Assignment_5_sql

Tasks 1: Database Design:

1. Create the database named "TicketBookingSystem"

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

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

```
mysql> use Ticketbookingsystem;
Database changed
mysql> create table venue
    -> (venue_id INT PRIMARY KEY,
    -> venue_name VARCHAR(50),
    -> address VARCHAR(50));
Query OK, 0 rows affected (0.04 sec)
```

```
mysql> create table customer
    -> (customer_id INT PRIMARY KEY,
    -> customer_name VARCHAR(50),
    -> email varchar(50),
    -> phone varchar(20),
    -> booking_id INT);
Query OK, 0 rows affected (0.02 sec)
```

```
mysql> create table booking
    -> (booking_id INT PRIMARY KEY,
    -> customer_id INT,
    -> event_id INT,
    -> num_tickets INT,
    -> total_cost DECIMAL(10,2),
    -> booking_date DATE,
    -> foreign key (customer_id) references customer(customer_id));
Query OK, 0 rows affected (0.05 sec)
```

```
mysql> CREATE TABLE Event (
    ->     event_id INT PRIMARY KEY,
    ->     event_name VARCHAR(255),
    ->     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),
    ->     FOREIGN KEY (booking_id) REFERENCES Booking(booking_id)
    -> );
Query OK, 0 rows affected (0.05 sec)
```

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

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

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

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

```
mysql> insert into venue(
    ->     venue_id,venue_name,address)values(1,'townhall','sohajan'),
    ->     (2,'mandirgate','bagranjchowk'),
    ->     (3,'gandhistatue','gandhichowk'),
    ->     (4,'collegegate','dsmcollege'),
    ->     (5,'shayammandir','barmasiya'),
    ->     (6,'stationclub','stationroad'),
    ->     (7,'chandwari','jc sah road'),
    ->     (8,'nagarpanchat','nawab road'),
    ->     (9,'DYZ','old bus stand'),
    ->     (10,'karpuri statue','karpuri chowk');
Query OK, 10 rows affected (0.01 sec)
Records: 10  Duplicates: 0  Warnings: 0
```

Assignment_5_sql

```
mysql> select * from venue;
+----------+-----------------+----------------+
| venue_id | venue_name      | address        |
+----------+-----------------+----------------+
|        1 | townhall        | sohajan        |
|        2 | mandirgate      | bagranjchowk   |
|        3 | gandhistatue    | gandhichowk    |
|        4 | collegegate     | dsmcollege     |
|        5 | shayammandir    | barmasiya      |
|        6 | stationclub     | stationroad    |
|        7 | chandwari       | jc sah road    |
|        8 | nagarpanchat    | nawab road     |
|        9 | DYZ             | old bus stand  |
|       10 | karpuri statue  | karpuri chowk  |
+----------+-----------------+----------------+
10 rows in set (0.00 sec)
```

```
mysql> insert into customer values
    -> (1,'sourav','sou@gmail.com','9119197656',01),
    -> (2,'Anurag','anu@gmail.com','8182282828',02),
    -> (3,'rishav','riv@gmail.com','8178282828',03),
    -> (4,'subham','sub@gmail.com','8198282828',04),
    -> (5,'mayank','may@gmail.com','8112282828',05),
    -> (6,'satyam','sat@gmail.com','8156282828',06),
    -> (7,'rahul','rah@gmail.com','8109282828',07),
    -> (8,'abhishek','abhi@gmail.com','8166282828',08),
    -> (9,'rocky','roc@gmail.com','8199282828',09),
    -> (10,'purusottam','puru@gmail.com','8922282828',010);
Query OK, 10 rows affected (0.00 sec)
Records: 10  Duplicates: 0  Warnings: 0
```

```
mysql> select * from customer;
+-------------+---------------+----------------+------------+------------+
| customer_id | customer_name | email          | phone      | booking_id |
+-------------+---------------+----------------+------------+------------+
|           1 | sourav        | sou@gmail.com  | 9119197656 |          1 |
|           2 | Anurag        | anu@gmail.com  | 8182282828 |          2 |
|           3 | rishav        | riv@gmail.com  | 8178282828 |          3 |
|           4 | subham        | sub@gmail.com  | 8198282828 |          4 |
|           5 | mayank        | may@gmail.com  | 8112282828 |          5 |
|           6 | satyam        | sat@gmail.com  | 8156282828 |          6 |
|           7 | rahul         | rah@gmail.com  | 8109282828 |          7 |
|           8 | abhishek      | abhi@gmail.com | 8166282828 |          8 |
|           9 | rocky         | roc@gmail.com  | 8199282828 |          9 |
|          10 | purusottam    | puru@gmail.com | 8922282828 |         10 |
+-------------+---------------+----------------+------------+------------+
10 rows in set (0.00 sec)
```

