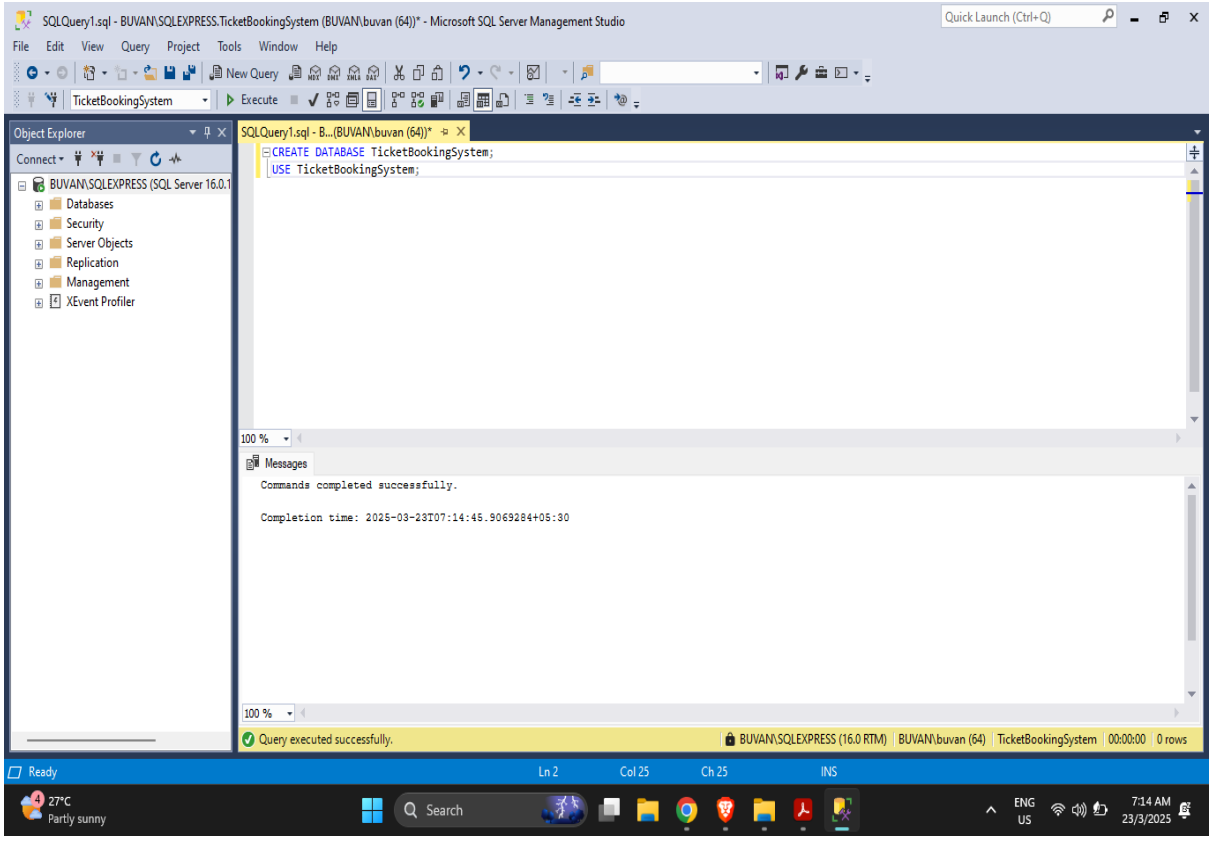
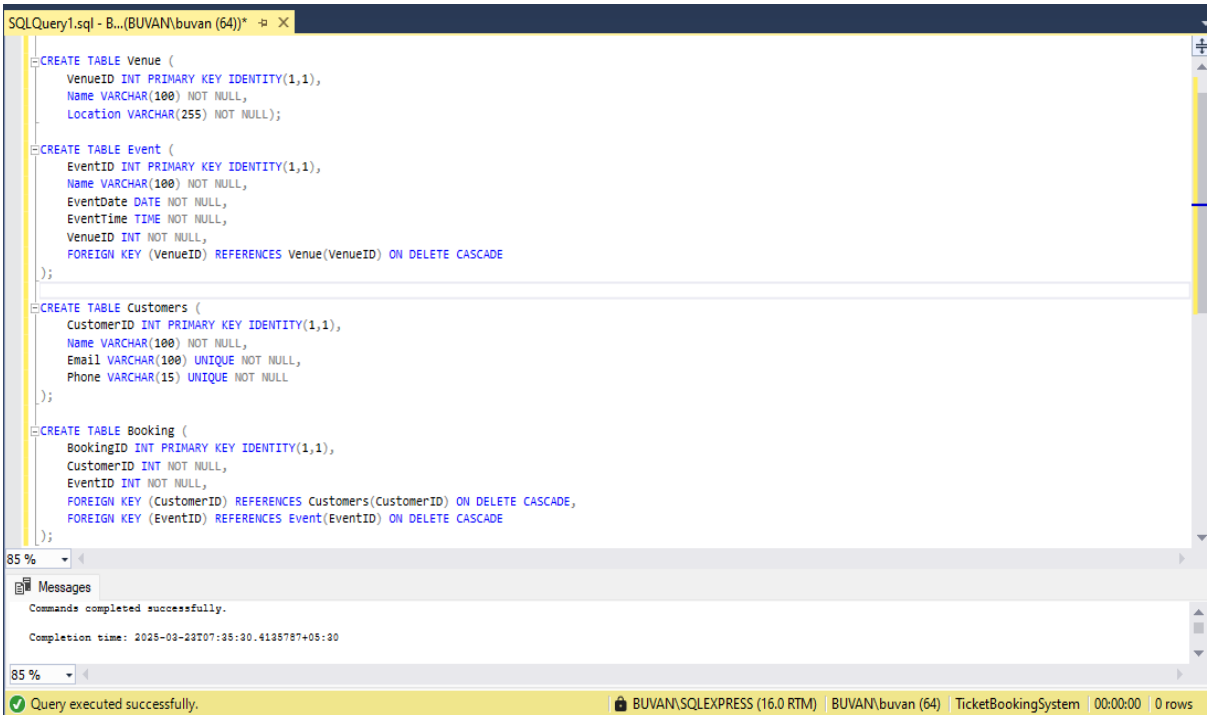
Ticket Booking System

Task 1:

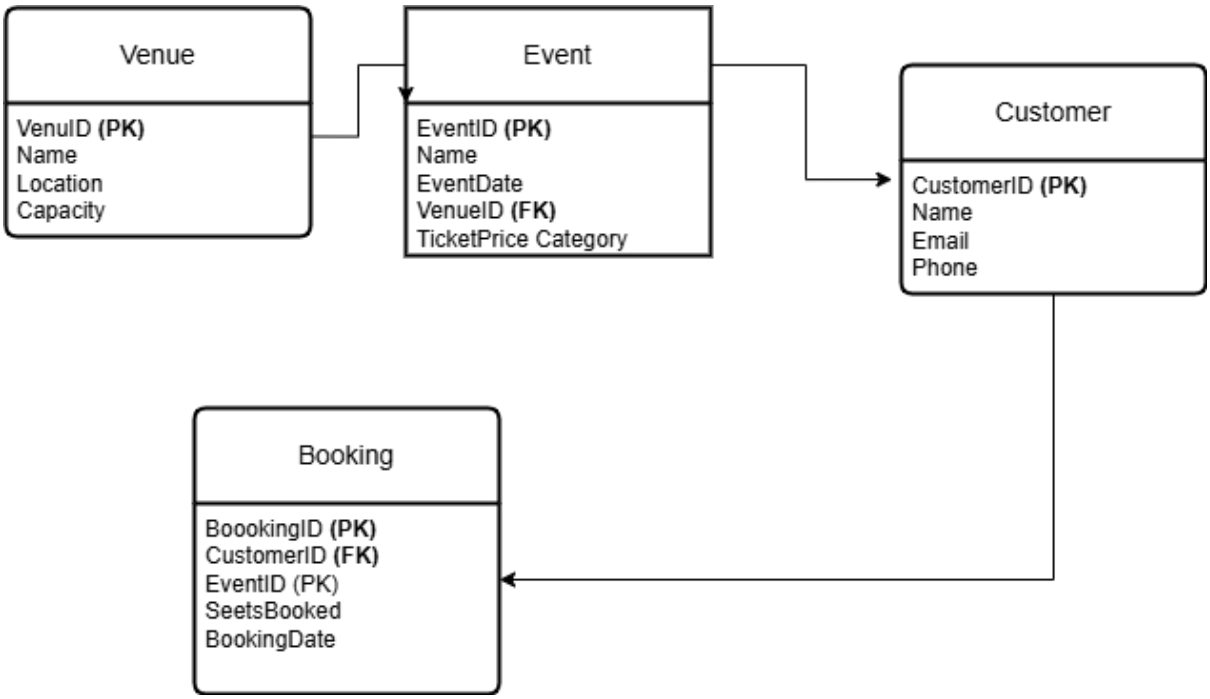
1. Create the database named "TicketBookingSystem"



2. Write SQL scripts to create the mentioned tables with appropriate data types, constraints, and relationships.
    - Venu
    - Event
    - Customers
    - Booking
4. Create appropriate Primary Key and Foreign Key constraints for referential integrity.



