**Ticket Booking System**

**TASK-1**

**1. Create the database named "TicketBookingSystem".**

create database TicketBookingSystem

use TicketBookingSystem

**2. Write SQL scripts to create the mentioned tables**

--To create Venue table

create table tbl\_venue (

venue\_id int primary key,

venue\_name varchar(100),

address varchar(100)

);

--To create Event table

create table tbl\_event (

event\_id int primary key,

event\_name varchar(100),

event\_date date,

event\_time time,

venue\_id int foreign key references tbl\_venue(venue\_id),

total\_seats int,

available\_seats int,

ticket\_price decimal(10,2),

event\_type varchar(50) check (event\_type in ('movie', 'sports', 'concert'))

);

--To create Customer table

create table tbl\_customer (

customer\_id int primary key,

customer\_name varchar(50),

email varchar(50),

phone\_number varchar(20)

)

--To create Booking table

create table tbl\_booking (

booking\_id int primary key,

customer\_id int foreign key references tbl\_customer(customer\_id),

event\_id int foreign key references tbl\_event(event\_id),

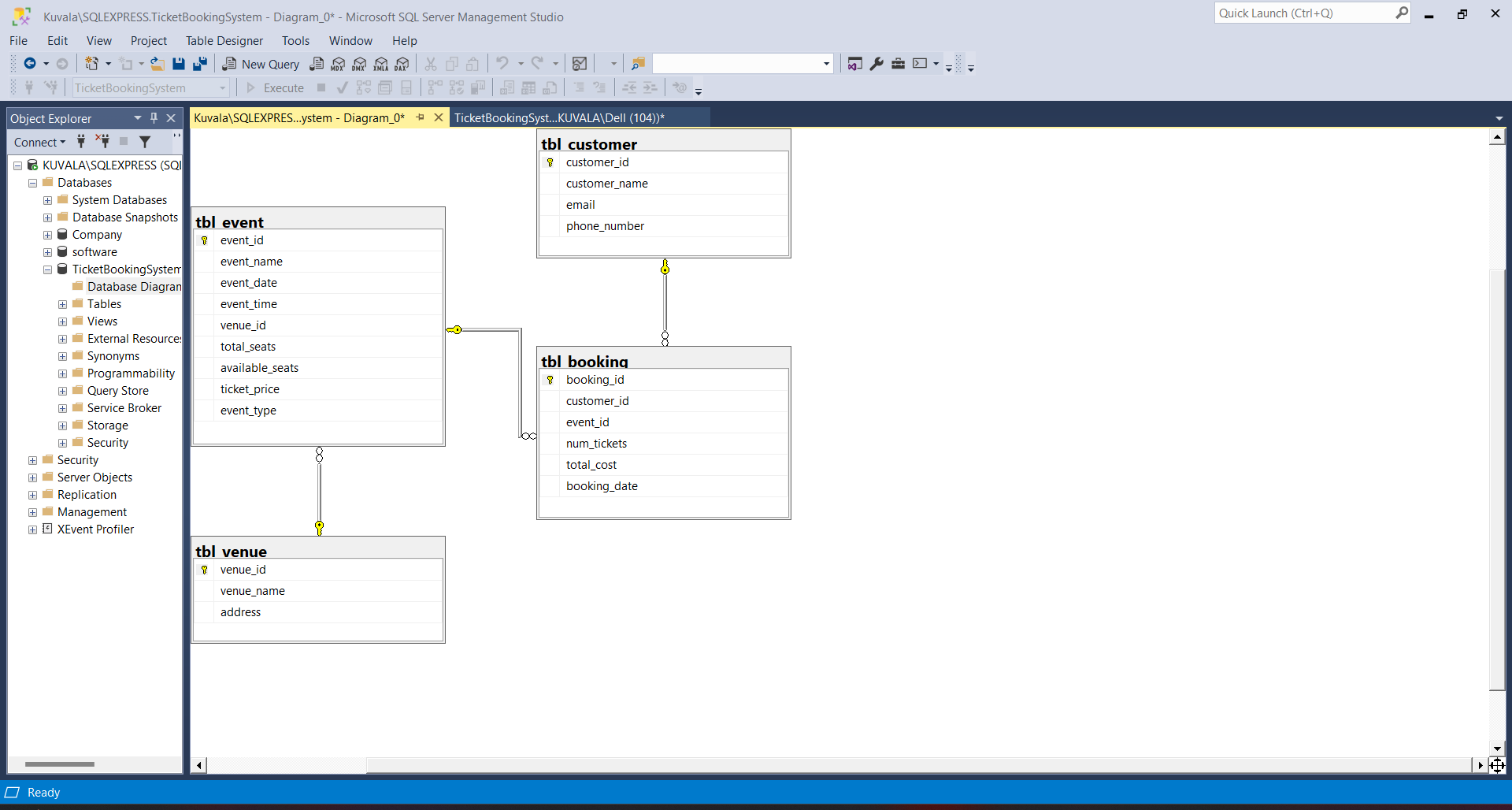
num\_tickets int,

total\_cost decimal(10,2),

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 tbl\_event

add constraint fk\_event\_venue

foreign key (venue\_id) references tbl\_venue(venue\_id);

alter table tbl\_booking

add constraint fk\_booking\_customer

foreign key (customer\_id) references tbl\_customer(customer\_id);

alter table tbl\_booking

add constraint fk\_booking\_event

foreign key (event\_id) references tbl\_event(event\_id);

