Task 1: Database Design

-- 1. Create the database named "TicketBookingSystem"

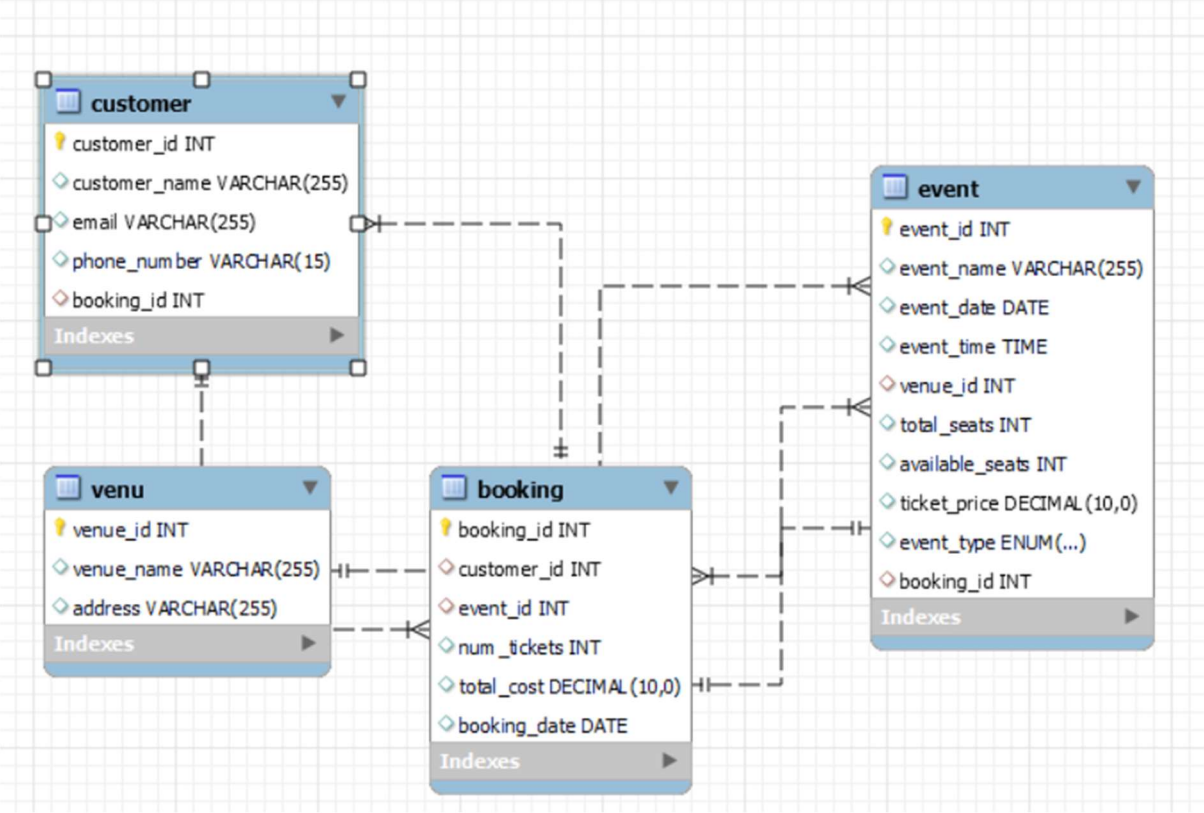
```
CREATE DATABASE TicketBookingSystem;
```

-- 2. Create tables with appropriate data types, constraints, and relationships

```
-- Venu Table
CREATE TABLE Venu (
    venue_id INT PRIMARY KEY,
    venue_name VARCHAR(255),
    address VARCHAR(255)
);

-- Event Table
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,
    event_type ENUM('Movie', 'Sports', 'Concert'),
    booking_id INT,
    FOREIGN KEY (venue_id) REFERENCES Venu(venue_id)

);
```

```
-- Customer Table
CREATE TABLE Customer (
    customer_id INT PRIMARY KEY,
    customer_name VARCHAR(255),
    email VARCHAR(255),
    phone_number VARCHAR(15),
    booking_id INT

);

-- Booking Table
CREATE TABLE Booking (
    booking_id INT PRIMARY KEY,
    customer_id INT,
    event_id INT,
    num_tickets INT,
    total_cost DECIMAL,
    booking_date DATE,
    FOREIGN KEY (customer_id) REFERENCES Customer(customer_id),
    FOREIGN KEY (event_id) REFERENCES Event(event_id)
);

ALTER TABLE Event
ADD CONSTRAINT fk_event_booking
FOREIGN KEY (booking_id) REFERENCES Booking(booking_id);


ALTER TABLE Customer
ADD CONSTRAINT fk_customer_booking
FOREIGN KEY (booking_id) REFERENCES Booking(booking_id);
```