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



Task 2:

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

```sql
INSERT INTO Venue (Name, Location, Capacity) VALUES
('Le Meridian', 'Guindy', 500),
('Taj', 'Uptown', 1000),
('Chepauk Stadium', 'Chepauk', 2000),
('Kamala Theatre', 'Midtown', 800),
('Convention Center', 'ECR', 1500),
('National Park', 'Adyar', 1200),
('Art Gallery', 'Central City', 600),
('Sports Complex', 'East Side', 2500),
('Exhibition Hall', 'South District', 1800),
('Cultural Center', 'North Zone', 700);
```

```
(10 rows affected)

Completion time: 2025-03-23T08:52:00.8205654+05:30
```

Query executed successfully. BUVAN\SQLEXPRESS (16.0 RTM) BUVAN\buvan (76) TicketBookingSystem 00:00:00 0 rows

```sql
INSERT INTO Customers (Name, Email, Phone) VALUES
('MS Gandhi', 'gandhi@gmail.com', '1234567890'),
('Jeevan', 'jaeevan@yahoo.com', '0987654321'),
('Parthiban', 'parthiban@outlook.com', '1122334455'),
('Leo Das', 'das@gmail.com', '2233445566'),
('Vijay Rajendran', 'vijay@yahoo.com', '3344556677'),
('Veera Raghavan ', 'veera@yahoo.com', '4455667788'),
('John Durairaj', 'durai@gmail.com', '5566778899'),
('Michael Rayappan ', 'bigil@gmail.com', '6677889900'),
('Sundar Ramasamy ', 'sundar@gmail.com', '7788990011'),
('Vetrimaaran', 'vetri@gmail.com', '8899001122');
```

```
(10 rows affected)

Completion time: 2025-03-23T08:52:44.8373610+05:30
```

Query executed successfully. BUVAN\SQLEXPRESS (16.0 RTM) BUVAN\buvan (76) TicketBookingSystem 00:00:00 0 rows

SQLQuery1.sql - B...(BUVAN\buvan (76))*

```sql
INSERT INTO Event (Name, EventDate, EventTime, VenueID) VALUES
('Rock Concert', '2025-06-10', '19:00:00', 1),
('Disco', '2025-07-20', '17:30:00', 3),
('IPL', '2025-08-05', '10:00:00', 5),
('Drama Play', '2025-06-25', '20:00:00', 4),
('Theater Play', '2025-09-10', '18:00:00', 2),
('Music Festival', '2025-07-15', '16:00:00', 7),
('Classical Arts', '2025-08-22', '11:30:00', 6),
('Basketball League', '2025-06-30', '14:00:00', 8),
('Orchestra Night', '2025-07-12', '21:00:00', 9),
('Food Carnival', '2025-08-18', '15:00:00', 10);
```

161 %

Messages

```
(10 rows affected)

Completion time: 2025-03-23T08:53:48.6260019+05:30
```

161 %

SQLQuery1.sql - B...(BUVAN\buvan (76))*

```sql
INSERT INTO Booking (CustomerID, EventID, SeatsBooked) VALUES
(1, 1, 2),
(2, 2, 4),
(3, 3, 1),
(4, 4, 3),
(5, 5, 5),
(6, 6, 2),
(7, 7, 6),
(8, 8, 1),
(9, 9, 3),
(10, 10, 7);
```

161 %

Messages

```
(10 rows affected)

Completion time: 2025-03-23T08:54:12.4796558+05:30
```

161 %

Query executed successfully. | BUVAN\SQLEXPRESS (16.0 RTM) | BUVAN\buvan (76) | TicketBookingSystem | 00:00:00 | 0 rows

2. Write a SQL query to list all Events.

SQLQuery1.sql - B...(BUVAN\buvan (76))*

```sql
Select * from Event;
```

140 %

Results | Messages

	EventID	Name	EventDate	EventTime	VenueID
1	1	Rock Concert	2025-06-10	19:00:00.0000000	1
2	4	Drama Play	2025-06-25	20:00:00.0000000	4
3	6	Music Festival	2025-07-15	16:00:00.0000000	7
4	8	Basketball League	2025-06-30	14:00:00.0000000	8
5	9	Orchestra Night	2025-07-12	21:00:00.0000000	9
6	10	Food Carnival	2025-08-18	15:00:00.0000000	10
7	12	Disco	2025-07-20	17:30:00.0000000	3
8	13	IPL	2025-08-05	10:00:00.0000000	5
9	15	Theater Play	2025-09-10	18:00:00.0000000	2
10	17	Classical Arts	2025-08-22	11:30:00.0000000	6

Query executed successfully. | BUVAN\SQLEXPRESS (16.0 RTM) | BUVAN\buvan (76) | TicketBookingSystem | 00:00:00 | 10 rows

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

SQLQuery1.sql - B...(BUVAN\buvan (76))*

```sql
SELECT e.*
FROM Event e
JOIN Venue v ON e.VenueID = v.VenueID
LEFT JOIN (
    SELECT EventID, SUM(SeatsBooked) AS TotalBooked
    FROM Booking
    GROUP BY EventID
) b ON e.EventID = b.EventID
WHERE (v.Capacity - COALESCE(b.TotalBooked, 0)) > 0;
```

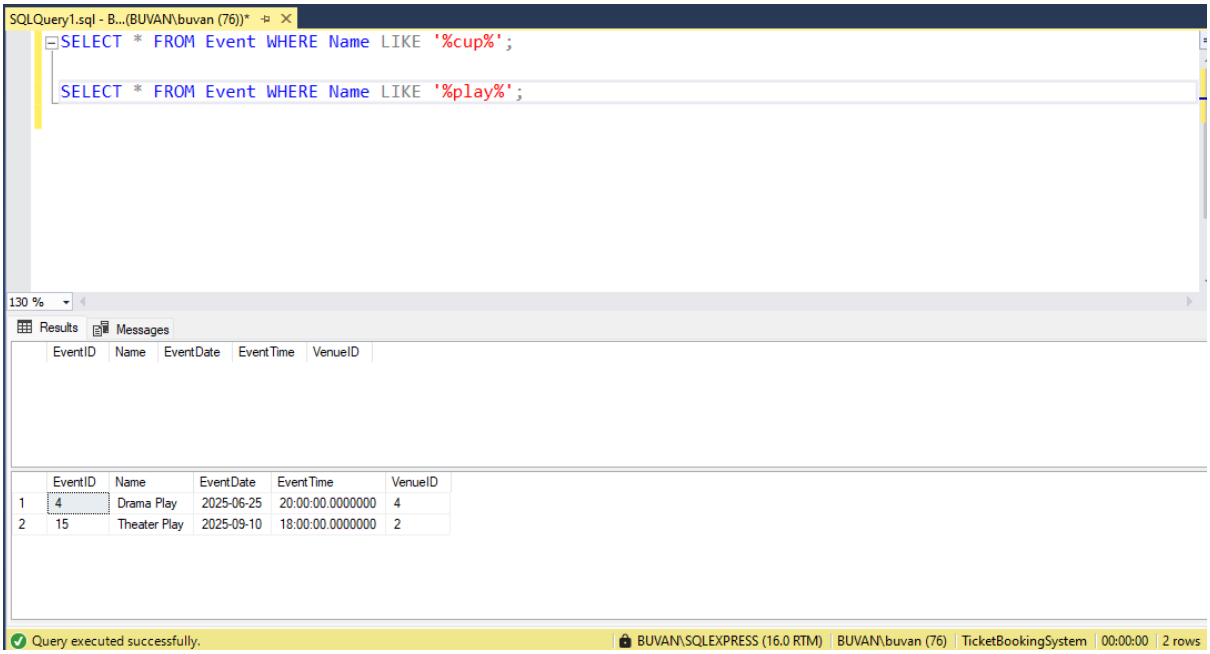
130 %

Results | Messages

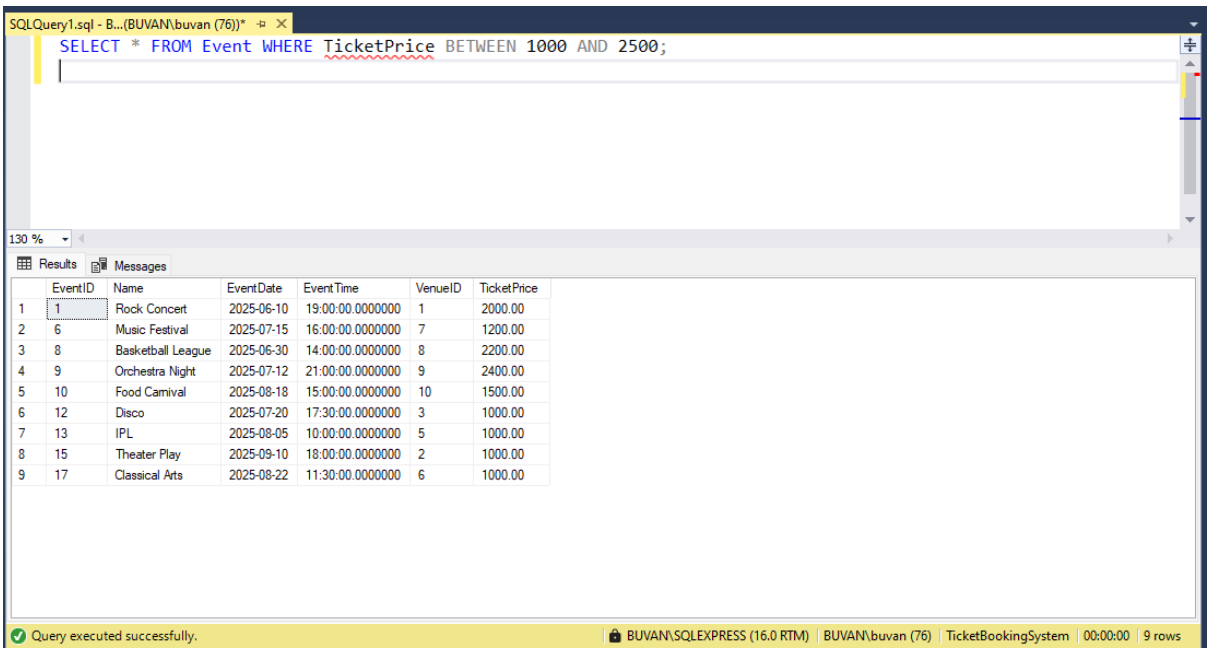
	EventID	Name	EventDate	EventTime	VenueID
1	1	Rock Concert	2025-06-10	19:00:00.0000000	1
2	4	Drama Play	2025-06-25	20:00:00.0000000	4
3	6	Music Festival	2025-07-15	16:00:00.0000000	7
4	8	Basketball League	2025-06-30	14:00:00.0000000	8
5	9	Orchestra Night	2025-07-12	21:00:00.0000000	9
6	10	Food Carnival	2025-08-18	15:00:00.0000000	10
7	12	Disco	2025-07-20	17:30:00.0000000	3
8	13	IPL	2025-08-05	10:00:00.0000000	5
9	15	Theater Play	2025-09-10	18:00:00.0000000	2
10	17	Classical Arts	2025-08-22	11:30:00.0000000	6

