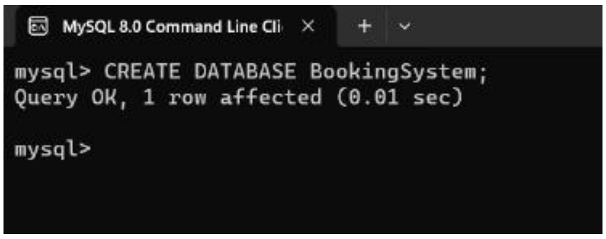
Assignment – 5

Madiha Aimon Tappal

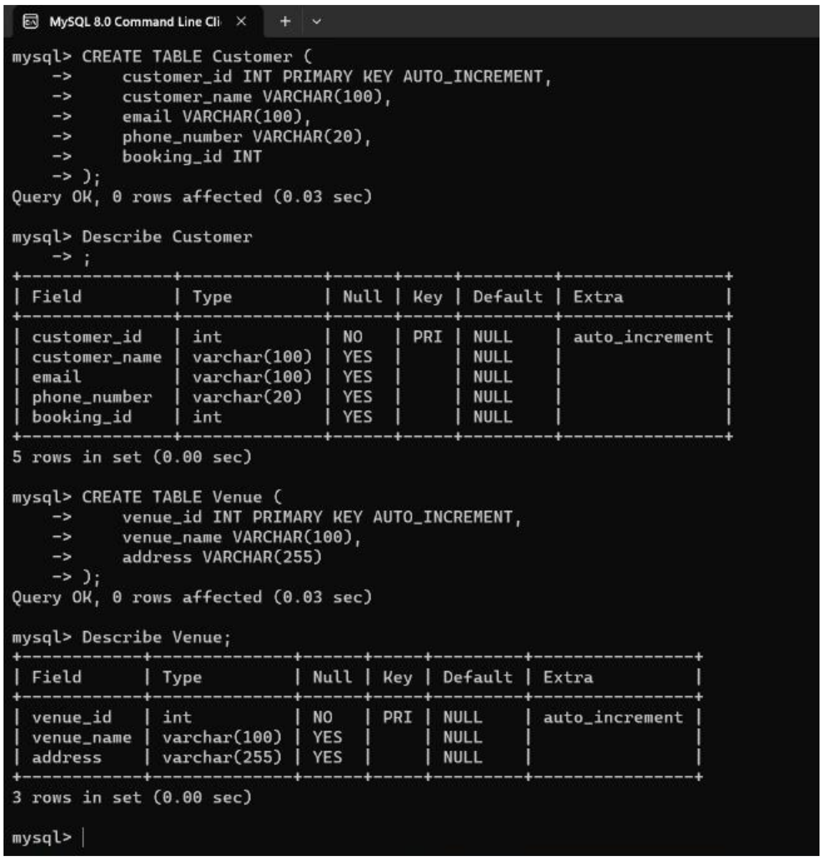
Database Table:



```
mysql> CREATE DATABASE BookingSystem;
Query OK, 1 row affected (0.01 sec)

mysql>
```

1. Venu Table

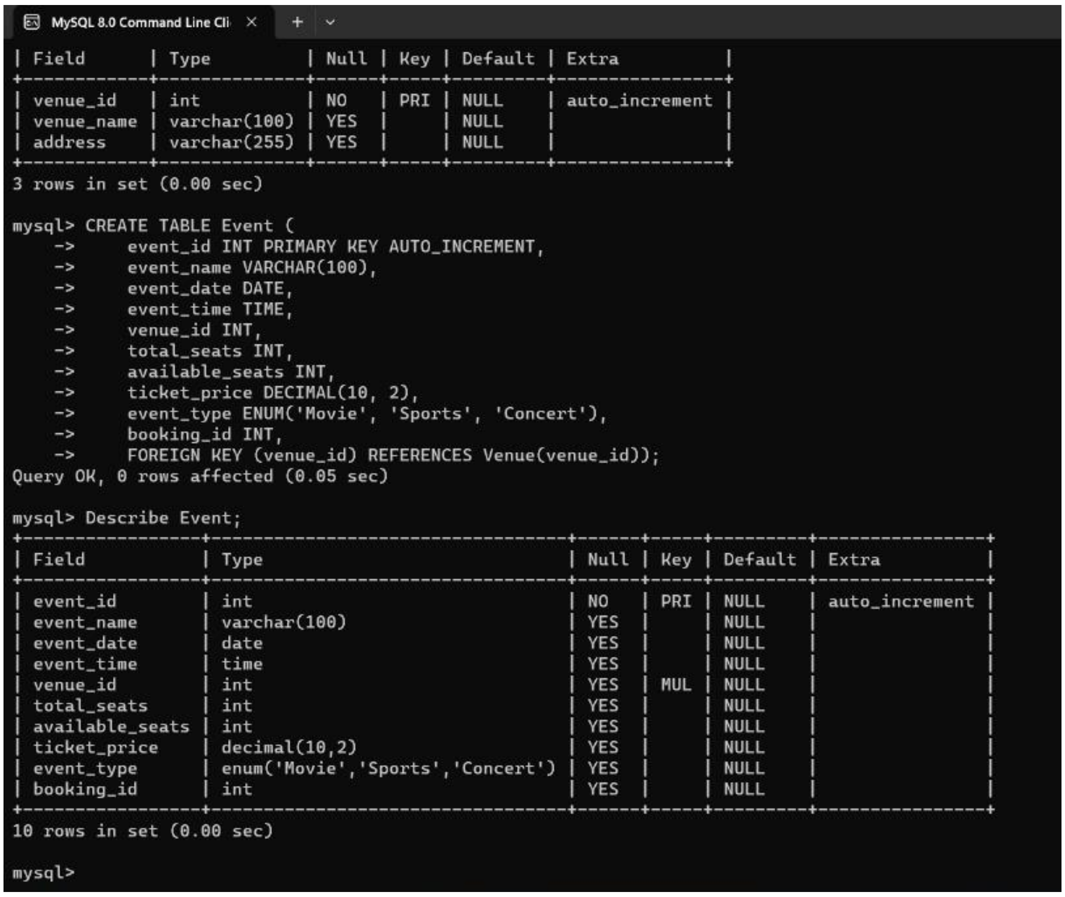
- venue_id (Primary Key)
- venue_name,
- address



```
mysql> CREATE TABLE Customer (
    ->     customer_id INT PRIMARY KEY AUTO_INCREMENT,
    ->     customer_name VARCHAR(100),
    ->     email VARCHAR(100),
    ->     phone_number VARCHAR(20),
    ->     booking_id INT
    -> );
Query OK, 0 rows affected (0.03 sec)

mysql> Describe Customer
    -> ;
+---------------+--------------+------+-----+---------+----------------+
| Field         | Type         | Null | Key | Default | Extra          |
+---------------+--------------+------+-----+---------+----------------+
| customer_id   | int          | NO   | PRI | NULL    | auto_increment |
| customer_name | varchar(100) | YES  |     | NULL    |                |
| email         | varchar(100) | YES  |     | NULL    |                |
| phone_number  | varchar(20)  | YES  |     | NULL    |                |
| booking_id    | int          | YES  |     | NULL    |                |
+---------------+--------------+------+-----+---------+----------------+
5 rows in set (0.00 sec)

mysql> CREATE TABLE Venue (
    ->     venue_id INT PRIMARY KEY AUTO_INCREMENT,
    ->     venue_name VARCHAR(100),
    ->     address VARCHAR(255)
    -> );
Query OK, 0 rows affected (0.03 sec)

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

mysql>
```

2. Event Table

- event_id (Primary Key)
- event_name,
- event_date DATE,
- event_time TIME,
- venue_id (Foreign Key),
- total_seats,
- available_seats,
- ticket_price DECIMAL,
- event_type ('Movie', 'Sports', 'Concert')
- booking_id (Foreign Key)

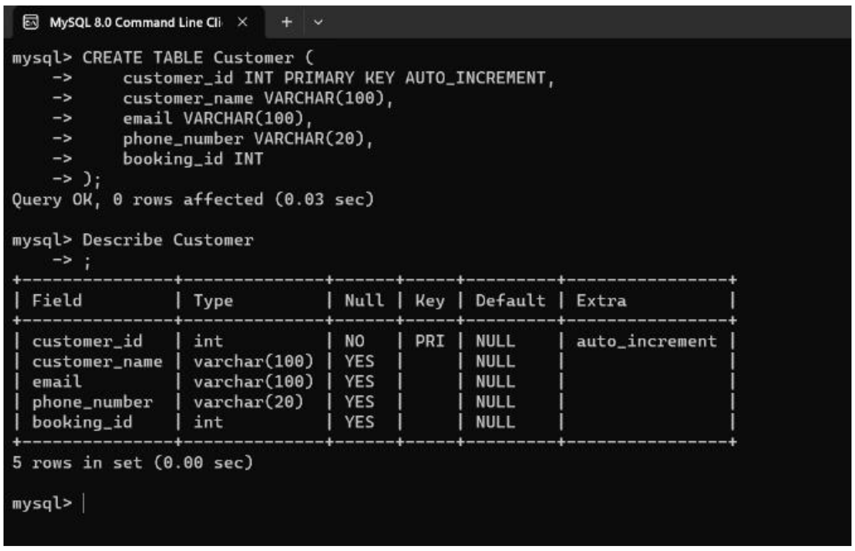


MySQL 8.0 Command Line Cli

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

mysql> CREATE TABLE Event (
    ->     event_id INT PRIMARY KEY AUTO_INCREMENT,
    ->     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> Describe Event;
+-----------------+-----------------------------------+------+-----+---------+----------------+
| Field           | Type                              | Null | Key | Default | Extra          |
+-----------------+-----------------------------------+------+-----+---------+----------------+
| event_id        | int                               | NO   | PRI | NULL    | auto_increment |
| 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>
```

3. Customer Table

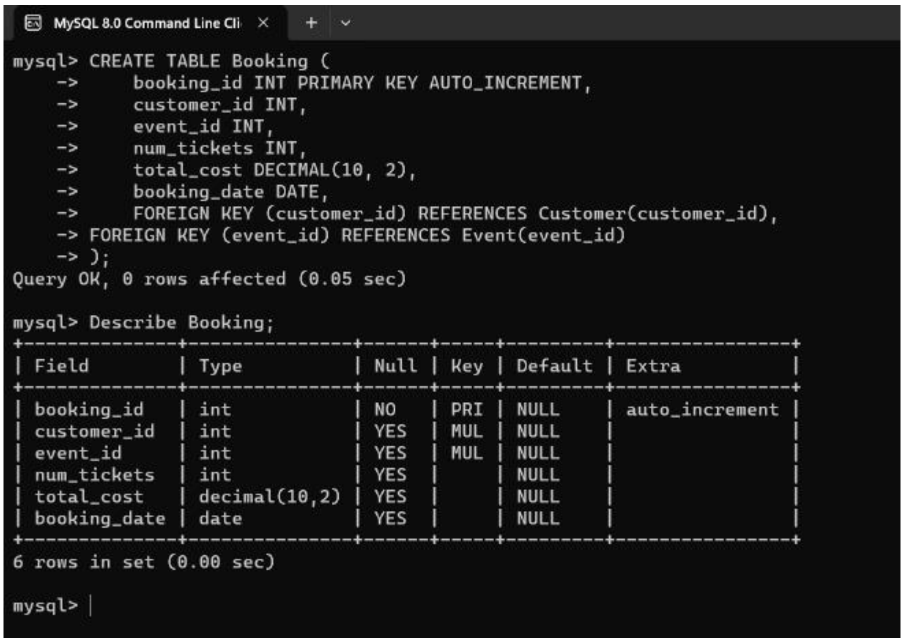
- customer_id (Primary key)
- customer_name,
- email,
- phone_number,
- booking_id (Foreign Key)



```
mysql> CREATE TABLE Customer (
    ->     customer_id INT PRIMARY KEY AUTO_INCREMENT,
    ->     customer_name VARCHAR(100),
    ->     email VARCHAR(100),
    ->     phone_number VARCHAR(20),
    ->     booking_id INT
    -> );
Query OK, 0 rows affected (0.03 sec)

mysql> Describe Customer
    -> ;
+---------------+--------------+------+-----+---------+----------------+
| Field         | Type         | Null | Key | Default | Extra          |
+---------------+--------------+------+-----+---------+----------------+
| customer_id   | int          | NO   | PRI | NULL    | auto_increment |
| customer_name | varchar(100) | YES  |     | NULL    |                |
| email         | varchar(100) | YES  |     | NULL    |                |
| phone_number  | varchar(20)  | YES  |     | NULL    |                |
| booking_id    | int          | YES  |     | NULL    |                |
+---------------+--------------+------+-----+---------+----------------+
5 rows in set (0.00 sec)

mysql> |
```

4. Booking Table

- booking_id (Primary Key),
- customer_id (Foreign Key),
- event_id (Foreign Key),
- num_tickets,
- total_cost,
- booking_date,



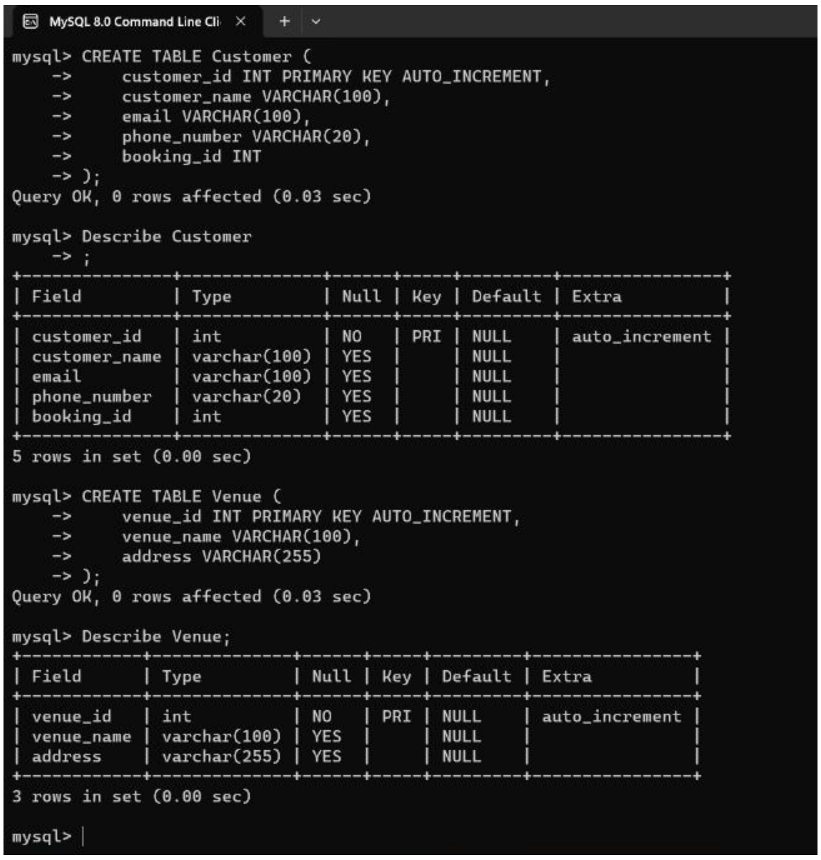
```
mysql> CREATE TABLE Booking (
    ->     booking_id INT PRIMARY KEY AUTO_INCREMENT,
    ->     customer_id INT,
    ->     event_id INT,
    ->     num_tickets INT,
    ->     total_cost DECIMAL(10, 2),
    ->     booking_date DATE,
    ->     FOREIGN KEY (customer_id) REFERENCES Customer(customer_id),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

mysql> Describe Booking;
+--------------+---------------+------+-----+---------+----------------+
| Field        | Type          | Null | Key | Default | Extra          |
+--------------+---------------+------+-----+---------+----------------+
| booking_id   | int           | NO   | PRI | NULL    | auto_increment |
| 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 | date          | YES  |     | NULL    |                |
+--------------+---------------+------+-----+---------+----------------+
6 rows in set (0.00 sec)

mysql> |
```

__Task – 1:__

1. Create the database named "TicketBookingSystem"

2. Write SQL scripts to create the mentioned tables with appropriate data types, constraints, and relationships.

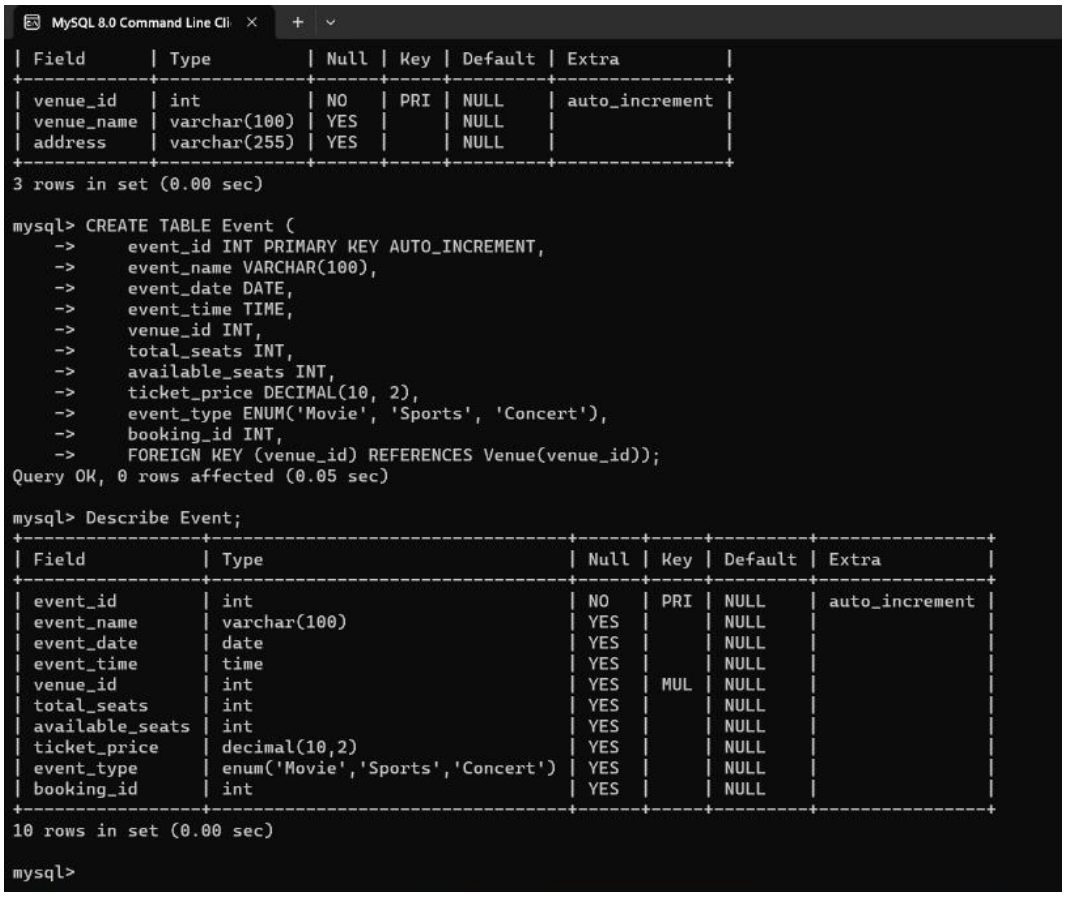
- Venu



MySQL 8.0 Command Line Cli

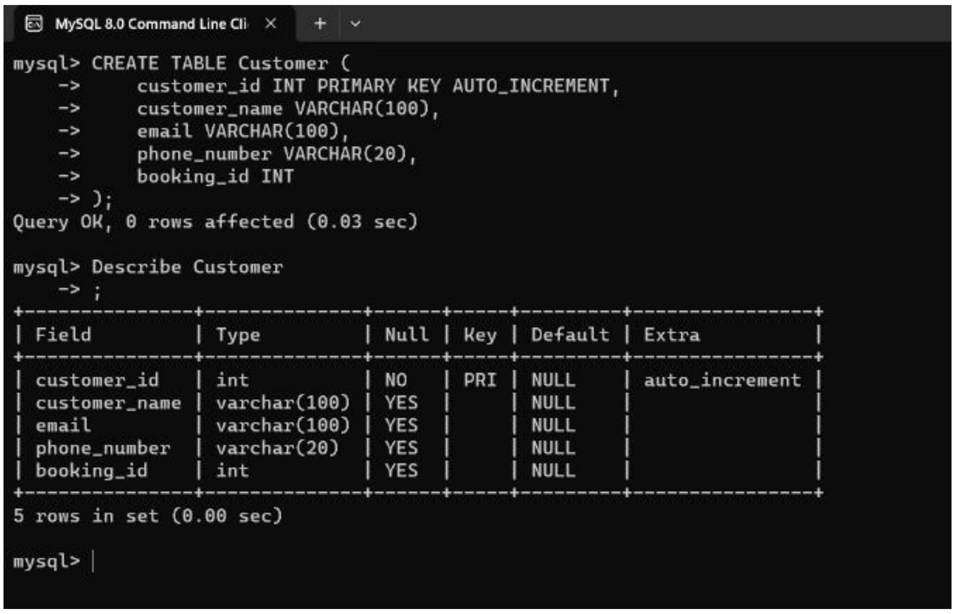
```
mysql> CREATE TABLE Customer (
    ->     customer_id INT PRIMARY KEY AUTO_INCREMENT,
    ->     customer_name VARCHAR(100),
    ->     email VARCHAR(100),
    ->     phone_number VARCHAR(20),
    ->     booking_id INT
    -> );
Query OK, 0 rows affected (0.03 sec)

mysql> Describe Customer
    -> ;
+---------------+--------------+------+-----+---------+----------------+
| Field         | Type         | Null | Key | Default | Extra          |
+---------------+--------------+------+-----+---------+----------------+
| customer_id   | int          | NO   | PRI | NULL    | auto_increment |
| customer_name | varchar(100) | YES  |     | NULL    |                |
| email         | varchar(100) | YES  |     | NULL    |                |
| phone_number  | varchar(20)  | YES  |     | NULL    |                |
| booking_id    | int          | YES  |     | NULL    |                |
+---------------+--------------+------+-----+---------+----------------+
5 rows in set (0.00 sec)

mysql> CREATE TABLE Venue (
    ->     venue_id INT PRIMARY KEY AUTO_INCREMENT,
    ->     venue_name VARCHAR(100),
    ->     address VARCHAR(255)
    -> );
Query OK, 0 rows affected (0.03 sec)

