NAME: ASENATH A

ASSIGNMENT

TICKET BOOKING SYSTEM

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.

- **Venu**
- **Event**
- **Customers**
- **Booking**

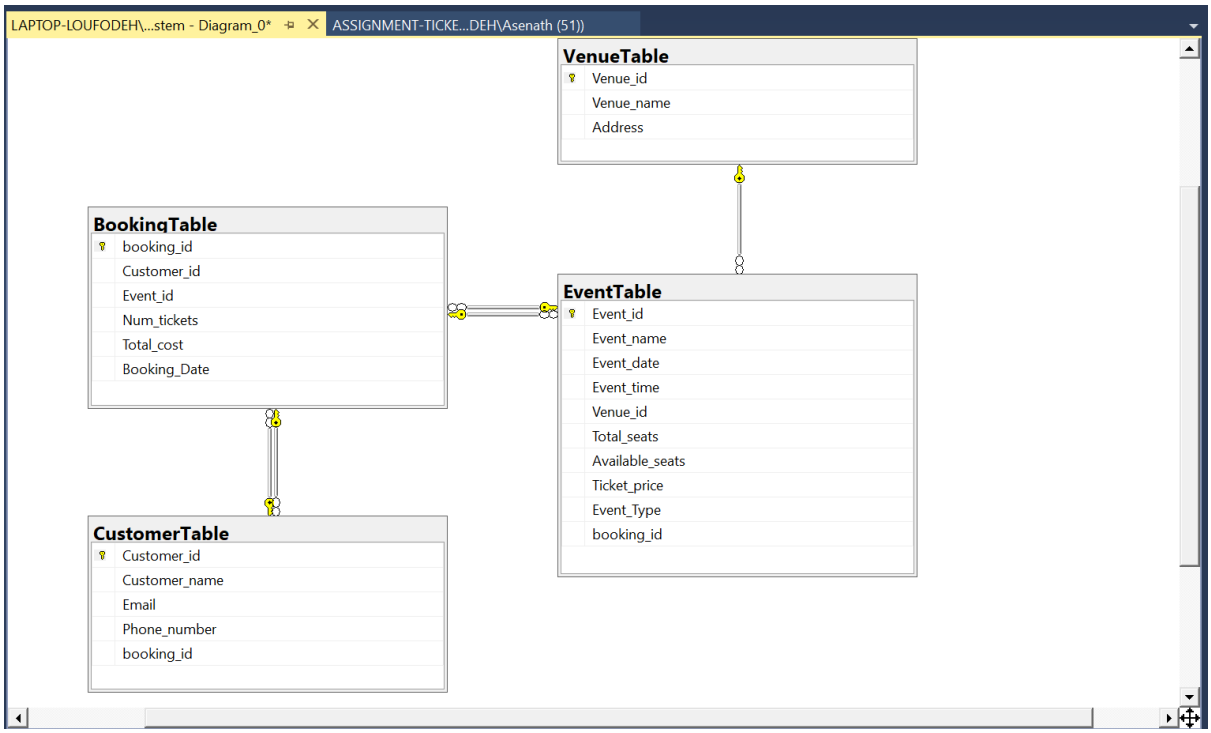
```
CREATE TABLE VenueTable(
Venue_id INT PRIMARY KEY,
Venue_name VARCHAR(255),
Address Text);

CREATE TABLE EventTable(
Event_id INT PRIMARY KEY,
Event_name VARCHAR(255),
Event_date DATE,
Event_time TIME,
Venue_id INT,
FOREIGN KEY (Venue_id) REFERENCES VenueTable(Venue_id),
Total_seats INT,
Available_seats INT,
Ticket_price DECIMAL,
Event_Type VARCHAR(30),
booking_id INT,
);

CREATE TABLE CustomerTable(
Customer_id INT PRIMARY KEY,
Customer_name VARCHAR(30),
Email VARCHAR(60),
Phone_number INT,
booking_id INT,
);

CREATE TABLE BookingTable(
booking_id INT PRIMARY KEY,
Customer_id INT,
FOREIGN KEY (Customer_id) REFERENCES CustomerTable(Customer_id),
Event_id INT,
FOREIGN KEY (Event_id) REFERENCES EventTable(Event_id),
Num_tickets INT,
Total_cost INT,
Booking_Date DATE);
```