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



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

--Constraints are already specified in the table definitions.

Task 2:

-- 1. Insert at least 10 sample records into each table

-- Insert sample data into Venu Table

INSERT INTO Venu (venue_id, venue_name, address) VALUES

(1, 'Raj Mahal', '12A MG Road, Mumbai, Maharashtra'),

(2, 'Taj Gardens', '34 Nehru Street, Delhi'),

(3, 'Lotus Convention Center', '56 Vivekananda Nagar, Bangalore, Karnataka'),

(4, 'Maratha Palace', '78 Shivaji Marg, Pune, Maharashtra'),

(5, 'Golden Sands Arena', '90 Beach Road, Goa'),

(6, 'Mysore Grand Hall', '45 Chamundi Hills, Mysuru, Karnataka'),

(7, 'Jaipur Pavilion', '23 Hawa Mahal Road, Jaipur, Rajasthan'),

(8, 'Chennai Convention Center', '67 Mount Road, Chennai, Tamil Nadu'),

(9, 'Kolkata Heights', '89 Park Street, Kolkata, West Bengal'),

(10, 'Hyderabad Hub', '78 Charminar Road, Hyderabad, Telangana');

-- Insert sample data into Event Table

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, 'Bollywood Extravaganza', '2023-12-15', '18:00:00', 1, 200, 150, 1500.00, 'Movie', 1),

(2, 'Cricket Championship', '2023-12-20', '19:30:00', 2, 5000, 4000, 250.00, 'Sports', 2),

(3, 'Classical Concert', '2023-12-25', '20:00:00', 3, 10000, 8000, 500.00, 'Concert', 3),

(4, 'Tech Summit', '2023-12-18', '09:00:00', 4, 500, 300, 10000.00, 'Conference', 4),

(5, 'Stand-up Comedy Night', '2023-12-22', '21:00:00', 5, 300, 250, 200.00, 'Comedy', 5),

(6, 'Cultural Fest', '2023-12-28', '17:00:00', 6, 1000, 800, 300.00, 'Festival', 6),

(7, 'Rajasthani Folk Evening', '2023-12-10', '19:30:00', 7, 800, 600, 400.00, 'Cultural', 7),

(8, 'Tech Workshop', '2023-12-15', '14:00:00', 8, 200, 150, 1200.00, 'Workshop', 8),

(9, 'Durga Puja Celebration', '2023-12-20', '18:30:00', 9, 1200, 1000, 300.00, 'Festival', 9),

(10, 'Hyderabadi Biryani Cook-off', '2023-12-22', '13:00:00', 10, 150, 120, 250.00, 'Culinary', 10);

-- Insert sample data into Customer Table

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

(1, 'Rahul Sharma', 'rahul.sharma@email.com', '9876543210', 1),

(2, 'Priya Patel', 'priya.patel@email.com', '8765432109', 2),

(3, 'Amit Singh', 'amit.singh@email.com', '7654321098', 3),

(4, 'Sneha Gupta', 'sneha.gupta@email.com', '6543210987', 4),

(5, 'Kunal Verma', 'kunal.verma@email.com', '5432109876', 5),

(6, 'Neha Kapoor', 'neha.kapoor@email.com', '4321098765', 6),

(7, 'Rajat Verma', 'rajat.verma@email.com', '3210987654', 7),

(8, 'Anika Das', 'anika.das@email.com', '2109876543', 8),

(9, 'Vikram Joshi', 'vikram.joshi@email.com', '1098765432', 9),

(10, 'Aishwarya Singh', 'aishwarya.singh@email.com', '0987654321', 10);