**TASK-2**

**Q1.Write a SQL query to insert at least 10 sample records into each table.**

**--Venue table**

insert into tbl\_venue (venue\_id, venue\_name, address) values

(1, 'City Hall', 'MG Road, Bengaluru'),

(2, 'Grand Arena', 'Marine Drive, Mumbai'),

(3, 'Sunset Park', 'Gachibowli, Hyderabad'),

(4, 'Ocean Center', 'RK Beach, Visakhapatnam'),

(5, 'Green Garden', 'Lodhi Road, New Delhi'),

(6, 'Tech Hub', 'Hinjewadi, Pune'),

(7, 'Royal Theatre', 'Park Street, Kolkata'),

(8, 'Sky Dome', 'Anna Salai, Chennai'),

(9, 'Cultural Center', 'Sector 17, Chandigarh'),

(10, 'Mega Grounds', 'Bistupur, Jamshedpur');

**--Event table**

insert into tbl\_event (event\_id, event\_name, event\_date, event\_time, venue\_id, total\_seats, available\_seats, ticket\_price, event\_type) values

(101, 'Rock Concert', '2025-04-10', '18:00', 2, 500, 300, 50.00, 'Concert'),

(102, 'Football Match', '2025-05-01', '16:00', 10, 1000, 800, 75.00, 'Sports'),

(103, 'Movie Night', '2025-03-25', '20:00', 1, 200, 150, 15.00, 'Movie'),

(104, 'Drama Play', '2025-04-15', '19:00', 7, 300, 100, 25.00, 'Concert'),

(105, 'Cricket Match', '2025-06-05', '14:00', 10, 1500, 1000, 80.00, 'Sports'),

(106, 'Jazz Night', '2025-04-20', '21:00', 4, 250, 50, 40.00, 'Concert'),

(107, 'Tech Expo', '2025-05-15', '10:00', 6, 600, 400, 20.00, 'Concert'),

(108, 'Indie Film', '2025-03-30', '18:30', 1, 150, 50, 12.00, 'Movie'),

(109, 'Basketball Game', '2025-04-22', '17:00', 10, 900, 600, 70.00, 'Sports'),

(110, 'Cultural Show', '2025-05-10', '19:30', 9, 400, 300, 30.00, 'Concert');

**--Customer table**

insert into tbl\_customer (customer\_id, customer\_name, email, phone\_number) VALUES

