Tasks 1: Database Design:

1. Create the database named "TicketBookingSystem"

CREATE DATABASE TicketBookingSystem;

USE TicketBookingSystem;

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

- **Create the Venue Table (Independent)**

```
CREATE TABLE Venue (
venue_id INT PRIMARY KEY,
venue_name VARCHAR(100) NOT NULL,
address VARCHAR(25));
```

	Field	Type	Null	Key	Default	Extra
▶	venue_id	int	NO	PRI	NULL	
	venue_name	varchar(100)	NO		NULL	
	address	varchar(255)	YES		NULL	

- **Create the Booking Table (Initially without customer_id and event_id)**

```
CREATE TABLE Booking (
booking_id INT PRIMARY KEY,
num_tickets INT NOT NULL,
total_cost DECIMAL(10, 2) NOT NULL,
booking_date DATE NOT NULL);
```

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

- **Create the Event Table (References Venue and Booking)**

```
CREATE TABLE Event (
event_id INT PRIMARY KEY,
event_name VARCHAR(100) NOT NULL,
event_date DATE NOT NULL,
event_time TIME NOT NULL,
venue_id INT,
total_seats INT NOT NULL,
available_seats INT NOT NULL,
ticket_price DECIMAL(10, 2) NOT NULL,
event_type ENUM('Movie', 'Sports', 'Concert'),
booking_id INT,
FOREIGN KEY (venue_id) REFERENCES Venue(venue_id)
ON UPDATE CASCADE
ON DELETE RESTRICT,
 FOREIGN KEY (booking_id) REFERENCES Booking(booking_id)
ON UPDATE CASCADE
ON DELETE SET NULL
  );
```

Field	Type	Null	Key	Default	Extra
event_name	varchar(100)	NO		NULL	
event_date	date	NO		NULL	
event_time	time	NO		NULL	
venue_id	int	YES	MUL	NULL	
total_seats	int	NO		NULL	
available_seats	int	NO		NULL	
ticket_price	decimal(10,2)	NO		NULL	
event_type	enum('Movie','Sports','Concert')	YES		NULL	
booking_id	int	YES	MUL	NULL	

- **Create the Customer Table (References Booking)**

```
CREATE TABLE Customer (
customer_id INT PRIMARY KEY,
customer_name VARCHAR(100) NOT NULL,
email VARCHAR(100),
phone_number VARCHAR(15),
booking_id INT,
FOREIGN KEY (booking_id) REFERENCES Booking(booking_id)
ON UPDATE CASCADE
ON DELETE SET NULL);
```