-- Insert sample data into Booking Table

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

(1, 1, 1, 3, 4500.00, '2023-12-10'),

(2, 2, 2, 5, 1250.00, '2023-12-12'),

(3, 3, 3, 2, 1000.00, '2023-12-14'),

(4, 4, 4, 2, 20000.00, '2023-12-16'),

(5, 5, 5, 4, 800.00, '2023-12-19'),

(6, 6, 6, 3, 900.00, '2023-12-22'),

(7, 7, 7, 2, 800.00, '2023-12-24'),

(8, 8, 8, 1, 1200.00, '2023-12-28'),

(9, 9, 9, 5, 1500.00, '2023-12-30'),

(10, 10, 10, 3, 750.00, '2023-12-31');

-- 2. 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	Bollywood Extravaganza	2023-12-15	18:00:00	1	200	150	1500	Movie	1
2	Cricket Championship	2023-12-20	19:30:00	2	5000	4000	250	Sports	2
3	Classical Concert	2023-12-25	20:00:00	3	10000	8000	500	Concert	3
4	Tech Summit	2023-12-18	09:00:00	4	500	300	10000	Movie	4
5	Stand-up Comedy Night	2023-12-22	21:00:00	5	300	250	200	Sports	5
6	Cultural Fest	2023-12-28	17:00:00	6	1000	800	300	Concert	6
7	Rajasthani Folk Evening	2023-12-10	19:30:00	7	800	600	400	Movie	7
8	Tech Workshop	2023-12-15	14:00:00	8	200	150	1200	Sports	8
9	Durga Puja Celebration	2023-12-20	18:30:00	9	1200	1000	300	Concert	9
10	Hyderabadi Biryani Cook-off	2023-12-22	13:00:00	10	150	120	250	Concert	10
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

-- 3. Select events with available tickets

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	Bollywood Extravaganza	2023-12-15	18:00:00	1	200	150	1500	Movie	1
2	Cricket Championship	2023-12-20	19:30:00	2	5000	4000	250	Sports	2
3	Classical Concert	2023-12-25	20:00:00	3	10000	8000	500	Concert	3
4	Tech Summit	2023-12-18	09:00:00	4	500	300	10000	Movie	4
5	Stand-up Comedy Night	2023-12-22	21:00:00	5	300	250	200	Sports	5
6	Cultural Fest	2023-12-28	17:00:00	6	1000	800	300	Concert	6
7	Rajasthani Folk Evening	2023-12-10	19:30:00	7	800	600	400	Movie	7
8	Tech Workshop	2023-12-15	14:00:00	8	200	150	1200	Sports	8
9	Durga Puja Celebration	2023-12-20	18:30:00	9	1200	1000	300	Concert	9
10	Hyderabadi Biryani Cook-off	2023-12-22	13:00:00	10	150	120	250	Concert	10
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

-- 4. Select events with name partial match 'cup'

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

-- 5. Select events with ticket price between 1000 and 2500

SELECT * FROM Event WHERE ticket_price BETWEEN 1000 AND 2500;

event_id	event_name	event_date	event_time	venue_id	total_seats	available_seats	ticket_price	event_type	booking_id
1	Bollywood Extravaganza	2023-12-15	18:00:00	1	200	150	1500	Movie	1
8	Tech Workshop	2023-12-15	14:00:00	8	200	150	1200	Sports	8
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

-- 6. Retrieve events with dates falling within a specific range

SELECT * FROM Event WHERE event_date BETWEEN '2023-12-15' AND '2023-12-20';

event_id	event_name	event_date	event_time	venue_id	total_seats	available_seats	ticket_price	event_type	booking_id
1	Bollywood Extravaganza	2023-12-15	18:00:00	1	200	150	1500	Movie	1
2	Cricket Championship	2023-12-20	19:30:00	2	5000	4000	250	Sports	2
4	Tech Summit	2023-12-18	09:00:00	4	500	300	10000	Movie	4
8	Tech Workshop	2023-12-15	14:00:00	8	200	150	1200	Sports	8
9	Durga Puja Celebration	2023-12-20	18:30:00	9	1200	1000	300	Concert	9
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

