michael Wilson | michael.wilson@email.com | 567-890-1234 |          5 |
+-------------+---------------+-------------------------+--------------+------------+
1 row in set (0.00 sec)

mysql>
```

Write a SQL query to retrieve the events in order whose seat capacity more than300

```
mysql> SELECT * FROM Event
    -> WHERE total_seats > 300
    -> ORDER BY total_seats;
+----------+------------------+------------+------------+----------+-------------+-----------------+--------------+------------+------------+
| event_id | event_name       | event_date | event_time | venue_id | total_seats | available_seats | ticket_price | event_type | booking_id |
+----------+------------------+------------+------------+----------+-------------+-----------------+--------------+------------+------------+
|        6 | Movie Premiere   | 2023-12-28 | 19:45:00   |        6 |         400 |             400 |        15.00 | Movie      |          6 |
|        1 | Concert A        | 2023-12-15 | 19:00:00   |        1 |         500 |             500 |        50.00 | Concert    |          1 |
|        7 | Concert Z        | 2023-12-30 | 21:00:00   |        7 |         600 |             600 |        25.00 | Concert    |          7 |
|        4 | Concert B        | 2023-12-22 | 18:30:00   |        4 |         700 |             700 |        40.00 | Concert    |          4 |
|        8 | Sports Game      | 2024-01-02 | 15:30:00   |        8 |         800 |             800 |        35.00 | Sports     |          8 |
|        9 | Concert C        | 2024-01-05 | 20:00:00   |        9 |         900 |             900 |        45.00 | Concert    |          9 |
|        3 | Sports Tournament | 2023-12-20 | 14:00:00   |        3 |        1000 |            1000 |        20.00 | Sports     |          3 |
+----------+------------------+------------+------------+----------+-------------+-----------------+--------------+------------+------------+
7 rows in set (0.00 sec)

mysql>
```

Write a SQL query to select events name not start with ‘x’, ‘y’, ‘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 | booking_id |
+----------+------------------+------------+------------+----------+-------------+-----------------+--------------+------------+------------+
|        1 | Concert A        | 2023-12-15 | 19:00:00   |        1 |         500 |             500 |        50.00 | Concert    |          1 |
|        2 | Movie Night      | 2023-12-18 | 20:30:00   |        2 |         300 |             300 |        10.00 | Movie      |          2 |
|        3 | Sports Tournament | 2023-12-20 | 14:00:00   |        3 |        1000 |            1000 |        20.00 | Sports     |          3 |
|        4 | Concert B        | 2023-12-22 | 18:30:00   |        4 |         700 |             700 |        40.00 | Concert    |          4 |
|        5 | Conference       | 2023-12-25 | 09:00:00   |        5 |         200 |             200 |        30.00 | Movie      |          5 |
|        6 | Movie Premiere   | 2023-12-28 | 19:45:00   |        6 |         400 |             400 |        15.00 | Movie      |          6 |
|        7 | Concert Z        | 2023-12-30 | 21:00:00   |        7 |         600 |             600 |        25.00 | Concert    |          7 |
|        8 | Sports Game      | 2024-01-02 | 15:30:00   |        8 |         800 |             800 |        35.00 | Sports     |          8 |
|        9 | Concert C        | 2024-01-05 | 20:00:00   |        9 |         900 |             900 |        45.00 | Concert    |          9 |
|       10 | Sports Show      | 2024-01-08 | 12:00:00   |       10 |         150 |             150 |         5.00 | Sports     |         10 |
+----------+------------------+------------+------------+----------+-------------+-----------------+--------------+------------+------------+
10 rows in set (0.00 sec)

mysql>
```