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



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

```
ALTER TABLE EventTable
ADD FOREIGN KEY (booking_id) REFERENCES BookingTable(booking_id);

ALTER TABLE CustomerTable
ADD FOREIGN KEY (booking_id) REFERENCES BookingTable(booking_id);
```

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

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

```
INSERT INTO VenueTable(Venue_id, Venue_name, Address)
VALUES
(1, 'Concert Hall A', '123 Main Street, Cityville'),
(2, 'Sports Arena B', '456 Oak Avenue, Townsville'),
(3, 'Exhibition Center C', '789 Pine Road, Villagetown'),
(4, 'Conference Hall D', '101 Elm Lane, Hamletville'),
(5, 'Stadium E', '222 Maple Street, Countryside'),
(6, 'Theater F', '333 Birch Avenue, Suburbia'),
(7, 'Club G', '444 Cedar Road, Metropolis'),
(8, 'Amphitheater H', '555 Redwood Lane, Downtown'),
(9, 'Auditorium I', '666 Spruce Avenue, Uptown'),
(10, 'Convention Center J', '777 Oak Lane, Riverside');

INSERT INTO EventTable(Event_id, Event_name, Event_date, Event_time, Venue_id,
Total_seats, Available_seats, Ticket_price, Event_Type, booking_id)
VALUES
(101, 'Music Concert', '2024-03-15', '18:30:00', 1, 500, 500, 50.00, 'Concert', 1),
(102, 'Basketball Game', '2024-03-20', '20:00:00', 2, 1000, 800, 20.00, 'Sports', 2),
(103, 'Art Exhibition', '2024-03-25', '10:00:00', 3, 200, 200, 10.00, 'Exhibition',
3),
```

```
(104, 'Tech Conference', '2024-04-05', '09:00:00', 4, 300, 250, 30.00, 'Conference',
4),
(105, 'Football Match', '2024-04-10', '15:30:00', 5, 800, 600, 25.00, 'Sports', 5),
(106, 'Drama Play', '2024-04-15', '19:00:00', 6, 150, 150, 15.00, 'Theater', 6),
(107, 'DJ Night', '2024-04-20', '22:00:00', 7, 300, 300, 15.00, 'Club', 7),
(108, 'Outdoor Concert', '2024-04-25', '17:00:00', 8, 1000, 800, 40.00, 'Concert', 8),
(109, 'Comedy Show', '2024-05-02', '20:30:00', 9, 200, 200, 20.00, 'Theater', 9),
(110, 'Business Expo', '2024-05-10', '09:00:00', 10, 500, 450, 25.00, 'Exhibition',
10);

ALTER TABLE CustomerTable
ALTER COLUMN Phone_number BIGINT;

INSERT INTO CustomerTable(Customer_id, Customer_name, Email, Phone_number, booking_id)
VALUES
(1001, 'John Doe', 'john.doe@example.com', 1234567890, 1),
(1002, 'Jane Smith', 'jane.smith@example.com', 9876543000, 2),
(1003, 'Bob Johnson', 'bob.johnson@example.com', 5551112233, 3),
(1004, 'Alice Brown', 'alice.brown@example.com', 4445556000, 4),
(1005, 'Charlie Davis', 'charlie.davis@example.com', 3338889999, 5),
(1006, 'Eva White', 'eva.white@example.com', 1112223344, 6),
(1007, 'Sam Miller', 'sam.miller@example.com', 6667778888, 7),
(1008, 'Olivia Wilson', 'olivia.wilson@example.com', 9998887776, 8),
(1009, 'David Lee', 'david.lee@example.com', 2223334455, 9),
(1010, 'Sophia Chen', 'sophia.chen@example.com', 7776665555, 10);

INSERT INTO BookingTable(booking_id, Customer_id, Event_id, Num_tickets, Total_cost,
Booking_Date)
VALUES
(1, 1001, 101, 2, 100.00, '2024-03-10'),
(2, 1002, 102, 4, 80.00, '2024-03-12'),
(3, 1003, 103, 1, 10.00, '2024-03-15'),
(4, 1004, 104, 3, 90.00, '2024-03-20'),
(5, 1005, 105, 5, 125.00, '2024-03-25'),
(6, 1006, 106, 2, 30.00, '2024-03-28'),
(7, 1007, 107, 2, 30.00, '2024-04-02'),
(8, 1008, 108, 4, 160.00, '2024-04-05'),
(9, 1009, 109, 1, 20.00, '2024-04-10'),
(10, 1010, 110, 3, 75.00, '2024-04-15');
```