Query executed successfully. | BUVAN\SQLEXPRESS (16.0 RTM) | BUVAN\buvan (76) | TicketBookingSystem | 00:00:00 | 10 rows

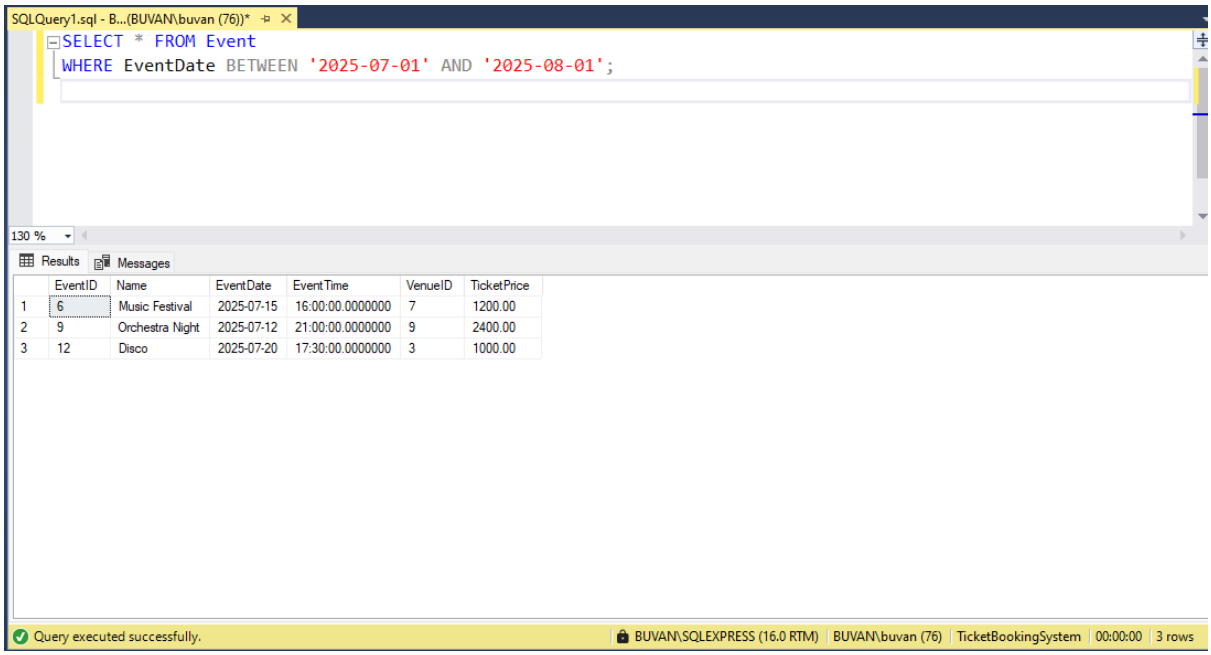
4. Write a SQL query to select events name partial match with ‘cup’.



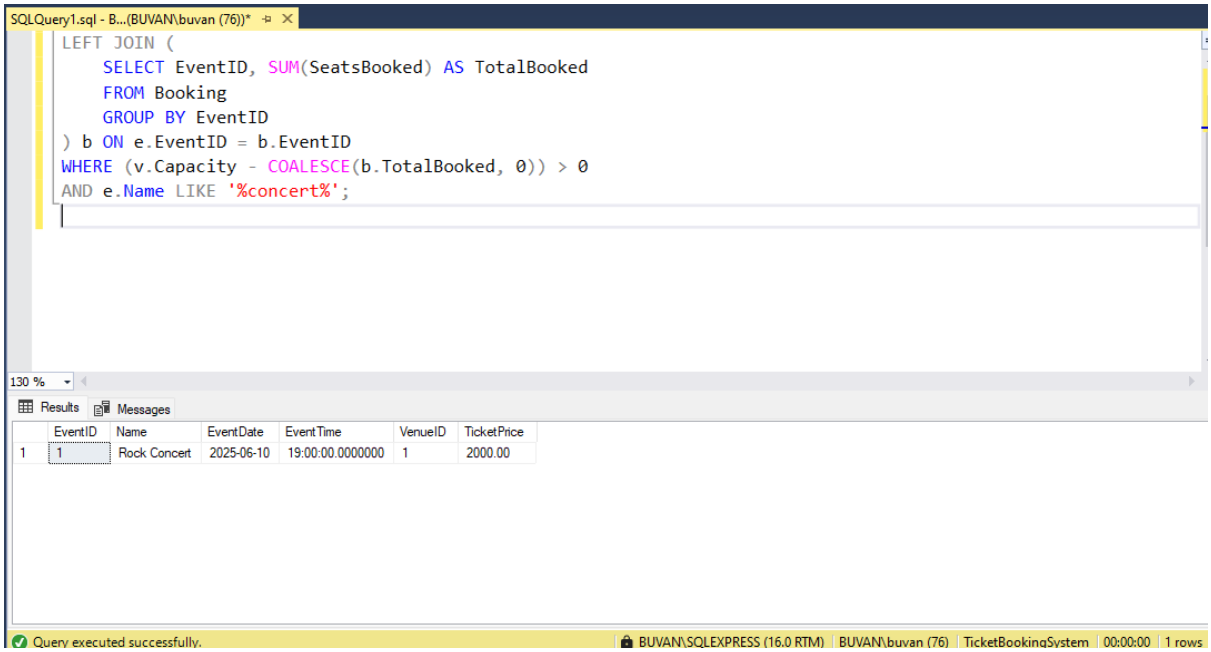
5. Write a SQL query to select events with ticket price range is between 1000 to 2500.



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



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



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

SQLQuery1.sql - B...(BUVAN\buvan (76))*

```sql
SELECT * FROM Customers
ORDER BY CustomerID
OFFSET 5 ROWS FETCH NEXT 5 ROWS ONLY;
```

130 %

Results | Messages

	CustomerID	Name	Email	Phone
1	6	Veera Raghavan	veera@yahoo.com	4455667788
2	7	John Durairaj	durai@gmail.com	5566778899
3	8	Michael Rayappan	bigil@gmail.com	6677889900
4	9	Sundar Ramasamy	sundar@gmail.com	7788990011
5	10	Vetrimaaran	vetri@gmail.com	8899001122

Query executed successfully. | BUVAN\SQLEXPRESS (16.0 RTM) | BUVAN\buvan (76) | TicketBookingSystem | 00:00:00 | 5 rows

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

SQLQuery1.sql - B...(BUVAN\buvan (76))*

```sql
SELECT * FROM Booking WHERE SeatsBooked > 4;
```