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

mysql> |
```

- Event



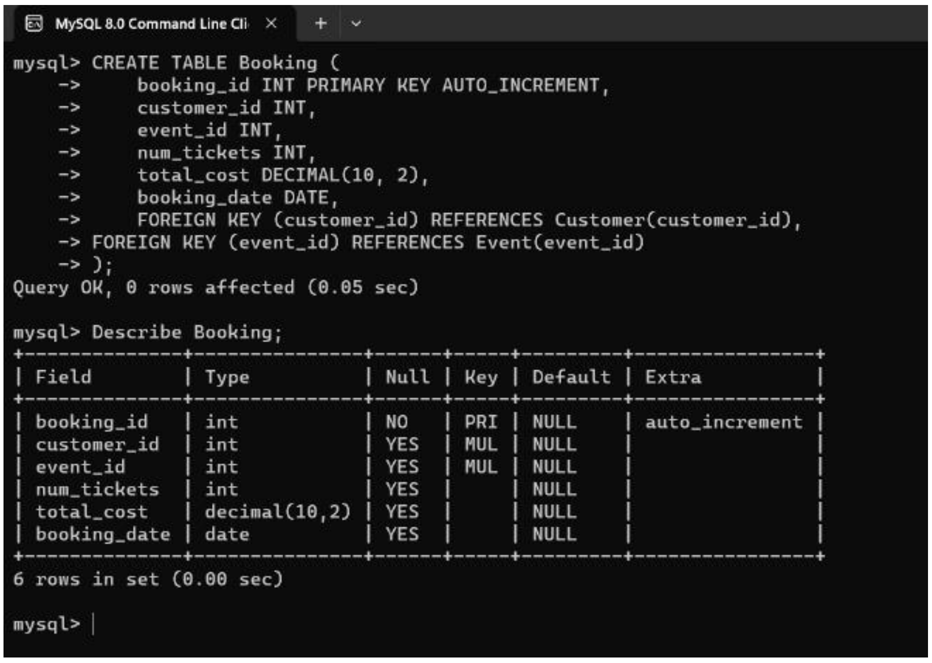
```
| Field      | Type         | Null | Key | Default | Extra          |
+------------+--------------+------+-----+---------+----------------+
| venue_id   | int          | NO   | PRI | NULL    | auto_increment |
| venue_name | varchar(100) | YES  |     | NULL    |                |
| address    | varchar(255)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

mysql> CREATE TABLE Event (
    ->     event_id INT PRIMARY KEY AUTO_INCREMENT,
    ->     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> Describe Event;
+-----------------+-----------------------------------+------+-----+---------+----------------+
| Field           | Type                              | Null | Key | Default | Extra          |
+-----------------+-----------------------------------+------+-----+---------+----------------+
| event_id        | int                               | NO   | PRI | NULL    | auto_increment |
| 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>
```

- Customers



```
mysql> CREATE TABLE Customer (
    ->     customer_id INT PRIMARY KEY AUTO_INCREMENT,
    ->     customer_name VARCHAR(100),
    ->     email VARCHAR(100),
    ->     phone_number VARCHAR(20),
    ->     booking_id INT
    -> );
Query OK, 0 rows affected (0.03 sec)

mysql> Describe Customer
    -> ;
+---------------+--------------+------+-----+---------+----------------+
| Field         | Type         | Null | Key | Default | Extra          |
+---------------+--------------+------+-----+---------+----------------+
| customer_id   | int          | NO   | PRI | NULL    | auto_increment |
| customer_name | varchar(100) | YES  |     | NULL    |                |
| email         | varchar(100) | YES  |     | NULL    |                |
| phone_number  | varchar(20)  | YES  |     | NULL    |                |
| booking_id    | int          | YES  |     | NULL    |                |
+---------------+--------------+------+-----+---------+----------------+
5 rows in set (0.00 sec)

mysql>
```

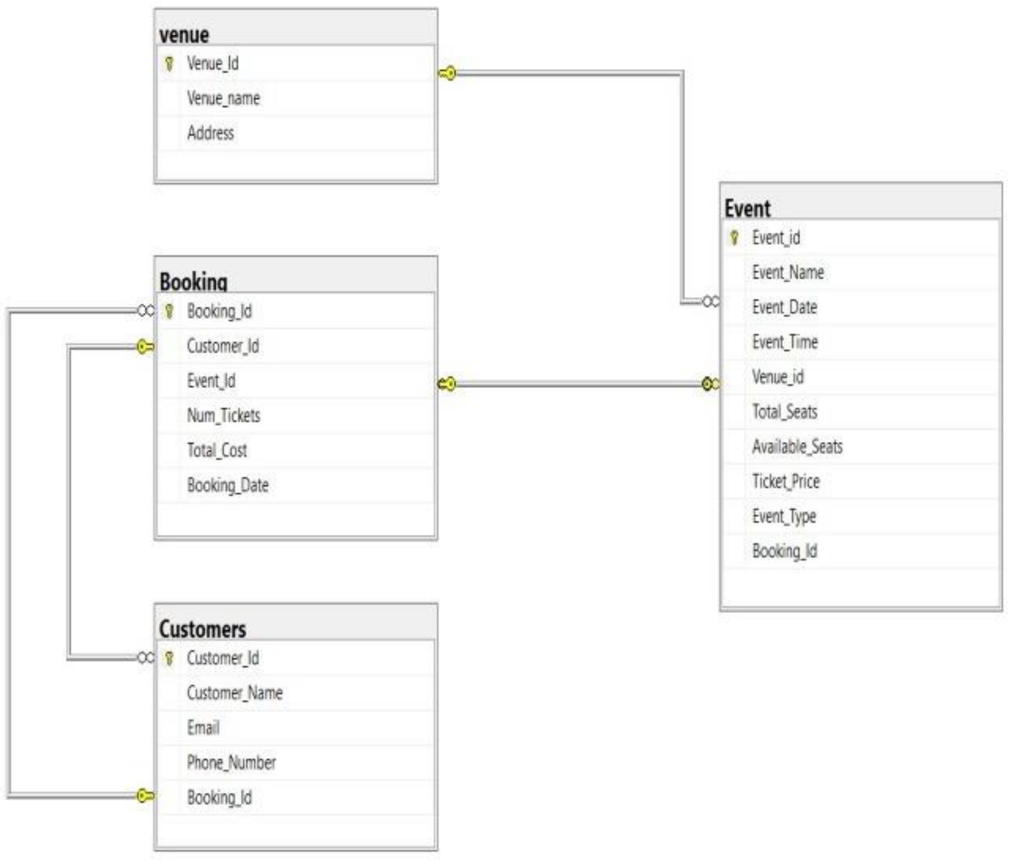
- Booking



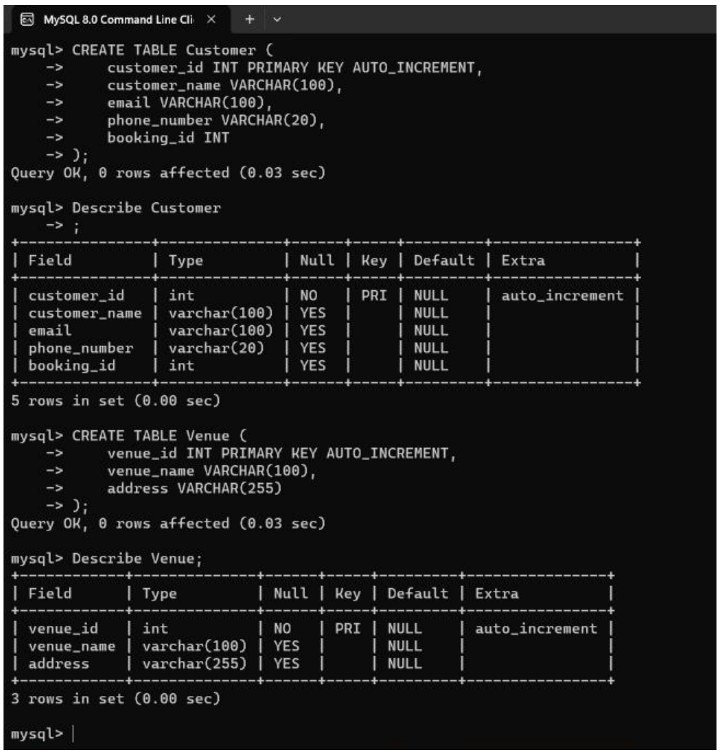
MySQL 8.0 Command Line Cli

```
mysql> CREATE TABLE Booking (
    ->     booking_id INT PRIMARY KEY AUTO_INCREMENT,
    ->     customer_id INT,
    ->     event_id INT,
    ->     num_tickets INT,
    ->     total_cost DECIMAL(10, 2),
    ->     booking_date DATE,
    ->     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> Describe Booking;
+--------------+---------------+------+-----+---------+----------------+
| Field        | Type          | Null | Key | Default | Extra          |
+--------------+---------------+------+-----+---------+----------------+
| booking_id   | int           | NO   | PRI | NULL    | auto_increment |
| 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 | date          | YES  |     | NULL    |                |
+--------------+---------------+------+-----+---------+----------------+
6 rows in set (0.00 sec)

mysql>
```

3. Create an ERD (Entity Relationship Diagram) for the database.

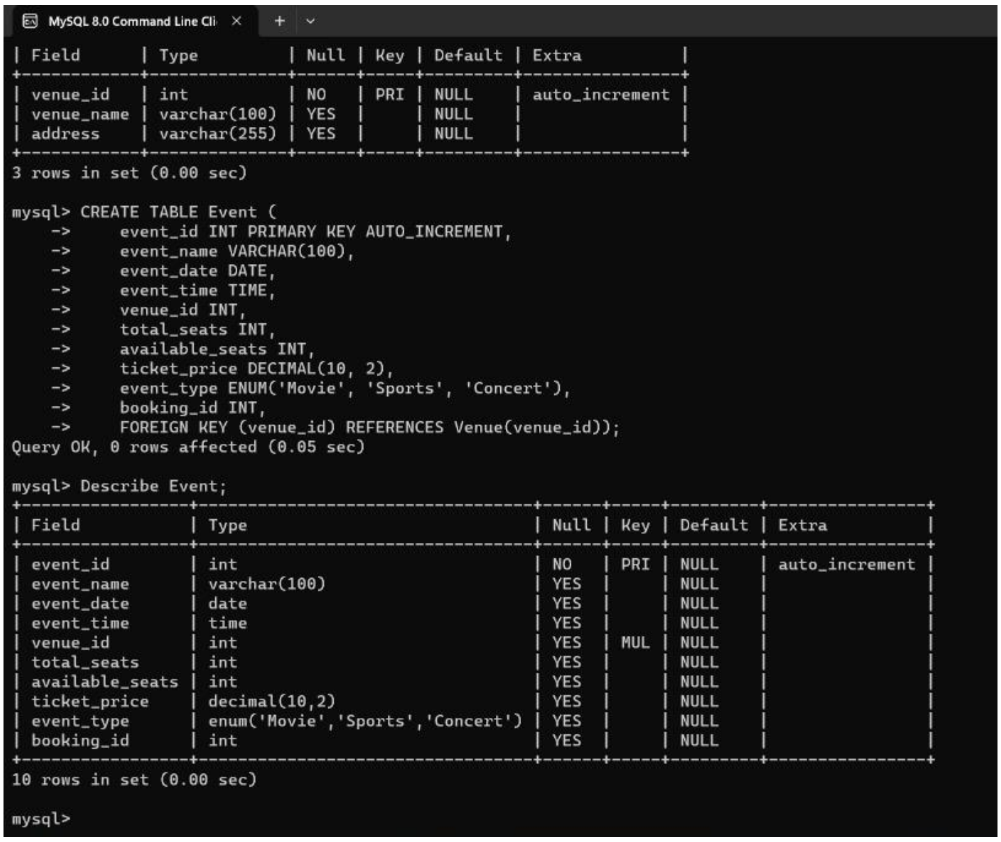


4. Create appropriate Primary Key and Foreign Key constraints for referential integrity.
   1. Venue



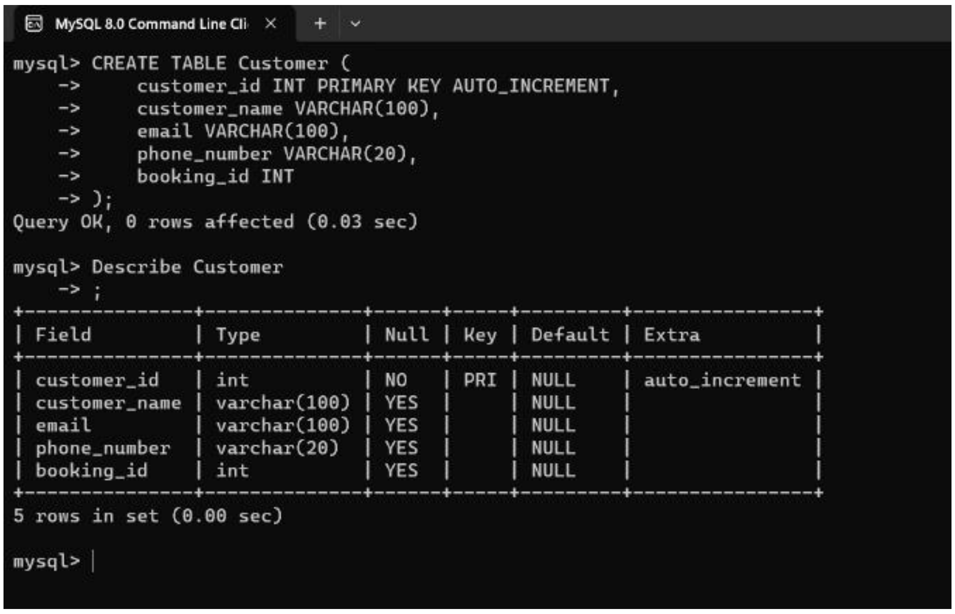
```
mysql> CREATE TABLE Customer (
    ->     customer_id INT PRIMARY KEY AUTO_INCREMENT,
    ->     customer_name VARCHAR(100),
    ->     email VARCHAR(100),
    ->     phone_number VARCHAR(20),
    ->     booking_id INT
    -> );
Query OK, 0 rows affected (0.03 sec)

mysql> Describe Customer
    -> ;
+---------------+--------------+------+-----+---------+----------------+
| Field         | Type         | Null | Key | Default | Extra          |
+---------------+--------------+------+-----+---------+----------------+
| customer_id   | int          | NO   | PRI | NULL    | auto_increment |
| customer_name | varchar(100) | YES  |     | NULL    |                |
| email         | varchar(100) | YES  |     | NULL    |                |
| phone_number  | varchar(20)  | YES  |     | NULL    |                |
| booking_id    | int          | YES  |     | NULL    |                |
+---------------+--------------+------+-----+---------+----------------+
5 rows in set (0.00 sec)

mysql> CREATE TABLE Venue (
    ->     venue_id INT PRIMARY KEY AUTO_INCREMENT,
    ->     venue_name VARCHAR(100),
    ->     address VARCHAR(255)
    -> );
Query OK, 0 rows affected (0.03 sec)

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

mysql>
```

   2. Event



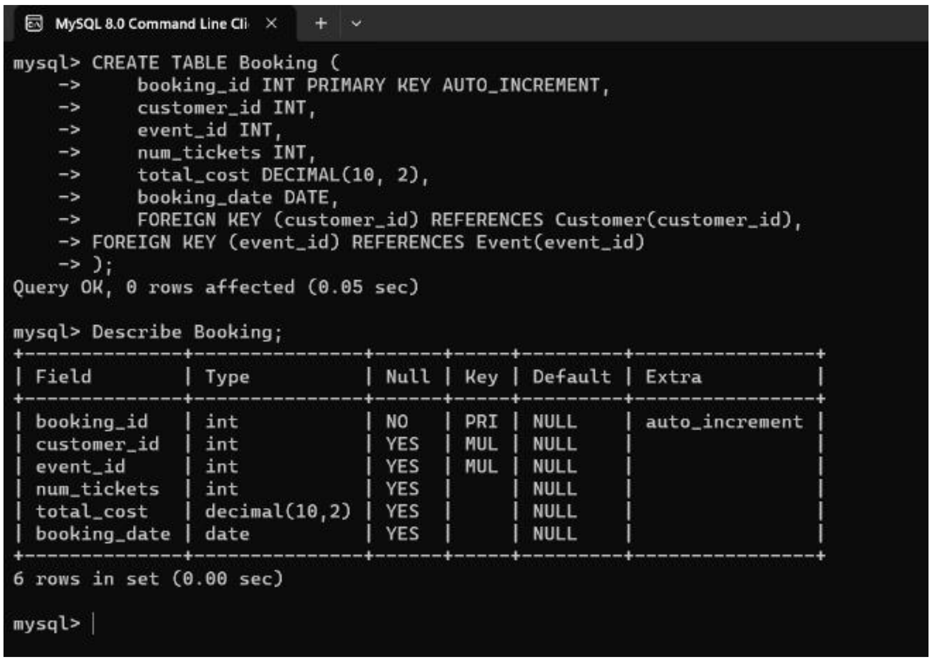
```
| Field      | Type         | Null | Key | Default | Extra          |
+------------+--------------+------+-----+---------+----------------+
| venue_id   | int          | NO   | PRI | NULL    | auto_increment |
| venue_name | varchar(100) | YES  |     | NULL    |                |
| address    | varchar(255)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

mysql> CREATE TABLE Event (
    ->     event_id INT PRIMARY KEY AUTO_INCREMENT,
    ->     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> Describe Event;
+-----------------+-----------------------------------+------+-----+---------+----------------+
| Field           | Type                              | Null | Key | Default | Extra          |
+-----------------+-----------------------------------+------+-----+---------+----------------+
| event_id        | int                               | NO   | PRI | NULL    | auto_increment |
| 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>
```

3. Customer



```
mysql> CREATE TABLE Customer (
    ->     customer_id INT PRIMARY KEY AUTO_INCREMENT,
    ->     customer_name VARCHAR(100),
    ->     email VARCHAR(100),
    ->     phone_number VARCHAR(20),
    ->     booking_id INT
    -> );
Query OK, 0 rows affected (0.03 sec)

mysql> Describe Customer
    -> ;
+---------------+--------------+------+-----+---------+----------------+
| Field         | Type         | Null | Key | Default | Extra          |
+---------------+--------------+------+-----+---------+----------------+
| customer_id   | int          | NO   | PRI | NULL    | auto_increment |
| customer_name | varchar(100) | YES  |     | NULL    |                |
| email         | varchar(100) | YES  |     | NULL    |                |
| phone_number  | varchar(20)  | YES  |     | NULL    |                |
| booking_id    | int          | YES  |     | NULL    |                |
+---------------+--------------+------+-----+---------+----------------+
5 rows in set (0.00 sec)

mysql>
```

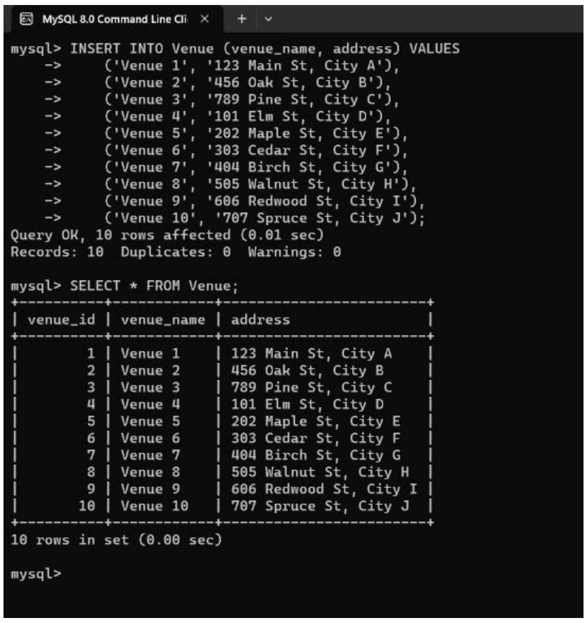
4. Booking



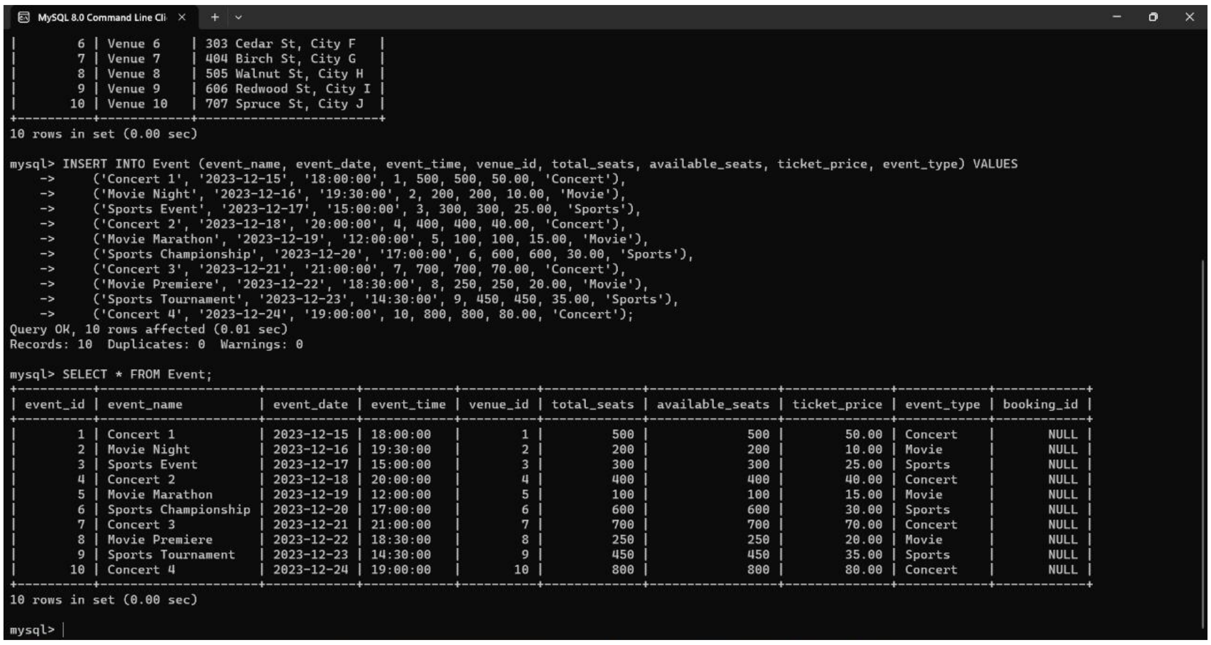
```
mysql> CREATE TABLE Booking (
    ->     booking_id INT PRIMARY KEY AUTO_INCREMENT,
    ->     customer_id INT,
    ->     event_id INT,
    ->     num_tickets INT,
    ->     total_cost DECIMAL(10, 2),
    ->     booking_date DATE,
    ->     FOREIGN KEY (customer_id) REFERENCES Customer(customer_id),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