-- 7. Retrieve events with available tickets that also have "Concert" in their name

SELECT * FROM Event WHERE available_seats > 0 AND event_type = 'Concert';

event_id	event_name	event_date	event_time	venue_id	total_seats	available_seats	ticket_price	event_type	booking_id
3	Classical Concert	2023-12-25	20:00:00	3	10000	8000	500	Concert	3
6	Cultural Fest	2023-12-28	17:00:00	6	1000	800	300	Concert	6
9	Durga Puja Celebration	2023-12-20	18:30:00	9	1200	1000	300	Concert	9
10	Hyderabadi Biryani Cook-off	2023-12-22	13:00:00	10	150	120	250	Concert	10
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

-- 8. 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	Neha Gupta	neha.gupta@example.com	5432109876	6
7	Rajesh Kumar	rajesh.kumar@example.com	4321098765	7
8	Anjali Singh	anjali.singh@example.com	3210987654	8
9	Suresh Menon	suresh.menon@example.com	2109876543	9
10	Pooja Verma	pooja.verma@example.com	1098765432	10
NULL	NULL	NULL	NULL	NULL

-- 9. Retrieve bookings details with booked no. of tickets more than 4

SELECT * FROM Booking WHERE num_tickets > 4;

booking_id	customer_id	event_id	num_tickets	total_cost	booking_date
2	2	2	5	1250	2023-12-08
9	9	9	5	1500	2023-12-22
NULL	NULL	NULL	NULL	NULL	NULL

-- 10. Retrieve customer information whose phone number ends with '000'

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

-- 11. Retrieve events in order whose seat capacity is more than 15000

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

-- 12. Select events name not starting 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	Bollywood Extravaganza	2023-12-15	18:00:00	1	200	150	1500	Movie	1
2	Cricket Championship	2023-12-20	19:30:00	2	5000	4000	250	Sports	2
3	Classical Concert	2023-12-25	20:00:00	3	10000	8000	500	Concert	3
4	Tech Summit	2023-12-18	09:00:00	4	500	300	10000	Movie	4
5	Stand-up Comedy Night	2023-12-22	21:00:00	5	300	250	200	Sports	5
6	Cultural Fest	2023-12-28	17:00:00	6	1000	800	300	Concert	6
7	Rajasthani Folk Evening	2023-12-10	19:30:00	7	800	600	400	Movie	7
8	Tech Workshop	2023-12-15	14:00:00	8	200	150	1200	Sports	8
9	Durga Puja Celebration	2023-12-20	18:30:00	9	1200	1000	300	Concert	9
10	Hyderabadi Biryani Cook-off	2023-12-22	13:00:00	10	150	120	250	Concert	10
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Task 3:

-- 1. List Events and Their Average Ticket Prices

SELECT event_id, event_name, AVG(ticket_price) AS average_ticket_price FROM Event GROUP BY event_id;

event_id	event_name	average_ticket_price
1	Bollywood Extravaganza	1500.0000
2	Cricket Championship	250.0000
3	Classical Concert	500.0000
4	Tech Summit	10000.0000
5	Stand-up Comedy Night	200.0000
6	Cultural Fest	300.0000
7	Rajasthani Folk Evening	400.0000
8	Tech Workshop	1200.0000
9	Durga Puja Celebration	300.0000
10	Hyderabadi Biryani Cook-off	250.0000

-- 2. Calculate the Total Revenue Generated by Events

SELECT SUM(total_cost) AS total_revenue FROM Booking;

total_revenue
33700

-- 3. Find the event with the highest ticket sales

SELECT e.event_id, e.event_name, SUM(b.num_tickets) AS total_ticket_sales

FROM Event e

JOIN Booking b ON e.event_id = b.event_id

GROUP BY e.event_id, e.event_name

ORDER BY total_ticket_sales DESC

LIMIT 1;

event_id	event_name	total_ticket_sales
2	Cricket Championship	5

-- 4. Calculate the Total Number of Tickets Sold for Each Event

SELECT e.event_id, e.event_name, SUM(b.num_tickets) AS total_tickets_sold

FROM Event e

JOIN Booking b ON e.event_id = b.event_id

GROUP BY e.event_id, e.event_name;

event_id	event_name	total_tickets_sold
1	Bollywood Extravaganza	2
2	Cricket Championship	5
3	Classical Concert	3
4	Tech Summit	2
5	Stand-up Comedy Night	4
6	Cultural Fest	3
7	Rajasthani Folk Evening	4
8	Tech Workshop	2
9	Durga Puja Celebration	5
10	Hyderabadi Biryani Cook-off	3