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

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

```
mysql> SELECT
    ->     e.event_id,
    ->     e.event_name,
    ->     AVG(e.ticket_price) AS average_ticket_price
    -> FROM
    ->     Event e
    -> JOIN
    ->     Booking b ON e.event_id = b.event_id
    -> GROUP BY
    ->     e.event_id, e.event_name;
+----------+------------------+----------------------+
| event_id | event_name       | average_ticket_price |
+----------+------------------+----------------------+
|        1 | Concert A        |            50.000000 |
|        2 | Movie Night      |            10.000000 |
|        3 | Sports Tournament |            20.000000 |
|        4 | Concert B        |            40.000000 |
|        5 | Conference       |            30.000000 |
|        6 | Movie Premiere   |            15.000000 |
|        7 | Concert Z        |            25.000000 |
|        8 | Sports Game      |            35.000000 |
|        9 | Concert C        |            45.000000 |
|       10 | Sports Show      |             5.000000 |
+----------+------------------+----------------------+
10 rows in set (0.00 sec)

mysql>
```

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

```
mysql> SELECT
    ->     SUM(total_cost) AS total_revenue
    -> FROM
    ->     Booking;
+---------------+
| total_revenue |
+---------------+
|        760.00 |
+---------------+
1 row in set (0.00 sec)

mysql>
```

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

```
mysql> SELECT
    ->     e.event_id,
    ->     e.event_name,
    ->     SUM(b.num_tickets) AS total_tickets_sold
    -> FROM
    ->     Event e
    -> JOIN
    ->     Booking b ON e.event_id = b.event_id
    -> GROUP BY
    ->     e.event_id, e.event_name
    -> ORDER BY
    ->     total_tickets_sold DESC
    -> LIMIT 1;
+----------+-------------+--------------------+
| event_id | event_name  | total_tickets_sold |
+----------+-------------+--------------------+
|        8 | Sports Game |                  6 |
+----------+-------------+--------------------+
1 row in set (0.00 sec)

mysql>
```

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

```
mysql> SELECT
    ->     e.event_id,
    ->     e.event_name,
    ->     SUM(b.num_tickets) AS total_tickets_sold
    -> FROM
    ->     Event e
    -> JOIN
    ->     Booking b ON e.event_id = b.event_id
    -> GROUP BY
    ->     e.event_id, e.event_name;
+----------+------------------+--------------------+
| event_id | event_name       | total_tickets_sold |
+----------+------------------+--------------------+
|        1 | Concert A        |                  2 |
|        2 | Movie Night      |                  3 |
|        3 | Sports Tournament |                 5 |
|        4 | Concert B        |                  1 |
|        5 | Conference       |                  4 |
|        6 | Movie Premiere   |                  2 |
|        7 | Concert Z        |                  3 |
|        8 | Sports Game      |                  6 |
|        9 | Concert C        |                  1 |
|       10 | Sports Show      |                  2 |
+----------+------------------+--------------------+
10 rows in set (0.00 sec)

mysql>
```

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

```
mysql> SELECT
    ->     e.event_id,
    ->     e.event_name
    -> FROM
    ->     Event e
    -> LEFT JOIN
    ->     Booking b ON e.event_id = b.event_id
    -> WHERE
    ->     b.booking_id IS NULL;
Empty set (0.00 sec)

mysql>
```

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

```
mysql> SELECT
    ->    c.customer_id,
    ->    c.customer_name,
    ->    SUM(b.num_tickets) AS total_tickets_booked
    -> FROM
    ->    Customer c
    -> JOIN
    ->    Booking b ON c.customer_id=b.customer_id
    -> GROUP BY
    ->    c.customer_id,c.customer_name
    ->  ORDER BY
    ->    total_tickets_booked DESC
    -> LIMIT 1;
+-------------+---------------+----------------------+
| customer_id | customer_name | total_tickets_booked |
+-------------+---------------+----------------------+
|           8 | Olivia Turner |                    6 |
+-------------+---------------+----------------------+
1 row in set (0.00 sec)

mysql>
```