Assignment_5_sql

```
mysql> insert into booking values
    -> (1,1,1,2,50.49,'2024-02-12'),
    -> (2,2,2,4,99.49,'2024-02-13'),
    -> (3,3,3,6,100.49,'2024-02-14'),
    -> (4,4,4,3,45.49,'2024-02-15'),
    -> (5,5,5,2,34.49,'2024-02-16'),
    -> (6,6,6,8,40.49,'2024-02-17'),
    -> (7,7,7,10,60.49,'2024-02-18'),
    -> (8,8,8,6,70.49,'2024-02-19'),
    -> (9,9,9,3,80.89,'2024-02-20'),
    -> (10,10,10,2,90.49,'2024-02-22');
Query OK, 10 rows affected (0.00 sec)
Records: 10  Duplicates: 0  Warnings: 0

mysql> select * from booking;
```

booking_id	customer_id	event_id	num_tickets	total_cost	booking_date
1	1	1	2	50.49	2024-02-12
2	2	2	4	99.49	2024-02-13
3	3	3	6	100.49	2024-02-14
4	4	4	3	45.49	2024-02-15
5	5	5	2	34.49	2024-02-16
6	6	6	8	40.49	2024-02-17
7	7	7	10	60.49	2024-02-18
8	8	8	6	70.49	2024-02-19
9	9	9	3	80.89	2024-02-20
10	10	10	2	90.49	2024-02-22

10 rows in set (0.00 sec)

```
mysql> insert into event values
    -> (101,'gadar','2024-03-1','01:50:30',1,50,25,99.50,'Movie',1),
    -> (102,'cricket','2024-03-2','01:51:30',2,40,15,49.50,'Sports',2),
    -> (103,'atif','2024-03-3','02:50:30',3,500,225,200.00,'Concert',3),
    -> (104,'solay','2024-03-4','06:30:30',4,50,25,99.50,'Movie',4),
    -> (105,'carrom','2024-03-5','07:51:30',5,40,15,49.50,'Sports',5),
    -> (106,'honeysingh','2024-03-6','03:50:30',6,500,225,200.00,'Concert',6),
    -> (107,'karan-arjun','2024-03-7','04:50:30',7,50,25,99.50,'Movie',7),
    -> (108,'badminton','2024-03-8','05:51:30',8,40,15,49.50,'Sports',8),
    -> (109,'jubin','2024-03-9','06:50:30',9,500,225,200.00,'Concert',9),
    -> (1010,'uditnarayan','2024-03-10','08:50:30',10,1000,150,250.00,'Concert',10);
Query OK, 10 rows affected (0.01 sec)
Records: 10  Duplicates: 0  Warnings: 0
```

```
mysql> select * from event;
+----------+------------+------------+------------+----------+-------------+----------------+--------------+------------+------------+
| event_id | event_name | event_date | event_time | venue_id | total_seats | available_seats | ticket_price | event_type | booking_id |
+----------+------------+------------+------------+----------+-------------+----------------+--------------+------------+------------+
|      101 | gadar      | 2024-03-01 | 01:50:30   |        1 |          50 |             25 |        99.50 | Movie      |          1 |
|      102 | cricket    | 2024-03-02 | 01:51:30   |        2 |          40 |             15 |        49.50 | Sports     |          2 |
|      103 | atif       | 2024-03-03 | 02:50:30   |        3 |         500 |            225 |       200.00 | Concert    |          3 |
|      104 | solay      | 2024-03-04 | 06:30:30   |        4 |          50 |             25 |        99.50 | Movie      |          4 |
|      105 | carrom     | 2024-03-05 | 07:51:30   |        5 |          40 |             15 |        49.50 | Sports     |          5 |
|      106 | honeysingh | 2024-03-06 | 03:50:30   |        6 |         500 |            225 |       200.00 | Concert    |          6 |
|      107 | karan-arjun | 2024-03-07 | 04:50:30   |        7 |          50 |             25 |        99.50 | Movie      |          7 |
|      108 | badminton  | 2024-03-08 | 05:51:30   |        8 |          40 |             15 |        49.50 | Sports     |          8 |
|      109 | jubin      | 2024-03-09 | 06:50:30   |        9 |         500 |            225 |       200.00 | Concert    |          9 |
|     1010 | uditnarayan | 2024-03-10 | 08:50:30   |       10 |        1000 |            150 |       250.00 | Concert    |         10 |
+----------+------------+------------+------------+----------+-------------+----------------+--------------+------------+------------+
10 rows in set (0.00 sec)
```