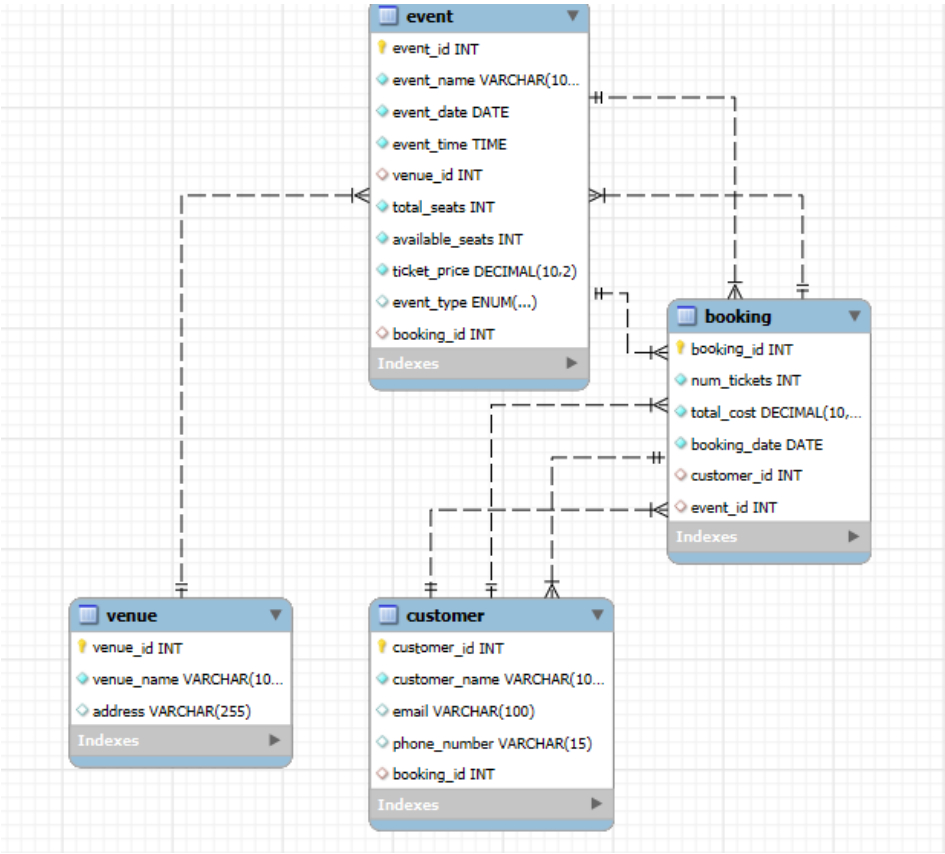
Field	Type	Null	Key	Default	Extra
customer_id	int	NO	PRI	NULL	
customer_name	varchar(100)	NO		NULL	
email	varchar(100)	YES		NULL	
phone_number	varchar(15)	YES		NULL	
booking_id	int	YES	MUL	NULL	

- **Alter the Booking Table to Add Customer and Event as Foreign Keys**

```
ALTER TABLE Booking
ADD CONSTRAINT fk_customer
FOREIGN KEY (customer_id) REFERENCES Customer(customer_id)
ON UPDATE CASCADE
ON DELETE SET NULL,
 ADD CONSTRAINT fk_event
FOREIGN KEY (event_id) REFERENCES Event(event_id)
ON UPDATE CASCADE
ON DELETE SET NULL;
```

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

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



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

Yes, the SQL provided correctly defines Primary keys and Foreign keys, ensuring referential integrity.

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

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

- **Inserting Sample Data into Venue**

INSERT INTO Venue (venue_id, venue_name, address) VALUES
(1, 'Nehru Stadium', 'Chennai'),
(2, 'Jawaharlal Stadium', 'Delhi'),
(3, 'Chinnaswamy Stadium', 'Bangalore'),
(4, 'Wankhede Stadium', 'Mumbai'),
(5, 'Eden Gardens', 'Kolkata'),
(6, 'Green Park', 'Kanpur'),
(7, 'Sardar Patel Stadium', 'Ahmedabad'),
(8, 'M.A. Chidambaram Stadium', 'Chennai'),
(9, 'DY Patil Stadium', 'Mumbai'),
(10, 'Kalinga Stadium', 'Bhubaneswar');

```
6 •    select* from venue;
```

sult Grid | Filter Rows: | Edit:

venue_id	venue_name	address
1	Nehru Stadium	Chennai
2	Jawaharlal Stadium	Delhi
3	Chinnaswamy Stadium	Bangalore
4	Wankhede Stadium	Mumbai
5	Eden Gardens	Kolkata
6	Green Park	Kanpur
7	Sardar Patel Stadium	Ahmedabad
8	M.A. Chidambaram Stadium	Chennai
9	DY Patil Stadium	Mumbai
10	Kalinga Stadium	Bhubaneswar

- **Inserting Sample Data into Event**

INSERT INTO Event (event_id, event_name, event_date, event_time, venue_id, total_seats, available_seats, ticket_price, event_type, booking_id) VALUES

(1, 'IPL Final', '2025-04-05', '18:30:00', 1, 25000, 1000, 2000.00, 'Sports', 1),