2. Write a SQL query to list all Events.

```
SELECT * FROM EventTable;
```

Results | Messages

	Event_id	Event_name	Event_date	Event_time	Venue_id	Total_seats	Available_seats	Ticket_price	Event_Type	booking_id
1	101	Music Concert	2024-03-15	18:30:00.0000000	1	500	500	50	Concert	1
2	102	Basketball Game	2024-03-20	20:00:00.0000000	2	1000	800	20	Sports	2
3	103	Art Exhibition	2024-03-25	10:00:00.0000000	3	200	200	10	Exhibition	3
4	104	Tech Conference	2024-04-05	09:00:00.0000000	4	300	250	30	Conference	4
5	105	Football Match	2024-04-10	15:30:00.0000000	5	800	600	25	Sports	5
6	106	Drama Play	2024-04-15	19:00:00.0000000	6	150	150	15	Theater	6
7	107	DJ Night	2024-04-20	22:00:00.0000000	7	300	300	15	Club	7
8	108	Outdoor Concert	2024-04-25	17:00:00.0000000	8	1000	800	40	Concert	8
9	109	Comedy Show	2024-05-02	20:30:00.0000000	9	200	200	20	Theater	9
10	110	Business Expo	2024-05-10	09:00:00.0000000	10	500	450	25	Exhibition	10

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

```
SELECT * FROM EventTable WHERE Available_seats>0;
```

Results Messages

	Event_id	Event_name	Event_date	Event_time	Venue_id	Total_seats	Available_seats	Ticket_price	Event_Type	booking_id
1	101	Music Concert	2024-03-15	18:30:00.0000000	1	500	500	50	Concert	1
2	102	Basketball Game	2024-03-20	20:00:00.0000000	2	1000	800	20	Sports	2
3	103	Art Exhibition	2024-03-25	10:00:00.0000000	3	200	200	10	Exhibition	3
4	104	Tech Conference	2024-04-05	09:00:00.0000000	4	300	250	30	Conference	4
5	105	Football Match	2024-04-10	15:30:00.0000000	5	800	600	25	Sports	5
6	106	Drama Play	2024-04-15	19:00:00.0000000	6	150	150	15	Theater	6
7	107	DJ Night	2024-04-20	22:00:00.0000000	7	300	300	15	Club	7
8	108	Outdoor Concert	2024-04-25	17:00:00.0000000	8	1000	800	40	Concert	8
9	109	Comedy Show	2024-05-02	20:30:00.0000000	9	200	200	20	Theater	9
10	110	Business Expo	2024-05-10	09:00:00.0000000	10	500	450	25	Exhibition	10

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

```
SELECT * FROM EventTable WHERE Event_name LIKE '%cup%';
```

Results Messages

Event_id	Event_name	Event_date	Event_time	Venue_id	Total_seats	Available_seats	Ticket_price	Event_Type	booking_id

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

```
SELECT * FROM EventTable WHERE Ticket_price BETWEEN 1000 AND 2500;
```

Results Messages

Event_id	Event_name	Event_date	Event_time	Venue_id	Total_seats	Available_seats	Ticket_price	Event_Type	booking_id