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

```
mysql> SELECT
    ->     e.event_id,
    ->     e.event_name,
    ->     MONTH(b.booking_date) AS month,
    ->     YEAR(b.booking_date) AS year,
    ->     SUM(b.num_tickets) AS total_tickets_sold
    -> FROM
    ->     Event e
    -> JOIN
    ->     Booking b ON e.event_id = b.event_id
    -> GROUP BY
    ->     e.event_id, e.event_name, MONTH(b.booking_date), YEAR(b.booking_date);
+----------+------------------+-------+------+--------------------+
| event_id | event_name       | month | year | total_tickets_sold |
+----------+------------------+-------+------+--------------------+
|        1 | Concert A        |    12 | 2023 |                  2 |
|        2 | Movie Night      |    12 | 2023 |                  3 |
|        3 | Sports Tournament |    12 | 2023 |                  5 |
|        4 | Concert B        |    12 | 2023 |                  1 |
|        5 | Conference       |    12 | 2023 |                  4 |
|        6 | Movie Premiere   |    12 | 2023 |                  2 |
|        7 | Concert Z        |    12 | 2023 |                  3 |
|        8 | Sports Game      |    12 | 2023 |                  6 |
|        9 | Concert C        |    12 | 2023 |                  1 |
|       10 | Sports Show      |    12 | 2023 |                  2 |
+----------+------------------+-------+------+--------------------+
10 rows in set (0.00 sec)

mysql>
```

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

```
mysql> SELECT
    ->     v.venue_id,
    ->     v.venue_name,
    ->     AVG(e.ticket_price) AS average_ticket_price
    -> FROM
    ->     Venue v
    -> JOIN
    ->     Event e ON v.venue_id = e.venue_id
    -> GROUP BY
    ->     v.venue_id, v.venue_name;
+----------+------------------+----------------------+
| venue_id | venue_name       | average_ticket_price |
+----------+------------------+----------------------+
|        1 | Example Venue 1  |            50.000000 |
|        2 | Example Venue 2  |            10.000000 |
|        3 | Example Venue 3  |            20.000000 |
|        4 | Example Venue 4  |            40.000000 |
|        5 | Example Venue 5  |            30.000000 |
|        6 | Example Venue 6  |            15.000000 |
|        7 | Example Venue 7  |            25.000000 |
|        8 | Example Venue 8  |            35.000000 |
|        9 | Example Venue 9  |            45.000000 |
|       10 | Example Venue 10 |             5.000000 |
+----------+------------------+----------------------+
10 rows in set (0.00 sec)

mysql>
```

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

```
mysql> SELECT
    ->     e.event_type,
    ->     SUM(b.num_tickets) AS total_tickets_sold
    -> FROM
    ->     Event e
    -> JOIN
    ->     Booking b ON e.event_id = b.event_id
    -> GROUP BY
    ->     e.event_type;
+------------+--------------------+
| event_type | total_tickets_sold |
+------------+--------------------+
| Concert    |                  7 |
| Movie      |                  9 |
| Sports     |                 13 |
+------------+--------------------+
3 rows in set (0.00 sec)

mysql>
```

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

```
mysql> SELECT
    ->     YEAR(b.booking_date) AS year,
    ->     SUM(b.total_cost) AS total_revenue
    -> FROM
    ->     Booking b
    -> JOIN
    ->     Event e ON b.event_id = e.event_id
    -> GROUP BY
    ->     YEAR(b.booking_date);
+------+---------------+
| year | total_revenue |
+------+---------------+
| 2023 |        760.00 |
+------+---------------+
1 row in set (0.00 sec)

mysql>
```

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

```
mysql> SELECT
    ->     c.Customer_id,
    ->     c.Customer_name
    -> FROM
    ->     Customer c
    -> JOIN
    ->     Booking b ON c.customer_id=b.customer_id
    -> GROUP BY
    ->     c.customer_id,c.customer_name
    -> HAVING
    ->     COUNT(DISTINCT b.event_id) > 1;
Empty set (0.00 sec)

mysql>
```

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

```
mysql> SELECT
    ->     c.Customer_id,
    ->     c.Customer_name,
    ->     SUM(b.total_cost) AS total_revenue
    -> FROM
    ->     Customer c
    -> JOIN
    ->     Booking b ON c.customer_id=b.customer_id
    -> GROUP BY
    ->     c.customer_id,c.customer_name;
+-------------+---------------+---------------+
| Customer_id | Customer_name | total_revenue |
+-------------+---------------+---------------+
|           1 | John Doe      |        100.00 |
|           2 | Alice Smith   |         30.00 |
|           3 | Bob Johnson   |        100.00 |
|           4 | Emily Davis   |         40.00 |
|           5 | Michael Wilson|        120.00 |
|           6 | Sophia Brown  |         30.00 |
|           7 | Daniel Lee    |         75.00 |
|           8 | Olivia Turner |        210.00 |
|           9 | Liam Miller   |         45.00 |
|          10 | Emma Harris   |         10.00 |
+-------------+---------------+---------------+
10 rows in set (0.00 sec)

mysql>
```