(1, 'Dharshini', 'dharshini@gmail.com', '6655443322'),

(2, 'Akshay', 'akshay@gmail.com', '8765432109'),

(3, 'Naveen', 'naveen@gmail.com', '6543210987'),

(4, 'Priya', 'priya@gmail.com', '7432109876'),

(5, 'Dhanashree', 'dhanashree@gmail.com', '7321098765'),

(6, 'Sharon Benny', 'sharon@gmail.com', '7210987654'),

(7, 'Karnika', 'karnika@gmail.com', '0109876543'),

(8, 'Abdhul', 'abdhul@gmail.com', '9098765432'),

(9, 'Nilofur', 'nilofur@gmail.com', '9988776655'),

(10, 'Dheva', 'dheva@gmail.com', '8877665544');

**--Booking table**

insert into tbl\_booking (booking\_id, customer\_id, event\_id, num\_tickets, total\_cost, booking\_date) values

(201, 1, 101, 2, 100.00, '2025-03-01'),

(202, 2, 102, 4, 300.00, '2025-03-05'),

(203, 3, 103, 3, 45.00, '2025-03-10'),

(204, 4, 104, 2, 50.00, '2025-03-12'),

(205, 5, 105, 5, 400.00, '2025-03-15'),

(206, 6, 106, 1, 40.00, '2025-03-18'),

(207, 7, 107, 2, 40.00, '2025-03-20'),

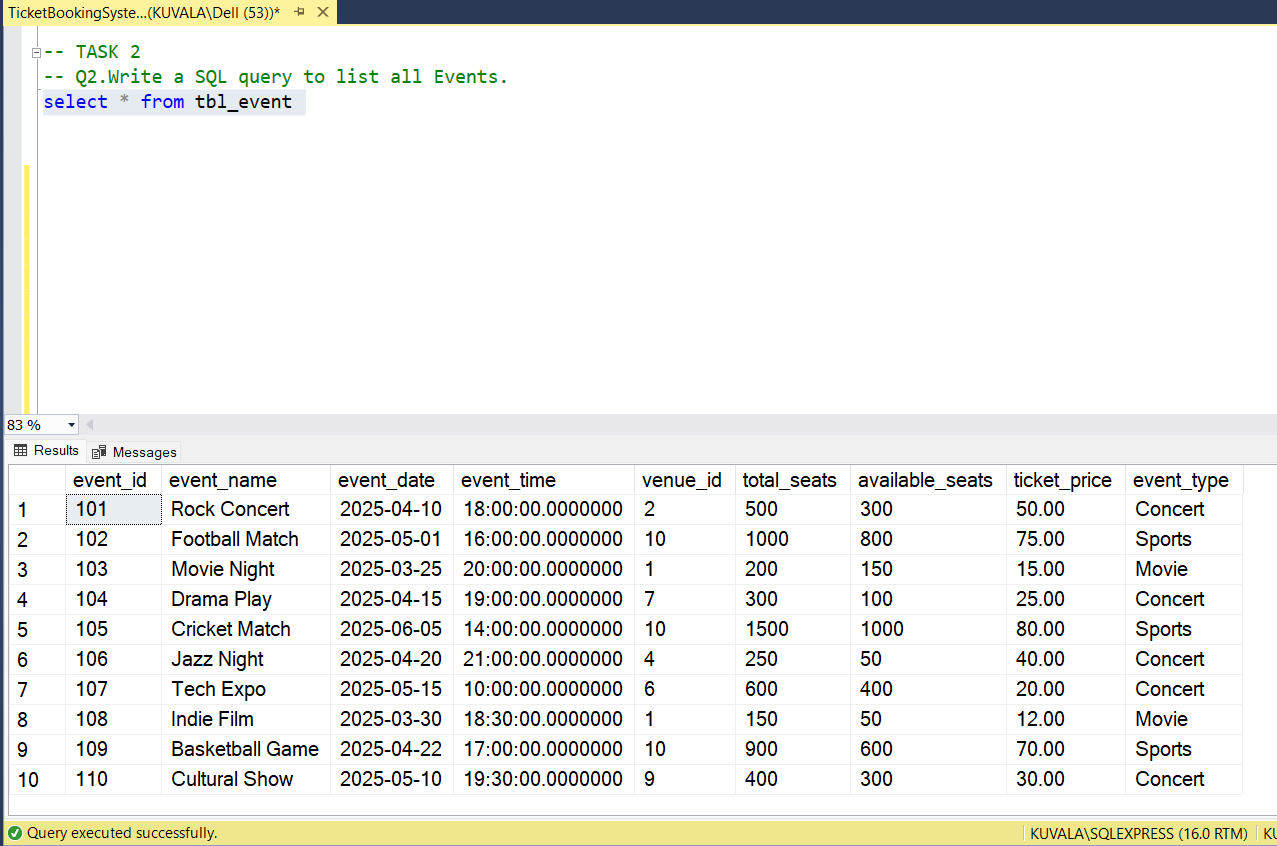
(208, 8, 108, 4, 48.00, '2025-03-22'),

(209, 9, 109, 3, 210.00, '2025-03-24'),

(210, 10, 110, 2, 60.00, '2025-03-26');

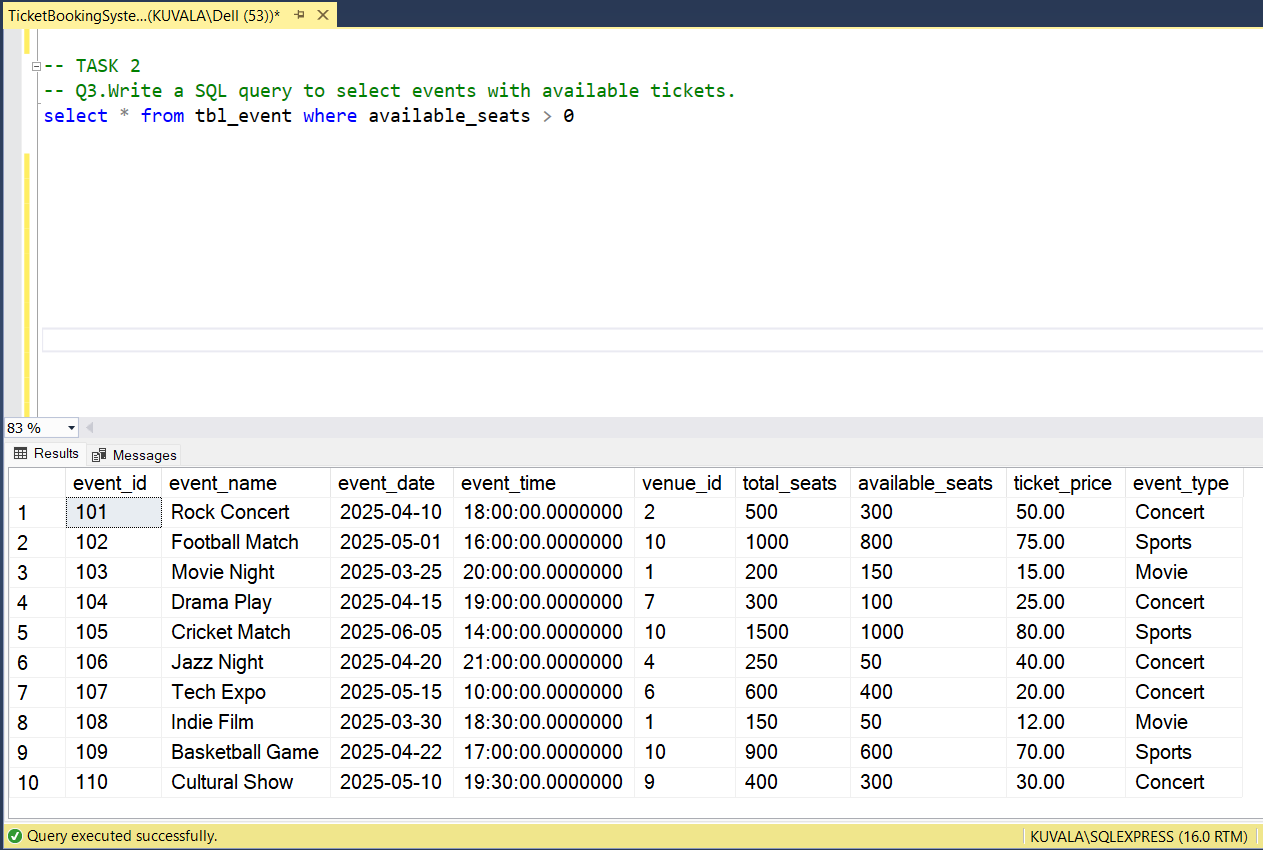
**2.Write a SQL query to list all Events.**