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

```
SELECT * FROM EventTable WHERE Event_date BETWEEN '2024-04-01' AND '2024-04-30';
```

Results Messages

	Event_id	Event_name	Event_date	Event_time	Venue_id	Total_seats	Available_seats	Ticket_price	Event_Type	booking_id
1	104	Tech Conference	2024-04-05	09:00:00.0000000	4	300	250	30	Conference	4
2	105	Football Match	2024-04-10	15:30:00.0000000	5	800	600	25	Sports	5
3	106	Drama Play	2024-04-15	19:00:00.0000000	6	150	150	15	Theater	6
4	107	DJ Night	2024-04-20	22:00:00.0000000	7	300	300	15	Club	7
5	108	Outdoor Concert	2024-04-25	17:00:00.0000000	8	1000	800	40	Concert	8

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

```
SELECT * FROM EventTable WHERE Available_seats<Total_seats AND Event_Type = 'Concert';
```

Results Messages

	Event_id	Event_name	Event_date	Event_time	Venue_id	Total_seats	Available_seats	Ticket_price	Event_Type	booking_id
1	108	Outdoor Concert	2024-04-25	17:00:00.0000000	8	1000	800	40	Concert	8

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

```
SELECT * FROM CustomerTable ORDER BY Customer_id OFFSET 5 ROWS FETCH NEXT 5 ROWS ONLY;
```

Results Messages

	Customer_id	Customer_name	Email	Phone_number	booking_id
1	1006	Eva White	eva.white@example.com	1112223344	6
2	1007	Sam Miller	sam.miller@example.com	6667778888	7
3	1008	Olivia Wilson	olivia.wilson@example.com	9998887776	8
4	1009	David Lee	david.lee@example.com	2223334455	9
5	1010	Sophia Chen	sophia.chen@example.com	7776665555	10

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

```
SELECT * FROM BookingTable WHERE Num_tickets>4;
```

Results Messages

	booking_id	Customer_id	Event_id	Num_tickets	Total_cost	Booking_Date
1	5	1005	105	5	125	2024-03-25

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

```
SELECT * FROM CustomerTable WHERE CAST(Phone_number AS VARCHAR) LIKE '%000';
```

Results Messages

	Customer_id	Customer_name	Email	Phone_number	booking_id
1	1002	Jane Smith	jane.smith@example.com	9876543000	2
2	1004	Alice Brown	alice.brown@example.com	4445556000	4

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

```
SELECT * FROM EventTable WHERE Total_seats>15000 ORDER BY Total_seats;
```

Results Messages

Event_id	Event_name	Event_date	Event_time	Venue_id	Total_seats	Available_seats	Ticket_price	Event_Type	booking_id

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

```
SELECT * FROM EventTable WHERE Event_Name NOT LIKE 'X%' AND Event_Name NOT LIKE 'Y%'
AND Event_Name NOT LIKE 'Z%';
```

Results Messages

	Event_id	Event_name	Event_date	Event_time	Venue_id	Total_seats	Available_seats	Ticket_price	Event_Type	booking_id
1	101	Music Concert	2024-03-15	18:30:00.0000000	1	500	500	50	Concert	1
2	102	Basketball Game	2024-03-20	20:00:00.0000000	2	1000	800	20	Sports	2
3	103	Art Exhibition	2024-03-25	10:00:00.0000000	3	200	200	10	Exhibition	3
4	104	Tech Conference	2024-04-05	09:00:00.0000000	4	300	250	30	Conference	4
5	105	Football Match	2024-04-10	15:30:00.0000000	5	800	600	25	Sports	5
6	106	Drama Play	2024-04-15	19:00:00.0000000	6	150	150	15	Theater	6
7	107	DJ Night	2024-04-20	22:00:00.0000000	7	300	300	15	Club	7
8	108	Outdoor Concert	2024-04-25	17:00:00.0000000	8	1000	800	40	Concert	8
9	109	Comedy Show	2024-05-02	20:30:00.0000000	9	200	200	20	Theater	9
10	110	Business Expo	2024-05-10	09:00:00.0000000	10	500	450	25	Exhibition	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_TICKET_PRICE FROM EventTable GROUP BY
Event_name;
```

Results | Messages

	Event_name	AVERAGE_TICKET_PRICE
1	Art Exhibition	10.000000
2	Basketball Game	20.000000
3	Business Expo	25.000000
4	Comedy Show	20.000000
5	DJ Night	15.000000
6	Drama Play	15.000000
7	Football Match	25.000000
8	Music Concert	50.000000
9	Outdoor Concert	40.000000
10	Tech Conference	30.000000

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