(2, 'Music Concert', '2025-04-10', '19:00:00', 2, 20000, 5000, 1500.00, 'Concert', 2),

(3, 'World Cup Match', '2025-04-15', '17:00:00', 3, 30000, 15000, 3000.00, 'Sports', 3),

(4, 'Rock Concert', '2025-04-20', '20:00:00', 4, 18000, 0, 2500.00, 'Concert', 4),

(5, 'Film Premiere', '2025-04-25', '18:00:00', 5, 15000, 2000, 1200.00, 'Movie', 5),

(6, 'Charity Cup', '2025-05-01', '16:00:00', 6, 10000, 4000, 1000.00, 'Sports', 6),

(7, 'Jazz Concert', '2025-05-05', '21:00:00', 7, 22000, 5000, 1800.00, 'Concert', 7),

(8, 'Cricket Cup', '2025-05-10', '14:00:00', 8, 28000, 10000, 2200.00, 'Sports', 8),

(9, 'Bollywood Night', '2025-05-15', '19:30:00', 9, 17000, 3000, 2000.00, 'Movie', 9),

(10, 'Athletics Meet', '2025-05-20', '10:00:00', 10, 25000, 8000, 1500.00, 'Sports', 10);

event_id	event_name	event_date	event_time	venue_id	total_seats	available_seats	ticket_price	event_type	booking_id
1	IPL Final	2025-04-05	18:30:00	1	25000	1000	2000.00	Sports	1
2	Music Concert	2025-04-10	19:00:00	2	20000	5000	1500.00	Concert	2
3	World Cup Match	2025-04-15	17:00:00	3	30000	15000	3000.00	Sports	3
4	Rock Concert	2025-04-20	20:00:00	4	18000	0	2500.00	Concert	4
5	Film Premiere	2025-04-25	18:00:00	5	15000	2000	1200.00	Movie	5
6	Charity Cup	2025-05-01	16:00:00	6	10000	4000	1000.00	Sports	6
7	Jazz Concert	2025-05-05	21:00:00	7	22000	5000	1800.00	Concert	7
8	Cricket Cup	2025-05-10	14:00:00	8	28000	10000	2200.00	Sports	8
9	Bollywood Night	2025-05-15	19:30:00	9	17000	3000	2000.00	Movie	9
10	Athletics Meet	2025-05-20	10:00:00	10	25000	8000	1500.00	Sports	10

■ Inserting Sample Data into Customer

INSERT INTO Customer (customer_id, customer_name, email, phone_number, booking_id) VALUES

(1, 'Rajesh Kumar', 'rajesh@example.com', '9876540000', 1),

(2, 'Anitha Reddy', 'anitha@example.com', '8765432100', 2),

(3, 'Manoj Das', 'manoj@example.com', '7654321000', 3),

(4, 'Suresh Iyer', 'suresh@example.com', '6543210000', 4),

(5, 'Priya Singh', 'priya@example.com', '9988770000', 5),

(6, 'Ganesh Babu', 'ganesh@example.com', '8899000000', 6),

(7, 'Deepa Nair', 'deepa@example.com', '7788990000', 7),

(8, 'Kiran Mehta', 'kiran@example.com', '6677880000', 8),

(9, 'Sakshi Jain', 'sakshi@example.com', '5566770000', 9),

(10, 'Rohit Sharma', 'rohit@example.com', '4455660000', 10);

customer_id	customer_name	email	phone_number	booking_id
1	Rajesh Kumar	rajesh@example.com	9876540000	1
2	Anitha Reddy	anitha@example.com	8765432100	2
3	Manoj Das	manoj@example.com	7654321000	3
4	Suresh Iyer	suresh@example.com	6543210000	4
5	Priya Singh	priya@example.com	9988770000	5
6	Ganesh Babu	ganesh@example.com	8899000000	6
7	Deepa Nair	deepa@example.com	7788990000	7
8	Kiran Mehta	kiran@example.com	6677880000	8
9	Sakshi Jain	sakshi@example.com	5566770000	9
10	Rohit Sharma	rohit@example.com	4455660000	10