**select \* from tbl\_event**



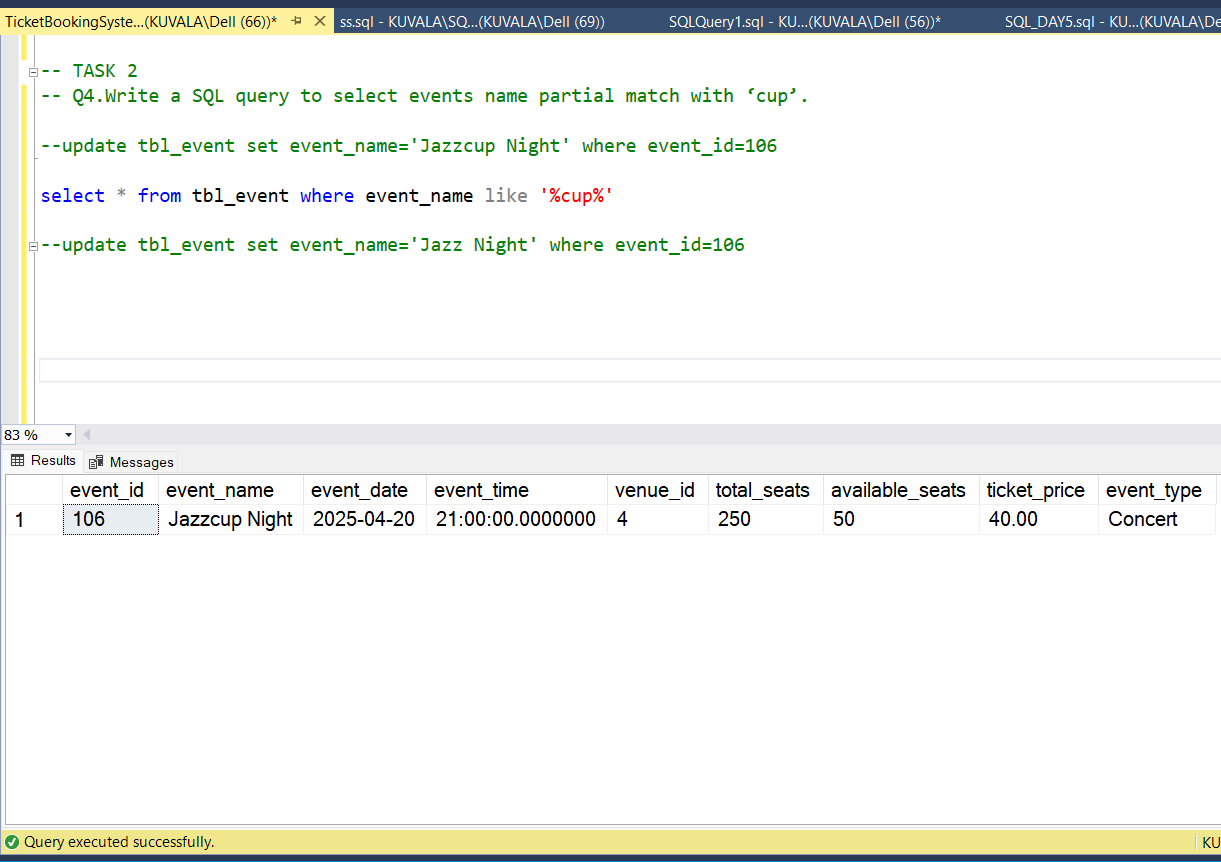
**3.Write a SQL query to select events with available tickets.**