```
SELECT
    E.Event_id,
    E.Event_name,
    SUM(E.Ticket_price * B.Num_tickets) AS Total_Revenue
FROM
    EventTable E
JOIN
    BookingTable B ON E.Event_id = B.Event_id
GROUP BY
    E.Event_id, E.Event_name;
```

Results | Messages

	Event_id	Event_name	Total_Revenue
1	101	Music Concert	100
2	102	Basketball Game	80
3	103	Art Exhibition	10
4	104	Tech Conference	90
5	105	Football Match	125
6	106	Drama Play	30
7	107	DJ Night	30
8	108	Outdoor Concert	160
9	109	Comedy Show	20
10	110	Business Expo	75

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

```
SELECT TOP 1 E.Event_id, E.event_name, SUM(B.Num_tickets*E.ticket_price) AS
Total_revenue
FROM EventTable E
JOIN BookingTable B
ON E.Event_id=B.Event_id
GROUP BY E.Event_id, E.Event_name
ORDER BY total_revenue DESC;
```

Results Messages

	Event_id	event_name	Total_revenue
1	108	Outdoor Concert	160

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

```
SELECT E.Event_id,E.Event_name, SUM(B.Num_Tickets) AS TOTAL_TICKETS
FROM EventTable E
JOIN BookingTable B
ON E.Event_id=B.Event_id GROUP BY E.Event_id,E.Event_name;
```

Results Messages

	Event_id	Event_name	TOTAL_TICKETS
1	101	Music Concert	2
2	102	Basketball Game	4
3	103	Art Exhibition	1
4	104	Tech Conference	3
5	105	Football Match	5
6	106	Drama Play	2
7	107	DJ Night	2
8	108	Outdoor Concert	4
9	109	Comedy Show	1
10	110	Business Expo	3

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

```
SELECT Event_id
FROM BookingTable
GROUP BY Event_id
HAVING COUNT(Num_tickets) = 0
ORDER BY Event_id;
```

Results Messages

Event_id

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

```
SELECT TOP 1 C.Customer_id, C.Customer_name FROM CustomerTable C
JOIN BookingTable B
ON C.Customer_id=B.Customer_id
ORDER BY B.Num_Tickets DESC;
```

Results Messages

	Customer_id	Customer_name
1	1005	Charlie Davis

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

```
SELECT E.Event_id, E.Event_name,MONTH(Booking_date) AS Month, SUM(Num_tickets) as
total_tickets
FROM EventTable E
```

```
LEFT JOIN BookingTable B ON E.Event_id=B.Event_id
GROUP BY E.Event_id, Event_name, MONTH(Booking_date)
ORDER BY Month, Event_id;
```

Results Messages

	Event_id	Event_name	Month	total_tickets
1	101	Music Concert	3	2
2	102	Basketball Game	3	4
3	103	Art Exhibition	3	1
4	104	Tech Conference	3	3
5	105	Football Match	3	5
6	106	Drama Play	3	2
7	107	DJ Night	4	2
8	108	Outdoor Concert	4	4
9	109	Comedy Show	4	1
10	110	Business Expo	4	3

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 VenueTable V
JOIN EventTable B ON V.Venue_id=B.Venue_id
GROUP BY Venue_name;
```