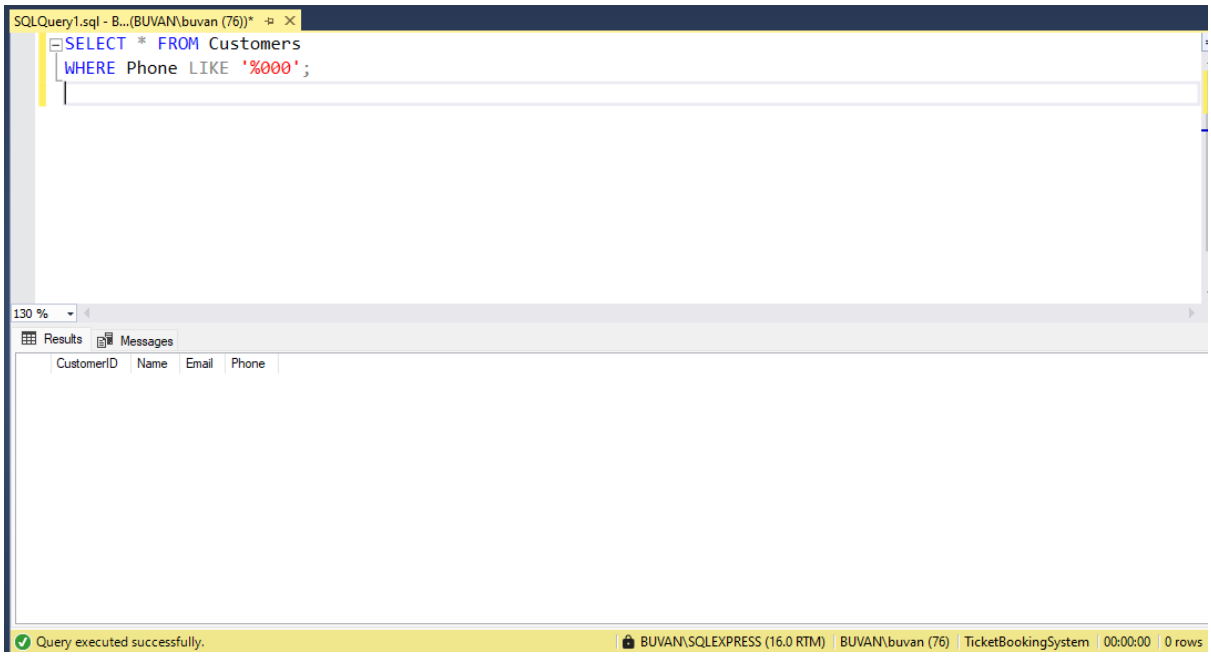
130 %

Results | Messages

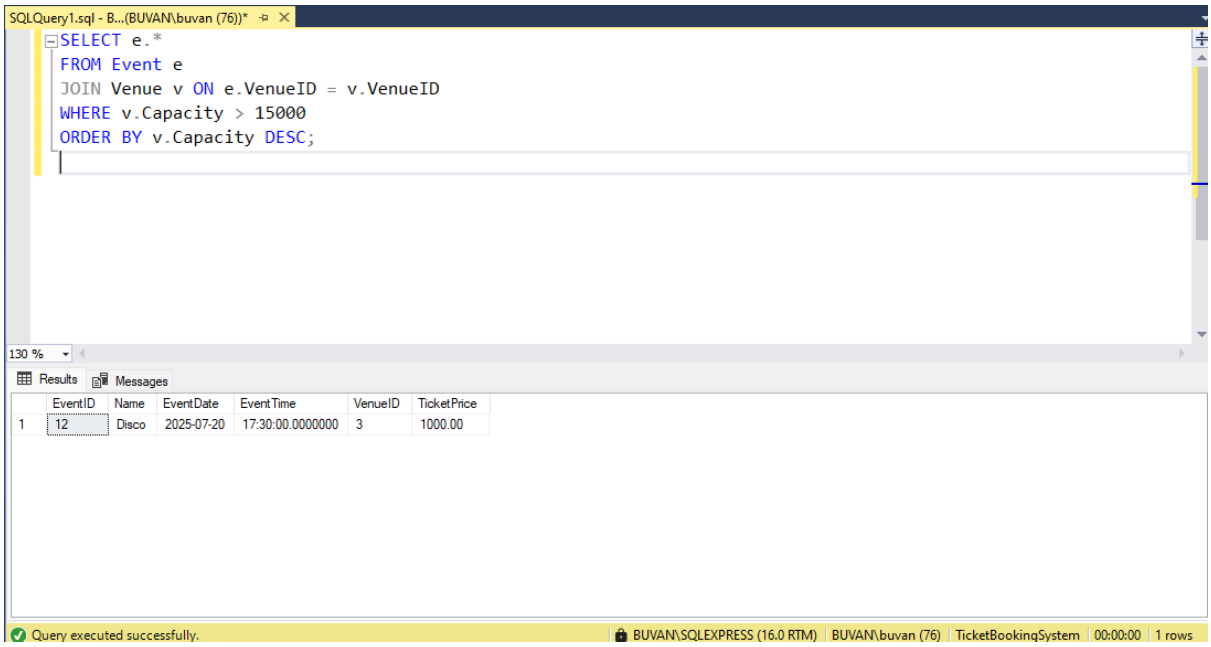
	BookingID	CustomerID	EventID	BookingDate	SeatsBooked
1	10	10	10	2025-03-23 08:54:12.470	7

Query executed successfully. | BUVAN\SQLEXPRESS (16.0 RTM) | BUVAN\buvan (76) | TicketBookingSystem | 00:00:00 | 1 rows

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



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



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

SQLQuery1.sql - B...(BUVAN\buvan (76))*

```sql
SELECT * FROM Event
WHERE Name NOT LIKE 'X%'
AND Name NOT LIKE 'Y%'
AND Name NOT LIKE 'Z%';
```

130 %

Results | Messages

	EventID	Name	EventDate	EventTime	VenueID	TicketPrice
1	1	Rock Concert	2025-06-10	19:00:00.0000000	1	2000.00
2	4	Drama Play	2025-06-25	20:00:00.0000000	4	500.00
3	6	Music Festival	2025-07-15	16:00:00.0000000	7	1200.00
4	8	Basketball League	2025-06-30	14:00:00.0000000	8	2200.00
5	9	Orchestra Night	2025-07-12	21:00:00.0000000	9	2400.00
6	10	Food Carnival	2025-08-18	15:00:00.0000000	10	1500.00
7	12	Disco	2025-07-20	17:30:00.0000000	3	1000.00
8	13	IPL	2025-08-05	10:00:00.0000000	5	1000.00
9	15	Theater Play	2025-09-10	18:00:00.0000000	2	1000.00
10	17	Classical Arts	2025-08-22	11:30:00.0000000	6	1000.00

Query executed successfully. | BUVAN\SQLEXPRESS (16.0 RTM) | BUVAN\buvan (76) | TicketBookingSystem | 00:00:00 | 10 rows

Task 3:

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

SQLQuery1.sql - B...(BUVAN\buvan (52))*

```sql
SELECT EventID, Name, AVG(TicketPrice) AS AvgTicketPrice
FROM Event
GROUP BY EventID, Name;
```

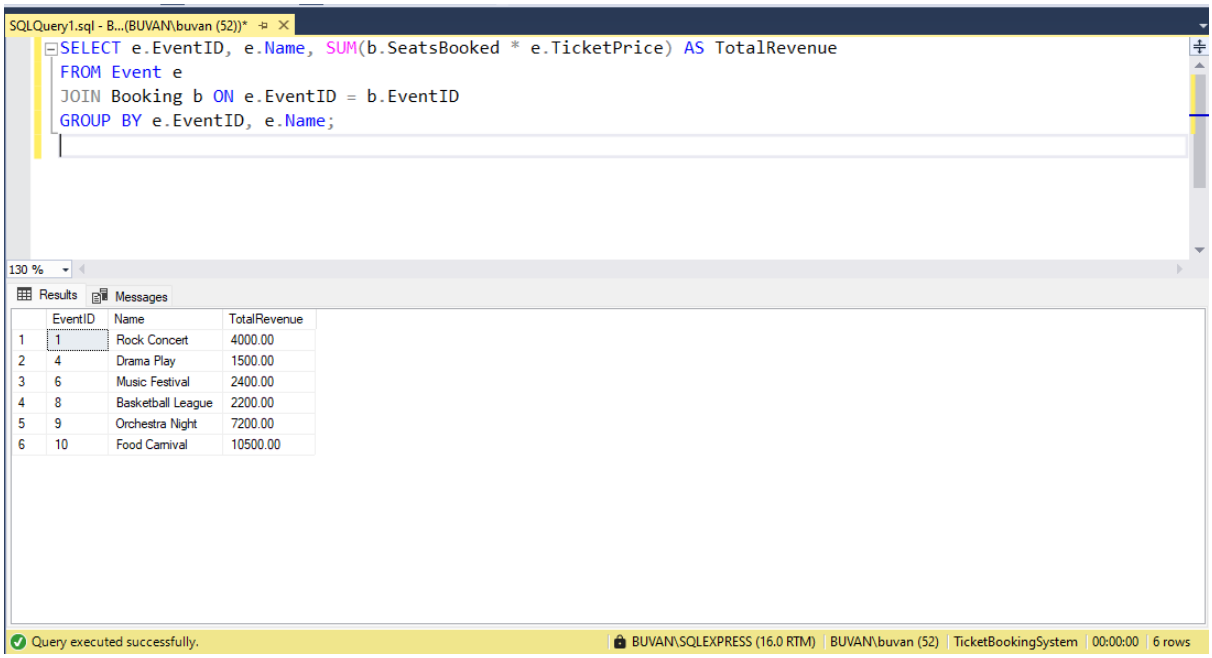
130 %

Results | Messages

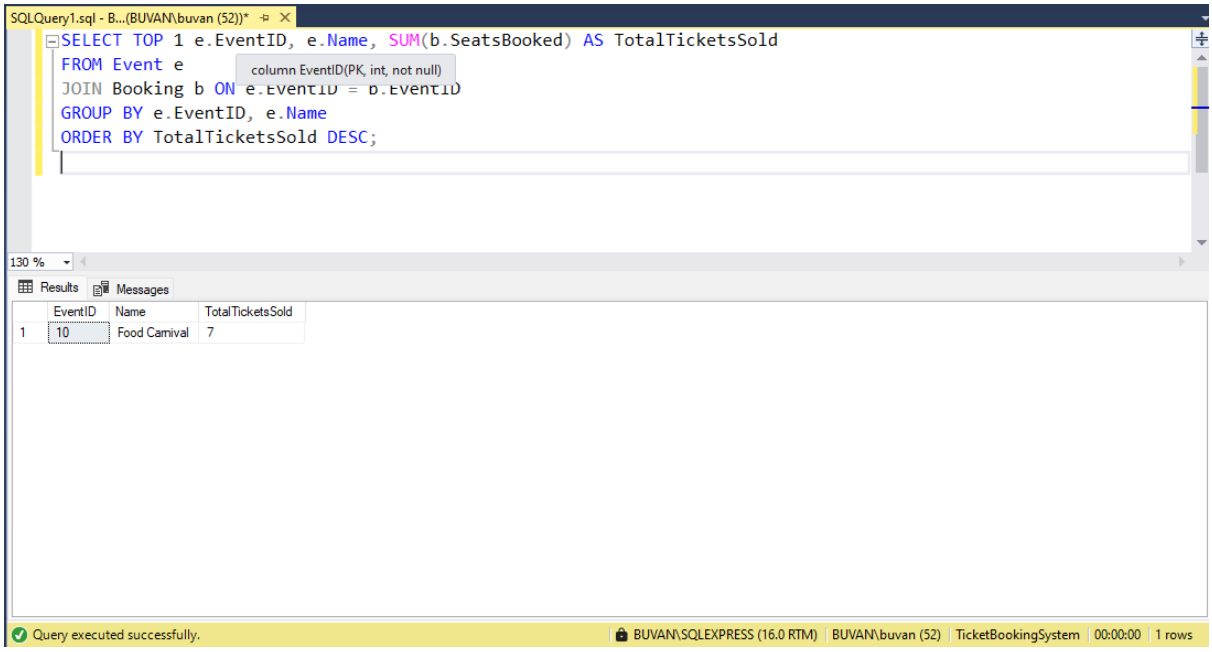
	EventID	Name	AvgTicketPrice
1	1	Rock Concert	2000.000000
2	4	Drama Play	500.000000
3	6	Music Festival	1200.000000
4	8	Basketball League	2200.000000
5	9	Orchestra Night	2400.000000
6	10	Food Carnival	1500.000000
7	12	Disco	1000.000000
8	13	IPL	1000.000000
9	15	Theater Play	1000.000000
10	17	Classical Arts	1000.000000