- **Inserting Sample Data into Booking**

INSERT INTO Booking (customer_id, event_id, num_tickets, total_cost, booking_date) VALUES

(1, 1, 5, 1500.00, '2025-03-10'),

(2, 2, 2, 800.00, '2025-03-12'),

(3, 3, 10, 5000.00, '2025-03-15'),

(4, 4, 3, 1200.00, '2025-03-18'),

(5, 5, 8, 3200.00, '2025-03-20'),

(6, 6, 4, 2000.00, '2025-03-22'),

(7, 7, 7, 6, 3000.00, '2025-03-25'),

(8, 8, 8, 1, 400.00, '2025-03-27'),

(9, 9, 9, 7, 3500.00, '2025-03-29'),

(10, 10, 10, 9, 4500.00, '2025-03-31');

booking_id	num_tickets	total_cost	booking_date	customer_id	event_id
1	5	1500.00	2025-03-10	1	1
2	2	800.00	2025-03-12	2	2
3	10	5000.00	2025-03-15	3	3
4	3	1200.00	2025-03-18	4	4
5	8	3200.00	2025-03-20	5	5
6	4	2000.00	2025-03-22	6	6
7	6	3000.00	2025-03-25	7	7
8	1	400.00	2025-03-27	8	8
9	7	3500.00	2025-03-29	9	9
10	9	4500.00	2025-03-31	10	10

2.Write a SQL query to list all Events.

SELECT * FROM Event;

event_id	event_name	event_date	event_time	venue_id	total_seats	available_seats	ticket_price	event_type	booking_id
1	IPL Final	2025-04-05	18:30:00	1	25000	1000	2000.00	Sports	1
2	Music Concert	2025-04-10	19:00:00	2	20000	5000	1500.00	Concert	2
3	World Cup Match	2025-04-15	17:00:00	3	30000	15000	3000.00	Sports	3
4	Rock Concert	2025-04-20	20:00:00	4	18000	0	2500.00	Concert	4
5	Film Premiere	2025-04-25	18:00:00	5	15000	2000	1200.00	Movie	5
6	Charity Cup	2025-05-01	16:00:00	6	10000	4000	1000.00	Sports	6
7	Jazz Concert	2025-05-05	21:00:00	7	22000	5000	1800.00	Concert	7
8	Cricket Cup	2025-05-10	14:00:00	8	28000	10000	2200.00	Sports	8
9	Bollywood Night	2025-05-15	19:30:00	9	17000	3000	2000.00	Movie	9
10	Athletics Meet	2025-05-20	10:00:00	10	25000	8000	1500.00	Sports	10

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

SELECT event_name , available_seats from Event WHERE available_seats > 0;

event_name	available_seats
IPL Final	1000
Music Concert	5000
World Cup Match	15000
Film Premiere	2000
Charity Cup	4000
Jazz Concert	5000
Cricket Cup	10000
Bollywood Night	3000
Athletics Meet	8000

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

SELECT * from Event WHERE event_name LIKE '%cup%';

event_id	event_name	event_date	event_time	venue_id	total_seats	available_seats	ticket_price	event_type	booking_id
3	World Cup Match	2025-04-15	17:00:00	3	30000	15000	3000.00	Sports	3
6	Charity Cup	2025-05-01	16:00:00	6	10000	4000	1000.00	Sports	6
8	Cricket Cup	2025-05-10	14:00:00	8	28000	10000	2200.00	Sports	8

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

SELECT event_name,ticket_price FROM Event WHERE ticket_price BETWEEN 1000 AND 2500;

event_name	ticket_price
IPL Final	2000.00
Music Concert	1500.00
Rock Concert	2500.00
Film Premiere	1200.00
Charity Cup	1000.00
Jazz Concert	1800.00
Cricket Cup	2200.00
Bollywood Night	2000.00
Athletics Meet	1500.00