2. Write a SQL query to list all Events.

```
mysql> select event_id,event_name from event;
+----------+-------------+
| event_id | event_name  |
+----------+-------------+
|      101 | gadar       |
|      102 | cricket     |
|      103 | atif        |
|      104 | solay       |
|      105 | carrom      |
|      106 | honeysingh  |
|      107 | karan-arjun |
|      108 | badminton   |
|      109 | jubin       |
|     1010 | uditnarayan |
+----------+-------------+
10 rows in set (0.00 sec)
```

Assignment_5_sql

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

```
mysql> select event_id,event_name,available_seats from event;
+----------+-------------+-----------------+
| event_id | event_name  | available_seats |
+----------+-------------+-----------------+
|      101 | gadar       |              25 |
|      102 | cricket     |              15 |
|      103 | atif        |             225 |
|      104 | solay       |              25 |
|      105 | carrom      |              15 |
|      106 | honeysingh  |             225 |
|      107 | karan-arjun |              25 |
|      108 | badminton   |              15 |
|      109 | jubin       |             225 |
|     1010 | uditnarayan |             150 |
+----------+-------------+-----------------+
10 rows in set (0.00 sec)
```

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

```
mysql> SELECT * FROM Event
    -> WHERE event_name LIKE '%cri%';
+----------+------------+------------+------------+----------+-------------+----------------+--------------+------------+------------+
| event_id | event_name | event_date | event_time | venue_id | total_seats | available_seats | ticket_price | event_type | booking_id |
+----------+------------+------------+------------+----------+-------------+----------------+--------------+------------+------------+
|      102 | cricket    | 2024-03-02 | 01:51:30   |        2 |          40 |              15 |        49.50 | Sports     |          2 |
+----------+------------+------------+------------+----------+-------------+----------------+--------------+------------+------------+
1 row in set (0.00 sec)
```

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

I have taken range of price between 50 to 300...

```
mysql> SELECT * FROM Event
    -> WHERE ticket_price BETWEEN 50 AND 300;
+----------+-------------+------------+------------+----------+-------------+----------------+--------------+------------+------------+
| event_id | event_name  | event_date | event_time | venue_id | total_seats | available_seats | ticket_price | event_type | booking_id |
+----------+-------------+------------+------------+----------+-------------+----------------+--------------+------------+------------+
|      101 | gadar       | 2024-03-01 | 01:50:30   |        1 |          50 |              25 |        99.50 | Movie      |          1 |
|      103 | atif        | 2024-03-03 | 02:50:30   |        3 |         500 |             225 |       200.00 | Concert    |          3 |
|      104 | solay       | 2024-03-04 | 06:30:30   |        4 |          50 |              25 |        99.50 | Movie      |          4 |
|      106 | honeysingh  | 2024-03-06 | 03:50:30   |        6 |         500 |             225 |       200.00 | Concert    |          6 |
|      107 | karan-arjun | 2024-03-07 | 04:50:30   |        7 |          50 |              25 |        99.50 | Movie      |          7 |
|      109 | jubin       | 2024-03-09 | 06:50:30   |        9 |         500 |             225 |       200.00 | Concert    |          9 |
|     1010 | uditnarayan | 2024-03-10 | 08:50:30   |       10 |        1000 |             150 |       250.00 | Concert    |         10 |
+----------+-------------+------------+------------+----------+-------------+----------------+--------------+------------+------------+
7 rows in set (0.00 sec)
```