Query executed successfully. | BUVAN\SQLEXPRESS (16.0 RTM) | BUVAN\buvan (52) | TicketBookingSystem | 00:00:00 | 10 rows

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



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



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

SQLQuery1.sql - B...(BUVAN\buvan (52))*

```sql
SELECT e.EventID, e.Name, SUM(b.SeatsBooked) AS TotalTicketsSold
FROM Event e
JOIN Booking b ON e.EventID = b.EventID
GROUP BY e.EventID, e.Name;
```

130 %

Results | Messages

	EventID	Name	TotalTicketsSold
1	1	Rock Concert	2
2	4	Drama Play	3
3	6	Music Festival	2
4	8	Basketball League	1
5	9	Orchestra Night	3
6	10	Food Carnival	7

Query executed successfully. | BUVAN\SQLEXPRESS (16.0 RTM) | BUVAN\buvan (52) | TicketBookingSystem | 00:00:00 | 6 rows

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

SQLQuery1.sql - B...(BUVAN\buvan (52))*

```sql
SELECT e.EventID, e.Name
FROM Event e
LEFT JOIN Booking b ON e.EventID = b.EventID
WHERE b.EventID IS NULL;
```

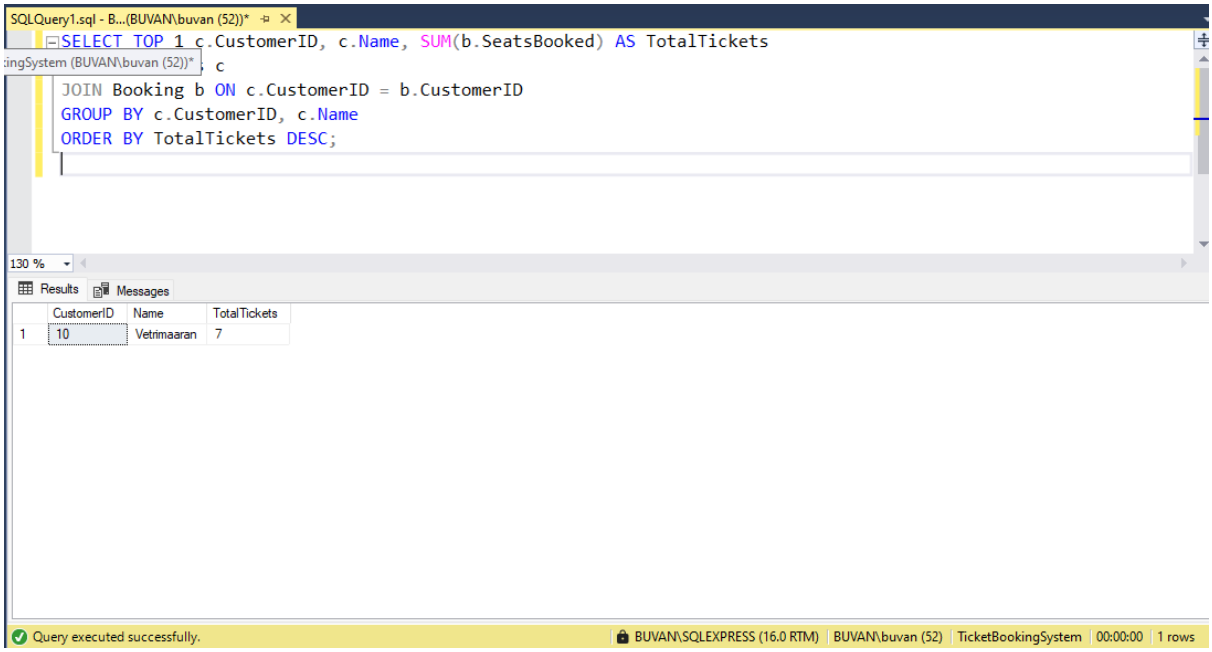
130 %

Results | Messages

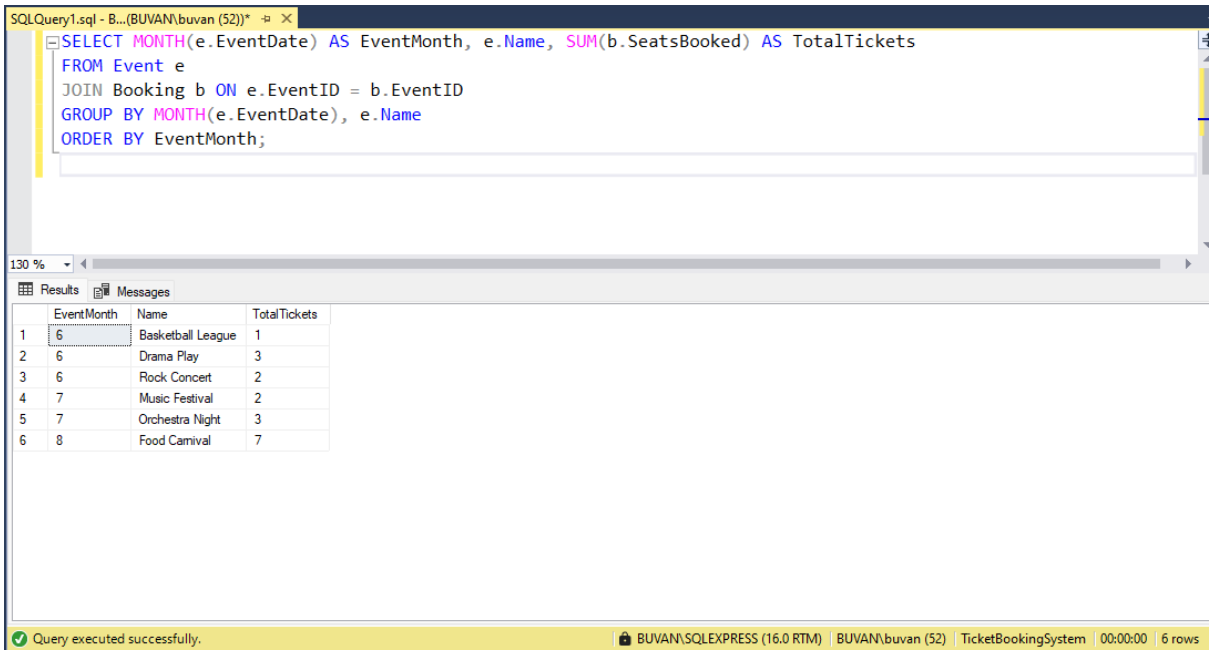
	EventID	Name
1	12	Disco
2	13	IPL
3	15	Theater Play
4	17	Classical Arts

Query executed successfully. | BUVAN\SQLEXPRESS (16.0 RTM) | BUVAN\buvan (52) | TicketBookingSystem | 00:00:00 | 4 rows

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



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



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

SQLQuery1.sql - B...(BUVAN\buvan (52))*

```sql
SELECT v.VenueID, v.Name AS VenueName, AVG(e.TicketPrice) AS AvgTicketPrice
FROM Venue v
JOIN Event e ON v.VenueID = e.VenueID
GROUP BY v.VenueID, v.Name;
```

130 %

Results | Messages

	VenueID	VenueName	AvgTicketPrice
1	1	Le Meridian	2000.000000
2	2	Taj	1000.000000
3	3	Chepauk Stadium	1000.000000
4	4	Kamala Theatre	500.000000
5	5	Convention Center	1000.000000
6	6	National Park	1000.000000
7	7	Art Gallery	1200.000000
8	8	Sports Complex	2200.000000
9	9	Exhibition Hall	2400.000000
10	10	Cultural Center	1500.000000

Query executed successfully. | BUVAN\SQLEXPRESS (16.0 RTM) | BUVAN\buvan (52) | TicketBookingSystem | 00:00:00 | 10 rows

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

SQLQuery1.sql - B...(BUVAN\buvan (52))*

```sql
SELECT e.EventType, SUM(b.SeatsBooked) AS TotalTicketsSold
FROM Event e
JOIN Booking b ON e.EventID = b.EventID
GROUP BY e.EventType;
```

