

ASSIGNMENT 5

TICKET BOOKING SYSTEM

AKILESH K

Task 1

1)

```
mysql> create database ticketbookingsystem;  
Query OK, 1 row affected (0.09 sec)  
  
mysql> use ticketbookingsystem;  
Database changed
```

2)

```
mysql> CREATE TABLE venu (  
->     venue_id INT PRIMARY KEY,  
->     venue_name VARCHAR(255),  
->     address VARCHAR(255)  
-> );  
Query OK, 0 rows affected (0.17 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 Venu(venue_id));  
Query OK, 0 rows affected (0.11 sec)
```

```
mysql> CREATE TABLE customer (  
  ->     customer_id INT PRIMARY KEY,  
  ->     customer_name VARCHAR(255),  
  ->     email VARCHAR(255),  
  ->     phone_number VARCHAR(20),  
  ->     booking_id INT);  
Query OK, 0 rows affected (0.04 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),  
  ->     FOREIGN KEY (event_id) REFERENCES event(event_id)  
  -> );  
Query OK, 0 rows affected (0.08 sec)
```

```
mysql> show tables;  
+-----+  
| Tables_in_ticketbookingsystem |  
+-----+  
| booking                        |  
| customer                      |  
| event                        |  
| venu                         |  
+-----+  
4 rows in set (0.07 sec)
```

3)

Venue	Event	Customer	Booking
venue_id 1-----*	event_id 1-----*	customer_id 1-----*	booking_id
venue_name	event_name	customer_name	num_tickets
address	event_date	email	total_cost
	event_time	phone_number	booking_date
	total_seats		
	avail_seats		
	ticket_price		
	event_type		
	venue_id		

4)

```
mysql> desc booking;
```

Field	Type	Null	Key	Default	Extra
booking_id	int	NO	PRI	NULL	
customer_id	int	YES	MUL	NULL	
event_id	int	YES	MUL	NULL	
num_tickets	int	YES		NULL	
total_cost	decimal(10,2)	YES		NULL	
booking_date	date	YES		NULL	

6 rows in set (0.03 sec)

1)

```
mysql> select * from event;
```

event_id	event_name	event_date	event_time	venue_id	total_seats	available_seats	ticket_price	event_type
10	Event1	2023-01-01	12:00:00	1	200	150	50.00	Movie
21	Event2	2023-02-15	18:30:00	2	300	250	75.00	Sports
31	Event3	2023-03-20	20:00:00	3	150	100	100.00	Concert
41	Event4	2023-04-10	15:45:00	4	250	200	60.00	Movie
51	Event5	2023-05-05	14:00:00	5	180	120	90.00	Concert

5 rows in set (0.03 sec)

2)

```
mysql> SELECT event_id, event_name, event_date
-> FROM event;
+-----+-----+-----+
| event_id | event_name | event_date |
+-----+-----+-----+
|      11 | Event1     | 2023-01-01 |
|      21 | Event2     | 2023-02-15 |
|      31 | Event3     | 2023-03-20 |
|      41 | Event4     | 2023-04-10 |
|      51 | Event5     | 2023-05-05 |
+-----+-----+-----+
5 rows in set (0.00 sec)
```

3)

```
mysql> SELECT
->     event_id,
->     event_name,
->     event_date,
->     available_seats
-> FROM
->     event
-> WHERE
->     available_seats > 0;
+-----+-----+-----+-----+
| event_id | event_name | event_date | available_seats |
+-----+-----+-----+-----+
|      11 | Event1     | 2023-01-01 |          150 |
|      21 | Event2     | 2023-02-15 |          250 |
|      31 | Event3     | 2023-03-20 |          100 |
|      41 | Event4     | 2023-04-10 |          200 |
|      51 | Event5     | 2023-05-05 |          120 |
+-----+-----+-----+-----+
5 rows in set (0.00 sec)
```

4)

```
mysql> SELECT
->     event_id,
->     event_name
-> FROM
->     event
-> WHERE
->     event_name LIKE '%cup%';
Empty set (0.01 sec)
```

5)

```
mysql> SELECT
->     event_id,
->     event_name,
->     ticket_price
-> FROM
->     event
-> WHERE
->     ticket_price BETWEEN 1000 AND 2500;
Empty set (0.00 sec)
```