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

```
mysql> SELECT * FROM Event
    -> WHERE event_date BETWEEN '2024-01-01' AND '2024-12-31';
+----------+------------+------------+------------+----------+-------------+----------------+--------------+------------+------------+
| event_id | event_name | event_date | event_time | venue_id | total_seats | available_seats | ticket_price | event_type | booking_id |
+----------+------------+------------+------------+----------+-------------+----------------+--------------+------------+------------+
|      101 | gadar      | 2024-03-01 | 01:50:30   |        1 |          50 |             25 |        99.50 | Movie      |          1 |
|      102 | cricket    | 2024-03-02 | 01:51:30   |        2 |          40 |             15 |        49.50 | Sports     |          2 |
|      103 | atif       | 2024-03-03 | 02:50:30   |        3 |         500 |            225 |       200.00 | Concert    |          3 |
|      104 | solay      | 2024-03-04 | 06:30:30   |        4 |          50 |             25 |        99.50 | Movie      |          4 |
|      105 | carrom     | 2024-03-05 | 07:51:30   |        5 |          40 |             15 |        49.50 | Sports     |          5 |
|      106 | honeysingh | 2024-03-06 | 03:50:30   |        6 |         500 |            225 |       200.00 | Concert    |          6 |
|      107 | karan-arjun | 2024-03-07 | 04:50:30   |        7 |          50 |             25 |        99.50 | Movie      |          7 |
|      108 | badminton  | 2024-03-08 | 05:51:30   |        8 |          40 |             15 |        49.50 | Sports     |          8 |
|      109 | jubin      | 2024-03-09 | 06:50:30   |        9 |         500 |            225 |       200.00 | Concert    |          9 |
|     1010 | uditnarayan | 2024-03-10 | 08:50:30   |       10 |        1000 |            150 |       250.00 | Concert    |         10 |
+----------+------------+------------+------------+----------+-------------+----------------+--------------+------------+------------+
10 rows in set (0.00 sec)
```

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

```
mysql> select event_id,event_name,available_seats from event
    -> where event_type = 'Concert';
+----------+-------------+-----------------+
| event_id | event_name  | available_seats |
+----------+-------------+-----------------+
|      103 | atif        |             225 |
|      106 | honeysingh  |             225 |
|      109 | jubin       |             225 |
|     1010 | uditnarayan |             150 |
+----------+-------------+-----------------+
4 rows in set (0.00 sec)
```

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

```
ysql> CREATE PROCEDURE printInBatches()
    -> BEGIN
    ->     DECLARE start_val INT DEFAULT 5;
    ->     DECLARE size_val INT DEFAULT 5;
    ->     DECLARE total_val INT;
    ->     SELECT COUNT(customer_id) INTO total_val FROM customer;
    ->     WHILE start_val <= total_val DO
    ->         SELECT * FROM customer
    ->         ORDER BY customer_id
    ->         LIMIT size_val OFFSET start_val;
    ->         SET start_val = start_val + size_val;
    ->     END WHILE;
    -> END ##
uery OK, 0 rows affected (0.01 sec)

ysql>
ysql> DELIMITER ;
ysql> call printInBatches();
+-------------+---------------+-----------------+------------+-------------+
| customer_id | customer_name | email           | phone      | booking_id  |
+-------------+---------------+-----------------+------------+-------------+
|           6 | satyam        | sat@gmail.com   | 8156282828 |           6 |
|           7 | rahul         | rah@gmail.com   | 8109282828 |           7 |
|           8 | abhishek      | abhi@gmail.com  | 8166282828 |           8 |
|           9 | rocky         | roc@gmail.com   | 8199282828 |           9 |
|          10 | purusottam    | puru@gmail.com  | 8922282828 |          10 |
+-------------+---------------+-----------------+------------+-------------+
 rows in set (0.01 sec)
```

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

```
mysql> select * from booking
    -> where num_tickets>4;
+------------+-------------+----------+-------------+------------+--------------+
| booking_id | customer_id | event_id | num_tickets | total_cost | booking_date |
+------------+-------------+----------+-------------+------------+--------------+
|          3 |           3 |        3 |           6 |     100.49 | 2024-02-14   |
|          6 |           6 |        6 |           8 |      40.49 | 2024-02-17   |
|          7 |           7 |        7 |          10 |      60.49 | 2024-02-18   |
|          8 |           8 |        8 |           6 |      70.49 | 2024-02-19   |
+------------+-------------+----------+-------------+------------+--------------+
4 rows in set (0.00 sec)

mysql>
```