mysql> Describe Booking;
+--------------+---------------+------+-----+---------+----------------+
| Field        | Type          | Null | Key | Default | Extra          |
+--------------+---------------+------+-----+---------+----------------+
| booking_id   | int           | NO   | PRI | NULL    | auto_increment |
| 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 | date          | YES  |     | NULL    |                |
+--------------+---------------+------+-----+---------+----------------+
6 rows in set (0.00 sec)

mysql>
```

Task-2:

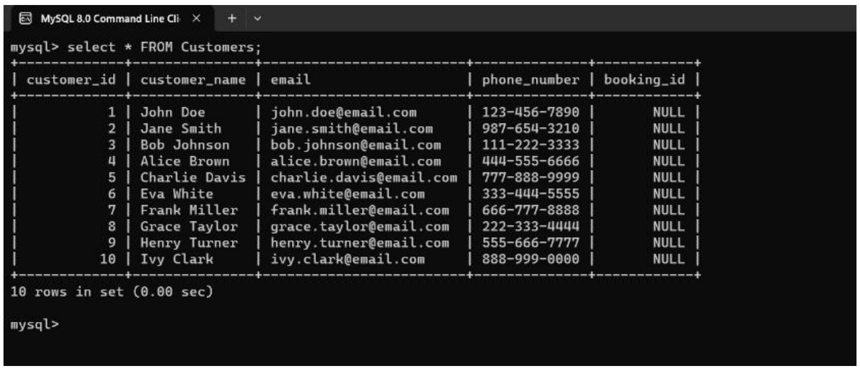
1. Write a SQL query to insert at least 10 sample records into each table.



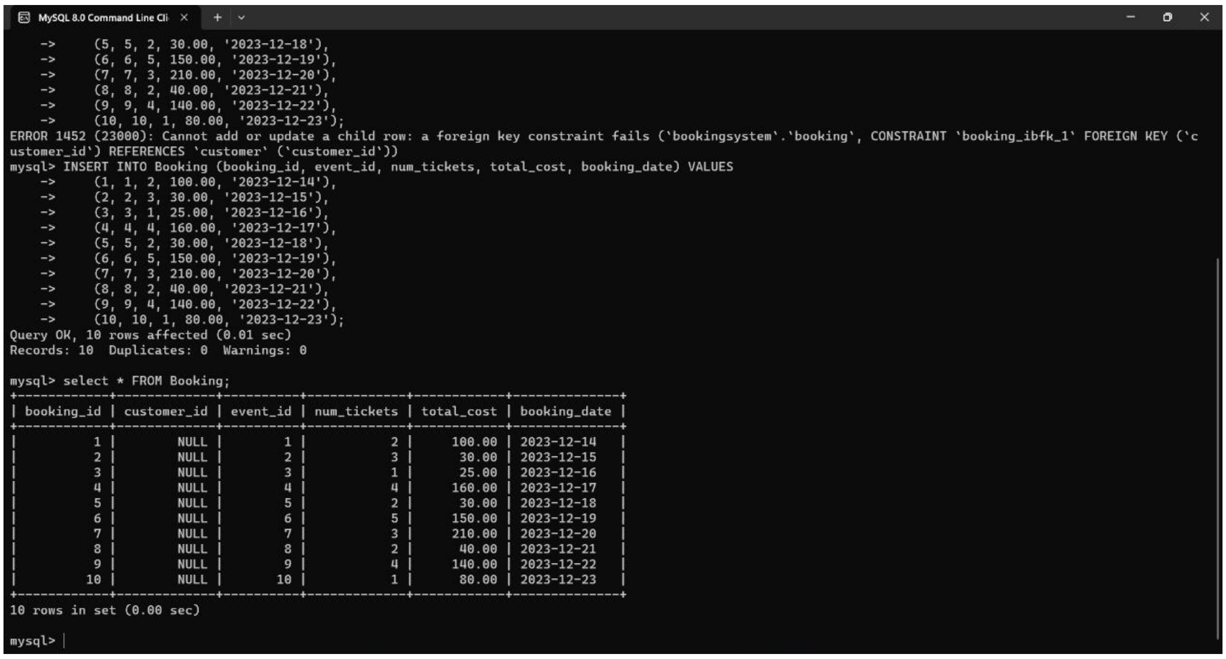
```
mysql> INSERT INTO Venue (venue_name, address) VALUES
    ->     ('Venue 1', '123 Main St, City A'),
    ->     ('Venue 2', '456 Oak St, City B'),
    ->     ('Venue 3', '789 Pine St, City C'),
    ->     ('Venue 4', '101 Elm St, City D'),
    ->     ('Venue 5', '202 Maple St, City E'),
    ->     ('Venue 6', '303 Cedar St, City F'),
    ->     ('Venue 7', '404 Birch St, City G'),
    ->     ('Venue 8', '505 Walnut St, City H'),
    ->     ('Venue 9', '606 Redwood St, City I'),
    ->     ('Venue 10', '707 Spruce St, City J');
Query OK, 10 rows affected (0.01 sec)
Records: 10  Duplicates: 0  Warnings: 0

mysql> SELECT * FROM Venue;
+----------+------------+------------------------+
| venue_id | venue_name | address                |
+----------+------------+------------------------+
|        1 | Venue 1    | 123 Main St, City A    |
|        2 | Venue 2    | 456 Oak St, City B     |
|        3 | Venue 3    | 789 Pine St, City C    |
|        4 | Venue 4    | 101 Elm St, City D     |
|        5 | Venue 5    | 202 Maple St, City E   |
|        6 | Venue 6    | 303 Cedar St, City F   |
|        7 | Venue 7    | 404 Birch St, City G   |
|        8 | Venue 8    | 505 Walnut St, City H  |
|        9 | Venue 9    | 606 Redwood St, City I |
|       10 | Venue 10   | 707 Spruce St, City J  |
+----------+------------+------------------------+
10 rows in set (0.00 sec)

mysql>
```



```
|        6 | Venue 6    | 303 Cedar St, City F   |
|        7 | Venue 7    | 404 Birch St, City G   |
|        8 | Venue 8    | 505 Walnut St, City H  |
|        9 | Venue 9    | 606 Redwood St, City I |
|       10 | Venue 10   | 707 Spruce St, City J  |
+----------+------------+------------------------+
10 rows in set (0.00 sec)

mysql> INSERT INTO Event (event_name, event_date, event_time, venue_id, total_seats, available_seats, ticket_price, event_type) VALUES
    ->     ('Concert 1', '2023-12-15', '18:00:00', 1, 500, 500, 50.00, 'Concert'),
    ->     ('Movie Night', '2023-12-16', '19:30:00', 2, 200, 200, 10.00, 'Movie'),
    ->     ('Sports Event', '2023-12-17', '15:00:00', 3, 300, 300, 25.00, 'Sports'),
    ->     ('Concert 2', '2023-12-18', '20:00:00', 4, 400, 400, 40.00, 'Concert'),
    ->     ('Movie Marathon', '2023-12-19', '12:00:00', 5, 100, 100, 15.00, 'Movie'),
    ->     ('Sports Championship', '2023-12-20', '17:00:00', 6, 600, 600, 30.00, 'Sports'),
    ->     ('Concert 3', '2023-12-21', '21:00:00', 7, 700, 700, 70.00, 'Concert'),
    ->     ('Movie Premiere', '2023-12-22', '18:30:00', 8, 250, 250, 20.00, 'Movie'),
    ->     ('Sports Tournament', '2023-12-23', '14:30:00', 9, 450, 450, 35.00, 'Sports'),
    ->     ('Concert 4', '2023-12-24', '19:00:00', 10, 800, 800, 80.00, 'Concert');
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 1           | 2023-12-15 | 18:00:00   |        1 |         500 |             500 |        50.00 | Concert    |       NULL |
|        2 | Movie Night         | 2023-12-16 | 19:30:00   |        2 |         200 |             200 |        10.00 | Movie      |       NULL |
|        3 | Sports Event        | 2023-12-17 | 15:00:00   |        3 |         300 |             300 |        25.00 | Sports     |       NULL |
|        4 | Concert 2           | 2023-12-18 | 20:00:00   |        4 |         400 |             400 |        40.00 | Concert    |       NULL |
|        5 | Movie Marathon      | 2023-12-19 | 12:00:00   |        5 |         100 |             100 |        15.00 | Movie      |       NULL |
|        6 | Sports Championship | 2023-12-20 | 17:00:00   |        6 |         600 |             600 |        30.00 | Sports     |       NULL |
|        7 | Concert 3           | 2023-12-21 | 21:00:00   |        7 |         700 |             700 |        70.00 | Concert    |       NULL |
|        8 | Movie Premiere      | 2023-12-22 | 18:30:00   |        8 |         250 |             250 |        20.00 | Movie      |       NULL |
|        9 | Sports Tournament   | 2023-12-23 | 14:30:00   |        9 |         450 |             450 |        35.00 | Sports     |       NULL |
|       10 | Concert 4           | 2023-12-24 | 19:00:00   |       10 |         800 |             800 |        80.00 | Concert    |       NULL |
+----------+---------------------+------------+------------+----------+-------------+-----------------+--------------+------------+------------+
10 rows in set (0.00 sec)

mysql> |
```

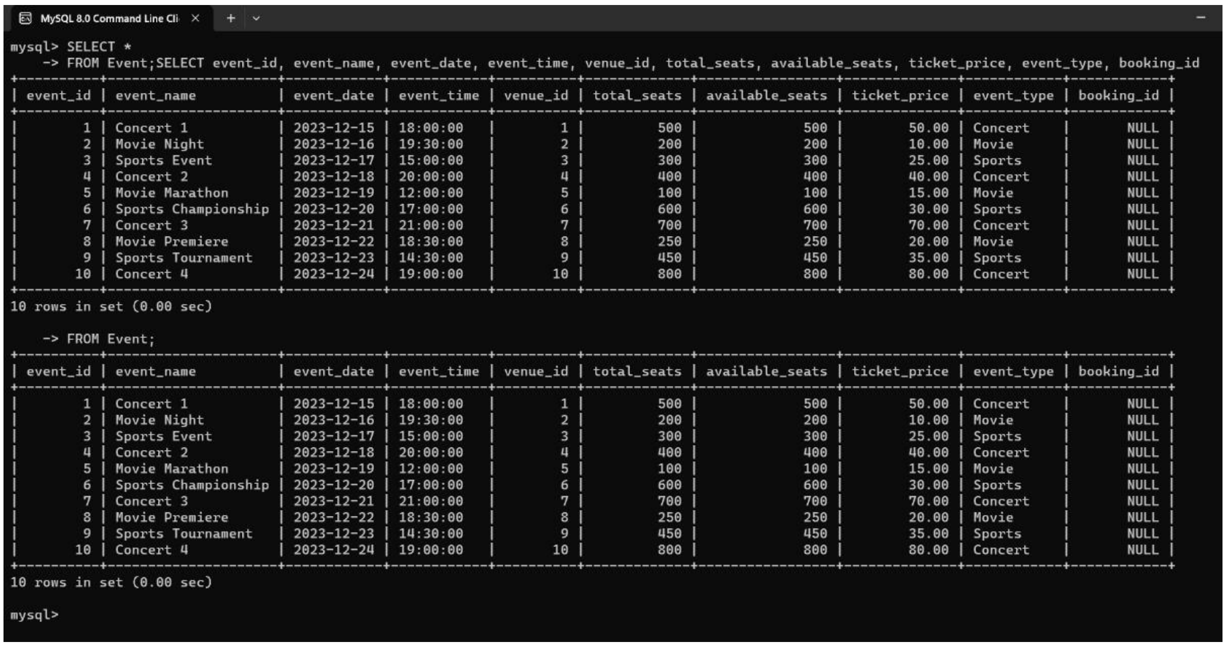


```
mysql> select * FROM Customers;
+-------------+---------------+--------------------------+--------------+------------+
| customer_id | customer_name | email                    | phone_number | booking_id |
+-------------+---------------+--------------------------+--------------+------------+
|           1 | John Doe      | john.doe@email.com       | 123-456-7890 |       NULL |
|           2 | Jane Smith    | jane.smith@email.com     | 987-654-3210 |       NULL |
|           3 | Bob Johnson   | bob.johnson@email.com    | 111-222-3333 |       NULL |
|           4 | Alice Brown   | alice.brown@email.com    | 444-555-6666 |       NULL |
|           5 | Charlie Davis | charlie.davis@email.com  | 777-888-9999 |       NULL |
|           6 | Eva White     | eva.white@email.com      | 333-444-5555 |       NULL |
|           7 | Frank Miller  | frank.miller@email.com   | 666-777-8888 |       NULL |
|           8 | Grace Taylor  | grace.taylor@email.com   | 222-333-4444 |       NULL |
|           9 | Henry Turner  | henry.turner@email.com   | 555-666-7777 |       NULL |
|          10 | Ivy Clark     | ivy.clark@email.com      | 888-999-0000 |       NULL |
+-------------+---------------+--------------------------+--------------+------------+
10 rows in set (0.00 sec)

mysql>
```



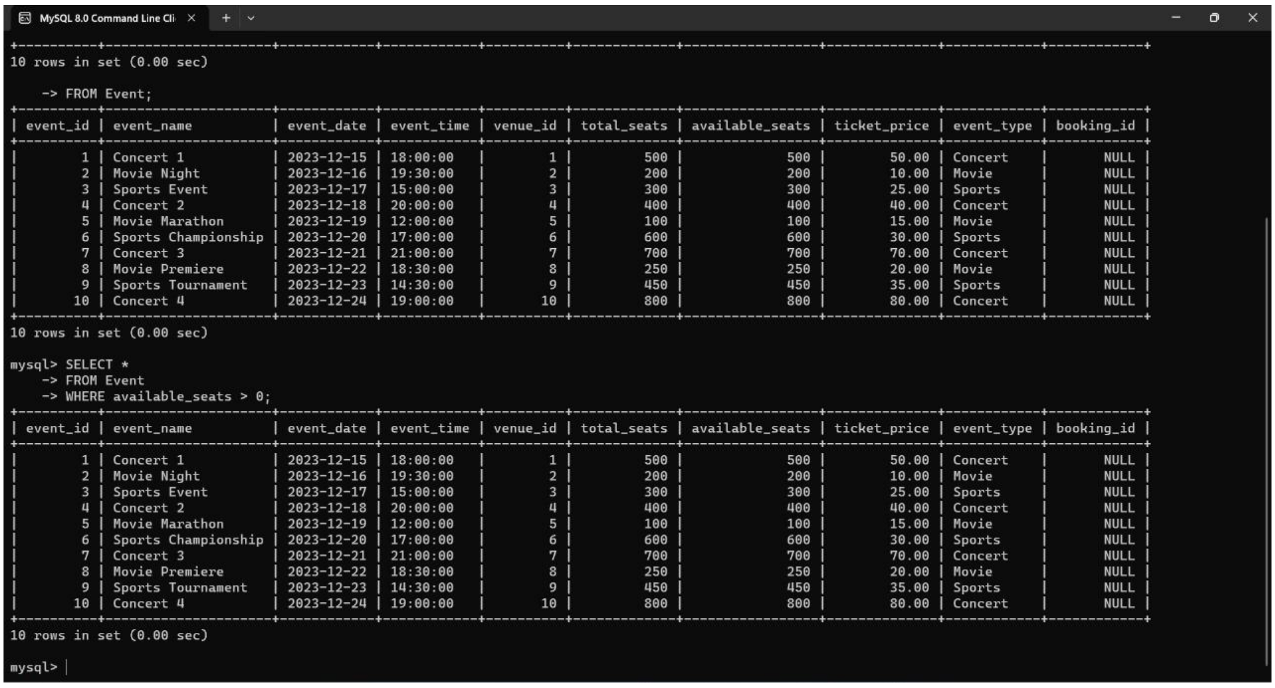
```
    ->     (5, 5, 2, 30.00, '2023-12-18'),
    ->     (6, 6, 5, 150.00, '2023-12-19'),
    ->     (7, 7, 3, 210.00, '2023-12-20'),
    ->     (8, 8, 2, 40.00, '2023-12-21'),
    ->     (9, 9, 4, 140.00, '2023-12-22'),
    ->     (10, 10, 1, 80.00, '2023-12-23');
ERROR 1452 (23000): Cannot add or update a child row: a foreign key constraint fails (`bookingsystem`.`booking`, CONSTRAINT `booking_ibfk_1` FOREIGN KEY (`c
ustomer_id`) REFERENCES `customer` (`customer_id`))
mysql> INSERT INTO Booking (booking_id, event_id, num_tickets, total_cost, booking_date) VALUES
    ->     (1, 1, 2, 100.00, '2023-12-14'),
    ->     (2, 2, 3, 30.00, '2023-12-15'),
    ->     (3, 3, 1, 25.00, '2023-12-16'),
    ->     (4, 4, 4, 160.00, '2023-12-17'),
    ->     (5, 5, 2, 30.00, '2023-12-18'),
    ->     (6, 6, 5, 150.00, '2023-12-19'),
    ->     (7, 7, 3, 210.00, '2023-12-20'),
    ->     (8, 8, 2, 40.00, '2023-12-21'),
    ->     (9, 9, 4, 140.00, '2023-12-22'),
    ->     (10, 10, 1, 80.00, '2023-12-23');
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 |        NULL |        1 |           2 |     100.00 | 2023-12-14   |
|          2 |        NULL |        2 |           3 |      30.00 | 2023-12-15   |
|          3 |        NULL |        3 |           1 |      25.00 | 2023-12-16   |
|          4 |        NULL |        4 |           4 |     160.00 | 2023-12-17   |
|          5 |        NULL |        5 |           2 |      30.00 | 2023-12-18   |
|          6 |        NULL |        6 |           5 |     150.00 | 2023-12-19   |
|          7 |        NULL |        7 |           3 |     210.00 | 2023-12-20   |
|          8 |        NULL |        8 |           2 |      40.00 | 2023-12-21   |
|          9 |        NULL |        9 |           4 |     140.00 | 2023-12-22   |
|         10 |        NULL |       10 |           1 |      80.00 | 2023-12-23   |
+------------+-------------+----------+-------------+------------+--------------+
10 rows in set (0.00 sec)

mysql>
```

2. Write a SQL query to list all Events.



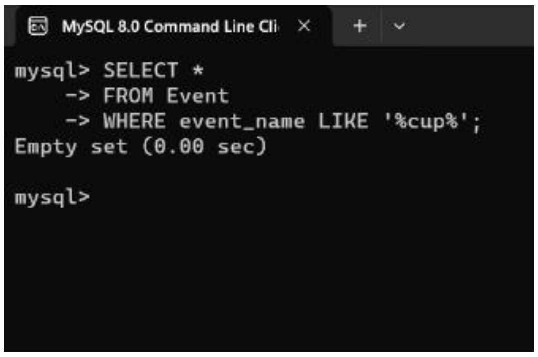
```
mysql> SELECT *
    -> FROM Event;SELECT event_id, event_name, event_date, event_time, venue_id, total_seats, available_seats, ticket_price, event_type, booking_id
+----------+---------------------+------------+------------+----------+-------------+-----------------+--------------+------------+------------+
| event_id | event_name          | event_date | event_time | venue_id | total_seats | available_seats | ticket_price | event_type | booking_id |
+----------+---------------------+------------+------------+----------+-------------+-----------------+--------------+------------+------------+
|        1 | Concert 1           | 2023-12-15 | 18:00:00   |        1 |         500 |             500 |        50.00 | Concert    |       NULL |
|        2 | Movie Night         | 2023-12-16 | 19:30:00   |        2 |         200 |             200 |        10.00 | Movie      |       NULL |
|        3 | Sports Event        | 2023-12-17 | 15:00:00   |        3 |         300 |             300 |        25.00 | Sports     |       NULL |
|        4 | Concert 2           | 2023-12-18 | 20:00:00   |        4 |         400 |             400 |        40.00 | Concert    |       NULL |
|        5 | Movie Marathon      | 2023-12-19 | 12:00:00   |        5 |         100 |             100 |        15.00 | Movie      |       NULL |
|        6 | Sports Championship | 2023-12-20 | 17:00:00   |        6 |         600 |             600 |        30.00 | Sports     |       NULL |
|        7 | Concert 3           | 2023-12-21 | 21:00:00   |        7 |         700 |             700 |        70.00 | Concert    |       NULL |
|        8 | Movie Premiere      | 2023-12-22 | 18:30:00   |        8 |         250 |             250 |        20.00 | Movie      |       NULL |
|        9 | Sports Tournament   | 2023-12-23 | 14:30:00   |        9 |         450 |             450 |        35.00 | Sports     |       NULL |
|       10 | Concert 4           | 2023-12-24 | 19:00:00   |       10 |         800 |             800 |        80.00 | Concert    |       NULL |
+----------+---------------------+------------+------------+----------+-------------+-----------------+--------------+------------+------------+
10 rows in set (0.00 sec)

    -> FROM Event;
+----------+---------------------+------------+------------+----------+-------------+-----------------+--------------+------------+------------+
| event_id | event_name          | event_date | event_time | venue_id | total_seats | available_seats | ticket_price | event_type | booking_id |
+----------+---------------------+------------+------------+----------+-------------+-----------------+--------------+------------+------------+
|        1 | Concert 1           | 2023-12-15 | 18:00:00   |        1 |         500 |             500 |        50.00 | Concert    |       NULL |
|        2 | Movie Night         | 2023-12-16 | 19:30:00   |        2 |         200 |             200 |        10.00 | Movie      |       NULL |
|        3 | Sports Event        | 2023-12-17 | 15:00:00   |        3 |         300 |             300 |        25.00 | Sports     |       NULL |
|        4 | Concert 2           | 2023-12-18 | 20:00:00   |        4 |         400 |             400 |        40.00 | Concert    |       NULL |
|        5 | Movie Marathon      | 2023-12-19 | 12:00:00   |        5 |         100 |             100 |        15.00 | Movie      |       NULL |
|        6 | Sports Championship | 2023-12-20 | 17:00:00   |        6 |         600 |             600 |        30.00 | Sports     |       NULL |
|        7 | Concert 3           | 2023-12-21 | 21:00:00   |        7 |         700 |             700 |        70.00 | Concert    |       NULL |
|        8 | Movie Premiere      | 2023-12-22 | 18:30:00   |        8 |         250 |             250 |        20.00 | Movie      |       NULL |
|        9 | Sports Tournament   | 2023-12-23 | 14:30:00   |        9 |         450 |             450 |        35.00 | Sports     |       NULL |
|       10 | Concert 4           | 2023-12-24 | 19:00:00   |       10 |         800 |             800 |        80.00 | Concert    |       NULL |
+----------+---------------------+------------+------------+----------+-------------+-----------------+--------------+------------+------------+
10 rows in set (0.00 sec)

mysql>
```

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



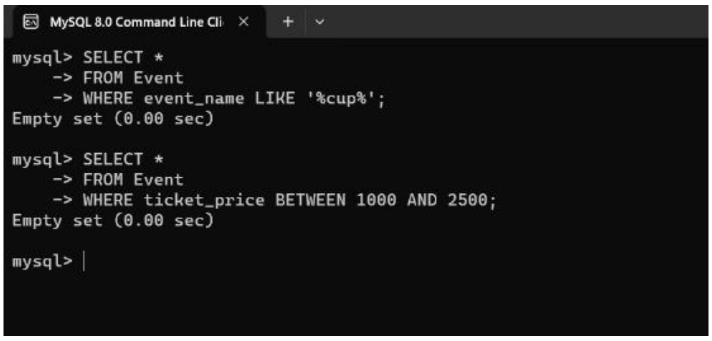
```
+----------+---------------------+------------+------------+----------+-------------+-----------------+--------------+------------+------------+
10 rows in set (0.00 sec)