6)

```
mysql> SELECT
->     event_id,
->     event_name,
->     event_date
-> FROM
->     event
-> WHERE
->     event_date BETWEEN '2023-01-01' AND '2023-12-31';
+-----+-----+-----+
| event_id | event_name | event_date |
+-----+-----+-----+
|      11 | Event1     | 2023-01-01 |
|      21 | Event2     | 2023-02-15 |
|      31 | Event3     | 2023-03-20 |
|      41 | Event4     | 2023-04-10 |
|      51 | Event5     | 2023-05-05 |
+-----+-----+-----+
5 rows in set (0.00 sec)
```

7)

```
mysql> SELECT
->     e.event_id,
->     e.event_name,
->     e.event_date,
->     e.available_seats
-> FROM
->     event e
-> JOIN
->     booking b ON e.event_id = b.event_id
-> WHERE
->     e.event_name LIKE '%Concert%' ;
Empty set (0.00 sec)
```

8)

```
mysql> SELECT
->     customer_id,
->     customer_name,
->     email,
->     phone_number
-> FROM
->     customer
-> ORDER BY
->     customer_id
-> LIMIT 2 OFFSET 2;
+-----+-----+-----+-----+
| customer_id | customer_name | email                | phone_number |
+-----+-----+-----+-----+
|          333 | mike          | mike3@example.com    | 456-789-0123 |
|          444 | varun         | varun4@example.com   | 789-012-3456 |
+-----+-----+-----+-----+
2 rows in set (0.00 sec)
```

9)

```
mysql> SELECT
->     booking_id,
->     customer_id,
->     event_id,
->     num_tickets,
->     total_cost,
->     booking_date
-> FROM
->     booking
-> WHERE
->     num_tickets > 4;
Empty set (0.00 sec)
```

10)

```
mysql> SELECT
->     customer_id,
->     customer_name,
->     phone_number
-> FROM
->     customer
-> WHERE
->     phone_number LIKE '%000';
Empty set (0.00 sec)
```

11)

```
mysql> SELECT
->     event_id,
->     event_name,
->     total_seats
-> FROM
->     event
-> WHERE
->     total_seats > 15000
-> ORDER BY
->     total_seats;
Empty set (0.00 sec)
```

12)

```
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
11	Event1	2023-01-01	12:00:00	1	200	150	50.00	Movie	10
21	Event2	2023-02-15	18:30:00	2	300	250	75.00	Sports	22
31	Event3	2023-03-20	20:00:00	3	150	100	100.00	Concert	33
41	Event4	2023-04-10	15:45:00	4	250	200	60.00	Movie	44
51	Event5	2023-05-05	14:00:00	5	180	120	90.00	Concert	55

```
5 rows in set (0.00 sec)

mysql>
```

Task 3

1)

```
mysql> SELECT
->     e.event_id,
->     e.event_name,
->     AVG(e.ticket_price) AS average_ticket_price
-> FROM
->     event e
-> JOIN
->     booking b ON e.event_id = b.event_id
-> GROUP BY
->     e.event_id, e.event_name;
```

event_id	event_name	average_ticket_price
11	Event1	50.000000
21	Event2	75.000000
31	Event3	100.000000
41	Event4	60.000000
51	Event5	90.000000

```
5 rows in set (0.00 sec)
```

2)

```
mysql> SELECT
->     SUM(total_cost) AS total_revenue
-> FROM
->     booking;
```

total_revenue
845.00

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


3)

```
mysql> SELECT
->     e.event_id,
->     e.event_name,
->     e.event_date,
->     SUM(b.num_tickets) AS total_tickets_sold,
->     e.ticket_price,
->     SUM(b.total_cost) AS total_revenue
-> FROM
->     event e
-> JOIN
->     booking b ON e.event_id = b.event_id
-> GROUP BY
->     e.event_id, e.event_name, e.event_date, e.ticket_price
-> ORDER BY
->     total_revenue DESC
-> LIMIT 1;
```

event_id	event_name	event_date	total_tickets_sold	ticket_price	total_revenue
41	Event4	2023-04-10	4	60.00	240.00

1 row in set (0.00 sec)

4)

```
mysql> SELECT
->     e.event_id,
->     e.event_name,
->     e.event_date,
->     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, e.event_date;
```

event_id	event_name	event_date	total_tickets_sold
11	Event1	2023-01-01	2
21	Event2	2023-02-15	3
31	Event3	2023-03-20	1
41	Event4	2023-04-10	4
51	Event5	2023-05-05	2

5 rows in set (0.00 sec)

5)

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

6)