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

```
mysql> SELECT *
    -> FROM customers
    -> WHERE phone LIKE '%656';
ERROR 1146 (42S02): Table 'ticketbookingsystem.customers' doesn't exist
mysql> SELECT *
    -> FROM customer
    -> WHERE phone LIKE '%656';
+-------------+---------------+---------------+------------+------------+
| customer_id | customer_name | email         | phone      | booking_id |
+-------------+---------------+---------------+------------+------------+
|           1 | sourav        | sou@gmail.com | 9119197656 |          1 |
+-------------+---------------+---------------+------------+------------+
1 row in set (0.00 sec)
```

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

```
mysql> SELECT event_id, event_name, total_seats
    -> FROM event
    -> WHERE total_seats> 50
    -> ORDER BY total_seats ASC;
+----------+-------------+-------------+
| event_id | event_name  | total_seats |
+----------+-------------+-------------+
|      103 | atif        |         500 |
|      106 | honeysingh  |         500 |
|      109 | jubin       |         500 |
|     1010 | uditnarayan |        1000 |
+----------+-------------+-------------+
4 rows in set (0.00 sec)
```

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

```
mysql> SELECT event_id, event_name, event_date
    -> FROM event
    -> WHERE event_name NOT LIKE 'a%' AND event_name NOT LIKE 'g%' AND event_name NOT LIKE 't%';
+----------+------------+------------+
| event_id | event_name | event_date |
+----------+------------+------------+
|      102 | cricket    | 2024-03-02 |
|      104 | solay      | 2024-03-04 |
|      105 | carrom     | 2024-03-05 |
|      106 | honeysingh | 2024-03-06 |
|      107 | karan-arjun | 2024-03-07 |
|      108 | badminton  | 2024-03-08 |
|      109 | jubin      | 2024-03-09 |
|     1010 | uditnarayan | 2024-03-10 |
+----------+------------+------------+
8 rows in set (0.00 sec)
```

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

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

```
ysql> 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 |
----------+-------------+----------------------+
      101 | gadar       |            99.500000 |
      102 | cricket     |            49.500000 |
      103 | atif        |           200.000000 |
      104 | solay       |            99.500000 |
      105 | carrom      |            49.500000 |
      106 | honeysingh  |           200.000000 |
      107 | karan-arjun |            99.500000 |
      108 | badminton   |            49.500000 |
      109 | jubin       |           200.000000 |
     1010 | uditnarayan |           250.000000 |
----------+-------------+----------------------+
10 rows in set (0.00 sec)
```

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

```
ysql> SELECT
    ->     e.event_id,
    ->     e.event_name,
    ->     SUM(ticket_price) AS total_revenue
    -> FROM
    ->     event e
    -> GROUP BY
    ->     e.event_id, e.event_name;
----------+-------------+-----------------+
 event_id | event_name  | total_revenue   |
----------+-------------+-----------------+
      101 | gadar       |           99.50 |
      102 | cricket     |           49.50 |
      103 | atif        |          200.00 |
      104 | solay       |           99.50 |
      105 | carrom      |           49.50 |
      106 | honeysingh  |          200.00 |
      107 | karan-arjun |           99.50 |
      108 | badminton   |           49.50 |
      109 | jubin       |          200.00 |
     1010 | uditnarayan |          250.00 |
----------+-------------+-----------------+
0 rows in set (0.00 sec)
```

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

```
ysql> SELECT
   ->     e.event_id,
   ->     e.event_name,
   ->     booking.num_tickets
   -> FROM
   ->     event e
   -> join booking on e.event_id=booking.event_id
   -> GROUP BY
   ->     e.event_id, e.event_name
   -> ORDER BY
   ->     booking.num_tickets DESC
   -> LIMIT 1;
mpty set (0.00 sec)
```