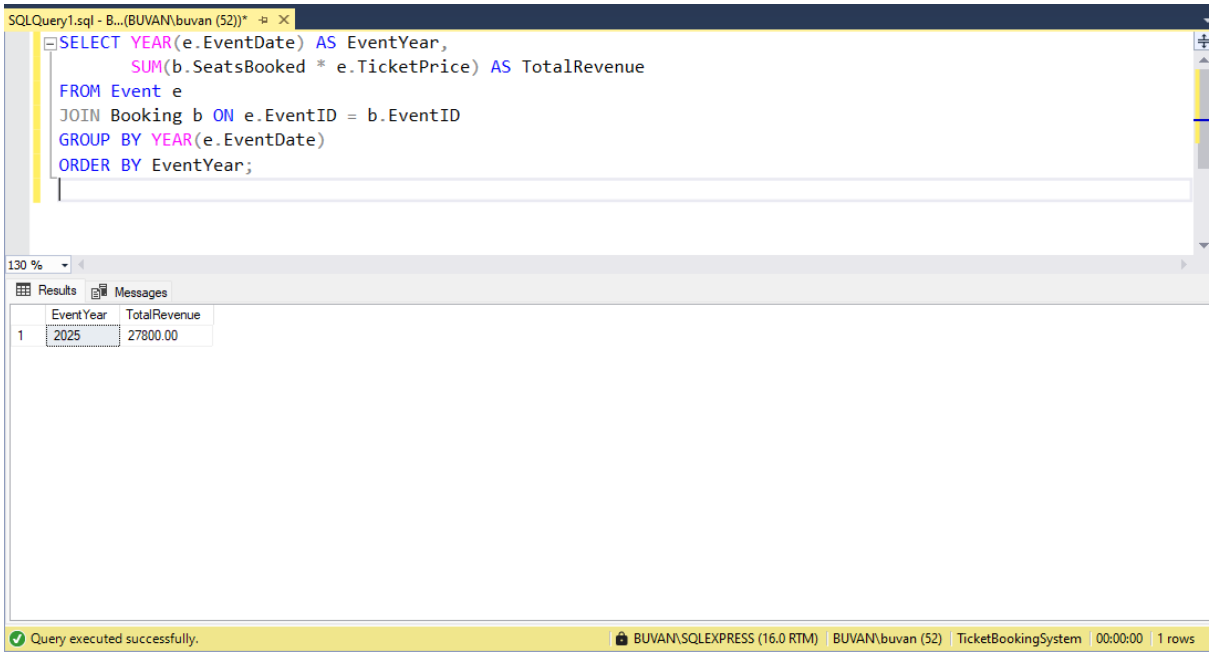
130 %

Results | Messages

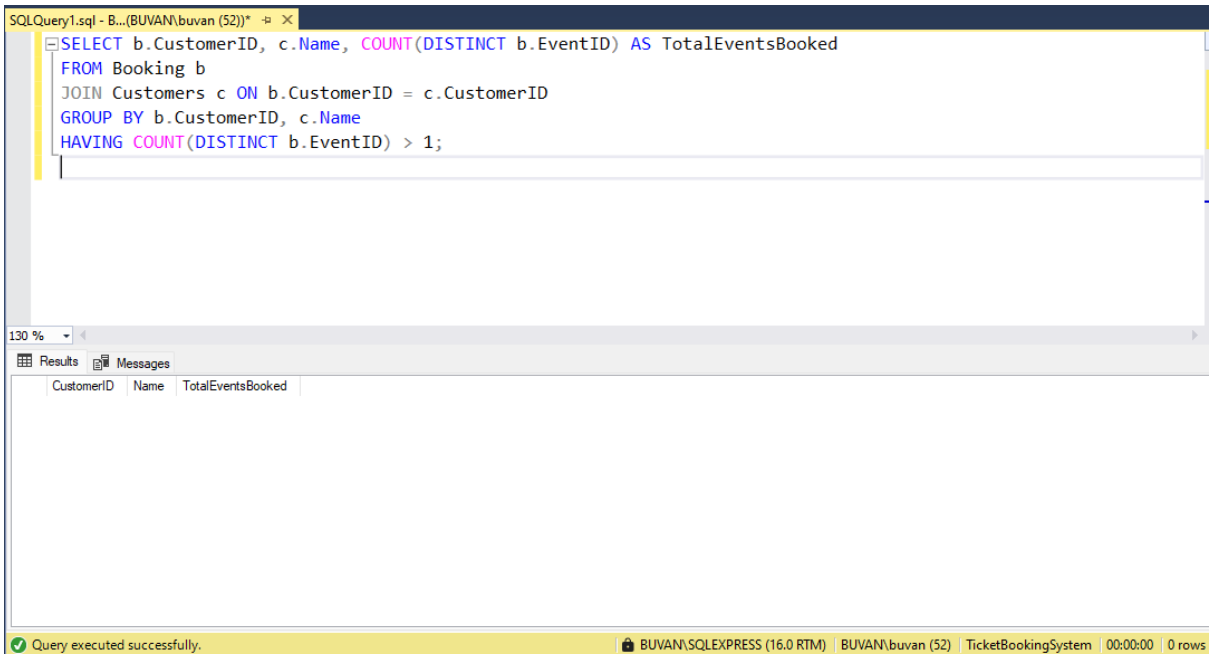
	EventType	TotalTicketsSold
1	Concert	2
2	Other	13
3	Theater	3

Query executed successfully. | BUVAN\SQLEXPRESS (16.0 RTM) | BUVAN\buvan (52) | TicketBookingSystem | 00:00:00 | 3 rows

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



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



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

SQLQuery1.sql - B...(BUVAN\buvan (78))*

```sql
SELECT b.CustomerID, c.Name, SUM(b.SeatsBooked) AS TotalTickets
FROM Booking b
JOIN Customers c ON b.CustomerID = c.CustomerID
WHERE b.BookingDate >= DATEADD(DAY, -30, GETDATE())
GROUP BY b.CustomerID, c.Name;
```

130 %

Results | Messages

	CustomerID	Name	TotalTickets
1	1	MS Gandhi	2
2	4	Leo Das	3
3	6	Veera Raghavan	2
4	8	Michael Rayappan	1
5	9	Sundar Ramasamy	3
6	10	Vetrimaaran	7

Query executed successfully. | BUVAN\SQLEXPRESS (16.0 RTM) | BUVAN\buvan (78) | TicketBookingSystem | 00:00:00 | 6 rows

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

SQLQuery1.sql - B...(BUVAN\buvan (78))*

```sql
SELECT e.Category, v.Name AS VenueName, AVG(e.TicketPrice) AS AvgTicketPrice
FROM Event e
JOIN Venue v ON e.VenueID = v.VenueID
GROUP BY e.Category, v.Name;
```

130 %

Results | Messages

	Category	VenueName	AvgTicketPrice
1	Other	Art Gallery	1200.000000
2	Other	Chepauk Stadium	1000.000000
3	Other	Convention Center	1000.000000
4	Other	Cultural Center	1500.000000
5	Other	Exhibition Hall	2400.000000
6	Theater	Kamala Theatre	500.000000
7	Concert	Le Meridian	2000.000000
8	Other	National Park	1000.000000
9	Sports	Sports Complex	2200.000000
10	Theater	Taj	1000.000000

Query executed successfully. | BUVAN\SQLEXPRESS (16.0 RTM) | BUVAN\buvan (78) | TicketBookingSystem | 00:00:00 | 10 rows

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

SQLQuery1.sql - B...(BUVAN\buvan (78))*

```sql
SELECT b.CustomerID, c.Name, SUM(b.SeatsBooked * e.TicketPrice) AS TotalRevenue
FROM Booking b
JOIN Customers c ON b.CustomerID = c.CustomerID
JOIN Event e ON b.EventID = e.EventID
GROUP BY b.CustomerID, c.Name;
```

130 %

Results | Messages

	CustomerID	Name	TotalRevenue
1	1	MS Gandhi	4000.00
2	4	Leo Das	1500.00
3	6	Veera Raghavan	2400.00
4	8	Michael Rayappan	2200.00
5	9	Sundar Ramasamy	7200.00
6	10	Vetrimaaran	10500.00

Query executed successfully. | BUVAN\SQLEXPRESS (16.0 RTM) | BUVAN\buvan (78) | TicketBookingSystem | 00:00:00 | 6 rows

Task 4:

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

SQLQuery1.sql - B...(BUVAN\buvan (78))*

```sql
SELECT VenueID, Name,
    (SELECT AVG(TicketPrice)
     FROM Event e
     WHERE e.VenueID = v.VenueID) AS AvgTicketPrice
FROM Venue v;
```