```
mysql> SELECT
->     c.customer_id,
->     c.customer_name,
->     c.email,
->     c.phone_number,
->     SUM(b.num_tickets) AS total_tickets_booked
-> FROM
->     customer c
-> JOIN
->     booking b ON c.customer_id = b.customer_id
-> GROUP BY
->     c.customer_id, c.customer_name, c.email, c.phone_number
-> ORDER BY
->     total_tickets_booked DESC
-> LIMIT 1;
+-----+-----+-----+-----+-----+
| customer_id | customer_name | email                | phone_number | total_tickets_booked |
+-----+-----+-----+-----+-----+
|          444 | varun         | varun4@example.com   | 789-012-3456 | 4                    |
+-----+-----+-----+-----+-----+
1 row in set (0.00 sec)
```

7)

```
mysql> SELECT
->     DATE_FORMAT(b.booking_date, '%Y-%m') AS month,
->     e.event_id,
->     e.event_name,
->     COUNT(b.num_tickets) AS total_tickets_sold
-> FROM
->     event e
-> LEFT JOIN
->     booking b ON e.event_id = b.event_id
-> GROUP BY
->     month, e.event_id, e.event_name
-> ORDER BY
->     month, e.event_id;
```

month	event_id	event_name	total_tickets_sold
2023-01	11	Event1	1
2023-02	21	Event2	1
2023-03	31	Event3	1
2023-04	41	Event4	1
2023-05	51	Event5	1

5 rows in set (0.01 sec)

8)

```
mysql> SELECT
->     v.venue_id,
->     v.venue_name,
->     AVG(e.ticket_price) AS average_ticket_price
-> FROM
->     venu 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	theatre1	50.000000
2	red stadium	75.000000
3	madison avenue	100.000000
4	Mi6	60.000000
5	openwalk	90.000000

5 rows in set (0.00 sec)

9)

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

event_type	total_tickets_sold
Movie	6
Sports	3
Concert	3

3 rows in set (0.00 sec)

10)

```
mysql> SELECT
->     YEAR(b.booking_date) AS year,
->     SUM(b.total_cost) AS total_revenue
-> FROM
->     booking b
-> JOIN
->     event e ON b.event_id = e.event_id
-> GROUP BY
->     YEAR(b.booking_date)
-> ORDER BY
->     year;
+-----+-----+
| year | total_revenue |
+-----+-----+
| 2023 |          845.00 |
+-----+-----+
1 row in set (0.01 sec)
```

11)

```
mysql> SELECT
->     c.customer_id,
->     c.customer_name,
->     COUNT(DISTINCT b.event_id) AS num_events_booked
-> FROM
->     customer c
-> JOIN
->     booking b ON c.customer_id = b.customer_id
-> GROUP BY
->     c.customer_id
-> HAVING
->     COUNT(DISTINCT b.event_id) > 1;
Empty set (0.01 sec)
```

12)

```
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
-> ORDER BY
->     total_revenue DESC;
```

customer_id	customer_name	total_revenue
444	varun	240.00
222	arun	225.00
555	leo	180.00
111	vijay	100.00
333	mike	100.00

5 rows in set (0.00 sec)

13)

```
mysql> SELECT
->     v.venue_id,
->     v.venue_name,
->     e.event_type,
->     AVG(e.ticket_price) AS average_ticket_price
-> FROM
->     venu v
-> JOIN
->     event e ON v.venue_id = e.venue_id
-> GROUP BY
->     v.venue_id, v.venue_name, e.event_type;
```

venue_id	venue_name	event_type	average_ticket_price
1	theatre1	Movie	50.000000
2	red stadium	Sports	75.000000
3	madison avenue	Concert	100.000000
4	Mi6	Movie	60.000000
5	openwalk	Concert	90.000000

5 rows in set (0.00 sec)

14)

```
mysql> SELECT
->     c.customer_id,
->     c.customer_name,
->     SUM(b.num_tickets) AS total
-> 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;
Empty set (0.01 sec)
```

Task 4

1)

```
mysql> SELECT
->     v.venue_id,
->     v.venue_name,
->     (SELECT AVG(e.ticket_price) FROM event e WHERE e.venue_id = v.venue_id) AS average_ticket_price
-> FROM
->     venue v;
```

venue_id	venue_name	average_ticket_price
1	theatre1	50.000000
2	red stadium	75.000000
3	madison avenue	100.000000
4	Mi6	60.000000
5	openwalk	90.000000

5 rows in set (0.00 sec)