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

```
mysql> SELECT
    ->     e.event_id,
    ->     e.event_name,
    ->     SUM(b.num_tickets) AS total_tickets_sold
    -> FROM
    ->     event e
    -> JOIN
    ->     booking b ON e.event_id = b.event_id
    -> GROUP BY
    ->     e.event_id, e.event_name;
Empty set (0.00 sec)
```

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

```
mysql> SELECT
    ->     e.event_id,
    ->     e.event_name
    -> FROM
    ->     event e
    -> LEFT JOIN
    ->     booking b ON e.event_id = b.event_id
    -> WHERE
    ->     b.event_id IS NULL;
+----------+--------------+
| event_id | event_name   |
+----------+--------------+
|      101 | gadar        |
|      102 | cricket      |
|      103 | atif         |
|      104 | solay        |
|      105 | carrom       |
|      106 | honeysingh   |
|      107 | karan-arjun  |
|      108 | badminton    |
|      109 | jubin        |
|     1010 | uditnarayan  |
+----------+--------------+
10 rows in set (0.00 sec)
```

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

```
mysql> SELECT
    ->     c.customer_id,
    ->     c.customer_name,
    ->     c.email,
    ->     MAX(b.num_tickets) AS max_tickets_booked
    -> FROM
    ->     customer c
    -> JOIN
    ->     booking b ON c.customer_id = b.customer_id
    -> GROUP BY
    ->     c.customer_id, c.customer_name, c.email
    -> ORDER BY
    ->     max_tickets_booked DESC
    -> LIMIT 1;
+-------------+---------------+---------------+--------------------+
| customer_id | customer_name | email         | max_tickets_booked |
+-------------+---------------+---------------+--------------------+
|           7 | rahul         | rah@gmail.com |                 10 |
+-------------+---------------+---------------+--------------------+
1 row in set (0.00 sec)
```

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

```
mysql> SELECT
    ->     e.event_id,
    ->     e.event_name,
    ->     MONTH(b.booking_date) AS booking_month,
    ->     SUM(b.num_tickets) AS total_tickets_sold
    -> FROM
    ->     event e
    -> JOIN
    ->     booking b ON e.event_id = b.event_id
    -> GROUP BY
    ->     e.event_id, e.event_name, booking_month
    -> ORDER BY
    ->     booking_month, e.event_id;
Empty set (0.00 sec)
```

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

```
mysql> SELECT
    ->     v.venue_id,
    ->     v.venue_name,
    ->     AVG(e.ticket_price) AS average_ticket_price
    -> FROM
    ->     venue v
    -> JOIN
    ->     event e ON v.venue_id = e.venue_id
    -> GROUP BY
    ->     v.venue_id, v.venue_name;
+----------+----------------+----------------------+
| venue_id | venue_name     | average_ticket_price |
+----------+----------------+----------------------+
|        1 | townhall       |            99.500000 |
|        2 | mandirgate     |            49.500000 |
|        3 | gandhistatue   |           200.000000 |
|        4 | collegegate    |            99.500000 |
|        5 | shayammandir   |            49.500000 |
|        6 | stationclub    |           200.000000 |
|        7 | chandwari      |            99.500000 |
|        8 | nagarpanchat   |            49.500000 |
|        9 | DYZ            |           200.000000 |
|       10 | karpuri statue |           250.000000 |
+----------+----------------+----------------------+
10 rows in set (0.00 sec)
```

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

```
mysql> SELECT
    ->     event_type,
    ->     SUM(num_tickets) AS total_tickets_sold
    -> FROM
    ->     Event e
    -> JOIN
    ->     Booking b ON e.event_id = b.event_id
    -> GROUP BY
    ->     event_type;
Empty set (0.00 sec)
```

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

```
mysql> SELECT
    ->     YEAR(e.event_date) AS event_year,
    ->     SUM(b.total_cost) AS total_revenue
    -> FROM
    ->     Event e
    -> JOIN
    ->     Booking b ON e.event_id = b.event_id
    -> GROUP BY
    ->     event_year;
Empty set (0.00 sec)
```