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

```
mysql> SELECT
    ->     v.venue_id,
    ->     v.venue_name,
    ->     e.event_type,
    ->     AVG(e.ticket_price) AS average_ticket_price
    -> FROM
    ->     Venue v
    -> JOIN
    ->     Event e ON v.venue_id = e.venue_id
    -> GROUP BY
    ->     v.venue_id, v.venue_name, e.event_type;
+----------+-----------------+------------+----------------------+
| venue_id | venue_name      | event_type | average_ticket_price |
+----------+-----------------+------------+----------------------+
|        1 | Example Venue 1 | Concert    |            50.000000 |
|        2 | Example Venue 2 | Movie      |            10.000000 |
|        3 | Example Venue 3 | Sports     |            20.000000 |
|        4 | Example Venue 4 | Concert    |            40.000000 |
|        5 | Example Venue 5 | Movie      |            30.000000 |
|        6 | Example Venue 6 | Movie      |            15.000000 |
|        7 | Example Venue 7 | Concert    |            25.000000 |
|        8 | Example Venue 8 | Sports     |            35.000000 |
|        9 | Example Venue 9 | Concert    |            45.000000 |
|       10 | Example Venue 10| Sports     |             5.000000 |
+----------+-----------------+------------+----------------------+
10 rows in set (0.00 sec)

mysql>
```

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

```
mysql>  SELECT
    ->    c.Customer_id,
    ->    c.Customer_name,
    ->    SUM(b.num_tickets) AS total_tickets_purchased
    -> FROM
    ->    Customer c
    -> JOIN
    ->    Booking b ON c.customer_id=b.customer_id
    -> WHERE
    ->    b.booking_date >= CURDATE() - INTERVAL 30 DAY
    -> GROUP BY
    ->    c.customer_id,c.customer_name;
+-------------+---------------+-------------------------+
| Customer_id | Customer_name | total_tickets_purchased |
+-------------+---------------+-------------------------+
|           1 | John Doe      |                       2 |
|           2 | Alice Smith   |                       3 |
|           3 | Bob Johnson   |                       5 |
|           4 | Emily Davis   |                       1 |
|           5 | Michael Wilson|                       4 |
|           6 | Sophia Brown  |                       2 |
|           7 | Daniel Lee    |                       3 |
|           8 | Olivia Turner |                       6 |
|           9 | Liam Miller   |                       1 |
|          10 | Emma Harris   |                       2 |
+-------------+---------------+-------------------------+
10 rows in set (0.00 sec)

mysql>
```

Tasks 4: Subquery and its types

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

```
mysql> SELECT
    ->     v.venue_id,
    ->     v.venue_name,
    ->     (
    ->         SELECT AVG(e.ticket_price)
    ->         FROM Event e
    ->         WHERE e.venue_id = v.venue_id
    ->     ) AS average_ticket_price
    -> FROM
    ->     Venue v;
+----------+------------------+----------------------+
| venue_id | venue_name       | average_ticket_price |
+----------+------------------+----------------------+
|        1 | Example Venue 1  |            50.000000 |
|        2 | Example Venue 2  |            10.000000 |
|        3 | Example Venue 3  |            20.000000 |
|        4 | Example Venue 4  |            40.000000 |
|        5 | Example Venue 5  |            30.000000 |
|        6 | Example Venue 6  |            15.000000 |
|        7 | Example Venue 7  |            25.000000 |
|        8 | Example Venue 8  |            35.000000 |
|        9 | Example Venue 9  |            45.000000 |
|       10 | Example Venue 10 |             5.000000 |
+----------+------------------+----------------------+
10 rows in set (0.00 sec)

mysql>
```