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

SELECT * FROM Event WHERE event_date BETWEEN '2025-04-01' AND '2025-04-30';

event_id	event_name	event_date	event_time	venue_id	total_seats	available_seats	ticket_price	event_type	booking_id
1	IPL Final	2025-04-05	18:30:00	1	25000	1000	2000.00	Sports	1
2	Music Concert	2025-04-10	19:00:00	2	20000	5000	1500.00	Concert	2
3	World Cup Match	2025-04-15	17:00:00	3	30000	15000	3000.00	Sports	3
4	Rock Concert	2025-04-20	20:00:00	4	18000	0	2500.00	Concert	4
5	Film Premiere	2025-04-25	18:00:00	5	15000	2000	1200.00	Movie	5

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

SELECT * FROM Event WHERE available_seats > 0 AND event_name LIKE '%Concert%';

event_id	event_name	event_date	event_time	venue_id	total_seats	available_seats	ticket_price	event_type	booking_id
2	Music Concert	2025-04-10	19:00:00	2	20000	5000	1500.00	Concert	2
7	Jazz Concert	2025-05-05	21:00:00	7	22000	5000	1800.00	Concert	7

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

SELECT * FROM Customer LIMIT 5 OFFSET 5;

customer_id	customer_name	email	phone_number	booking_id
6	Ganesh Babu	ganesh@example.com	8899000000	6
7	Deepa Nair	deepa@example.com	7788990000	7
8	Kiran Mehta	kiran@example.com	6677880000	8
9	Sakshi Jain	sakshi@example.com	5566770000	9
10	Rohit Sharma	rohit@example.com	4455660000	10

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

SELECT * FROM Booking WHERE num_tickets > 4;

booking_id	num_tickets	total_cost	booking_date	customer_id	event_id
1	5	1500.00	2025-03-10	1	1
3	10	5000.00	2025-03-15	3	3
5	8	3200.00	2025-03-20	5	5
7	6	3000.00	2025-03-25	7	7
9	7	3500.00	2025-03-29	9	9
10	9	4500.00	2025-03-31	10	10

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

SELECT * FROM Customer WHERE phone_number LIKE '%000';

customer_id	customer_name	email	phone_number	booking_id
1	Rajesh Kumar	rajesh@example.com	9876540000	1
3	Manoj Das	manoj@example.com	7654321000	3
4	Suresh Iyer	suresh@example.com	6543210000	4
5	Priya Singh	priya@example.com	9988770000	5
6	Ganesh Babu	ganesh@example.com	8899000000	6
7	Deepa Nair	deepa@example.com	7788990000	7
8	Kiran Mehta	kiran@example.com	6677880000	8
9	Sakshi Jain	sakshi@example.com	5566770000	9
10	Rohit Sharma	rohit@example.com	4455660000	10

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

SELECT * FROM Event WHERE total_seats > 15000 ORDER BY total_seats;

event_id	event_name	event_date	event_time	venue_id	total_seats	available_seats	ticket_price	event_type	booking_id
9	Bollywood Night	2025-05-15	19:30:00	9	17000	3000	2000.00	Movie	9
4	Rock Concert	2025-04-20	20:00:00	4	18000	0	2500.00	Concert	4
2	Music Concert	2025-04-10	19:00:00	2	20000	5000	1500.00	Concert	2
7	Jazz Concert	2025-05-05	21:00:00	7	22000	5000	1800.00	Concert	7
1	IPL Final	2025-04-05	18:30:00	1	25000	1000	2000.00	Sports	1
10	Athletics Meet	2025-05-20	10:00:00	10	25000	8000	1500.00	Sports	10
8	Cricket Cup	2025-05-10	14:00:00	8	28000	10000	2200.00	Sports	8
3	World Cup Match	2025-04-15	17:00:00	3	30000	15000	3000.00	Sports	3