-- 5. Find Events with No Ticket Sales

SELECT Event.event_id, Event.event_name

FROM Event

LEFT JOIN Booking ON Event.event_id = Booking.event_id

WHERE Booking.event_id IS NULL

LIMIT 0, 1000;

-- 6. Find the User Who Has Booked the Most Tickets

SELECT c.customer_id, c.customer_name, SUM(b.num_tickets) AS total_tickets_booked

FROM Customer c

JOIN Booking b ON c.customer_id = b.customer_id

GROUP BY c.customer_id, c.customer_name

ORDER BY total_tickets_booked DESC

LIMIT 1;1;

customer_id	customer_name	total_tickets_booked
2	Jane Smith	5

-- 7. List Events and the total number of tickets sold for each month

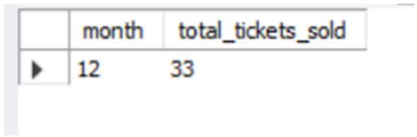
SELECT MONTH(e.event_date) AS month, SUM(b.num_tickets) AS total_tickets_sold

FROM Booking b

JOIN Event e ON b.event_id = e.event_id

GROUP BY month

LIMIT 0, 1000;



	month	total_tickets_sold
▶	12	33

-- 8. Calculate the average Ticket Price for Events in Each Venue

SELECT venue_id, AVG(ticket_price) AS average_ticket_price FROM Event GROUP BY venue_id;

	venue_id	average_ticket_price
▶	1	1500.0000
	2	250.0000
	3	500.0000
	4	10000.0000
	5	200.0000
	6	300.0000
	7	400.0000
	8	1200.0000
	9	300.0000
	10	250.0000

-- 9. Calculate the total Number of Tickets Sold for Each Event Type

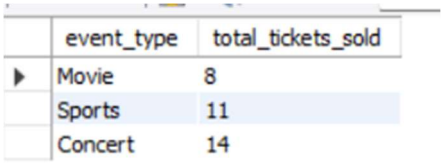
SELECT e.event_type, SUM(b.num_tickets) AS total_tickets_sold

FROM Event e

JOIN Booking b ON e.event_id = b.event_id

GROUP BY e.event_type

LIMIT 0, 1000;



	event_type	total_tickets_sold
▶	Movie	8
	Sports	11
	Concert	14

-- 10. Calculate the total Revenue Generated by Events in Each Year

SELECT YEAR(e.event_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

LIMIT 0, 1000;

	year	total_revenue
▶	2023	33700

-- 11. List users who have booked tickets for multiple events

```
SELECT Customer.customer_id, Customer.customer_name
FROM Customer
JOIN Booking ON Customer.customer_id = Booking.customer_id
GROUP BY Customer.customer_id
HAVING COUNT(DISTINCT Booking.event_id) > 1
LIMIT 0, 1000;
```

-- 12. Calculate the Total Revenue Generated by Events for Each User

```
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
LIMIT 0, 1000;
```

	customer_id	customer_name	total_revenue
▶	1	John Doe	3000
	2	Jane Smith	1250
	3	Amit Patel	1500
	4	Priya Sharma	20000
	5	Michael Brown	800
	6	Neha Gupta	900
	7	Rajesh Kumar	1600
	8	Anjali Singh	2400
	9	Suresh Menon	1500
	10	Pooja Verma	750

-- 13. Calculate the Average Ticket Price for Events in Each Category and Venue

```
SELECT venue_id, event_type, AVG(ticket_price) AS average_ticket_price FROM Event GROUP BY
venue_id, event_type;
```

venue_id	event_type	average_ticket_price
1	Movie	1500.0000
2	Sports	250.0000
3	Concert	500.0000
4	Movie	10000.0000
5	Sports	200.0000
6	Concert	300.0000
7	Movie	400.0000
8	Sports	1200.0000
9	Concert	300.0000
10	Concert	250.0000