2)

```
mysql> SELECT
->     e.event_id,
->     e.event_name,
->     e.event_date,
->     e.total_seats,
->     COUNT(b.booking_id) AS booked_tickets,
->     (COUNT(b.booking_id) / e.total_seats) * 100 AS percentage_sold
-> FROM
->     event e
-> JOIN
->     booking b ON e.event_id = b.event_id
-> GROUP BY
->     e.event_id, e.event_name, e.event_date, e.total_seats
-> HAVING
->     (COUNT(b.booking_id) / e.total_seats) * 100 > 50;
Empty set (0.01 sec)
```

3)

```
mysql> SELECT
->     e.event_id,
->     e.event_name,
->     (
->         SELECT SUM(b.num_tickets)
->         FROM booking b
->         WHERE b.event_id = e.event_id
->     ) AS total_tickets_sold
-> FROM
->     event e;
```

event_id	event_name	total_tickets_sold
11	Event1	2
21	Event2	3
31	Event3	1
41	Event4	4
51	Event5	2

```
5 rows in set (0.00 sec)
```

4)


```
mysql> SELECT
->     c.customer_id,
->     c.customer_name
->
-> FROM
->     customer c
-> WHERE
->     NOT EXISTS (
->         SELECT 1
->         FROM booking b
->         WHERE b.customer_id = c.customer_id
->     );
Empty set (0.01 sec)
```

5)

```
mysql> SELECT
->     event_id,
->     event_name,
->     event_date
-> FROM
->     event
-> WHERE
->     event_id NOT IN (
->         SELECT DISTINCT event_id
->         FROM booking
->     );
Empty set (0.01 sec)
```

6)

```
mysql> SELECT
->     sub.event_type,
->     SUM(sub.num_tickets) AS total_tickets_sold
-> FROM (
->     SELECT
->         e.event_type,
->         b.num_tickets
->     FROM
->         event e
->     JOIN
->         booking b ON e.event_id = b.event_id
-> ) AS sub
-> GROUP BY
->     sub.event_type;
```

event_type	total_tickets_sold
Movie	6
Sports	3
Concert	3

3 rows in set (0.00 sec)

7)

```
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
31	Event3	100.00
51	Event5	90.00

2 rows in set (0.00 sec)

8)

```
mysql> SELECT
->     c.customer_id,
->     c.customer_name,
->     (
->         SELECT SUM(b.total_cost)
->         FROM booking b
->         WHERE b.customer_id = c.customer_id
->     ) AS total_revenue
-> FROM
->     customer c;
```

customer_id	customer_name	total_revenue
111	vijay	100.00
222	arun	225.00
333	mike	100.00
444	varun	240.00
555	leo	180.00

5 rows in set (0.00 sec)

9)

```
mysql> SELECT
->     c.customer_id,
->     c.customer_name,
->     c.email,
->     c.phone_number
-> FROM
->     customer c
-> WHERE
->     EXISTS (
->         SELECT 1
->         FROM booking b
->         JOIN event e ON b.event_id = e.event_id
->         WHERE e.venue_id = (SELECT venue_id FROM venu WHERE venue_name = 'openwalk')
->         AND b.customer_id = c.customer_id
->     );
```

customer_id	customer_name	email	phone_number
555	leo	leo5@example.com	321-654-9870

1 row in set (0.01 sec)

10)

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

event_type	total_tickets_sold
Movie	6
Sports	3
Concert	3

3 rows in set (0.00 sec)

11)

```
mysql> SELECT
->     c.customer_id,
->     c.customer_name,
->     DATE_FORMAT(b.booking_date, '%Y-%m') AS booking_month
-> FROM
->     customer c
-> JOIN
->     booking b ON c.customer_id = b.customer_id
-> GROUP BY
->     c.customer_id , booking_month;
```

customer_id	customer_name	booking_month
111	vijay	2023-01
222	arun	2023-02
333	mike	2023-03
444	varun	2023-04
555	leo	2023-05

5 rows in set (0.00 sec)

12)

```
mysql> SELECT
->     v.venue_id,
->     v.venue_name,
->     (
->         SELECT AVG(e.ticket_price)
->         FROM event e
->         WHERE e.venue_id = v.venue_id
->     ) AS average_ticket_price
-> FROM
->     venu v;
```

venue_id	venue_name	average_ticket_price
1	theatre1	50.000000
2	red stadium	75.000000
3	madison avenue	100.000000
4	Mi6	60.000000
5	openwalk	90.000000

```
5 rows in set (0.00 sec)
```