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

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

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

```
mysql> SELECT
    ->     c.customer_id,
    ->     c.customer_name,
    ->     SUM(b.total_cost) AS total_revenue
    -> FROM
    ->     Customer c
    -> JOIN
    ->     Booking b ON c.customer_id = b.customer_id
    -> GROUP BY
    ->     c.customer_id, c.customer_name;
+-------------+---------------+-----------------+
| customer_id | customer_name | total_revenue   |
+-------------+---------------+-----------------+
|           1 | sourav        |           50.49 |
|           2 | Anurag        |           99.49 |
|           3 | rishav        |          100.49 |
|           4 | subham        |           45.49 |
|           5 | mayank        |           34.49 |
|           6 | satyam        |           40.49 |
|           7 | rahul         |           60.49 |
|           8 | abhishek      |           70.49 |
|           9 | rocky         |           80.89 |
|          10 | purusottam    |           90.49 |
+-------------+---------------+-----------------+
10 rows in set (0.00 sec)
```

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

```
mysql> SELECT
    ->     e.event_type,
    ->     v.venue_id,
    ->     v.venue_name,
    ->     AVG(e.ticket_price) AS average_ticket_price
    -> FROM
    ->     Event e
    -> JOIN
    ->     booking b ON e.event_id = b.event_id
    -> JOIN
    ->     Venue v ON e.venue_id = v.venue_id
    -> GROUP BY
    ->     e.event_type, v.venue_id, v.venue_name;
Empty set (0.00 sec)
```

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

```
mysql> SELECT
    ->     c.customer_id,
    ->     c.customer_name,
    ->     COUNT(b.booking_id) AS total_tickets_purchased
    -> FROM
    ->     Customer c
    -> JOIN
    ->     Booking b ON c.customer_id = b.customer_id
    -> WHERE
    ->     b.booking_date >= CURDATE() - INTERVAL 30 DAY
    -> GROUP BY
    ->     c.customer_id, c.customer_name;
+-------------+---------------+--------------------------+
| customer_id | customer_name | total_tickets_purchased  |
+-------------+---------------+--------------------------+
|           1 | sourav        |                        1 |
|           2 | Anurag        |                        1 |
|           3 | rishav        |                        1 |
|           4 | subham        |                        1 |
|           5 | mayank        |                        1 |
|           6 | satyam        |                        1 |
|           7 | rahul         |                        1 |
|           8 | abhishek      |                        1 |
|           9 | rocky         |                        1 |
|          10 | purusottam    |                        1 |
+-------------+---------------+--------------------------+
10 rows in set (0.00 sec)
```

Tasks 4: Subquery and its types

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

```
mysql> select venue_id,avg(ticket_price)
    -> from event
    -> where venue_id in(select venue_id from venue)
    -> group by venue_id;
+----------+--------------------+
| venue_id | avg(ticket_price) |
+----------+--------------------+
|        1 |          99.500000 |
|        2 |          49.500000 |
|        3 |         200.000000 |
|        4 |          99.500000 |
|        5 |          49.500000 |
|        6 |         200.000000 |
|        7 |          99.500000 |
|        8 |          49.500000 |
|        9 |         200.000000 |
|       10 |         250.000000 |
+----------+--------------------+
10 rows in set (0.00 sec)
```

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

```
mysql> select event_id,event_name,total_seats-available_seats as totalsoldtickets
    -> from event
    -> where event_id in(
    -> select event_id from event where total_seats-available_seats>=total_seats*0.5);
+----------+-------------+------------------+
| event_id | event_name  | totalsoldtickets |
+----------+-------------+------------------+
|      101 | gadar       |               25 |
|      102 | cricket     |               25 |
|      103 | atif        |              275 |
|      104 | solay       |               25 |
|      105 | carrom      |               25 |
|      106 | honeysingh  |              275 |
|      107 | karan-arjun |               25 |
|      108 | badminton   |               25 |
|      109 | jubin       |              275 |
|     1010 | uditnarayan |              850 |
+----------+-------------+------------------+
10 rows in set (0.00 sec)
```

Assignment_5_sql

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

```
mysql> select event_id, event_name,total_seats-available_seats as TicketsSol
d from event;
+----------+-------------+-------------+
| event_id | event_name  | TicketsSold |
+----------+-------------+-------------+
|      101 | gadar       |          25 |
|      102 | cricket     |          25 |
|      103 | atif        |         275 |
|      104 | solay       |          25 |
|      105 | carrom      |          25 |
|      106 | honeysingh  |         275 |
|      107 | karan-arjun |          25 |
|      108 | badminton   |          25 |
|      109 | jubin       |         275 |
|     1010 | uditnarayan |         850 |
+----------+-------------+-------------+
10 rows in set (0.00 sec)
```