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

SELECT * FROM Event

WHERE event_name NOT LIKE 'x%'

AND event_name NOT LIKE 'y%'

AND event_name NOT LIKE 'z%';

event_id	event_name	event_date	event_time	venue_id	total_seats	available_seats	ticket_price	event_type	booking_id
1	IPL Final	2025-04-05	18:30:00	1	25000	1000	2000.00	Sports	1
2	Music Concert	2025-04-10	19:00:00	2	20000	5000	1500.00	Concert	2
3	World Cup Match	2025-04-15	17:00:00	3	30000	15000	3000.00	Sports	3
4	Rock Concert	2025-04-20	20:00:00	4	18000	0	2500.00	Concert	4
5	Film Premiere	2025-04-25	18:00:00	5	15000	2000	1200.00	Movie	5
6	Charity Cup	2025-05-01	16:00:00	6	10000	4000	1000.00	Sports	6
7	Jazz Concert	2025-05-05	21:00:00	7	22000	5000	1800.00	Concert	7
8	Cricket Cup	2025-05-10	14:00:00	8	28000	10000	2200.00	Sports	8
9	Bollywood Night	2025-05-15	19:30:00	9	17000	3000	2000.00	Movie	9
10	Athletics Meet	2025-05-20	10:00:00	10	25000	8000	1500.00	Sports	10

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

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

```
SELECT event_name, AVG(ticket_price) AS average_price
FROM Event
GROUP BY event_name;
```

event_name	average_price
IPL Final	2000.000000
Music Concert	1500.000000
World Cup Match	3000.000000
Rock Concert	2500.000000
Film Premiere	1200.000000
Charity Cup	1000.000000
Jazz Concert	1800.000000
Cricket Cup	2200.000000
Bollywood Night	2000.000000
Athletics Meet	1500.000000

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

```
SELECT event_name, SUM(total_cost) AS total_revenue
FROM Booking
JOIN Event ON Booking.event_id = Event.event_id
GROUP BY event_name;
```

event_name	total_revenue
IPL Final	1500.00
Music Concert	800.00
World Cup Match	5000.00
Rock Concert	1200.00
Film Premiere	3200.00
Charity Cup	2000.00
Jazz Concert	3000.00
Cricket Cup	400.00
Bollywood Night	3500.00
Athletics Meet	4500.00

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

```
SELECT event_name, SUM(num_tickets) AS total_tickets_sold
FROM Booking
JOIN Event ON Booking.event_id = Event.event_id
GROUP BY event_name
ORDER BY total_tickets_sold DESC
LIMIT 1;
```

event_name	total_tickets_sold
World Cup Match	10

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

```
SELECT event_name, SUM(num_tickets) AS total_tickets_sold
FROM Booking
JOIN Event ON Booking.event_id = Event.event_id
GROUP BY event_name;
```

event_name	total_tickets_sold
IPL Final	5
Music Concert	2
World Cup Match	10
Rock Concert	3
Film Premiere	8
Charity Cup	4
Jazz Concert	6
Cricket Cup	1
Bollywood Night	7
Athletics Meet	9

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

```
SELECT event_name

FROM Event

WHERE event_id NOT IN (SELECT event_id FROM Booking);
```

Result Grid

	event_name

-- its empty, there is no event like that.

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

```
SELECT customer_name, SUM(num_tickets) AS total_tickets

FROM Booking

JOIN Customer ON Booking.customer_id = Customer.customer_id

GROUP BY customer_name

ORDER BY total_tickets DESC

LIMIT 1;
```

customer_name	total_tickets
Manoj Das	10

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

```
SELECT event_name, MONTH(booking_date) AS month, SUM(num_tickets) AS
total_tickets_sold

FROM Booking

JOIN Event ON Booking.event_id = Event.event_id

GROUP BY event_name, month;
```

event_name	month	total_tickets_sold
IPL Final	3	5
Music Concert	3	2
World Cup Match	3	10
Rock Concert	3	3
Film Premiere	3	8
Charity Cup	3	4
Jazz Concert	3	6
Cricket Cup	3	1
Bollywood Night	3	7
Athletics Meet	3	9