Results Messages

	Venue_name	Average_ticket_price
1	Amphitheater H	40.000000
2	Auditorium I	20.000000
3	Club G	15.000000
4	Concert Hall A	50.000000
5	Conference Hall D	30.000000
6	Convention Center J	25.000000
7	Exhibition Center C	10.000000
8	Sports Arena B	20.000000
9	Stadium E	25.000000
10	Theater F	15.000000

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

```
SELECT Event_Type,SUM(Num_tickets) as Num_tickets_sold
FROM EventTable E
JOIN BookingTable B ON B.Event_id=E.Event_id
GROUP BY Event_Type;
```

Results Messages

	Event_Type	Num_tickets_sold
1	Club	2
2	Concert	6
3	Conference	3
4	Exhibition	4
5	Sports	9
6	Theater	3

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

```
SELECT YEAR(B.Booking_Date) AS Year, SUM(E.Ticket_price * b.Num_tickets) AS
Total_Revenue
FROM BookingTable B
JOIN EventTable E ON B.Event_id = E.Event_id
GROUP BY YEAR(B.Booking_Date);
```

Results Messages

	Year	Total_Revenue
1	2024	720

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

```
SELECT C.Customer_id, C.Customer_name, COUNT(DISTINCT B.Event_id) AS
Total_Events_Booked
FROM CustomerTable C
JOIN BookingTable B ON C.Customer_id = B.Customer_id
GROUP BY C.Customer_id, C.Customer_name
HAVING COUNT(DISTINCT B.Event_id) > 1;
```

Results Messages

Customer_id	Customer_name	Total_Events_Booked

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

```
SELECT C.Customer_id,C.Customer_name, SUM(E.Ticket_price*B.Num_tickets) AS
Total_Revenue
FROM CustomerTable C
JOIN BookingTable B ON C.booking_id = B.booking_id
JOIN EventTable E ON B.Event_id=E.Event_id
GROUP BY C.Customer_id, C.Customer_name;
```

Results Messages

	Customer_id	Customer_name	Total_Revenue
1	1001	John Doe	100
2	1002	Jane Smith	80
3	1003	Bob Johnson	10
4	1004	Alice Brown	90
5	1005	Charlie Davis	125
6	1006	Eva White	30
7	1007	Sam Miller	30
8	1008	Olivia Wilson	160
9	1009	David Lee	20
10	1010	Sophia Chen	75

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

```
SELECT V.Venue_name, E.Event_type, AVG(E.Ticket_price) AS AVERAGE_TICKET_PRICE
FROM EventTable E
JOIN VenueTable V
ON V.Venue_id=E.Venue_id
GROUP BY V.Venue_name, E.Event_type;
```

Results | Messages

	Venue_name	Event_type	AVERAGE_TICKET_PRICE
1	Club G	Club	15.000000
2	Amphitheater H	Concert	40.000000
3	Concert Hall A	Concert	50.000000
4	Conference Hall D	Conference	30.000000
5	Convention Center J	Exhibition	25.000000
6	Exhibition Center C	Exhibition	10.000000
7	Sports Arena B	Sports	20.000000
8	Stadium E	Sports	25.000000
9	Auditorium I	Theater	20.000000
10	Theater F	Theater	15.000000

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

```
SELECT
    CT.Customer_id,
    CT.Customer_name,
    SUM(BT.Num_tickets) AS Total_Tickets_Purchased
FROM
    CustomerTable CT
JOIN
    BookingTable BT ON CT.booking_id = BT.booking_id
WHERE
    BT.Booking_Date >= '2024-02-03'
GROUP BY
    CT.Customer_id, CT.Customer_name;
```

Results | Messages

	Customer_id	Customer_name	Total_Tickets_Purchased
1	1001	John Doe	2
2	1002	Jane Smith	4
3	1003	Bob Johnson	1
4	1004	Alice Brown	3
5	1005	Charlie Davis	5
6	1006	Eva White	2
7	1007	Sam Miller	2
8	1008	Olivia Wilson	4
9	1009	David Lee	1
10	1010	Sophia Chen	3