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

```
mysql> select customer_id, customer_name from customer c
    ->
    -> where not exists(select customer_id from booking b where b.customer_i
d=c.customer_id);
Empty set (0.00 sec)
```

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

```
ysql> select event_id,event_name
   -> from event e
   -> where event_id not in(select event_id from booking);
----------+-------------+
 event_id | event_name  |
----------+-------------+
      101 | gadar       |
      102 | cricket     |
      103 | atif        |
      104 | solay       |
      105 | carrom      |
      106 | honeysingh  |
      107 | karan-arjun |
      108 | badminton   |
      109 | jubin       |
     1010 | uditnarayan |
----------+-------------+
0 rows in set (0.00 sec)
```

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

```
mysql> select event_type
    -> from (select e.event_type,sum(b.num_tickets) as total_sold
    -> from event e
    -> join booking b on e.event_id=b.event_id
    -> group by event_type) as total_event_sumary;
Empty set (0.00 sec)
```

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

```
mysql> select event_id,event_name,ticket_price from event
    -> where ticket_price > (select avg(ticket_price) from event);
+----------+-------------+--------------+
| event_id | event_name  | ticket_price |
+----------+-------------+--------------+
|      103 | atif        |       200.00 |
|      106 | honeysingh  |       200.00 |
|      109 | jubin       |       200.00 |
|     1010 | uditnarayan |       250.00 |
+----------+-------------+--------------+
4 rows in set (0.00 sec)

mysql> select avg(ticket_price) from event;
+-------------------+
| avg(ticket_price) |
+-------------------+
|        129.700000 |
+-------------------+
1 row in set (0.00 sec)
```

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

```
mysql> select c.customer_id,c.customer_name,
    -> (select sum(b.num_tickets*e.ticket_price) from booking b
    -> join event e on b.event_id=e.event_id
    -> where b.customer_id=c.customer_id) as Totalrevenue
    -> from customer c;
+-------------+---------------+--------------+
| customer_id | customer_name | Totalrevenue |
+-------------+---------------+--------------+
|           1 | sourav        |         NULL |
|           2 | Anurag        |         NULL |
|           3 | rishav        |         NULL |
|           4 | subham        |         NULL |
|           5 | mayank        |         NULL |
|           6 | satyam        |         NULL |
|           7 | rahul         |         NULL |
|           8 | abhishek      |         NULL |
|           9 | rocky         |         NULL |
|          10 | purusottam    |         NULL |
+-------------+---------------+--------------+
10 rows in set (0.00 sec)
```

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

```
mysql> select c.customer_id,c.customer_name
    -> from customer c
    -> where exists(select c.customer_id from booking b
    -> join event e on b.event_id=e.event_id
    -> where b.customer_id=c.customer_id
    -> and e.venue_id=2);
Empty set (0.00 sec)
```

.

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

```
mysql> select event_type, sum(totalTickets) as totalTickets
    -> from
    -> (select e.event_type,sum(b.num_tickets) as totalTickets from event e
    -> join booking b on e.event_id=b.event_id
    -> group by e.event_type,b.event_id) as tickets
    -> group by event_type;
Empty set (0.00 sec)
```

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

```
mysql> select c.customer_id,c.customer_name
    -> from customer c
    -> where exists(select c.customer_id from booking b
    -> join event e on b.event_id=e.event_id
    -> where b.customer_id=c.customer_id
    -> and date_format(b.booking_date,'%Y-%m')='2024-04');
Empty set (0.00 sec)
```

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

```
ysql> select avg(ticket_price)as AveragePrice,venue_id
   -> from event
   -> where venue_id in(select venue_id from venue)
   -> group by venue_id;
--------------+----------+
 AveragePrice | venue_id |
--------------+----------+
    99.500000 |        1 |
    49.500000 |        2 |
   200.000000 |        3 |
    99.500000 |        4 |
    49.500000 |        5 |
   200.000000 |        6 |
    99.500000 |        7 |
    49.500000 |        8 |
   200.000000 |        9 |
   250.000000 |       10 |
--------------+----------+
0 rows in set (0.00 sec)
```