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

```
SELECT venue_name, AVG(ticket_price) AS average_ticket_price
FROM Event
JOIN Venue ON Event.venue_id = Venue.venue_id
GROUP BY venue_name;
```

venue_name	average_ticket_price
Nehru Stadium	2000.000000
Jawaharlal Stadium	1500.000000
Chinnaswamy Stadium	3000.000000
Wankhede Stadium	2500.000000
Eden Gardens	1200.000000
Green Park	1000.000000
Sardar Patel Stadium	1800.000000
M.A. Chidambaram Stadium	2200.000000
DY Patil Stadium	2000.000000
Kalinga Stadium	1500.000000

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

```
SELECT event_type, SUM(num_tickets) AS total_tickets_sold
FROM Event
JOIN Booking ON Event.event_id = Booking.event_id
GROUP BY event_type;
```

event_type	total_tickets_sold
Sports	29
Concert	11
Movie	15

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

```
SELECT YEAR(booking_date) AS year, SUM(total_cost) AS total_revenue
FROM Booking
GROUP BY year;
```

year	total_revenue
2025	25100.00

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

```
SELECT customer_name, COUNT(DISTINCT event_id) AS event_count
FROM Booking
JOIN Customer ON Booking.customer_id = Customer.customer_id
GROUP BY customer_name
HAVING event_count > 1;
```

customer_name	event_count

-- there is no customer in this case.

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

```
SELECT customer_name, SUM(total_cost) AS total_revenue
FROM Booking
JOIN Customer ON Booking.customer_id = Customer.customer_id
GROUP BY customer_name;
```

customer_name	total_revenue
Rajesh Kumar	1500.00
Anitha Reddy	800.00
Manoj Das	5000.00
Suresh Iyer	1200.00
Priya Singh	3200.00
Ganesh Babu	2000.00
Deepa Nair	3000.00
Kiran Mehta	400.00
Sakshi Jain	3500.00
Rohit Sharma	4500.00

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

```
SELECT event_type, venue_name, AVG(ticket_price) AS average_ticket_price

FROM Event

JOIN Venue ON Event.venue_id = Venue.venue_id

GROUP BY event_type, venue_name;
```

event_type	venue_name	average_ticket_price
Sports	Nehru Stadium	2000.000000
Concert	Jawaharlal Stadium	1500.000000
Sports	Chinnaswamy Stadium	3000.000000
Concert	Wankhede Stadium	2500.000000
Movie	Eden Gardens	1200.000000
Sports	Green Park	1000.000000
Concert	Sardar Patel Stadium	1800.000000
Sports	M.A. Chidambaram Stadium	2200.000000
Movie	DY Patil Stadium	2000.000000
Sports	Kalinga Stadium	1500.000000

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

Days.

```
SELECT customer_name, SUM(num_tickets) AS total_tickets_purchased

FROM Booking

JOIN Customer ON Booking.customer_id = Customer.customer_id

WHERE booking_date >= CURDATE() - INTERVAL 30 DAY

GROUP BY customer_name;
```

customer_name	total_tickets_purchased
Rajesh Kumar	5
Anitha Reddy	2
Manoj Das	10
Suresh Iyer	3
Priya Singh	8
Ganesh Babu	4
Deepa Nair	6
Kiran Mehta	1
Sakshi Jain	7
Rohit Sharma	9

Tasks 4: Subquery and its types

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

```
SELECT venue_name,
      (SELECT AVG(ticket_price)
       FROM Event
       WHERE Venue.venue_id = Event.venue_id) AS average_ticket_price
FROM Venue;
```

venue_name	average_ticket_price
Nehru Stadium	2000.000000
Jawaharlal Stadium	1500.000000
Chinnaswamy Stadium	3000.000000
Wankhede Stadium	2500.000000
Eden Gardens	1200.000000
Green Park	1000.000000
Sardar Patel Stadium	1800.000000
M.A. Chidambaram Stadium	2200.000000
DY Patil Stadium	2000.000000
Kalinga Stadium	1500.000000