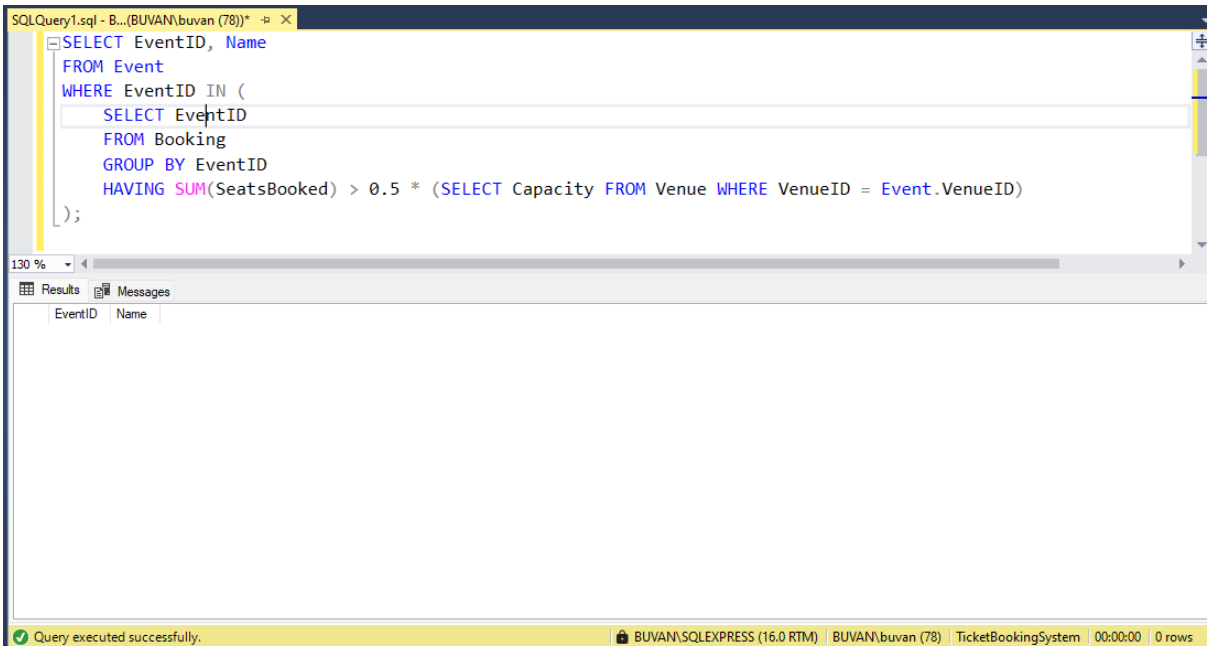
130 %

Results | Messages

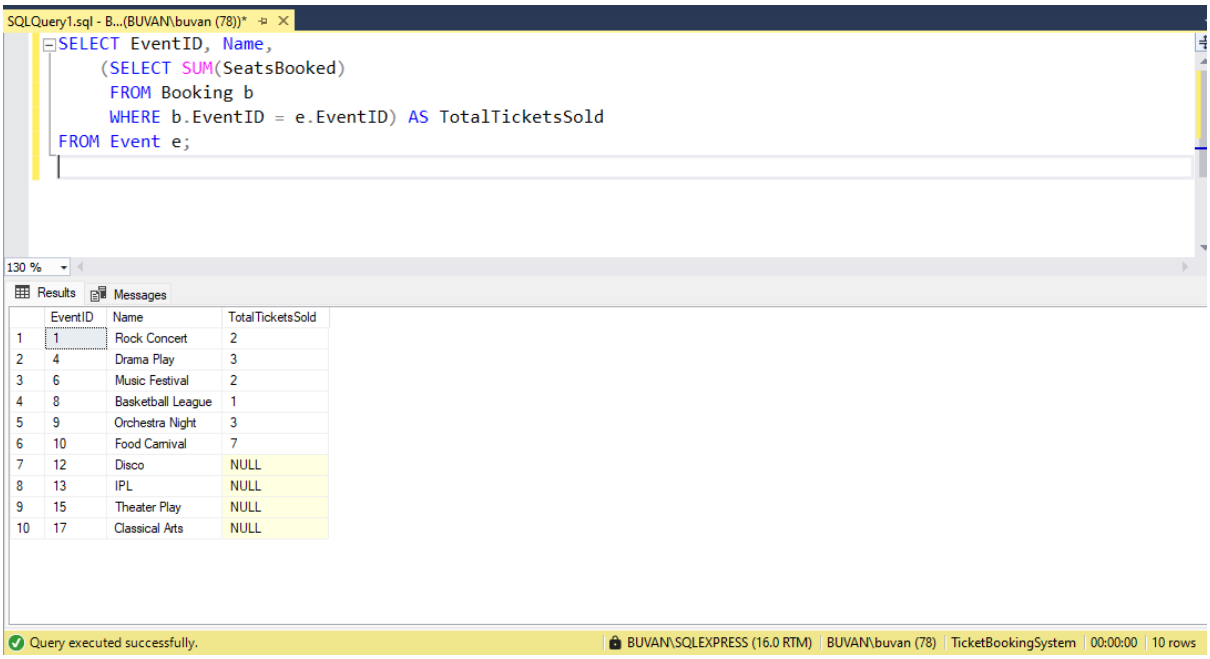
	VenueID	Name	AvgTicketPrice
1	1	Le Meridian	2000.000000
2	2	Taj	1000.000000
3	3	Chepauk Stadium	1000.000000
4	4	Kamala Theatre	500.000000
5	5	Convention Center	1000.000000
6	6	National Park	1000.000000
7	7	Art Gallery	1200.000000
8	8	Sports Complex	2200.000000
9	9	Exhibition Hall	2400.000000
10	10	Cultural Center	1500.000000

Query executed successfully. | BUVAN\SQLEXPRESS (16.0 RTM) | BUVAN\buvan (78) | TicketBookingSystem | 00:00:00 | 10 rows

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

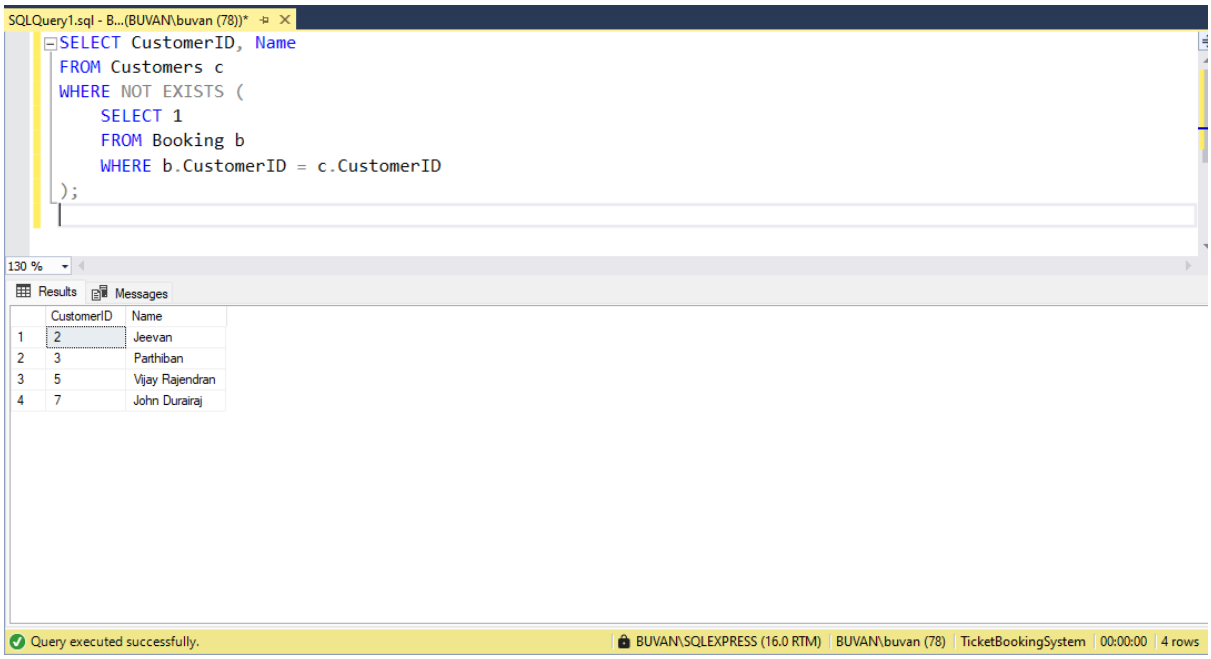


3. Calculate the Total Number of Tickets Sold for Each Event.

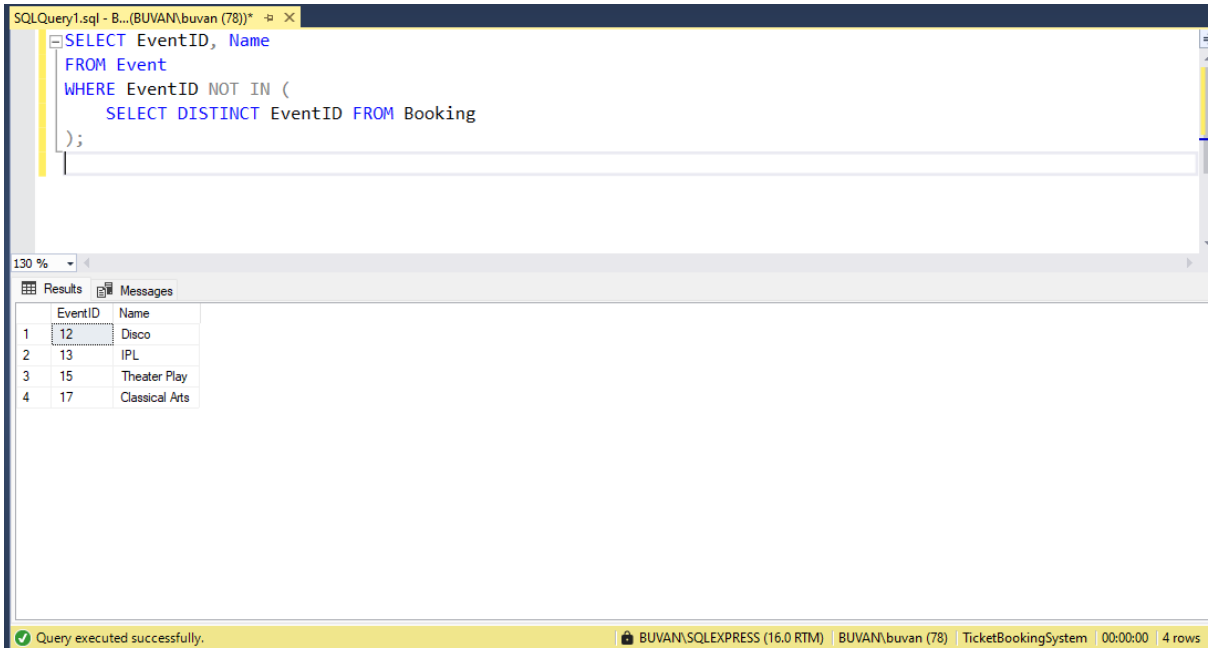


	EventID	Name	TotalTicketsSold
1	1	Rock Concert	2
2	4	Drama Play	3
3	6	Music Festival	2
4	8	Basketball League	1
5	9	Orchestra Night	3
6	10	Food Carnival	7
7	12	Disco	NULL
8	13	IPL	NULL
9	15	Theater Play	NULL
10	17	Classical Arts	NULL