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

```
mysql> SELECT
    ->     e.event_id,
    ->     e.event_name
    -> FROM
    ->     Event e
    -> WHERE
    ->     (
    ->         SELECT COUNT(*)
    ->         FROM Booking b
    ->         WHERE b.event_id = e.event_id
    ->     ) > 0.5 * e.total_seats;
Empty set (0.00 sec)

mysql>
```

Calculate the Total Number of Tickets Sold for Each Event.

```
mysql> SELECT
    ->     e.event_id,
    ->     e.event_name,
    ->     SUM(b.num_tickets) AS total_tickets_sold
    -> FROM
    ->     Event e
    -> JOIN
    ->     Booking b ON e.event_id = b.event_id
    -> GROUP BY
    ->     e.event_id, e.event_name;
+----------+------------------+--------------------+
| event_id | event_name       | total_tickets_sold |
+----------+------------------+--------------------+
|        1 | Concert A        |                  2 |
|        2 | Movie Night      |                  3 |
|        3 | Sports Tournament |                 5 |
|        4 | Concert B        |                  1 |
|        5 | Conference       |                  4 |
|        6 | Movie Premiere   |                  2 |
|        7 | Concert Z        |                  3 |
|        8 | Sports Game      |                  6 |
|        9 | Concert C        |                  1 |
|       10 | Sports Show      |                  2 |
+----------+------------------+--------------------+
10 rows in set (0.00 sec)

mysql>
```

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

```
mysql> SELECT
    ->     customer_id,
    ->     customer_name
    -> FROM
    ->     Customer c
    -> WHERE
    ->     NOT EXISTS (
    ->         SELECT 1
    ->         FROM Booking b
    ->         WHERE b.customer_id=c.customer_id);
Empty set (0.00 sec)

mysql>
```

List Events with No Ticket Sales Using a NOT IN Subquery

```
mysql> SELECT
    ->     event_id,
    ->     event_name
    -> FROM
    ->     Event
    -> WHERE
    ->     event_id NOT IN (
    ->         SELECT DISTINCT event_id
    ->         FROM Booking
    ->     );
Empty set (0.00 sec)

mysql>
```

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

```
mysql> SELECT
    ->     e.event_type,
    ->     SUM(subquery.num_tickets) AS total_tickets_sold
    -> FROM
    ->     Event e
    -> JOIN
    ->     (
    ->         SELECT
    ->             event_id,
    ->             num_tickets
    ->         FROM
    ->             Booking
    ->     ) subquery ON e.event_id = subquery.event_id
    -> GROUP BY
    ->     e.event_type;
+------------+--------------------+
| event_type | total_tickets_sold |
+------------+--------------------+
| Concert    |                  7 |
| Movie      |                  9 |
| Sports     |                 13 |
+------------+--------------------+
3 rows in set (0.00 sec)

mysql>
```

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

```
mysql> SELECT
    ->     event_id,
    ->     event_name,
    ->     ticket_price
    -> FROM
    ->     Event
    -> WHERE
    ->     ticket_price > (
    ->         SELECT AVG(ticket_price)
    ->         FROM Event
    ->     );
+----------+-------------+--------------+
| event_id | event_name  | ticket_price |
+----------+-------------+--------------+
|        1 | Concert A   |        50.00 |
|        4 | Concert B   |        40.00 |
|        5 | Conference  |        30.00 |
|        8 | Sports Game |        35.00 |
|        9 | Concert C   |        45.00 |
+----------+-------------+--------------+
5 rows in set (0.00 sec)

mysql>
```

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

```
mysql> SELECT
    ->    c.customer_id,
    ->    c.customer_name,
    ->    (
    ->       SELECT SUM(b.total_cost)
    ->         FROM Booking b
    ->         WHERE b.customer_id=c.customer_id
    ->    ) AS total_revenue
    -> FROM
    ->     Customer c;
+-------------+---------------+---------------+
| customer_id | customer_name | total_revenue |
+-------------+---------------+---------------+
|           1 | John Doe      |        100.00 |
|           2 | Alice Smith   |         30.00 |
|           3 | Bob Johnson   |        100.00 |
|           4 | Emily Davis   |         40.00 |
|           5 | Michael Wilson |        120.00 |
|           6 | Sophia Brown  |         30.00 |
|           7 | Daniel Lee    |         75.00 |
|           8 | Olivia Turner |        210.00 |
|           9 | Liam Miller   |         45.00 |
|          10 | Emma Harris   |         10.00 |
+-------------+---------------+---------------+
10 rows in set (0.00 sec)

mysql>
```