    -> FROM Event;
+----------+---------------------+------------+------------+----------+-------------+-----------------+--------------+------------+------------+
| event_id | event_name          | event_date | event_time | venue_id | total_seats | available_seats | ticket_price | event_type | booking_id |
+----------+---------------------+------------+------------+----------+-------------+-----------------+--------------+------------+------------+
|        1 | Concert 1           | 2023-12-15 | 18:00:00   |        1 |         500 |             500 |        50.00 | Concert    |       NULL |
|        2 | Movie Night         | 2023-12-16 | 19:30:00   |        2 |         200 |             200 |        10.00 | Movie      |       NULL |
|        3 | Sports Event        | 2023-12-17 | 15:00:00   |        3 |         300 |             300 |        25.00 | Sports     |       NULL |
|        4 | Concert 2           | 2023-12-18 | 20:00:00   |        4 |         400 |             400 |        40.00 | Concert    |       NULL |
|        5 | Movie Marathon      | 2023-12-19 | 12:00:00   |        5 |         100 |             100 |        15.00 | Movie      |       NULL |
|        6 | Sports Championship | 2023-12-20 | 17:00:00   |        6 |         600 |             600 |        30.00 | Sports     |       NULL |
|        7 | Concert 3           | 2023-12-21 | 21:00:00   |        7 |         700 |             700 |        70.00 | Concert    |       NULL |
|        8 | Movie Premiere      | 2023-12-22 | 18:30:00   |        8 |         250 |             250 |        20.00 | Movie      |       NULL |
|        9 | Sports Tournament   | 2023-12-23 | 14:30:00   |        9 |         450 |             450 |        35.00 | Sports     |       NULL |
|       10 | Concert 4           | 2023-12-24 | 19:00:00   |       10 |         800 |             800 |        80.00 | Concert    |       NULL |
+----------+---------------------+------------+------------+----------+-------------+-----------------+--------------+------------+------------+
10 rows in set (0.00 sec)

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 1           | 2023-12-15 | 18:00:00   |        1 |         500 |             500 |        50.00 | Concert    |       NULL |
|        2 | Movie Night         | 2023-12-16 | 19:30:00   |        2 |         200 |             200 |        10.00 | Movie      |       NULL |
|        3 | Sports Event        | 2023-12-17 | 15:00:00   |        3 |         300 |             300 |        25.00 | Sports     |       NULL |
|        4 | Concert 2           | 2023-12-18 | 20:00:00   |        4 |         400 |             400 |        40.00 | Concert    |       NULL |
|        5 | Movie Marathon      | 2023-12-19 | 12:00:00   |        5 |         100 |             100 |        15.00 | Movie      |       NULL |
|        6 | Sports Championship | 2023-12-20 | 17:00:00   |        6 |         600 |             600 |        30.00 | Sports     |       NULL |
|        7 | Concert 3           | 2023-12-21 | 21:00:00   |        7 |         700 |             700 |        70.00 | Concert    |       NULL |
|        8 | Movie Premiere      | 2023-12-22 | 18:30:00   |        8 |         250 |             250 |        20.00 | Movie      |       NULL |
|        9 | Sports Tournament   | 2023-12-23 | 14:30:00   |        9 |         450 |             450 |        35.00 | Sports     |       NULL |
|       10 | Concert 4           | 2023-12-24 | 19:00:00   |       10 |         800 |             800 |        80.00 | Concert    |       NULL |
+----------+---------------------+------------+------------+----------+-------------+-----------------+--------------+------------+------------+
10 rows in set (0.00 sec)

mysql>
```

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



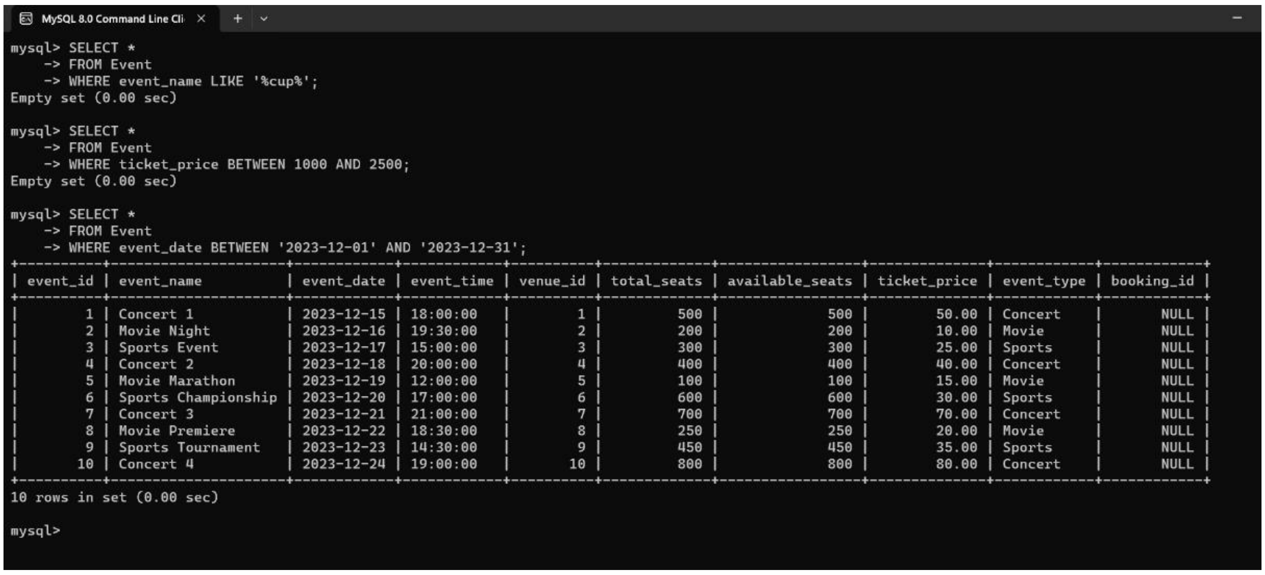
```
mysql> SELECT *
    -> FROM Event
    -> WHERE event_name LIKE '%cup%';
Empty set (0.00 sec)

mysql>
```

5. Write a SQL query to select events with ticket price range is between 1000 to 2500.



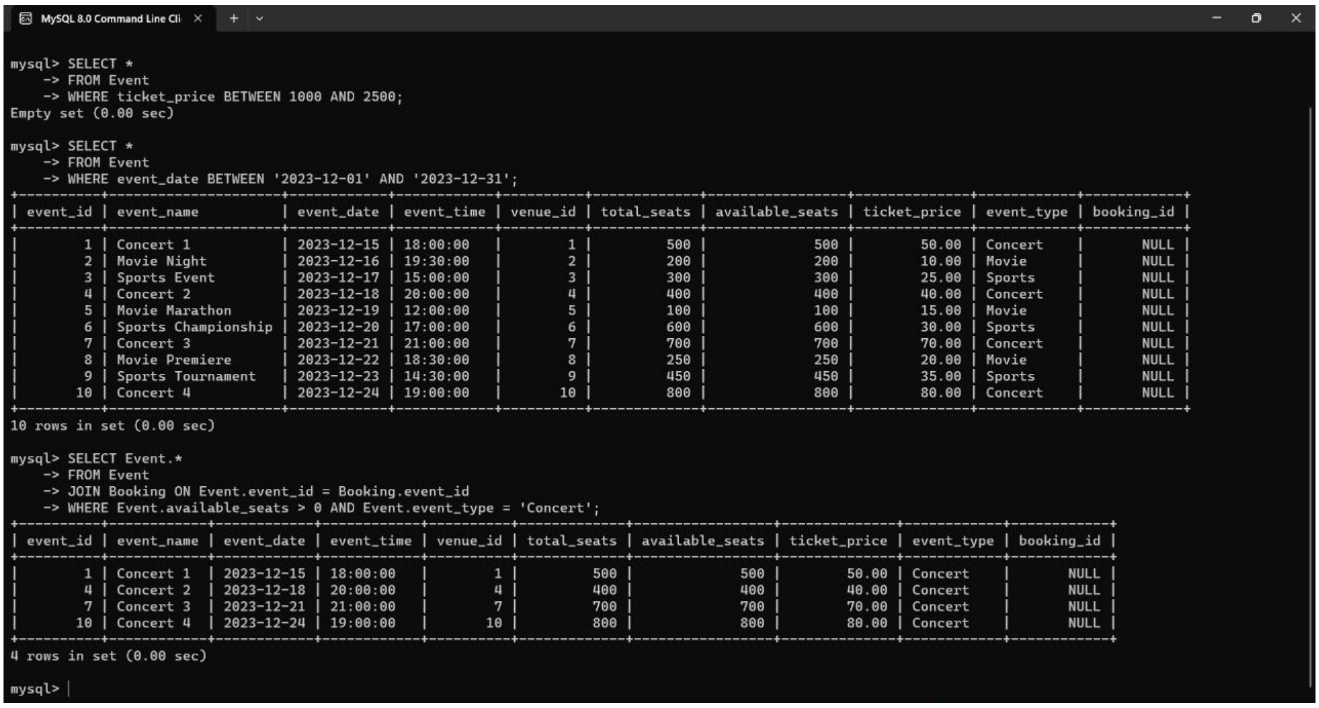
```
mysql> SELECT *
    -> FROM Event
    -> WHERE event_name LIKE '%cup%';
Empty set (0.00 sec)

mysql> SELECT *
    -> FROM Event
    -> WHERE ticket_price BETWEEN 1000 AND 2500;
Empty set (0.00 sec)

mysql>
```

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



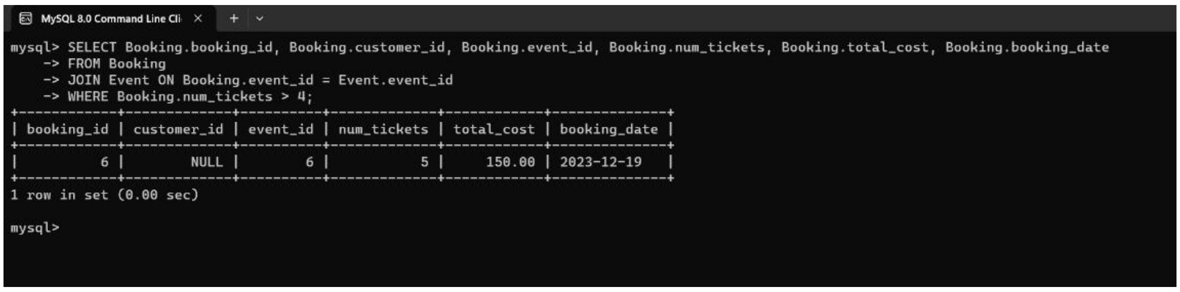
```
mysql> SELECT *
    -> FROM Event
    -> WHERE event_name LIKE '%cup%';
Empty set (0.00 sec)

mysql> SELECT *
    -> FROM Event
    -> WHERE ticket_price BETWEEN 1000 AND 2500;
Empty set (0.00 sec)

mysql> SELECT *
    -> FROM Event
    -> WHERE event_date BETWEEN '2023-12-01' 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 1           | 2023-12-15 | 18:00:00   |        1 |         500 |             500 |        50.00 | Concert    |       NULL |
|        2 | Movie Night         | 2023-12-16 | 19:30:00   |        2 |         200 |             200 |        10.00 | Movie      |       NULL |
|        3 | Sports Event        | 2023-12-17 | 15:00:00   |        3 |         300 |             300 |        25.00 | Sports     |       NULL |
|        4 | Concert 2           | 2023-12-18 | 20:00:00   |        4 |         400 |             400 |        40.00 | Concert    |       NULL |
|        5 | Movie Marathon      | 2023-12-19 | 12:00:00   |        5 |         100 |             100 |        15.00 | Movie      |       NULL |
|        6 | Sports Championship | 2023-12-20 | 17:00:00   |        6 |         600 |             600 |        30.00 | Sports     |       NULL |
|        7 | Concert 3           | 2023-12-21 | 21:00:00   |        7 |         700 |             700 |        70.00 | Concert    |       NULL |
|        8 | Movie Premiere      | 2023-12-22 | 18:30:00   |        8 |         250 |             250 |        20.00 | Movie      |       NULL |
|        9 | Sports Tournament   | 2023-12-23 | 14:30:00   |        9 |         450 |             450 |        35.00 | Sports     |       NULL |
|       10 | Concert 4           | 2023-12-24 | 19:00:00   |       10 |         800 |             800 |        80.00 | Concert    |       NULL |
+----------+---------------------+------------+------------+----------+-------------+-----------------+--------------+------------+------------+
10 rows in set (0.00 sec)

mysql>
```

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



```
mysql> SELECT *
    -> FROM Event
    -> WHERE ticket_price BETWEEN 1000 AND 2500;
Empty set (0.00 sec)

mysql> SELECT *
    -> FROM Event
    -> WHERE event_date BETWEEN '2023-12-01' 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 1           | 2023-12-15 | 18:00:00   |        1 |         500 |             500 |        50.00 | Concert    |       NULL |
|        2 | Movie Night         | 2023-12-16 | 19:30:00   |        2 |         200 |             200 |        10.00 | Movie      |       NULL |
|        3 | Sports Event        | 2023-12-17 | 15:00:00   |        3 |         300 |             300 |        25.00 | Sports     |       NULL |
|        4 | Concert 2           | 2023-12-18 | 20:00:00   |        4 |         400 |             400 |        40.00 | Concert    |       NULL |
|        5 | Movie Marathon      | 2023-12-19 | 12:00:00   |        5 |         100 |             100 |        15.00 | Movie      |       NULL |
|        6 | Sports Championship | 2023-12-20 | 17:00:00   |        6 |         600 |             600 |        30.00 | Sports     |       NULL |
|        7 | Concert 3           | 2023-12-21 | 21:00:00   |        7 |         700 |             700 |        70.00 | Concert    |       NULL |
|        8 | Movie Premiere      | 2023-12-22 | 18:30:00   |        8 |         250 |             250 |        20.00 | Movie      |       NULL |
|        9 | Sports Tournament   | 2023-12-23 | 14:30:00   |        9 |         450 |             450 |        35.00 | Sports     |       NULL |
|       10 | Concert 4           | 2023-12-24 | 19:00:00   |       10 |         800 |             800 |        80.00 | Concert    |       NULL |
+----------+---------------------+------------+------------+----------+-------------+-----------------+--------------+------------+------------+
10 rows in set (0.00 sec)

mysql> SELECT Event.*
    -> FROM Event
    -> JOIN Booking ON Event.event_id = Booking.event_id
    -> WHERE Event.available_seats > 0 AND Event.event_type = 'Concert';
+----------+------------+------------+------------+----------+-------------+-----------------+--------------+------------+------------+
| event_id | event_name | event_date | event_time | venue_id | total_seats | available_seats | ticket_price | event_type | booking_id |
+----------+------------+------------+------------+----------+-------------+-----------------+--------------+------------+------------+
|        1 | Concert 1  | 2023-12-15 | 18:00:00   |        1 |         500 |             500 |        50.00 | Concert    |       NULL |
|        4 | Concert 2  | 2023-12-18 | 20:00:00   |        4 |         400 |             400 |        40.00 | Concert    |       NULL |
|        7 | Concert 3  | 2023-12-21 | 21:00:00   |        7 |         700 |             700 |        70.00 | Concert    |       NULL |
|       10 | Concert 4  | 2023-12-24 | 19:00:00   |       10 |         800 |             800 |        80.00 | Concert    |       NULL |
+----------+------------+------------+------------+----------+-------------+-----------------+--------------+------------+------------+
4 rows in set (0.00 sec)

mysql>
```

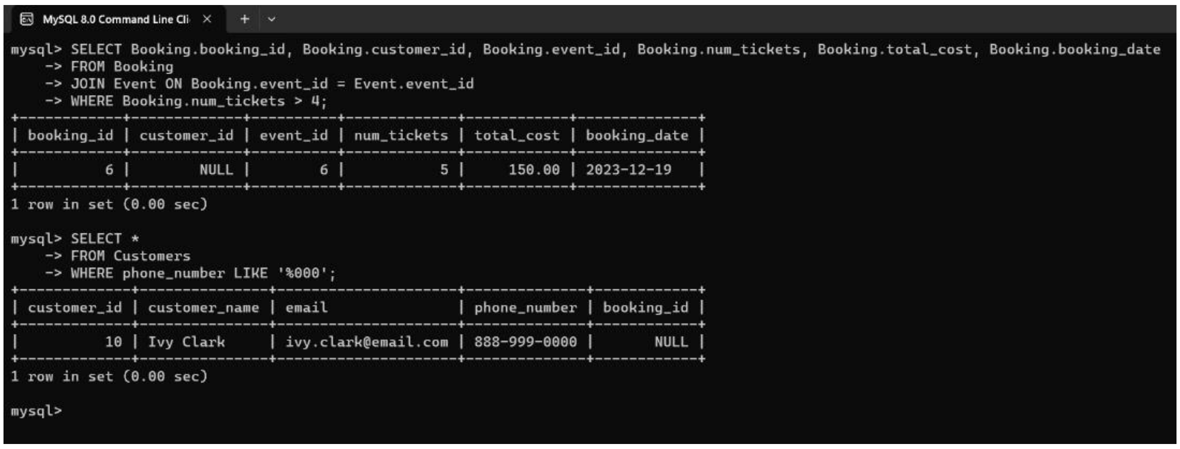
9 .Write a SQL query to retrieve bookings details contains booked no of ticket more than 4.



MySQL 8.0 Command Line Cli

```
mysql> SELECT Booking.booking_id, Booking.customer_id, Booking.event_id, Booking.num_tickets, Booking.total_cost, Booking.booking_date
    -> FROM Booking
    -> JOIN Event ON Booking.event_id = Event.event_id
    -> WHERE Booking.num_tickets > 4;
+------------+-------------+----------+-------------+------------+--------------+
| booking_id | customer_id | event_id | num_tickets | total_cost | booking_date |
+------------+-------------+----------+-------------+------------+--------------+
|          6 |        NULL |        6 |           5 |     150.00 | 2023-12-19   |
+------------+-------------+----------+-------------+------------+--------------+
1 row in set (0.00 sec)

mysql>
```

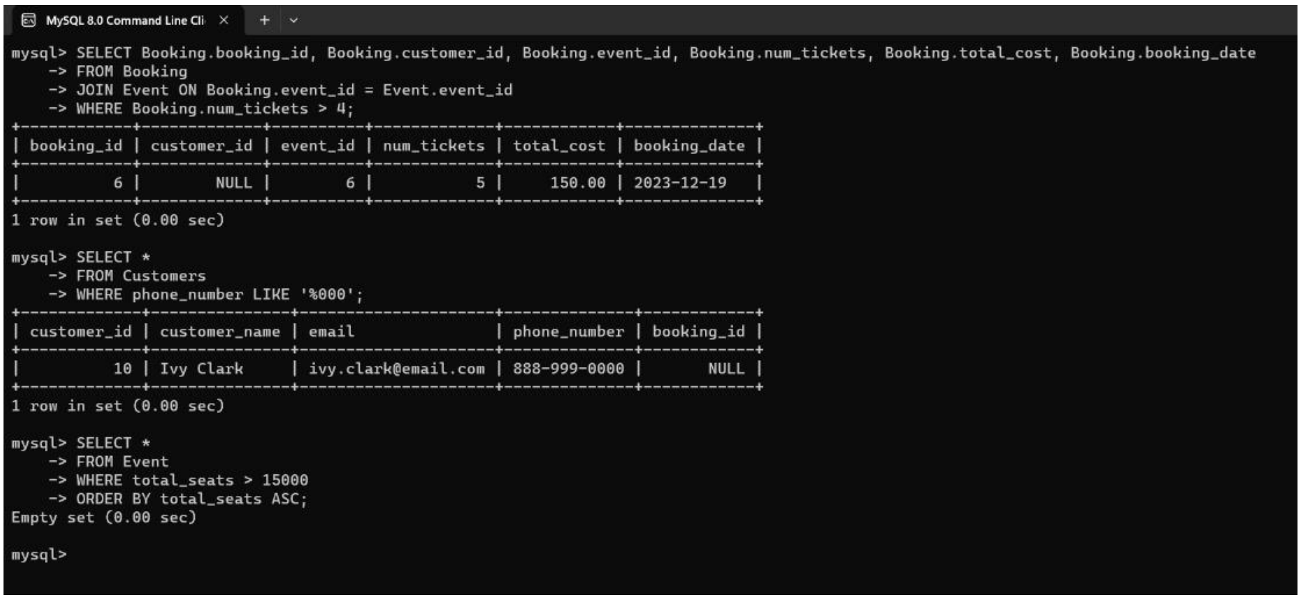
10. Write a SQL query to retrieve customer information whose phone number end with ‘000’



MySQL 8.0 Command Line Cli