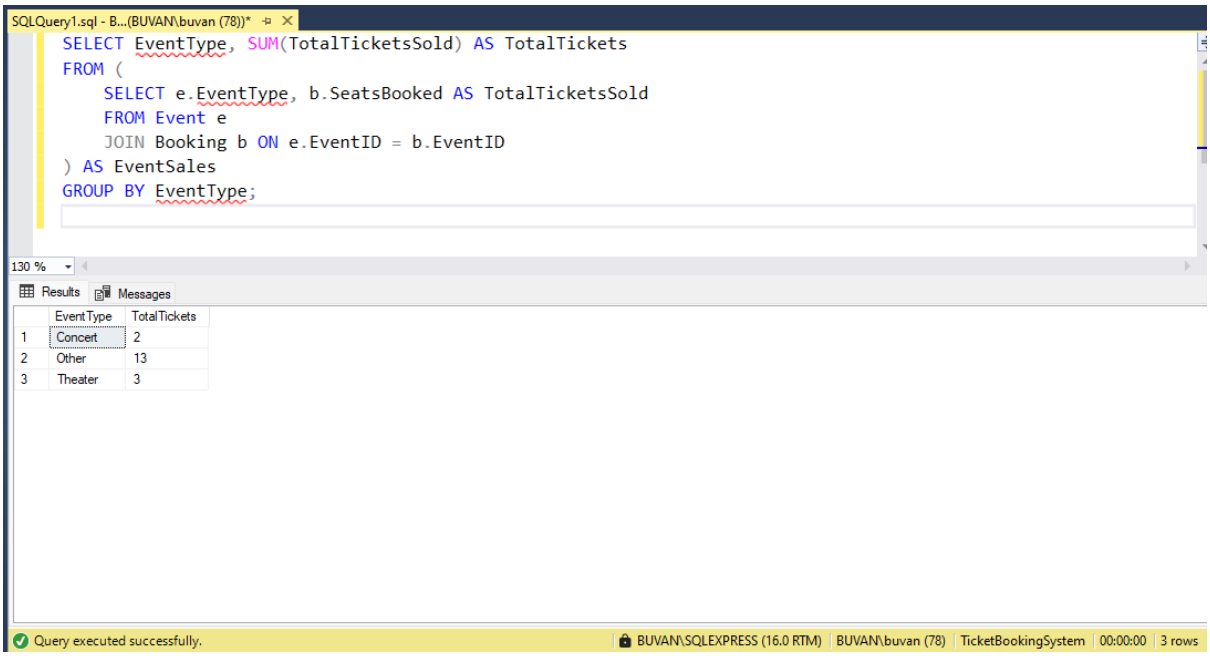
4. Find Users Who Have Not Booked Any Tickets Using a NOT EXISTS Subquery.



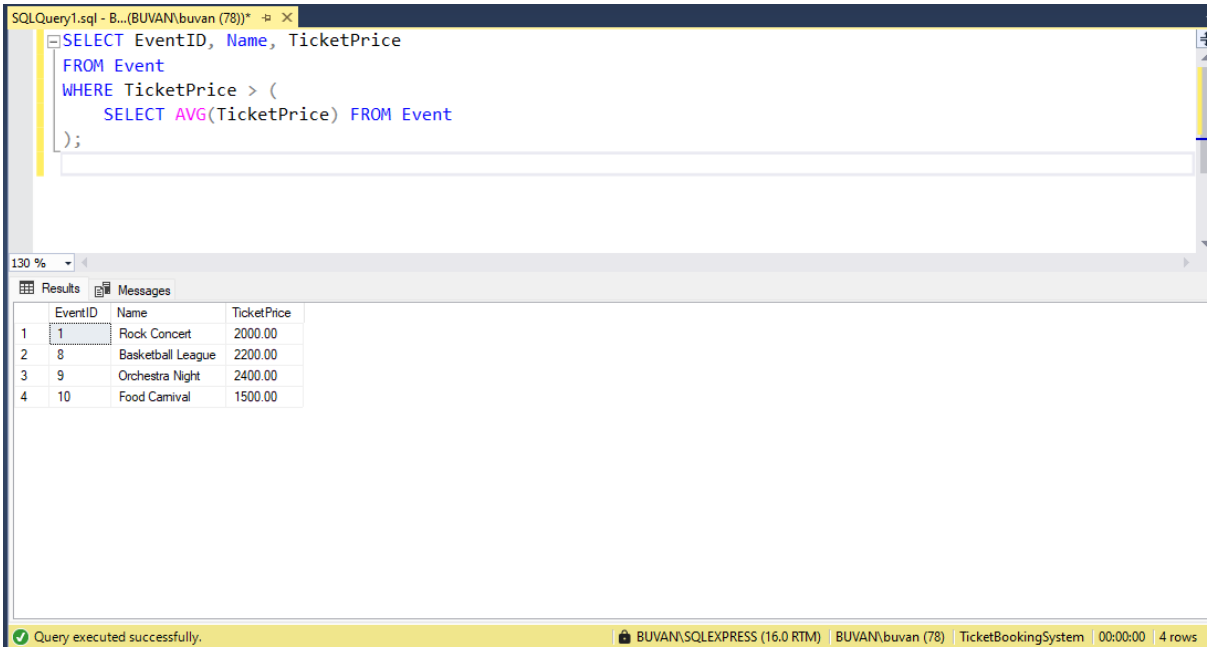
5. List Events with No Ticket Sales Using a NOT IN Subquery.



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



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



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

SQLQuery1.sql - B...(BUVAN\buvan (78))*

```sql
SELECT CustomerID, Name,
    (SELECT SUM(b.SeatsBooked * e.TicketPrice)
     FROM Booking b
     JOIN Event e ON b.EventID = e.EventID
     WHERE b.CustomerID = c.CustomerID) AS TotalRevenue
FROM Customers c;
```

130 %

Results | Messages

	CustomerID	Name	TotalRevenue
1	1	MS Gandhi	4000.00
2	2	Jeevan	NULL
3	3	Parthiban	NULL
4	4	Leo Das	1500.00
5	5	Vijay Rajendran	NULL
6	6	Veera Raghavan	2400.00
7	7	John Durairaj	NULL
8	8	Michael Rayappan	2200.00
9	9	Sundar Ramasamy	7200.00
10	10	Vetrimaaran	10500.00

Query executed successfully. | BUVAN\SQLEXPRESS (16.0 RTM) | BUVAN\buvan (78) | TicketBookingSystem | 00:00:00 | 10 rows

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

SQLQuery1.sql - B...(BUVAN\buvan (78))*

```sql
FROM Customers
WHERE CustomerID IN (
    SELECT DISTINCT CustomerID
    FROM Booking
    WHERE EventID IN (
        SELECT EventID FROM Event WHERE VenueID = 2
    )
);
```

130 %

Results | Messages

CustomerID	Name

Query executed successfully. | BUVAN\SQLEXPRESS (16.0 RTM) | BUVAN\buvan (78) | TicketBookingSystem | 00:00:00 | 0 rows

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

SQLQuery1.sql - B...(BUVAN\buvan (78))*

```sql
SELECT Category,
    (SELECT SUM(SeatsBooked)
     FROM Booking
     WHERE EventID IN (
         SELECT EventID FROM Event WHERE Category = e.Category
     )) AS TotalTicketsSold
FROM Event e
GROUP BY Category;
```

130 %

Results | Messages

	Category	TotalTicketsSold
1	Concert	2
2	Other	12
3	Sports	1
4	Theater	3

Query executed successfully. | BUVAN\SQLEXPRESS (16.0 RTM) | BUVAN\buvan (78) | TicketBookingSystem | 00:00:00 | 4 rows

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

SQLQuery1.sql - B...(BUVAN\buvan (78))*

```sql
SELECT FORMAT(b.BookingDate, 'yyyy-MM') AS BookingMonth, c.Name, SUM(b.SeatsBooked) AS TotalTickets
FROM Customers c
JOIN Booking b ON c.CustomerID = b.CustomerID
GROUP BY FORMAT(b.BookingDate, 'yyyy-MM'), c.Name;
```

130 %

Results | Messages

	BookingMonth	Name	TotalTickets
1	2025-03	Leo Das	3
2	2025-03	Michael Rayappan	1
3	2025-03	MS Gandhi	2
4	2025-03	Sundar Ramasamy	3
5	2025-03	Veera Raghavan	2
6	2025-03	Vetrimaaran	7

Query executed successfully. | BUVAN\SQLEXPRESS (16.0 RTM) | BUVAN\buvan (78) | TicketBookingSystem | 00:00:00 | 6 rows

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

SQLQuery1.sql - B...(BUVAN\buvan (78))*

```sql
SELECT VenueID, Name,
    (SELECT AVG(TicketPrice)
     FROM Event e
     WHERE e.VenueID = v.VenueID) AS AvgTicketPrice
FROM Venue v;
```

130 %

Results | Messages

	VenueID	Name	AvgTicketPrice
1	1	Le Meridian	2000.000000
2	2	Taj	1000.000000
3	3	Chepauk Stadium	1000.000000
4	4	Kamala Theatre	500.000000
5	5	Convention Center	1000.000000
6	6	National Park	1000.000000
7	7	Art Gallery	1200.000000
8	8	Sports Complex	2200.000000
9	9	Exhibition Hall	2400.000000
10	10	Cultural Center	1500.000000

Query executed successfully. | BUVAN\SQLEXPRESS (16.0 RTM) | BUVAN\buvan (78) | TicketBookingSystem | 00:00:00 | 10 rows