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

```
mysql> SELECT
    ->     c.customer_id,
    ->     c.customer_name
    -> FROM
    ->     Customer c
    -> WHERE
    ->     customer_id IN(
    ->         SELECT DISTINCT b.customer_id
    ->         FROM Booking b
    ->         JOIN Event e ON b.event_id = e.event_id
    ->          WHERE e.venue_id =6);
+-------------+---------------+
| customer_id | customer_name |
+-------------+---------------+
|           6 | Sophia Brown  |
+-------------+---------------+
1 row in set (0.00 sec)
```

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

```
mysql> SELECT
    ->     e.event_type,
    ->     (
    ->         SELECT SUM(b.num_tickets)
    ->         FROM Booking b
    ->         WHERE b.event_id IN (
    ->             SELECT e_inner.event_id
    ->             FROM Event e_inner
    ->             WHERE e_inner.event_type = e.event_type
    ->         )
    ->     ) AS total_tickets_sold
    -> FROM
    ->     Event e
    -> GROUP BY
    ->     e.event_type;
+------------+--------------------+
| event_type | total_tickets_sold |
+------------+--------------------+
| Concert    |                  7 |
| Movie      |                  9 |
| Sports     |                 13 |
+------------+--------------------+
3 rows in set (0.00 sec)

mysql>
```