```
mysql> SELECT Booking.booking_id, Booking.customer_id, Booking.event_id, Booking.num_tickets, Booking.total_cost, Booking.booking_date
    -> FROM Booking
    -> JOIN Event ON Booking.event_id = Event.event_id
    -> WHERE Booking.num_tickets > 4;
+------------+-------------+----------+-------------+------------+--------------+
| booking_id | customer_id | event_id | num_tickets | total_cost | booking_date |
+------------+-------------+----------+-------------+------------+--------------+
|          6 |        NULL |        6 |           5 |     150.00 | 2023-12-19   |
+------------+-------------+----------+-------------+------------+--------------+
1 row in set (0.00 sec)

mysql> SELECT *
    -> FROM Customers
    -> WHERE phone_number LIKE '%000';
+-------------+---------------+---------------------+--------------+------------+
| customer_id | customer_name | email               | phone_number | booking_id |
+-------------+---------------+---------------------+--------------+------------+
|          10 | Ivy Clark     | ivy.clark@email.com | 888-999-0000 |       NULL |
+-------------+---------------+---------------------+--------------+------------+
1 row in set (0.00 sec)

mysql>
```

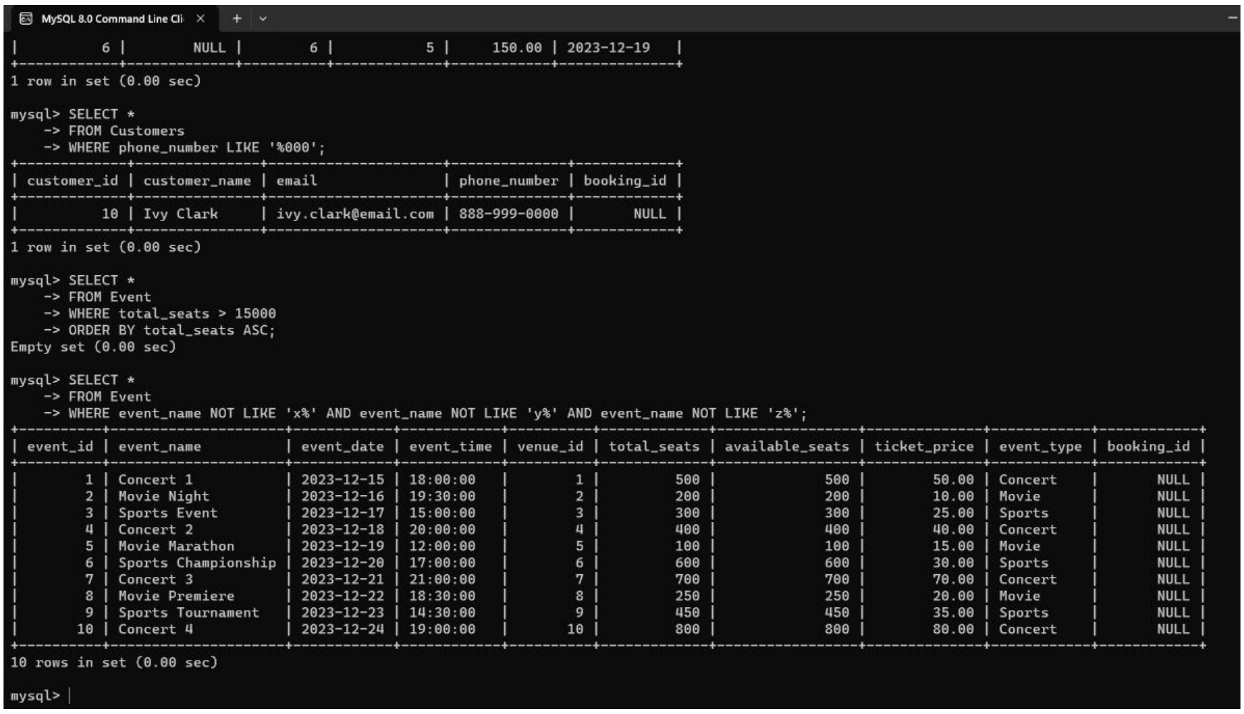
11. Write a SQL query to retrieve the events in order whose seat capacity more than 15000.



MySQL 8.0 Command Line Cli

```
mysql> SELECT Booking.booking_id, Booking.customer_id, Booking.event_id, Booking.num_tickets, Booking.total_cost, Booking.booking_date
    -> FROM Booking
    -> JOIN Event ON Booking.event_id = Event.event_id
    -> WHERE Booking.num_tickets > 4;
+------------+-------------+----------+-------------+------------+--------------+
| booking_id | customer_id | event_id | num_tickets | total_cost | booking_date |
+------------+-------------+----------+-------------+------------+--------------+
|          6 |        NULL |        6 |           5 |     150.00 | 2023-12-19   |
+------------+-------------+----------+-------------+------------+--------------+
1 row in set (0.00 sec)

mysql> SELECT *
    -> FROM Customers
    -> WHERE phone_number LIKE '%000';
+-------------+---------------+---------------------+--------------+------------+
| customer_id | customer_name | email               | phone_number | booking_id |
+-------------+---------------+---------------------+--------------+------------+
|          10 | Ivy Clark     | ivy.clark@email.com | 888-999-0000 |       NULL |
+-------------+---------------+---------------------+--------------+------------+
1 row in set (0.00 sec)

mysql> SELECT *
    -> FROM Event
    -> WHERE total_seats > 15000
    -> ORDER BY total_seats ASC;
Empty set (0.00 sec)

mysql>
```

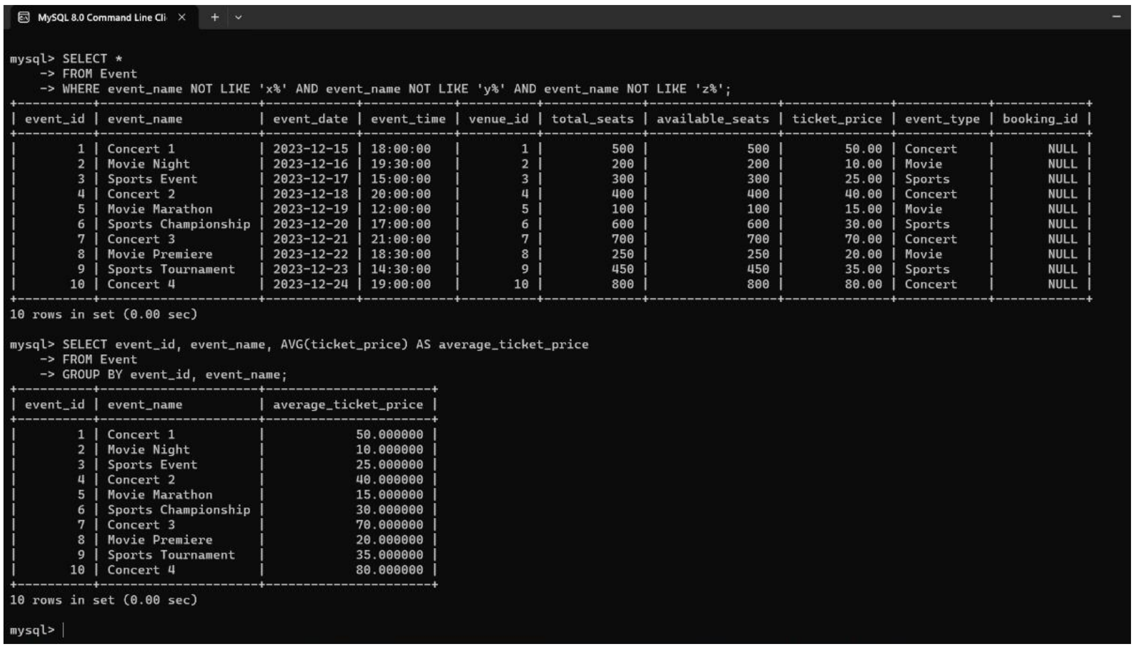
12. Write a SQL query to select events name not start with ‘x’, ‘y’, ‘z’



```
|          6 |        NULL |        6 |           5 |     150.00 | 2023-12-19   |
+------------+-------------+----------+-------------+------------+--------------+
1 row in set (0.00 sec)

mysql> SELECT *
    -> FROM Customers
    -> WHERE phone_number LIKE '%000';
+-------------+---------------+---------------------+--------------+------------+
| customer_id | customer_name | email               | phone_number | booking_id |
+-------------+---------------+---------------------+--------------+------------+
|          10 | Ivy Clark     | ivy.clark@email.com | 888-999-0000 |       NULL |
+-------------+---------------+---------------------+--------------+------------+
1 row in set (0.00 sec)

mysql> SELECT *
    -> FROM Event
    -> WHERE total_seats > 15000
    -> ORDER BY total_seats ASC;
Empty set (0.00 sec)

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 1           | 2023-12-15 | 18:00:00   |        1 |         500 |             500 |        50.00 | Concert    |       NULL |
|        2 | Movie Night         | 2023-12-16 | 19:30:00   |        2 |         200 |             200 |        10.00 | Movie      |       NULL |
|        3 | Sports Event        | 2023-12-17 | 15:00:00   |        3 |         300 |             300 |        25.00 | Sports     |       NULL |
|        4 | Concert 2           | 2023-12-18 | 20:00:00   |        4 |         400 |             400 |        40.00 | Concert    |       NULL |
|        5 | Movie Marathon      | 2023-12-19 | 12:00:00   |        5 |         100 |             100 |        15.00 | Movie      |       NULL |
|        6 | Sports Championship | 2023-12-20 | 17:00:00   |        6 |         600 |             600 |        30.00 | Sports     |       NULL |
|        7 | Concert 3           | 2023-12-21 | 21:00:00   |        7 |         700 |             700 |        70.00 | Concert    |       NULL |
|        8 | Movie Premiere      | 2023-12-22 | 18:30:00   |        8 |         250 |             250 |        20.00 | Movie      |       NULL |
|        9 | Sports Tournament   | 2023-12-23 | 14:30:00   |        9 |         450 |             450 |        35.00 | Sports     |       NULL |
|       10 | Concert 4           | 2023-12-24 | 19:00:00   |       10 |         800 |             800 |        80.00 | Concert    |       NULL |
+----------+---------------------+------------+------------+----------+-------------+-----------------+--------------+------------+------------+
10 rows in set (0.00 sec)

mysql>
```

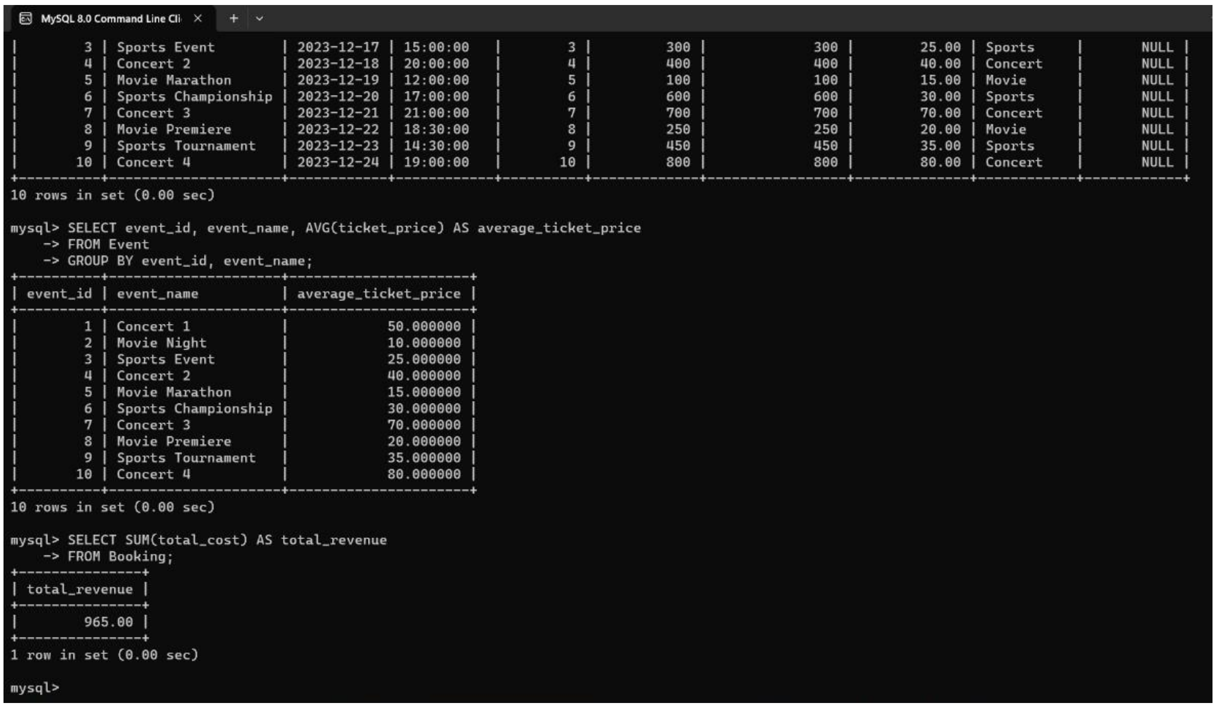
Task-3:

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



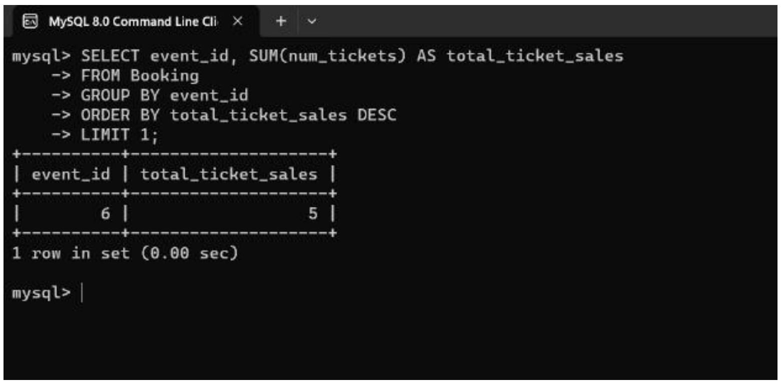
```
mysql> SELECT *
    -> FROM Event
    -> WHERE event_name NOT LIKE 'x%' AND event_name NOT LIKE 'y%' AND event_name NOT LIKE 'z%';
+----------+---------------------+------------+------------+----------+-------------+-----------------+--------------+------------+------------+
| event_id | event_name          | event_date | event_time | venue_id | total_seats | available_seats | ticket_price | event_type | booking_id |
+----------+---------------------+------------+------------+----------+-------------+-----------------+--------------+------------+------------+
|        1 | Concert 1           | 2023-12-15 | 18:00:00   |        1 |         500 |             500 |        50.00 | Concert    |       NULL |
|        2 | Movie Night         | 2023-12-16 | 19:30:00   |        2 |         200 |             200 |        10.00 | Movie      |       NULL |
|        3 | Sports Event        | 2023-12-17 | 15:00:00   |        3 |         300 |             300 |        25.00 | Sports     |       NULL |
|        4 | Concert 2           | 2023-12-18 | 20:00:00   |        4 |         400 |             400 |        40.00 | Concert    |       NULL |
|        5 | Movie Marathon      | 2023-12-19 | 12:00:00   |        5 |         100 |             100 |        15.00 | Movie      |       NULL |
|        6 | Sports Championship | 2023-12-20 | 17:00:00   |        6 |         600 |             600 |        30.00 | Sports     |       NULL |
|        7 | Concert 3           | 2023-12-21 | 21:00:00   |        7 |         700 |             700 |        70.00 | Concert    |       NULL |
|        8 | Movie Premiere      | 2023-12-22 | 18:30:00   |        8 |         250 |             250 |        20.00 | Movie      |       NULL |
|        9 | Sports Tournament   | 2023-12-23 | 14:30:00   |        9 |         450 |             450 |        35.00 | Sports     |       NULL |
|       10 | Concert 4           | 2023-12-24 | 19:00:00   |       10 |         800 |             800 |        80.00 | Concert    |       NULL |
+----------+---------------------+------------+------------+----------+-------------+-----------------+--------------+------------+------------+
10 rows in set (0.00 sec)

mysql> SELECT event_id, event_name, AVG(ticket_price) AS average_ticket_price
    -> FROM Event
    -> GROUP BY event_id, event_name;
+----------+---------------------+----------------------+
| event_id | event_name          | average_ticket_price |
+----------+---------------------+----------------------+
|        1 | Concert 1           |            50.000000 |
|        2 | Movie Night         |            10.000000 |
|        3 | Sports Event        |            25.000000 |
|        4 | Concert 2           |            40.000000 |
|        5 | Movie Marathon      |            15.000000 |
|        6 | Sports Championship |            30.000000 |
|        7 | Concert 3           |            70.000000 |
|        8 | Movie Premiere      |            20.000000 |
|        9 | Sports Tournament   |            35.000000 |
|       10 | Concert 4           |            80.000000 |
+----------+---------------------+----------------------+
10 rows in set (0.00 sec)

mysql>
```

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



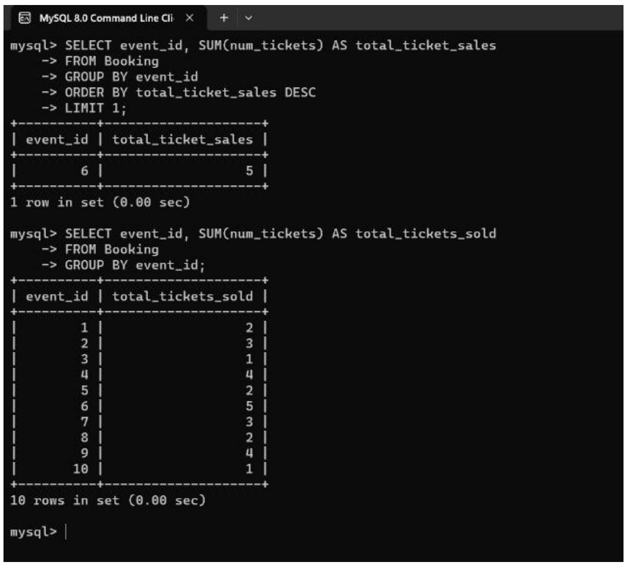
```
|        3 | Sports Event         | 2023-12-17 | 15:00:00   |         3 |         300 |             300 |        25.00 | Sports      |       NULL |
|        4 | Concert 2            | 2023-12-18 | 20:00:00   |         4 |         400 |             400 |        40.00 | Concert     |       NULL |
|        5 | Movie Marathon       | 2023-12-19 | 12:00:00   |         5 |         100 |             100 |        15.00 | Movie       |       NULL |
|        6 | Sports Championship  | 2023-12-20 | 17:00:00   |         6 |         600 |             600 |        30.00 | Sports      |       NULL |
|        7 | Concert 3            | 2023-12-21 | 21:00:00   |         7 |         700 |             700 |        70.00 | Concert     |       NULL |
|        8 | Movie Premiere       | 2023-12-22 | 18:30:00   |         8 |         250 |             250 |        20.00 | Movie       |       NULL |
|        9 | Sports Tournament    | 2023-12-23 | 14:30:00   |         9 |         450 |             450 |        35.00 | Sports      |       NULL |
|       10 | Concert 4            | 2023-12-24 | 19:00:00   |        10 |         800 |             800 |        80.00 | Concert     |       NULL |
+----------+----------------------+------------+------------+-----------+-------------+-----------------+--------------+-------------+------------+
10 rows in set (0.00 sec)

mysql> SELECT event_id, event_name, AVG(ticket_price) AS average_ticket_price
    -> FROM Event
    -> GROUP BY event_id, event_name;
+----------+----------------------+----------------------+
| event_id | event_name           | average_ticket_price |
+----------+----------------------+----------------------+
|        1 | Concert 1            |            50.000000 |
|        2 | Movie Night          |            10.000000 |
|        3 | Sports Event         |            25.000000 |
|        4 | Concert 2            |            40.000000 |
|        5 | Movie Marathon       |            15.000000 |
|        6 | Sports Championship  |            30.000000 |
|        7 | Concert 3            |            70.000000 |
|        8 | Movie Premiere       |            20.000000 |
|        9 | Sports Tournament    |            35.000000 |
|       10 | Concert 4            |            80.000000 |
+----------+----------------------+----------------------+
10 rows in set (0.00 sec)

mysql> SELECT SUM(total_cost) AS total_revenue
    -> FROM Booking;
+---------------+
| total_revenue |
+---------------+
|        965.00 |
+---------------+
1 row in set (0.00 sec)

mysql>
```

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



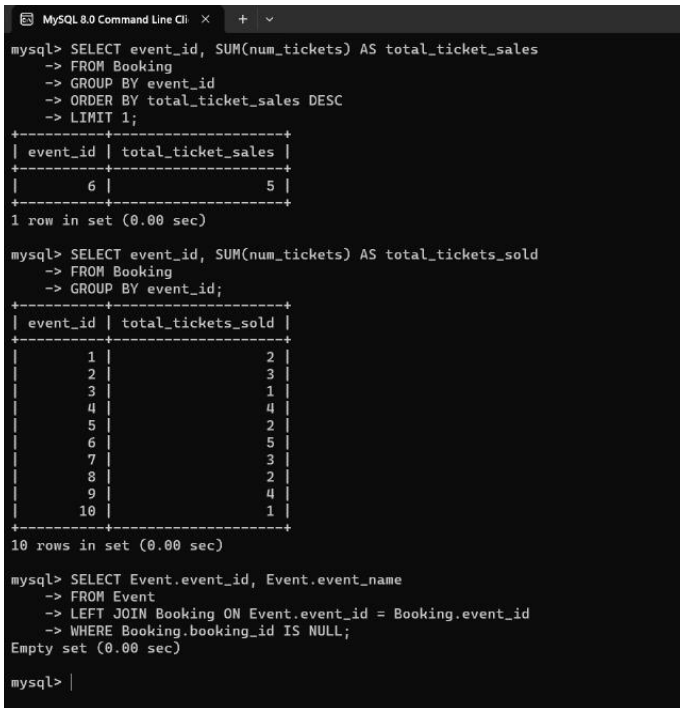
```
mysql> SELECT event_id, SUM(num_tickets) AS total_ticket_sales
    -> FROM Booking
    -> GROUP BY event_id
    -> ORDER BY total_ticket_sales DESC
    -> LIMIT 1;
+----------+--------------------+
| event_id | total_ticket_sales |
+----------+--------------------+
|        6 |                  5 |
+----------+--------------------+
1 row in set (0.00 sec)

mysql>
```

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



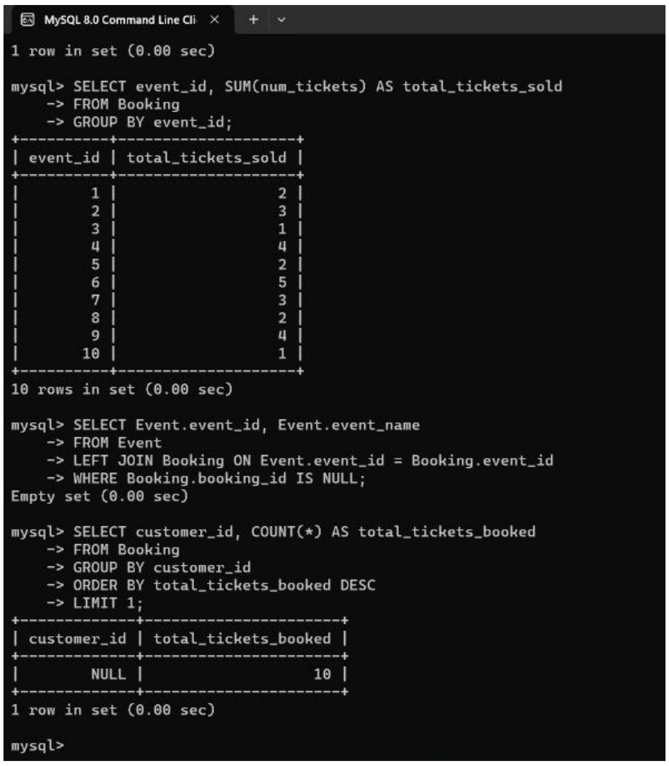
```
mysql> SELECT event_id, SUM(num_tickets) AS total_ticket_sales
    -> FROM Booking
    -> GROUP BY event_id
    -> ORDER BY total_ticket_sales DESC
    -> LIMIT 1;
+----------+--------------------+
| event_id | total_ticket_sales |
+----------+--------------------+
|        6 |                  5 |
+----------+--------------------+
1 row in set (0.00 sec)