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 EventTable
WHERE EventTable.Venue_id=VenueTable.Venue_id)
FROM VenueTable;
```

Results | Messages

	Venue_name	(No column name)
1	Concert Hall A	50.000000
2	Sports Arena B	20.000000
3	Exhibition Center C	10.000000
4	Conference Hall D	30.000000
5	Stadium E	25.000000
6	Theater F	15.000000
7	Club G	15.000000
8	Amphitheater H	40.000000
9	Auditorium I	20.000000
10	Convention Center J	25.000000

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

```
SELECT Event_id, Event_name
FROM EventTable
WHERE (SELECT SUM(Num_tickets) FROM BookingTable WHERE EventTable.Event_id =
BookingTable.Event_id) > 0.5 * Total_seats;
```

Results | Messages

	Event_id	Event_name

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

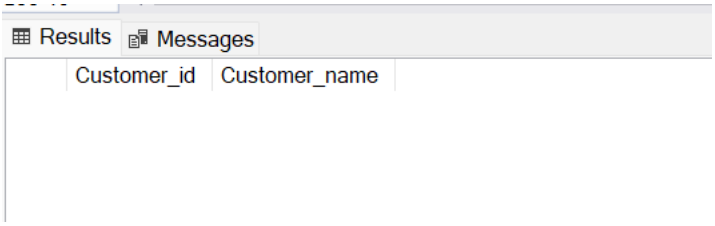
```
SELECT E.Event_id, E.Event_name, (SELECT SUM(B.Num_tickets) FROM BookingTable B WHERE
B.Event_id = E.Event_id) AS Total_Tickets_Sold
FROM EventTable E;
```

Results | Messages

	Event_id	Event_name	Total_Tickets_Sold
1	101	Music Concert	2
2	102	Basketball Game	4
3	103	Art Exhibition	1
4	104	Tech Conference	3
5	105	Football Match	5
6	106	Drama Play	2
7	107	DJ Night	2
8	108	Outdoor Concert	4
9	109	Comedy Show	1
10	110	Business Expo	3

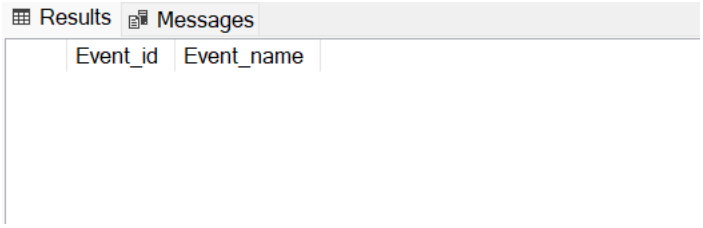
4. Find Users Who Have Not Booked Any Tickets Using a NOT EXISTS Subquery.

```
SELECT Customer_id, Customer_name
FROM CustomerTable C
WHERE NOT EXISTS (SELECT 1 FROM BookingTable B WHERE C.booking_id = B.booking_id);
```



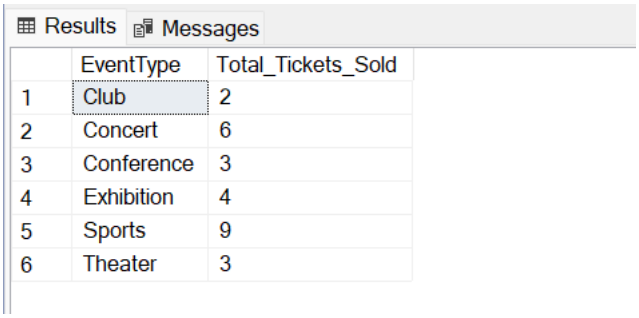
5. List Events with No Ticket Sales Using a NOT IN Subquery.