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

```
SELECT event_name
FROM Event
WHERE available_seats < (total_seats / 2);
```

event_name
IPL Final
Music Concert
Rock Concert
Film Premiere
Charity Cup
Jazz Concert
Cricket Cup
Bollywood Night
Athletics Meet

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

```
SELECT event_name,
      (SELECT SUM(num_tickets)
       FROM Booking
       WHERE Event.event_id = Booking.event_id) AS total_tickets_sold
FROM Event;
```

event_name	total_tickets_sold
IPL Final	5
Music Concert	2
World Cup Match	10
Rock Concert	3
Film Premiere	8
Charity Cup	4
Jazz Concert	6
Cricket Cup	1
Bollywood Night	7
Athletics Meet	9

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

```
SELECT customer_name
FROM Customer
WHERE NOT EXISTS (
   SELECT 1
   FROM Booking
   WHERE Customer.customer_id = Booking.customer_id
);
```

customer_name

-- there is no customer, in such case.

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

```
SELECT event_name
FROM Event
WHERE event_id NOT IN (
   SELECT event_id
   FROM Booking
);
```

Result Grid

	event_name

-- there is no event, in such condition.

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

Clause.

```
SELECT event_type, total_tickets_sold

FROM (
```

```
    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
) AS subquery;
```

event_type	total_tickets_sold
Sports	29
Concert	11
Movie	15

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

```
SELECT event_name, ticket_price
FROM Event
WHERE ticket_price > (SELECT AVG(ticket_price) FROM Event);
```

event_name	ticket_price
IPL Final	2000.00
World Cup Match	3000.00
Rock Concert	2500.00
Cricket Cup	2200.00
Bollywood Night	2000.00

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

```
SELECT customer_name,
    (SELECT SUM(total_cost)
     FROM Booking
     WHERE Customer.customer_id = Booking.customer_id) AS total_revenue
FROM Customer;
```

customer_name	total_revenue
Rajesh Kumar	1500.00
Anitha Reddy	800.00
Manoj Das	5000.00
Suresh Iyer	1200.00
Priya Singh	3200.00
Ganesh Babu	2000.00
Deepa Nair	3000.00
Kiran Mehta	400.00
Sakshi Jain	3500.00
Rohit Sharma	4500.00

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

```
SELECT customer_name
FROM Customer
WHERE customer_id IN (
   SELECT customer_id
   FROM Booking
   WHERE event_id IN (
    SELECT event_id
    FROM Event
     WHERE venue_id = 1  -- Change '1' to your desired venue_id
   )
);
```

customer_name
Rajesh Kumar

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

```
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
Sports	29
Concert	11
Movie	15

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

```
SELECT customer_name
FROM Customer
WHERE customer_id IN (
   SELECT customer_id
   FROM Booking
   WHERE DATE_FORMAT(booking_date, '%Y-%m') = '2025-04'
);
```

customer_name

-- there is no customer in this case.

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

```
SELECT venue_name,
      (SELECT AVG(ticket_price)
       FROM Event
       WHERE Venue.venue_id = Event.venue_id) AS average_ticket_price
        FROM Venue;
```

venue_name	average_ticket_price
Nehru Stadium	2000.000000
Jawaharlal Stadium	1500.000000
Chinnaswamy Stadium	3000.000000
Wankhede Stadium	2500.000000
Eden Gardens	1200.000000
Green Park	1000.000000
Sardar Patel Stadium	1800.000000
M.A. Chidambaram Stadium	2200.000000
DY Patil Stadium	2000.000000
Kalinga Stadium	1500.000000