mysql> SELECT event_id, SUM(num_tickets) AS total_tickets_sold
    -> FROM Booking
    -> GROUP BY event_id;
+----------+--------------------+
| event_id | total_tickets_sold |
+----------+--------------------+
|        1 |                  2 |
|        2 |                  3 |
|        3 |                  1 |
|        4 |                  4 |
|        5 |                  2 |
|        6 |                  5 |
|        7 |                  3 |
|        8 |                  2 |
|        9 |                  4 |
|       10 |                  1 |
+----------+--------------------+
10 rows in set (0.00 sec)

mysql>
```

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



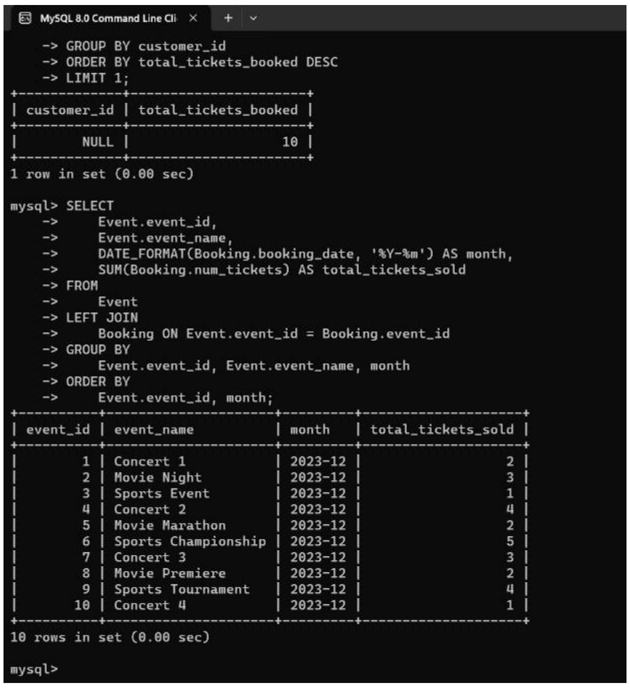
```
mysql> SELECT event_id, SUM(num_tickets) AS total_ticket_sales
    -> FROM Booking
    -> GROUP BY event_id
    -> ORDER BY total_ticket_sales DESC
    -> LIMIT 1;
+----------+--------------------+
| event_id | total_ticket_sales |
+----------+--------------------+
|        6 |                  5 |
+----------+--------------------+
1 row in set (0.00 sec)

mysql> SELECT event_id, SUM(num_tickets) AS total_tickets_sold
    -> FROM Booking
    -> GROUP BY event_id;
+----------+--------------------+
| event_id | total_tickets_sold |
+----------+--------------------+
|        1 |                  2 |
|        2 |                  3 |
|        3 |                  1 |
|        4 |                  4 |
|        5 |                  2 |
|        6 |                  5 |
|        7 |                  3 |
|        8 |                  2 |
|        9 |                  4 |
|       10 |                  1 |
+----------+--------------------+
10 rows in set (0.00 sec)

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

mysql> |
```

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



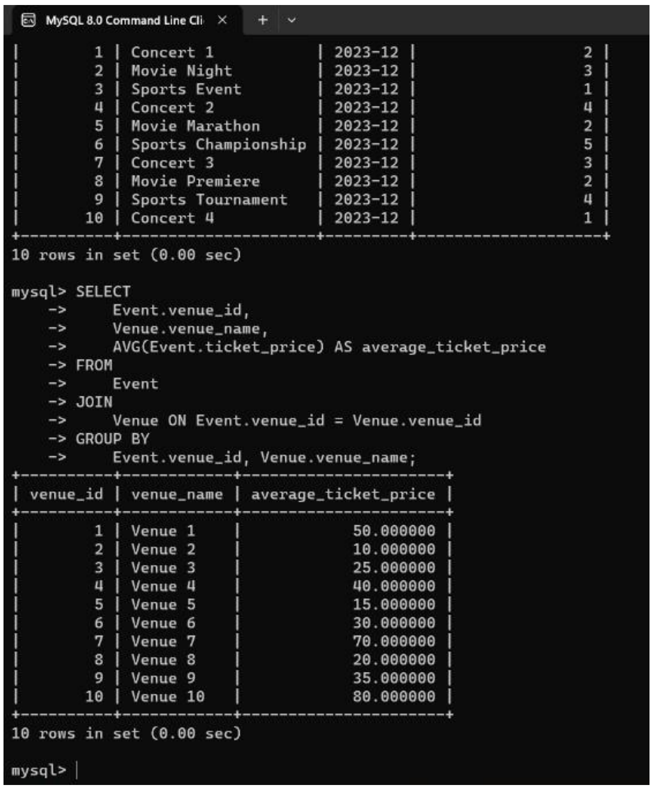
```
1 row in set (0.00 sec)

mysql> SELECT event_id, SUM(num_tickets) AS total_tickets_sold
    -> FROM Booking
    -> GROUP BY event_id;
+----------+--------------------+
| event_id | total_tickets_sold |
+----------+--------------------+
|        1 |                  2 |
|        2 |                  3 |
|        3 |                  1 |
|        4 |                  4 |
|        5 |                  2 |
|        6 |                  5 |
|        7 |                  3 |
|        8 |                  2 |
|        9 |                  4 |
|       10 |                  1 |
+----------+--------------------+
10 rows in set (0.00 sec)

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

mysql> SELECT customer_id, COUNT(*) AS total_tickets_booked
    -> FROM Booking
    -> GROUP BY customer_id
    -> ORDER BY total_tickets_booked DESC
    -> LIMIT 1;
+-------------+----------------------+
| customer_id | total_tickets_booked |
+-------------+----------------------+
|        NULL |                   10 |
+-------------+----------------------+
1 row in set (0.00 sec)

mysql>
```

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



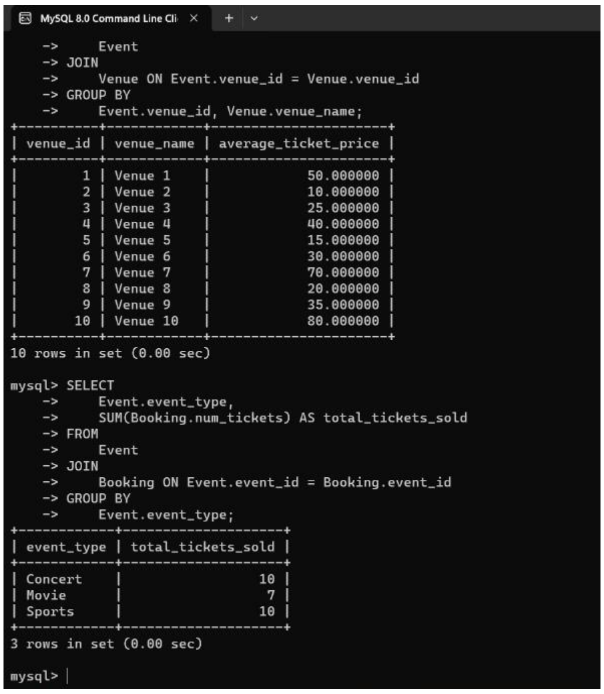
```
    -> GROUP BY customer_id
    -> ORDER BY total_tickets_booked DESC
    -> LIMIT 1;
+-------------+----------------------+
| customer_id | total_tickets_booked |
+-------------+----------------------+
|        NULL |                   10 |
+-------------+----------------------+
1 row in set (0.00 sec)

mysql> SELECT
    ->     Event.event_id,
    ->     Event.event_name,
    ->     DATE_FORMAT(Booking.booking_date, '%Y-%m') AS month,
    ->     SUM(Booking.num_tickets) AS total_tickets_sold
    -> FROM
    ->     Event
    -> LEFT JOIN
    ->     Booking ON Event.event_id = Booking.event_id
    -> GROUP BY
    ->     Event.event_id, Event.event_name, month
    -> ORDER BY
    ->     Event.event_id, month;
+----------+---------------------+---------+--------------------+
| event_id | event_name          | month   | total_tickets_sold |
+----------+---------------------+---------+--------------------+
|        1 | Concert 1           | 2023-12 |                  2 |
|        2 | Movie Night         | 2023-12 |                  3 |
|        3 | Sports Event        | 2023-12 |                  1 |
|        4 | Concert 2           | 2023-12 |                  4 |
|        5 | Movie Marathon      | 2023-12 |                  2 |
|        6 | Sports Championship | 2023-12 |                  5 |
|        7 | Concert 3           | 2023-12 |                  3 |
|        8 | Movie Premiere      | 2023-12 |                  2 |
|        9 | Sports Tournament   | 2023-12 |                  4 |
|       10 | Concert 4           | 2023-12 |                  1 |
+----------+---------------------+---------+--------------------+
10 rows in set (0.00 sec)

mysql>
```

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



```
|        1 | Concert 1           | 2023-12 |                  2 |
|        2 | Movie Night         | 2023-12 |                  3 |
|        3 | Sports Event        | 2023-12 |                  1 |
|        4 | Concert 2           | 2023-12 |                  4 |
|        5 | Movie Marathon      | 2023-12 |                  2 |
|        6 | Sports Championship | 2023-12 |                  5 |
|        7 | Concert 3           | 2023-12 |                  3 |
|        8 | Movie Premiere      | 2023-12 |                  2 |
|        9 | Sports Tournament   | 2023-12 |                  4 |
|       10 | Concert 4           | 2023-12 |                  1 |
+----------+---------------------+---------+--------------------+
10 rows in set (0.00 sec)

mysql> SELECT
    ->     Event.venue_id,
    ->     Venue.venue_name,
    ->     AVG(Event.ticket_price) AS average_ticket_price
    -> FROM
    ->     Event
    -> JOIN
    ->     Venue ON Event.venue_id = Venue.venue_id
    -> GROUP BY
    ->     Event.venue_id, Venue.venue_name;
+----------+------------+----------------------+
| venue_id | venue_name | average_ticket_price |
+----------+------------+----------------------+
|        1 | Venue 1    |            50.000000 |
|        2 | Venue 2    |            10.000000 |
|        3 | Venue 3    |            25.000000 |
|        4 | Venue 4    |            40.000000 |
|        5 | Venue 5    |            15.000000 |
|        6 | Venue 6    |            30.000000 |
|        7 | Venue 7    |            70.000000 |
|        8 | Venue 8    |            20.000000 |
|        9 | Venue 9    |            35.000000 |
|       10 | Venue 10   |            80.000000 |
+----------+------------+----------------------+
10 rows in set (0.00 sec)

mysql>
```

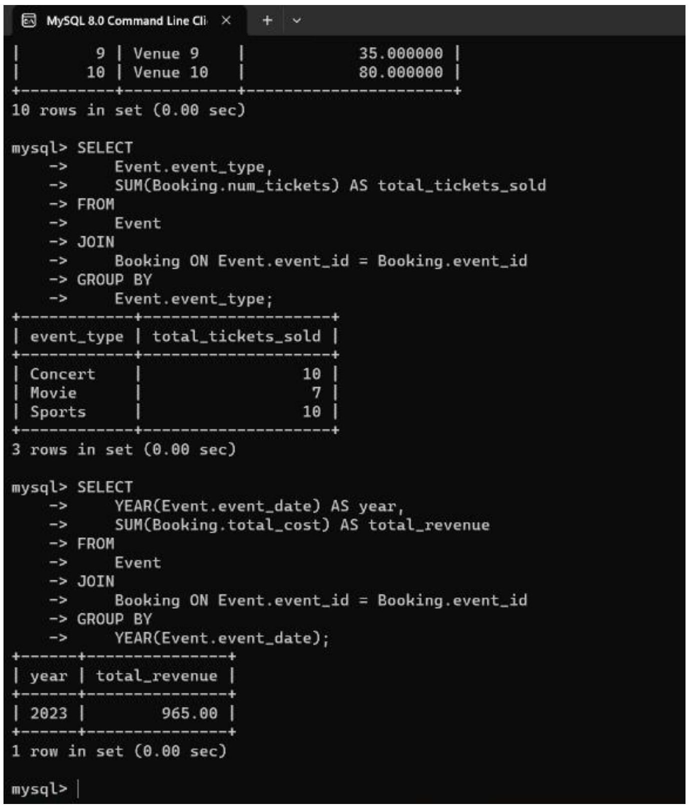
9. Write a SQL query to calculate the total Number of Tickets Sold for Each Event Type.



MySQL 8.0 Command Line Cli

```
    ->     Event
    -> JOIN
    ->     Venue ON Event.venue_id = Venue.venue_id
    -> GROUP BY
    ->     Event.venue_id, Venue.venue_name;
+----------+------------+----------------------+
| venue_id | venue_name | average_ticket_price |
+----------+------------+----------------------+
|        1 | Venue 1    |            50.000000 |
|        2 | Venue 2    |            10.000000 |
|        3 | Venue 3    |            25.000000 |
|        4 | Venue 4    |            40.000000 |
|        5 | Venue 5    |            15.000000 |
|        6 | Venue 6    |            30.000000 |
|        7 | Venue 7    |            70.000000 |
|        8 | Venue 8    |            20.000000 |
|        9 | Venue 9    |            35.000000 |
|       10 | Venue 10   |            80.000000 |
+----------+------------+----------------------+
10 rows in set (0.00 sec)

mysql> SELECT
    ->     Event.event_type,
    ->     SUM(Booking.num_tickets) AS total_tickets_sold
    -> FROM
    ->     Event
    -> JOIN
    ->     Booking ON Event.event_id = Booking.event_id
    -> GROUP BY
    ->     Event.event_type;
+------------+--------------------+
| event_type | total_tickets_sold |
+------------+--------------------+
| Concert    |                 10 |
| Movie      |                  7 |
| Sports     |                 10 |
+------------+--------------------+
3 rows in set (0.00 sec)

mysql>
```

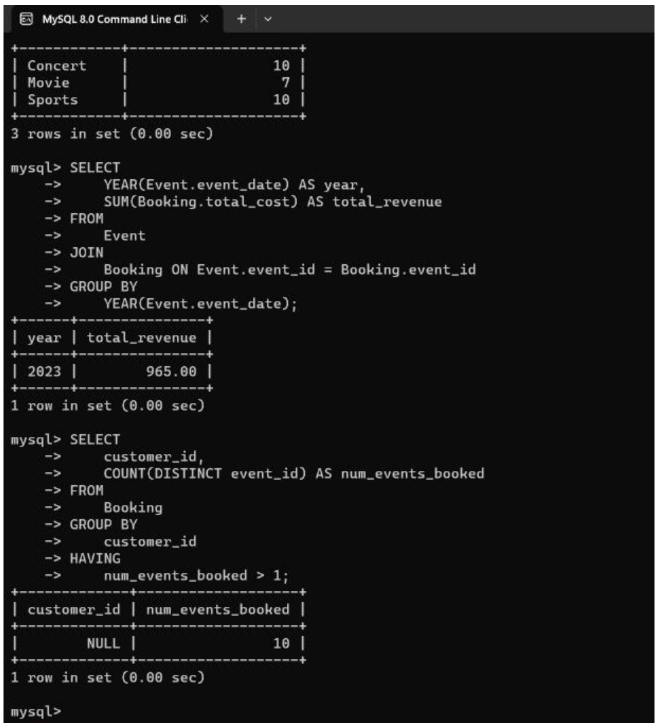
10. Write a SQL query to calculate the total Revenue Generated by Events in Each Year.



MySQL 8.0 Command Line Cli

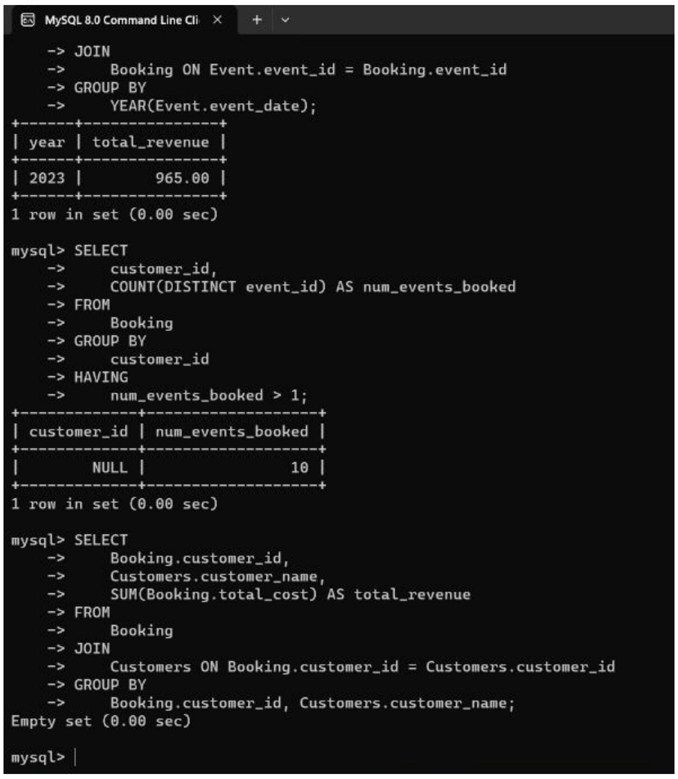
```
|        9 | Venue 9    |            35.000000 |
|       10 | Venue 10   |            80.000000 |
+----------+------------+----------------------+
10 rows in set (0.00 sec)

mysql> SELECT
    ->     Event.event_type,
    ->     SUM(Booking.num_tickets) AS total_tickets_sold
    -> FROM
    ->     Event
    -> JOIN
    ->     Booking ON Event.event_id = Booking.event_id
    -> GROUP BY
    ->     Event.event_type;
+------------+--------------------+
| event_type | total_tickets_sold |
+------------+--------------------+
| Concert    |                 10 |
| Movie      |                  7 |
| Sports     |                 10 |
+------------+--------------------+
3 rows in set (0.00 sec)

mysql> SELECT
    ->     YEAR(Event.event_date) AS year,
    ->     SUM(Booking.total_cost) AS total_revenue
    -> FROM
    ->     Event
    -> JOIN
    ->     Booking ON Event.event_id = Booking.event_id
    -> GROUP BY
    ->     YEAR(Event.event_date);
+------+---------------+
| year | total_revenue |
+------+---------------+
| 2023 |        965.00 |
+------+---------------+
1 row in set (0.00 sec)

mysql>
```

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



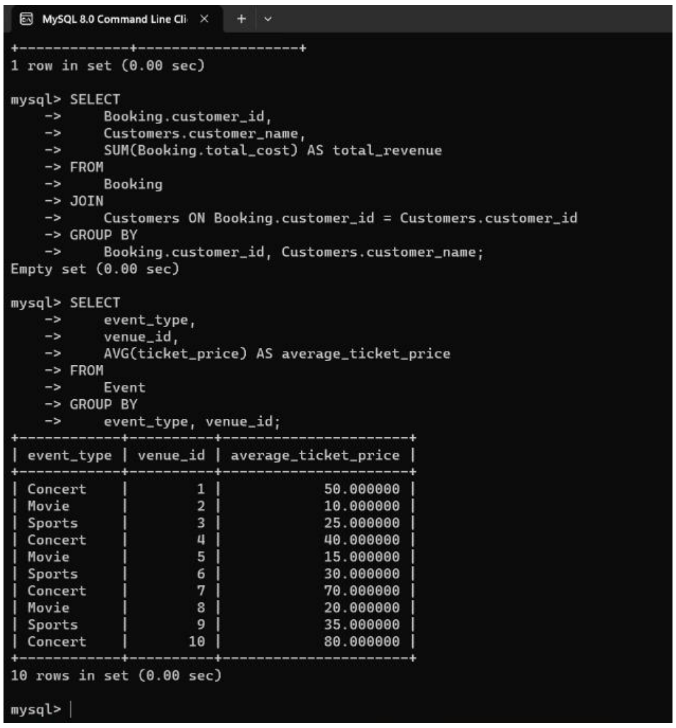
```
+------------+--------------------+
| Concert    |                 10 |
| Movie      |                  7 |
| Sports     |                 10 |
+------------+--------------------+
3 rows in set (0.00 sec)

mysql> SELECT
    ->     YEAR(Event.event_date) AS year,
    ->     SUM(Booking.total_cost) AS total_revenue
    -> FROM
    ->     Event
    -> JOIN
    ->     Booking ON Event.event_id = Booking.event_id
    -> GROUP BY
    ->     YEAR(Event.event_date);
+------+---------------+
| year | total_revenue |
+------+---------------+
| 2023 |        965.00 |
+------+---------------+
1 row in set (0.00 sec)

mysql> SELECT
    ->     customer_id,
    ->     COUNT(DISTINCT event_id) AS num_events_booked
    -> FROM
    ->     Booking
    -> GROUP BY
    ->     customer_id
    -> HAVING
    ->     num_events_booked > 1;
+-------------+-------------------+
| customer_id | num_events_booked |
+-------------+-------------------+
|        NULL |                10 |
+-------------+-------------------+
1 row in set (0.00 sec)