-- 14. List Users and the Total Number of Tickets They've Purchased in the Last 30 Days

```
SELECT c.customer_id, c.customer_name, SUM(b.num_tickets) AS total_tickets_purchased
FROM Customer c
JOIN Booking b ON c.customer_id = b.customer_id
WHERE b.booking_date >= CURDATE() - INTERVAL 30 DAY
GROUP BY c.customer_id
LIMIT 0, 1000;
```

customer_id	customer_name	total_tickets_purchased
1	John Doe	2
2	Jane Smith	5
3	Amit Patel	3
4	Priya Sharma	2
5	Michael Brown	4
6	Neha Gupta	3
7	Rajesh Kumar	4
8	Anjali Singh	2
9	Suresh Menon	5
10	Pooja Verma	3

Task 4

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

```
SELECT venue_id, AVG(ticket_price) AS average_ticket_price FROM Event GROUP BY venue_id;
```

venue_id	average_ticket_price
1	1500.0000
2	250.0000
3	500.0000
4	10000.0000
5	200.0000
6	300.0000
7	400.0000
8	1200.0000
9	300.0000
10	250.0000

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

SELECT event_id, event_name FROM Event WHERE (SELECT SUM(num_tickets) FROM Booking WHERE Booking.event_id = Event.event_id) > 0.5 * total_seats;

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

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

event_id	event_name	total_tickets_sold
1	Bollywood Extravaganza	2
2	Cricket Championship	5
3	Classical Concert	3
4	Tech Summit	2
5	Stand-up Comedy Night	4
6	Cultural Fest	3
7	Rajasthani Folk Evening	4
8	Tech Workshop	2
9	Durga Puja Celebration	5
10	Hyderabadi Biryani Cook-off	3

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

SELECT customer_id, customer_name FROM Customer WHERE NOT EXISTS (SELECT * FROM Booking WHERE Booking.customer_id = Customer.customer_id);

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

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

-- 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
    LEFT JOIN Booking B ON E.event_id = B.event_id
    GROUP BY E.event_type
) AS EventTypeTotals;
```

event_type	total_tickets_sold
Movie	8
Sports	11
Concert	14

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

```
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	Bollywood Extravaganza	1500
4	Tech Summit	10000
NULL	NULL	NULL

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

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

customer_id	customer_name	total_revenue
1	John Doe	3000
2	Jane Smith	1250
3	Amit Patel	1500
4	Priya Sharma	20000
5	Michael Brown	800
6	Neha Gupta	900
7	Rajesh Kumar	1600
8	Anjali Singh	2400
9	Suresh Menon	1500
10	Pooja Verma	750

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

```
SELECT customer_id, customer_name
FROM Customer
WHERE EXISTS (
    SELECT 1
    FROM Booking
    JOIN Event ON Booking.event_id = Event.event_id
    WHERE Event.venue_id = 1
    AND Booking.customer_id = Customer.customer_id
);
```

customer_id	customer_name
1	John Doe
NULL	NULL

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

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

event_type	total_tickets_sold
Movie	8
Sports	11
Concert	14

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

```
SELECT customer_id, customer_name, (
    SELECT GROUP_CONCAT(DISTINCT DATE_FORMAT(booking_date, '%M') ORDER BY booking_date)
    FROM Booking
    WHERE Booking.customer_id = Customer.customer_id
) AS booked_months
FROM Customer;
```

customer_id	customer_name	booked_months
1	John Doe	December
2	Jane Smith	December
3	Amit Patel	December
4	Priya Sharma	December
5	Michael Brown	December
6	Neha Gupta	December
7	Rajesh Kumar	December
8	Anjali Singh	December
9	Suresh Menon	December
10	Pooja Verma	December

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

```
SELECT venue_id, AVG(ticket_price) AS average_ticket_price
FROM Event
GROUP BY venue_id;
```

venue_id	average_ticket_price
1	1500.0000
2	250.0000
3	500.0000
4	10000.0000
5	200.0000
6	300.0000
7	400.0000
8	1200.0000
9	300.0000
10	250.0000