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

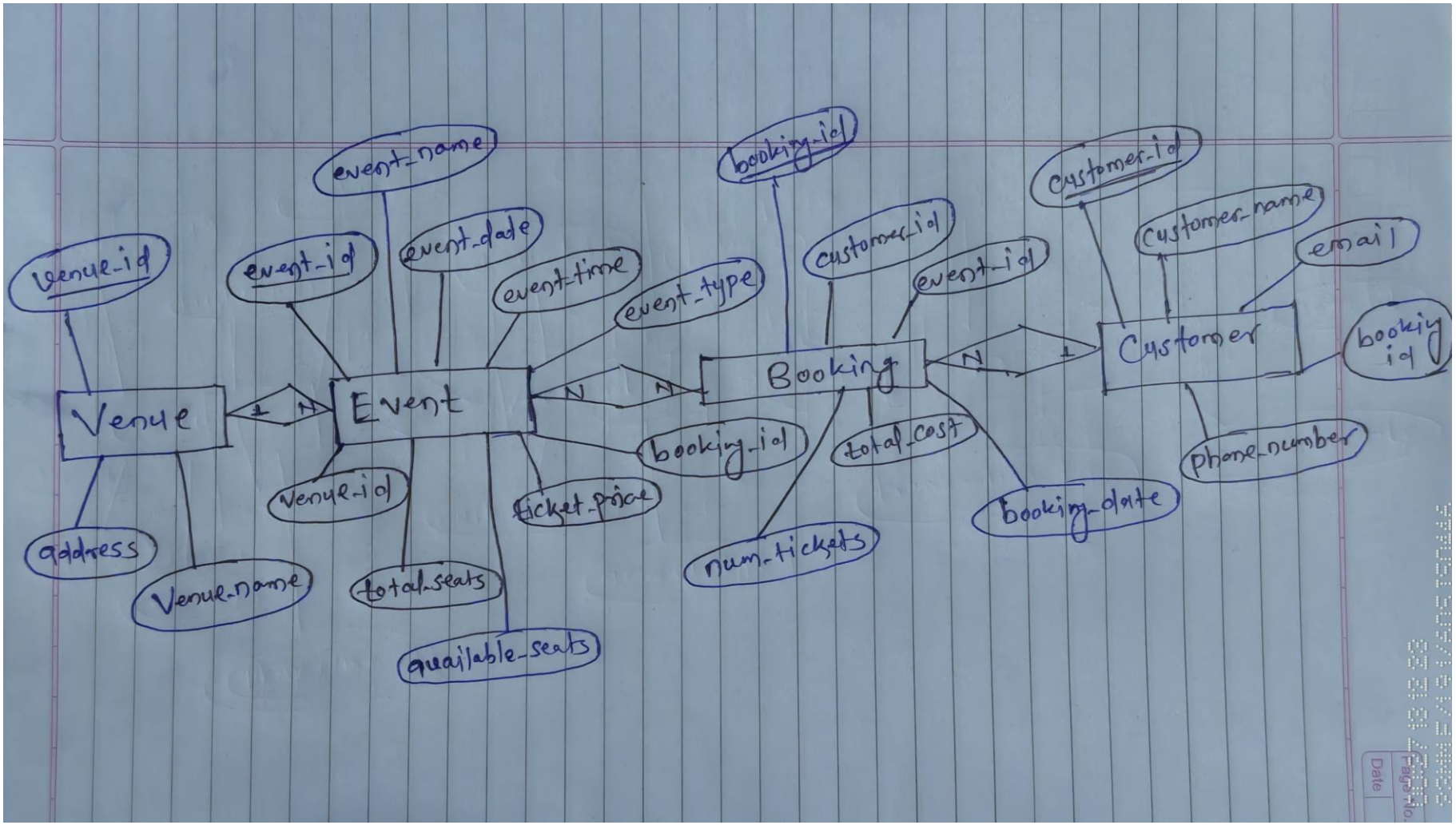
```
mysql> SELECT
    ->     c.customer_id,
    ->     c.customer_name,
    ->     DATE_FORMAT(b.booking_date, '%Y-%m') AS booking_month
    -> FROM
    ->     Customer c
    -> JOIN
    ->     Booking b ON c.customer_id=b.customer_id
    -> GROUP BY
    ->     c.customer_id,c.customer_name,booking_month;
+-------------+---------------+---------------+
| customer_id | customer_name  | booking_month |
+-------------+---------------+---------------+
|           1 | John Doe       | 2023-12       |
|           2 | Alice Smith    | 2023-12       |
|           3 | Bob Johnson    | 2023-12       |
|           4 | Emily Davis    | 2023-12       |
|           5 | Michael Wilson | 2023-12       |
|           6 | Sophia Brown   | 2023-12       |
|           7 | Daniel Lee     | 2023-12       |
|           8 | Olivia Turner  | 2023-12       |
|           9 | Liam Miller    | 2023-12       |
|          10 | Emma Harris    | 2023-12       |
+-------------+---------------+---------------+
10 rows in set (0.00 sec)
```

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

```
mysql> SELECT
    ->     v.venue_id,
    ->     v.venue_name,
    ->     (
    ->         SELECT AVG(e.ticket_price)
    ->         FROM Event e
    ->         WHERE e.venue_id = v.venue_id
    ->     ) AS average_ticket_price
    -> FROM
    ->     Venue v;
+----------+-----------------+---------------------+
| venue_id | venue_name       | average_ticket_price |
+----------+-----------------+---------------------+
|        1 | Example Venue 1  |           50.000000 |
|        2 | Example Venue 2  |           10.000000 |
|        3 | Example Venue 3  |           20.000000 |
|        4 | Example Venue 4  |           40.000000 |
|        5 | Example Venue 5  |           30.000000 |
|        6 | Example Venue 6  |           15.000000 |
|        7 | Example Venue 7  |           25.000000 |
|        8 | Example Venue 8  |           35.000000 |
|        9 | Example Venue 9  |           45.000000 |
|       10 | Example Venue 10 |            5.000000 |
+----------+-----------------+---------------------+
10 rows in set (0.00 sec)

mysql>
```

Entity Relationship Diagram


Venue
Event
Booking
Customer
venue_id
venue_name
address
event_id
event_name
event_date
event_time
event_type
venue_id
total_seats
available_seats
ticket_price
booking_id
customer_id
event_id
booking_id
num_tickets
total_cost
booking_date
customer_id
customer_name
email
phone_number
booking id
1
N
N
N
N
1