mysql>
```

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



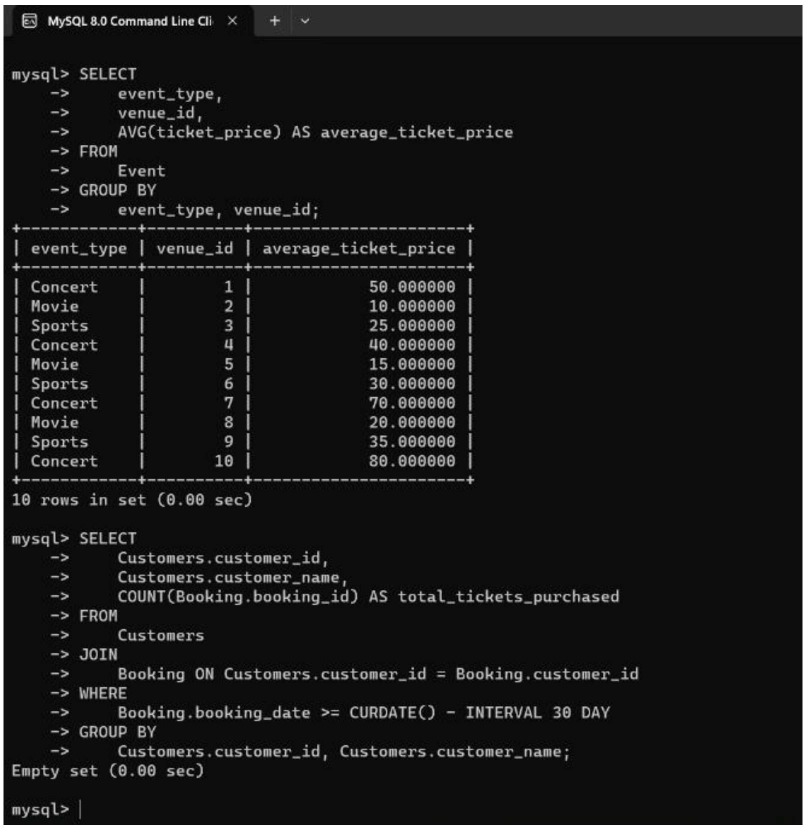
```
    -> JOIN
    ->     Booking ON Event.event_id = Booking.event_id
    -> GROUP BY
    ->     YEAR(Event.event_date);
+------+---------------+
| year | total_revenue |
+------+---------------+
| 2023 |        965.00 |
+------+---------------+
1 row in set (0.00 sec)

mysql> SELECT
    ->     customer_id,
    ->     COUNT(DISTINCT event_id) AS num_events_booked
    -> FROM
    ->     Booking
    -> GROUP BY
    ->     customer_id
    -> HAVING
    ->     num_events_booked > 1;
+-------------+-------------------+
| customer_id | num_events_booked |
+-------------+-------------------+
|        NULL |                10 |
+-------------+-------------------+
1 row in set (0.00 sec)

mysql> SELECT
    ->     Booking.customer_id,
    ->     Customers.customer_name,
    ->     SUM(Booking.total_cost) AS total_revenue
    -> FROM
    ->     Booking
    -> JOIN
    ->     Customers ON Booking.customer_id = Customers.customer_id
    -> GROUP BY
    ->     Booking.customer_id, Customers.customer_name;
Empty set (0.00 sec)

mysql>
```

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



```
+-------------+-------------------+
1 row in set (0.00 sec)

mysql> SELECT
    ->     Booking.customer_id,
    ->     Customers.customer_name,
    ->     SUM(Booking.total_cost) AS total_revenue
    -> FROM
    ->     Booking
    -> JOIN
    ->     Customers ON Booking.customer_id = Customers.customer_id
    -> GROUP BY
    ->     Booking.customer_id, Customers.customer_name;
Empty set (0.00 sec)

mysql> SELECT
    ->     event_type,
    ->     venue_id,
    ->     AVG(ticket_price) AS average_ticket_price
    -> FROM
    ->     Event
    -> GROUP BY
    ->     event_type, venue_id;
+------------+----------+----------------------+
| event_type | venue_id | average_ticket_price |
+------------+----------+----------------------+
| Concert    |        1 |            50.000000 |
| Movie      |        2 |            10.000000 |
| Sports     |        3 |            25.000000 |
| Concert    |        4 |            40.000000 |
| Movie      |        5 |            15.000000 |
| Sports     |        6 |            30.000000 |
| Concert    |        7 |            70.000000 |
| Movie      |        8 |            20.000000 |
| Sports     |        9 |            35.000000 |
| Concert    |       10 |            80.000000 |
+------------+----------+----------------------+
10 rows in set (0.00 sec)

mysql>
```

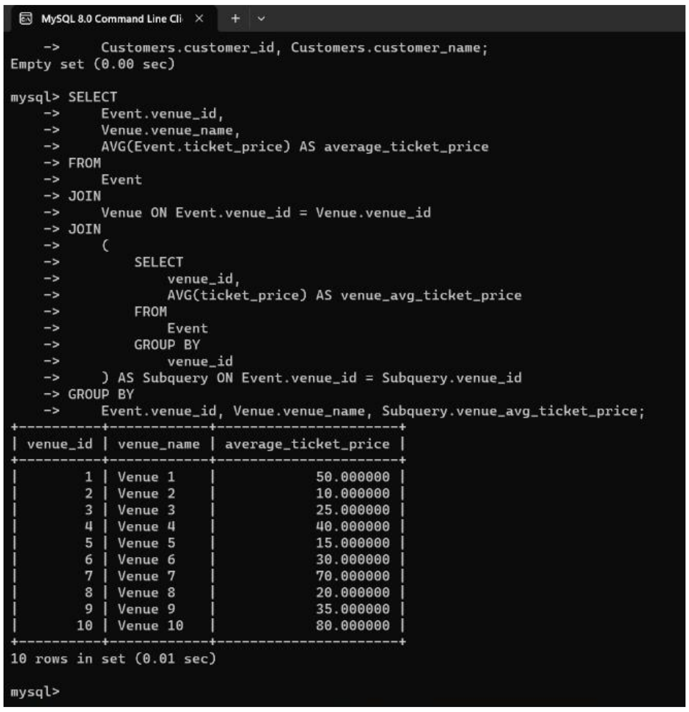
14. Write a SQL query to list Users and the Total Number of Tickets They've Purchased in the Last 30 Days.



```
mysql> SELECT
    ->     event_type,
    ->     venue_id,
    ->     AVG(ticket_price) AS average_ticket_price
    -> FROM
    ->     Event
    -> GROUP BY
    ->     event_type, venue_id;
+------------+----------+----------------------+
| event_type | venue_id | average_ticket_price |
+------------+----------+----------------------+
| Concert    |        1 |            50.000000 |
| Movie      |        2 |            10.000000 |
| Sports     |        3 |            25.000000 |
| Concert    |        4 |            40.000000 |
| Movie      |        5 |            15.000000 |
| Sports     |        6 |            30.000000 |
| Concert    |        7 |            70.000000 |
| Movie      |        8 |            20.000000 |
| Sports     |        9 |            35.000000 |
| Concert    |       10 |            80.000000 |
+------------+----------+----------------------+
10 rows in set (0.00 sec)

mysql> SELECT
    ->     Customers.customer_id,
    ->     Customers.customer_name,
    ->     COUNT(Booking.booking_id) AS total_tickets_purchased
    -> FROM
    ->     Customers
    -> JOIN
    ->     Booking ON Customers.customer_id = Booking.customer_id
    -> WHERE
    ->     Booking.booking_date >= CURDATE() - INTERVAL 30 DAY
    -> GROUP BY
    ->     Customers.customer_id, Customers.customer_name;
Empty set (0.00 sec)

mysql>
```

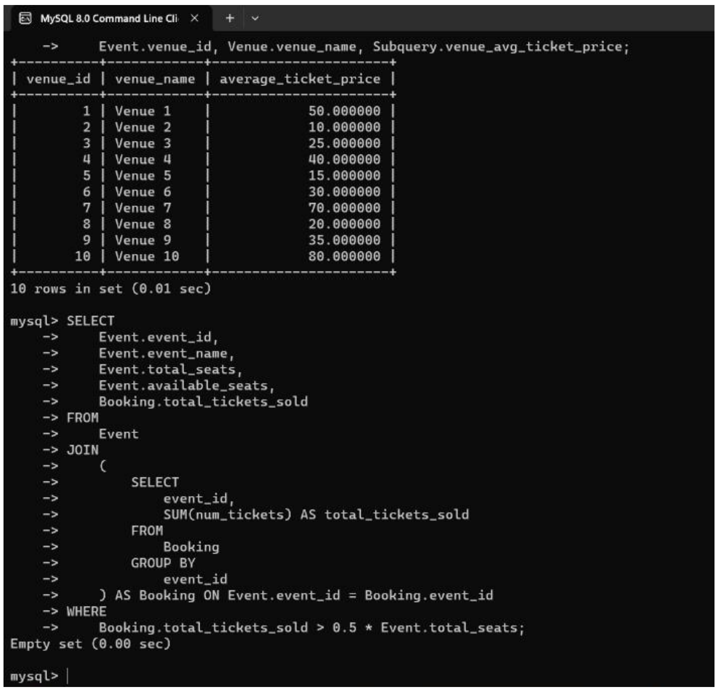
Task – 4:

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



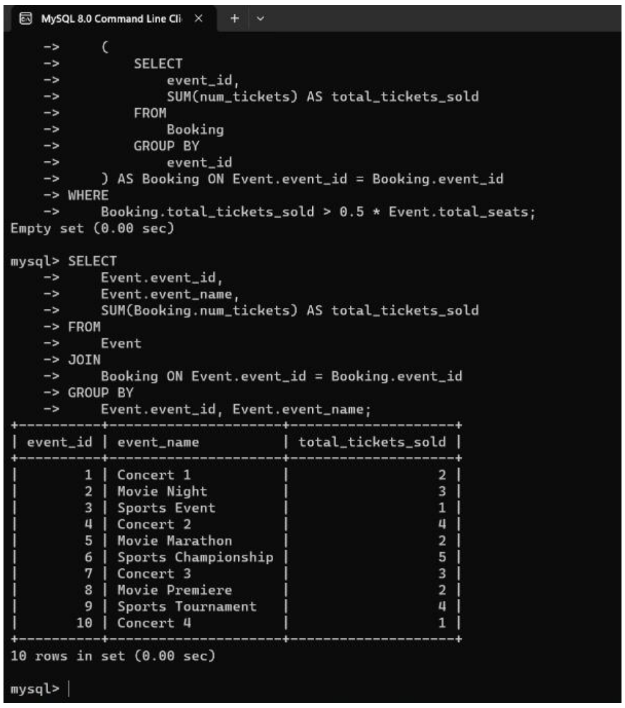
```
    ->     Customers.customer_id, Customers.customer_name;
Empty set (0.00 sec)

mysql> SELECT
    ->     Event.venue_id,
    ->     Venue.venue_name,
    ->     AVG(Event.ticket_price) AS average_ticket_price
    -> FROM
    ->     Event
    -> JOIN
    ->     Venue ON Event.venue_id = Venue.venue_id
    -> JOIN
    ->     (
    ->         SELECT
    ->             venue_id,
    ->             AVG(ticket_price) AS venue_avg_ticket_price
    ->         FROM
    ->             Event
    ->         GROUP BY
    ->             venue_id
    ->     ) AS Subquery ON Event.venue_id = Subquery.venue_id
    -> GROUP BY
    ->     Event.venue_id, Venue.venue_name, Subquery.venue_avg_ticket_price;
+----------+------------+----------------------+
| venue_id | venue_name | average_ticket_price |
+----------+------------+----------------------+
|        1 | Venue 1    |            50.000000 |
|        2 | Venue 2    |            10.000000 |
|        3 | Venue 3    |            25.000000 |
|        4 | Venue 4    |            40.000000 |
|        5 | Venue 5    |            15.000000 |
|        6 | Venue 6    |            30.000000 |
|        7 | Venue 7    |            70.000000 |
|        8 | Venue 8    |            20.000000 |
|        9 | Venue 9    |            35.000000 |
|       10 | Venue 10   |            80.000000 |
+----------+------------+----------------------+
10 rows in set (0.01 sec)

mysql>
```

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



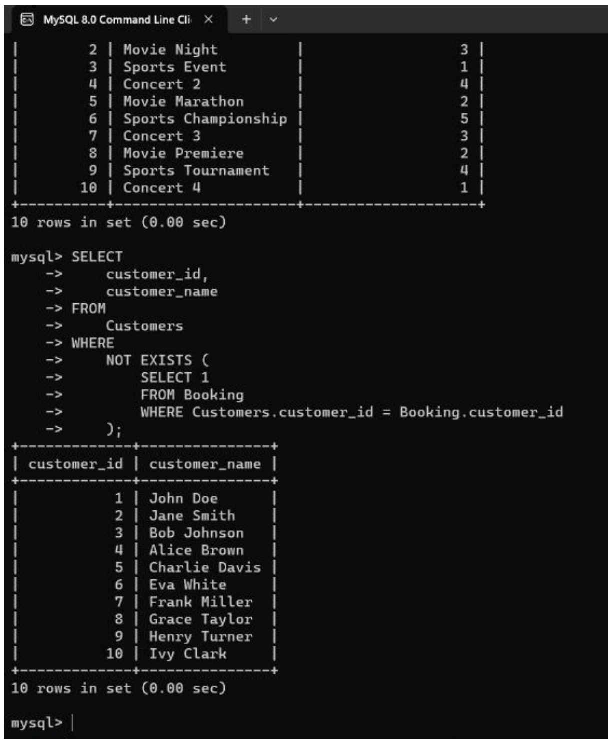
```
    ->     Event.venue_id, Venue.venue_name, Subquery.venue_avg_ticket_price;
+----------+------------+----------------------+
| venue_id | venue_name | average_ticket_price |
+----------+------------+----------------------+
|        1 | Venue 1    |            50.000000 |
|        2 | Venue 2    |            10.000000 |
|        3 | Venue 3    |            25.000000 |
|        4 | Venue 4    |            40.000000 |
|        5 | Venue 5    |            15.000000 |
|        6 | Venue 6    |            30.000000 |
|        7 | Venue 7    |            70.000000 |
|        8 | Venue 8    |            20.000000 |
|        9 | Venue 9    |            35.000000 |
|       10 | Venue 10   |            80.000000 |
+----------+------------+----------------------+
10 rows in set (0.01 sec)

mysql> SELECT
    ->     Event.event_id,
    ->     Event.event_name,
    ->     Event.total_seats,
    ->     Event.available_seats,
    ->     Booking.total_tickets_sold
    -> FROM
    ->     Event
    -> JOIN
    ->     (
    ->         SELECT
    ->             event_id,
    ->             SUM(num_tickets) AS total_tickets_sold
    ->         FROM
    ->             Booking
    ->         GROUP BY
    ->             event_id
    ->     ) AS Booking ON Event.event_id = Booking.event_id
    -> WHERE
    ->     Booking.total_tickets_sold > 0.5 * Event.total_seats;
Empty set (0.00 sec)

mysql>
```

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



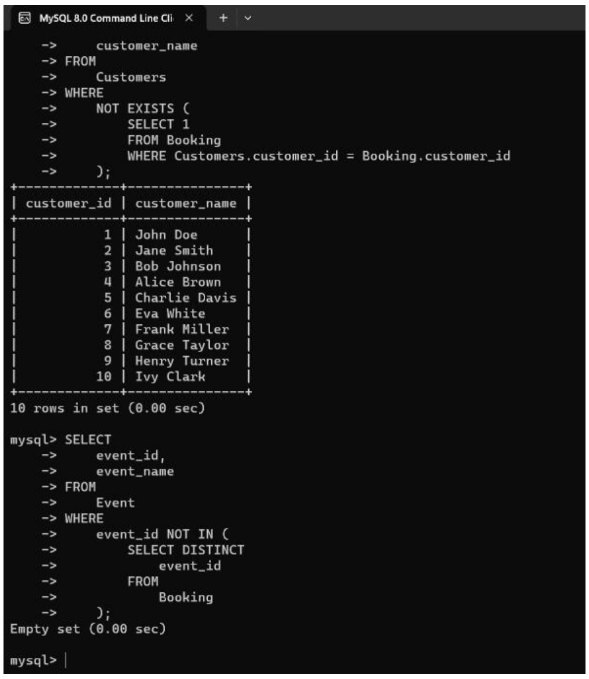
```
    ->     (
    ->         SELECT
    ->             event_id,
    ->             SUM(num_tickets) AS total_tickets_sold
    ->         FROM
    ->             Booking
    ->         GROUP BY
    ->             event_id
    ->     ) AS Booking ON Event.event_id = Booking.event_id
    -> WHERE
    ->     Booking.total_tickets_sold > 0.5 * Event.total_seats;
Empty set (0.00 sec)

mysql> SELECT
    ->     Event.event_id,
    ->     Event.event_name,
    ->     SUM(Booking.num_tickets) AS total_tickets_sold
    -> FROM
    ->     Event
    -> JOIN
    ->     Booking ON Event.event_id = Booking.event_id
    -> GROUP BY
    ->     Event.event_id, Event.event_name;
+----------+---------------------+--------------------+
| event_id | event_name          | total_tickets_sold |
+----------+---------------------+--------------------+
|        1 | Concert 1           |                  2 |
|        2 | Movie Night         |                  3 |
|        3 | Sports Event        |                  1 |
|        4 | Concert 2           |                  4 |
|        5 | Movie Marathon      |                  2 |
|        6 | Sports Championship |                  5 |
|        7 | Concert 3           |                  3 |
|        8 | Movie Premiere      |                  2 |
|        9 | Sports Tournament   |                  4 |
|       10 | Concert 4           |                  1 |
+----------+---------------------+--------------------+
10 rows in set (0.00 sec)

mysql> |
```

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



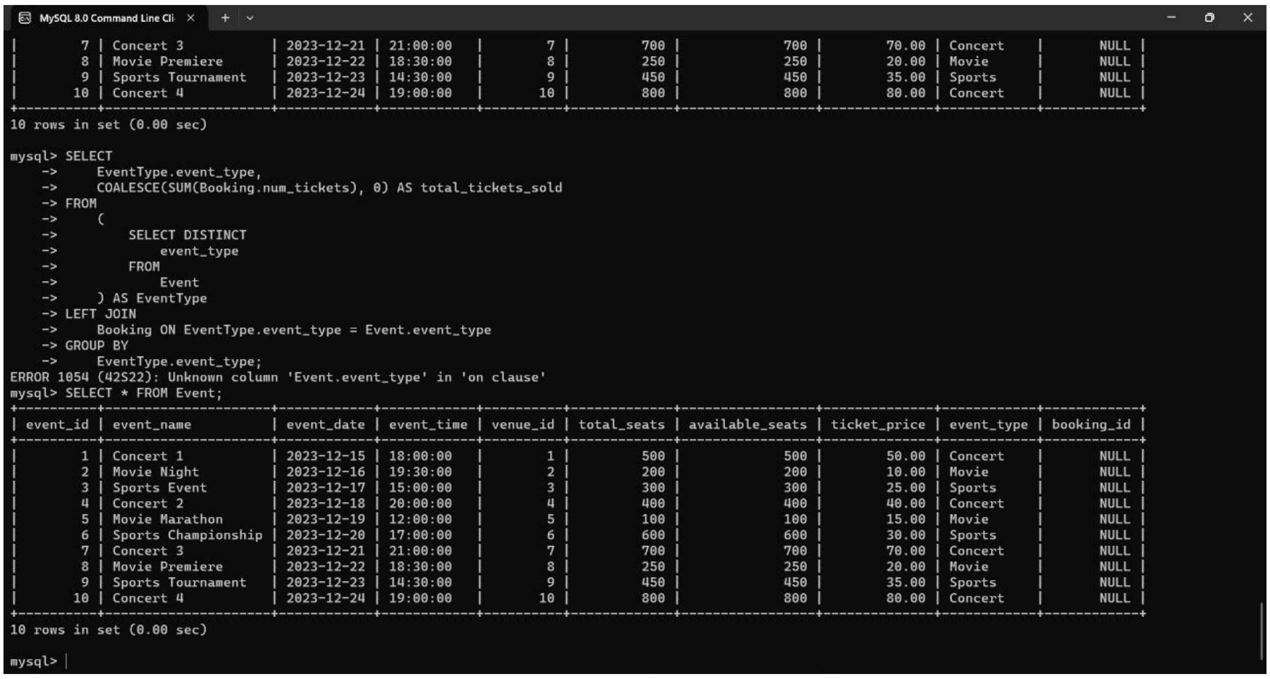
```
|        2 | Movie Night         |                  3 |
|        3 | Sports Event        |                  1 |
|        4 | Concert 2           |                  4 |
|        5 | Movie Marathon      |                  2 |
|        6 | Sports Championship |                  5 |
|        7 | Concert 3           |                  3 |
|        8 | Movie Premiere      |                  2 |
|        9 | Sports Tournament   |                  4 |
|       10 | Concert 4           |                  1 |
+----------+---------------------+--------------------+
10 rows in set (0.00 sec)

mysql> SELECT
    ->     customer_id,
    ->     customer_name
    -> FROM
    ->     Customers
    -> WHERE
    ->     NOT EXISTS (
    ->         SELECT 1
    ->         FROM Booking
    ->         WHERE Customers.customer_id = Booking.customer_id
    ->     );
+-------------+---------------+
| customer_id | customer_name |
+-------------+---------------+
|           1 | John Doe      |
|           2 | Jane Smith    |
|           3 | Bob Johnson   |
|           4 | Alice Brown   |
|           5 | Charlie Davis |
|           6 | Eva White     |
|           7 | Frank Miller  |
|           8 | Grace Taylor  |
|           9 | Henry Turner  |
|          10 | Ivy Clark     |
+-------------+---------------+
10 rows in set (0.00 sec)

mysql> |
```

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



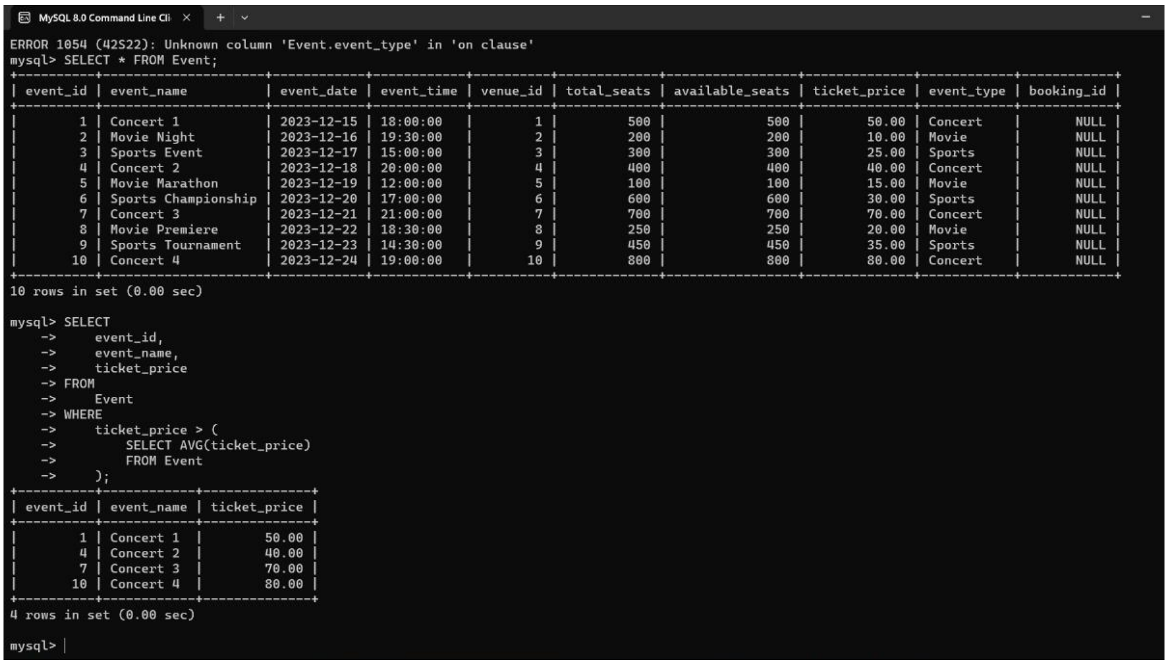
```
    ->     customer_name
    -> FROM
    ->     Customers
    -> WHERE
    ->     NOT EXISTS (
    ->         SELECT 1
    ->         FROM Booking
    ->         WHERE Customers.customer_id = Booking.customer_id
    ->     );
+-------------+---------------+
| customer_id | customer_name |
+-------------+---------------+
|           1 | John Doe      |
|           2 | Jane Smith    |
|           3 | Bob Johnson   |
|           4 | Alice Brown   |
|           5 | Charlie Davis |
|           6 | Eva White     |
|           7 | Frank Miller  |
|           8 | Grace Taylor  |
|           9 | Henry Turner  |
|          10 | Ivy Clark     |
+-------------+---------------+
10 rows in set (0.00 sec)