```
SELECT Event_id, Event_name
FROM EventTable
WHERE Event_id NOT IN (SELECT DISTINCT Event_id FROM BookingTable);
```



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

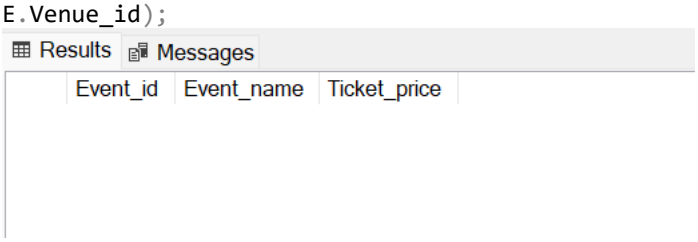
```
SELECT EventType, SUM(Num_tickets) AS Total_Tickets_Sold
FROM (SELECT E.Event_id, E.Event_Type AS EventType, B.Num_tickets FROM EventTable E
JOIN BookingTable B ON E.Event_id = B.Event_id) AS Subquery
GROUP BY EventType;
```



	EventType	Total_Tickets_Sold
1	Club	2
2	Concert	6
3	Conference	3
4	Exhibition	4
5	Sports	9
6	Theater	3

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 EventTable E
WHERE Ticket_price > (SELECT AVG(Ticket_price) FROM EventTable WHERE Venue_id =
E.Venue_id);
```



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

```
SELECT Customer_id, Customer_name,
(SELECT SUM(E.Ticket_price * B.Num_tickets) FROM EventTable E
JOIN BookingTable B ON E.Event_id = B.Event_id
```

```
WHERE C.booking_id = B.booking_id) AS Total_Revenue
FROM CustomerTable C;
```

Results | Messages

	Customer_id	Customer_name	Total_Revenue
1	1001	John Doe	100
2	1002	Jane Smith	80
3	1003	Bob Johnson	10
4	1004	Alice Brown	90
5	1005	Charlie Davis	125
6	1006	Eva White	30
7	1007	Sam Miller	30
8	1008	Olivia Wilson	160
9	1009	David Lee	20
10	1010	Sophia Chen	75

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 CustomerTable C
WHERE C.booking_id IN (SELECT B.booking_id FROM BookingTable B WHERE B.Event_id
IN (SELECT Event_id FROM EventTable WHERE Venue_id = 1));
```

Results | Messages

	Customer_id	Customer_name
1	1001	John Doe

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_id, E.Event_Type, B.Num_tickets FROM EventTable E
JOIN BookingTable B ON E.Event_id = B.Event_id) AS Subquery
GROUP BY Event_Type;
```

Results | Messages

	Event_Type	Total_Tickets_Sold
1	Club	2
2	Concert	6
3	Conference	3
4	Exhibition	4
5	Sports	9
6	Theater	3

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

```
SELECT C.Customer_id, C.Customer_name
```

```
FROM CustomerTable C
WHERE
    EXISTS (
        SELECT 1
        FROM BookingTable B
        WHERE C.booking_id = B.booking_id
        AND FORMAT(B.Booking_Date, 'yyyy-MM') IN (
            SELECT DISTINCT FORMAT(Booking_Date, 'yyyy-MM')
            FROM BookingTable
        )
    );
```

Results Messages

	Customer_id	Customer_name
1	1001	John Doe
2	1002	Jane Smith
3	1003	Bob Johnson
4	1004	Alice Brown
5	1005	Charlie Davis
6	1006	Eva White
7	1007	Sam Miller
8	1008	Olivia Wilson
9	1009	David Lee
10	1010	Sophia Chen

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

```
SELECT Venue_name, (SELECT AVG(Ticket_price) FROM EventTable WHERE Venue_id =
V.Venue_id) AS Average_Ticket_Price
FROM VenueTable V;
```

Results Messages

	Venue_name	Average_Ticket_Price
1	Concert Hall A	50.000000
2	Sports Arena B	20.000000
3	Exhibition Center C	10.000000
4	Conference Hall D	30.000000
5	Stadium E	25.000000
6	Theater F	15.000000
7	Club G	15.000000
8	Amphitheater H	40.000000
9	Auditorium I	20.000000
10	Convention Center J	25.000000