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>
```

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



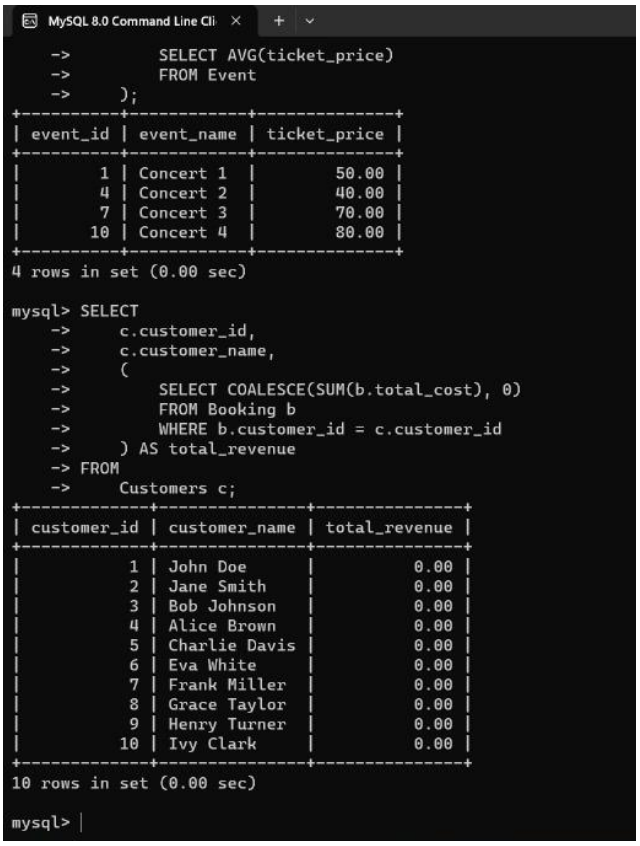
```
|        7 | Concert 3             | 2023-12-21 | 21:00:00   |        7 |         700 |             700 |        70.00 | Concert    |       NULL |
|        8 | Movie Premiere        | 2023-12-22 | 18:30:00   |        8 |         250 |             250 |        20.00 | Movie      |       NULL |
|        9 | Sports Tournament     | 2023-12-23 | 14:30:00   |        9 |         450 |             450 |        35.00 | Sports     |       NULL |
|       10 | Concert 4             | 2023-12-24 | 19:00:00   |       10 |         800 |             800 |        80.00 | Concert    |       NULL |
+----------+-----------------------+------------+------------+----------+-------------+-----------------+--------------+------------+------------+
10 rows in set (0.00 sec)

mysql> SELECT
    ->     EventType.event_type,
    ->     COALESCE(SUM(Booking.num_tickets), 0) AS total_tickets_sold
    -> FROM
    ->     (
    ->         SELECT DISTINCT
    ->             event_type
    ->         FROM
    ->             Event
    ->     ) AS EventType
    -> LEFT JOIN
    ->     Booking ON EventType.event_type = Event.event_type
    -> GROUP BY
    ->     EventType.event_type;
ERROR 1054 (42S22): Unknown column 'Event.event_type' in 'on clause'
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 1             | 2023-12-15 | 18:00:00   |        1 |         500 |             500 |        50.00 | Concert    |       NULL |
|        2 | Movie Night           | 2023-12-16 | 19:30:00   |        2 |         200 |             200 |        10.00 | Movie      |       NULL |
|        3 | Sports Event          | 2023-12-17 | 15:00:00   |        3 |         300 |             300 |        25.00 | Sports     |       NULL |
|        4 | Concert 2             | 2023-12-18 | 20:00:00   |        4 |         400 |             400 |        40.00 | Concert    |       NULL |
|        5 | Movie Marathon        | 2023-12-19 | 12:00:00   |        5 |         100 |             100 |        15.00 | Movie      |       NULL |
|        6 | Sports Championship   | 2023-12-20 | 17:00:00   |        6 |         600 |             600 |        30.00 | Sports     |       NULL |
|        7 | Concert 3             | 2023-12-21 | 21:00:00   |        7 |         700 |             700 |        70.00 | Concert    |       NULL |
|        8 | Movie Premiere        | 2023-12-22 | 18:30:00   |        8 |         250 |             250 |        20.00 | Movie      |       NULL |
|        9 | Sports Tournament     | 2023-12-23 | 14:30:00   |        9 |         450 |             450 |        35.00 | Sports     |       NULL |
|       10 | Concert 4             | 2023-12-24 | 19:00:00   |       10 |         800 |             800 |        80.00 | Concert    |       NULL |
+----------+-----------------------+------------+------------+----------+-------------+-----------------+--------------+------------+------------+
10 rows in set (0.00 sec)

mysql>
```

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



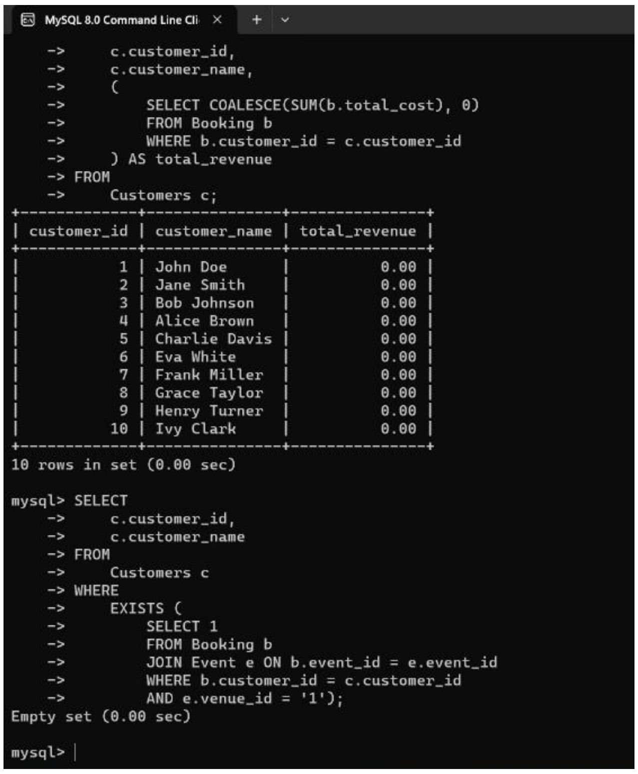
```
ERROR 1054 (42S22): Unknown column 'Event.event_type' in 'on clause'
mysql> SELECT * FROM Event;
+----------+----------------------+------------+------------+----------+-------------+-----------------+--------------+------------+------------+
| event_id | event_name           | event_date | event_time | venue_id | total_seats | available_seats | ticket_price | event_type | booking_id |
+----------+----------------------+------------+------------+----------+-------------+-----------------+--------------+------------+------------+
|        1 | Concert 1            | 2023-12-15 | 18:00:00   |        1 |         500 |             500 |        50.00 | Concert    |       NULL |
|        2 | Movie Night          | 2023-12-16 | 19:30:00   |        2 |         200 |             200 |        10.00 | Movie      |       NULL |
|        3 | Sports Event         | 2023-12-17 | 15:00:00   |        3 |         300 |             300 |        25.00 | Sports     |       NULL |
|        4 | Concert 2            | 2023-12-18 | 20:00:00   |        4 |         400 |             400 |        40.00 | Concert    |       NULL |
|        5 | Movie Marathon       | 2023-12-19 | 12:00:00   |        5 |         100 |             100 |        15.00 | Movie      |       NULL |
|        6 | Sports Championship  | 2023-12-20 | 17:00:00   |        6 |         600 |             600 |        30.00 | Sports     |       NULL |
|        7 | Concert 3            | 2023-12-21 | 21:00:00   |        7 |         700 |             700 |        70.00 | Concert    |       NULL |
|        8 | Movie Premiere       | 2023-12-22 | 18:30:00   |        8 |         250 |             250 |        20.00 | Movie      |       NULL |
|        9 | Sports Tournament    | 2023-12-23 | 14:30:00   |        9 |         450 |             450 |        35.00 | Sports     |       NULL |
|       10 | Concert 4            | 2023-12-24 | 19:00:00   |       10 |         800 |             800 |        80.00 | Concert    |       NULL |
+----------+----------------------+------------+------------+----------+-------------+-----------------+--------------+------------+------------+
10 rows in set (0.00 sec)

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 1  |        50.00 |
|        4 | Concert 2  |        40.00 |
|        7 | Concert 3  |        70.00 |
|       10 | Concert 4  |        80.00 |
+----------+------------+--------------+
4 rows in set (0.00 sec)

mysql>
```

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



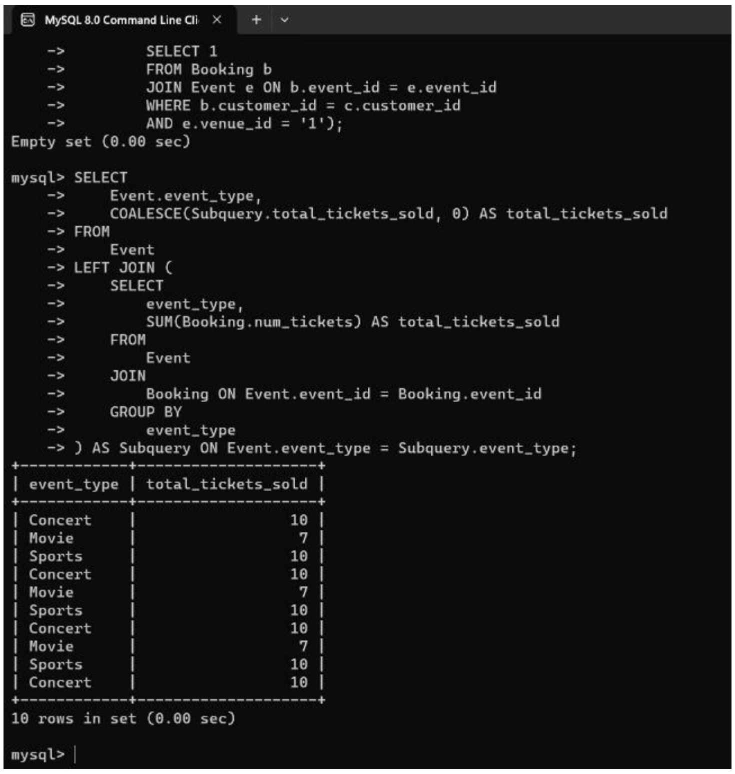
```
    ->         SELECT AVG(ticket_price)
    ->         FROM Event
    ->     );
+----------+------------+--------------+
| event_id | event_name | ticket_price |
+----------+------------+--------------+
|        1 | Concert 1  |        50.00 |
|        4 | Concert 2  |        40.00 |
|        7 | Concert 3  |        70.00 |
|       10 | Concert 4  |        80.00 |
+----------+------------+--------------+
4 rows in set (0.00 sec)

mysql> SELECT
    ->     c.customer_id,
    ->     c.customer_name,
    ->     (
    ->         SELECT COALESCE(SUM(b.total_cost), 0)
    ->         FROM Booking b
    ->         WHERE b.customer_id = c.customer_id
    ->     ) AS total_revenue
    -> FROM
    ->     Customers c;
+-------------+---------------+---------------+
| customer_id | customer_name | total_revenue |
+-------------+---------------+---------------+
|           1 | John Doe      |          0.00 |
|           2 | Jane Smith    |          0.00 |
|           3 | Bob Johnson   |          0.00 |
|           4 | Alice Brown   |          0.00 |
|           5 | Charlie Davis |          0.00 |
|           6 | Eva White     |          0.00 |
|           7 | Frank Miller  |          0.00 |
|           8 | Grace Taylor  |          0.00 |
|           9 | Henry Turner  |          0.00 |
|          10 | Ivy Clark     |          0.00 |
+-------------+---------------+---------------+
10 rows in set (0.00 sec)

mysql>
```

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



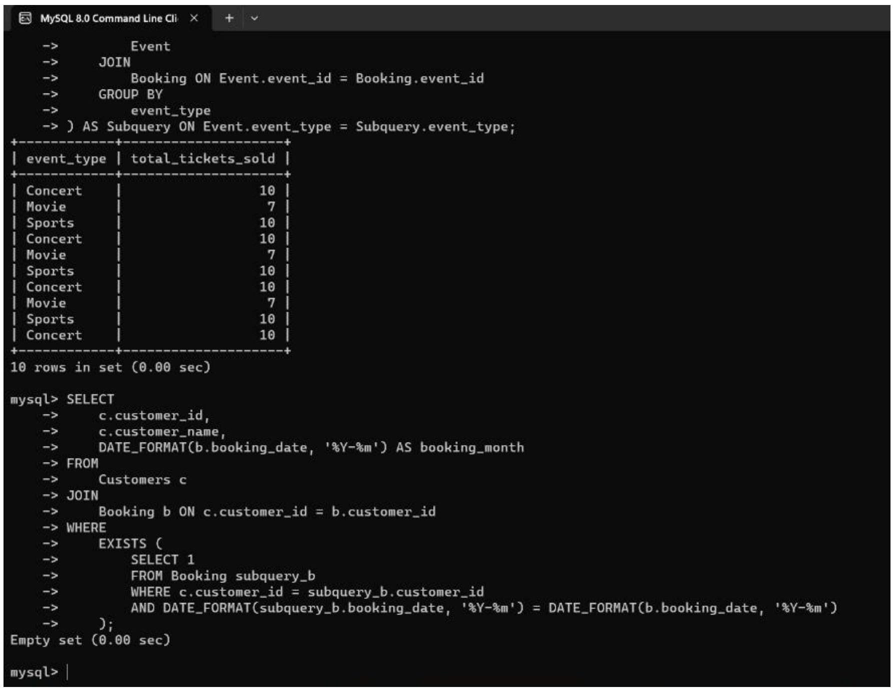
```
    ->     c.customer_id,
    ->     c.customer_name,
    ->     (
    ->         SELECT COALESCE(SUM(b.total_cost), 0)
    ->         FROM Booking b
    ->         WHERE b.customer_id = c.customer_id
    ->     ) AS total_revenue
    -> FROM
    ->     Customers c;
+-------------+---------------+---------------+
| customer_id | customer_name | total_revenue |
+-------------+---------------+---------------+
|           1 | John Doe      |          0.00 |
|           2 | Jane Smith    |          0.00 |
|           3 | Bob Johnson   |          0.00 |
|           4 | Alice Brown   |          0.00 |
|           5 | Charlie Davis |          0.00 |
|           6 | Eva White     |          0.00 |
|           7 | Frank Miller  |          0.00 |
|           8 | Grace Taylor  |          0.00 |
|           9 | Henry Turner  |          0.00 |
|          10 | Ivy Clark     |          0.00 |
+-------------+---------------+---------------+
10 rows in set (0.00 sec)

mysql> SELECT
    ->     c.customer_id,
    ->     c.customer_name
    -> FROM
    ->     Customers c
    -> WHERE
    ->     EXISTS (
    ->         SELECT 1
    ->         FROM Booking b
    ->         JOIN Event e ON b.event_id = e.event_id
    ->         WHERE b.customer_id = c.customer_id
    ->         AND e.venue_id = '1');
Empty set (0.00 sec)

mysql>
```

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



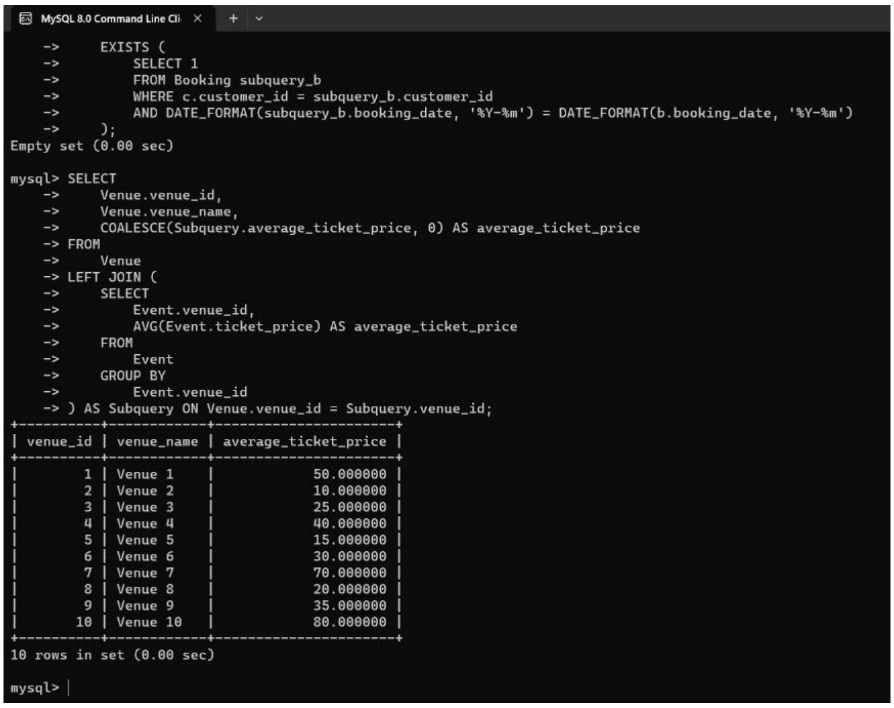
```
    ->         SELECT 1
    ->         FROM Booking b
    ->         JOIN Event e ON b.event_id = e.event_id
    ->         WHERE b.customer_id = c.customer_id
    ->         AND e.venue_id = '1');
Empty set (0.00 sec)

mysql> SELECT
    ->     Event.event_type,
    ->     COALESCE(Subquery.total_tickets_sold, 0) AS total_tickets_sold
    -> FROM
    ->     Event
    -> LEFT JOIN (
    ->     SELECT
    ->         event_type,
    ->         SUM(Booking.num_tickets) AS total_tickets_sold
    ->     FROM
    ->         Event
    ->     JOIN
    ->         Booking ON Event.event_id = Booking.event_id
    ->     GROUP BY
    ->         event_type
    -> ) AS Subquery ON Event.event_type = Subquery.event_type;
+------------+--------------------+
| event_type | total_tickets_sold |
+------------+--------------------+
| Concert    |                 10 |
| Movie      |                  7 |
| Sports     |                 10 |
| Concert    |                 10 |
| Movie      |                  7 |
| Sports     |                 10 |
| Concert    |                 10 |
| Movie      |                  7 |
| Sports     |                 10 |
| Concert    |                 10 |
+------------+--------------------+
10 rows in set (0.00 sec)

mysql>
```

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



```
    ->         Event
    ->     JOIN
    ->         Booking ON Event.event_id = Booking.event_id
    ->     GROUP BY
    ->         event_type
    -> ) AS Subquery ON Event.event_type = Subquery.event_type;
+------------+--------------------+
| event_type | total_tickets_sold |
+------------+--------------------+
| Concert    |                 10 |
| Movie      |                  7 |
| Sports     |                 10 |
| Concert    |                 10 |
| Movie      |                  7 |
| Sports     |                 10 |
| Concert    |                 10 |
| Movie      |                  7 |
| Sports     |                 10 |
| Concert    |                 10 |
+------------+--------------------+
10 rows in set (0.00 sec)

mysql> SELECT
    ->     c.customer_id,
    ->     c.customer_name,
    ->     DATE_FORMAT(b.booking_date, '%Y-%m') AS booking_month
    -> FROM
    ->     Customers c
    -> JOIN
    ->     Booking b ON c.customer_id = b.customer_id
    -> WHERE
    ->     EXISTS (
    ->         SELECT 1
    ->         FROM Booking subquery_b
    ->         WHERE c.customer_id = subquery_b.customer_id
    ->         AND DATE_FORMAT(subquery_b.booking_date, '%Y-%m') = DATE_FORMAT(b.booking_date, '%Y-%m')
    ->     );
Empty set (0.00 sec)

mysql> |
```

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



```
    ->     EXISTS (
    ->         SELECT 1
    ->         FROM Booking subquery_b
    ->         WHERE c.customer_id = subquery_b.customer_id
    ->         AND DATE_FORMAT(subquery_b.booking_date, '%Y-%m') = DATE_FORMAT(b.booking_date, '%Y-%m')
    ->     );
Empty set (0.00 sec)

mysql> SELECT
    ->     Venue.venue_id,
    ->     Venue.venue_name,
    ->     COALESCE(Subquery.average_ticket_price, 0) AS average_ticket_price
    -> FROM
    ->     Venue
    -> LEFT JOIN (
    ->     SELECT
    ->         Event.venue_id,
    ->         AVG(Event.ticket_price) AS average_ticket_price
    ->     FROM
    ->         Event
    ->     GROUP BY
    ->         Event.venue_id
    -> ) AS Subquery ON Venue.venue_id = Subquery.venue_id;
+----------+------------+----------------------+
| venue_id | venue_name | average_ticket_price |
+----------+------------+----------------------+
|        1 | Venue 1    |            50.000000 |
|        2 | Venue 2    |            10.000000 |
|        3 | Venue 3    |            25.000000 |
|        4 | Venue 4    |            40.000000 |
|        5 | Venue 5    |            15.000000 |
|        6 | Venue 6    |            30.000000 |
|        7 | Venue 7    |            70.000000 |
|        8 | Venue 8    |            20.000000 |
|        9 | Venue 9    |            35.000000 |
|       10 | Venue 10   |            80.000000 |
+----------+------------+----------------------+
10 rows in set (0.00 sec)

mysql> |
```