**select \* from tbl\_event where available\_seats > 0**



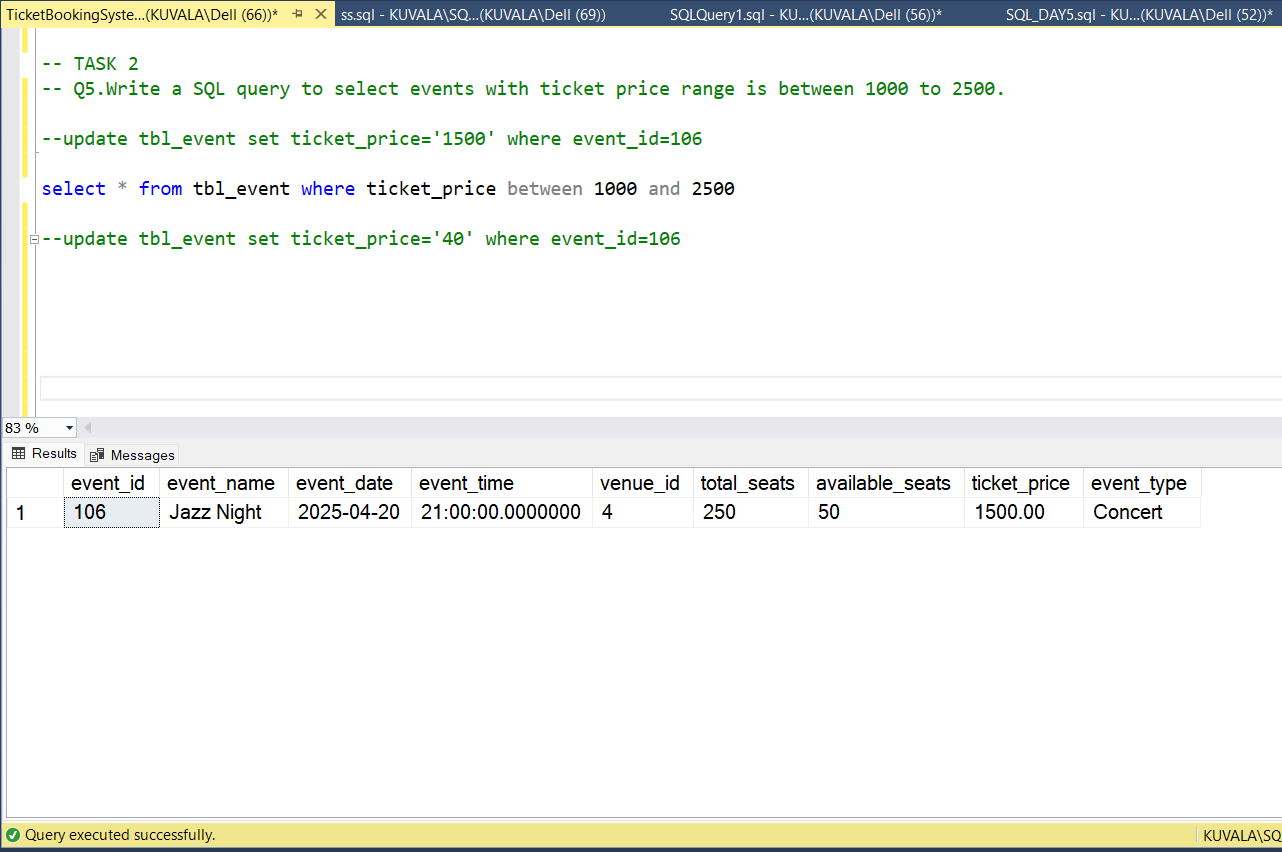
**4.Write a SQL query to select events name partial match with ‘cup’.**

**select \* from tbl\_event where event\_name like '%cup%'**



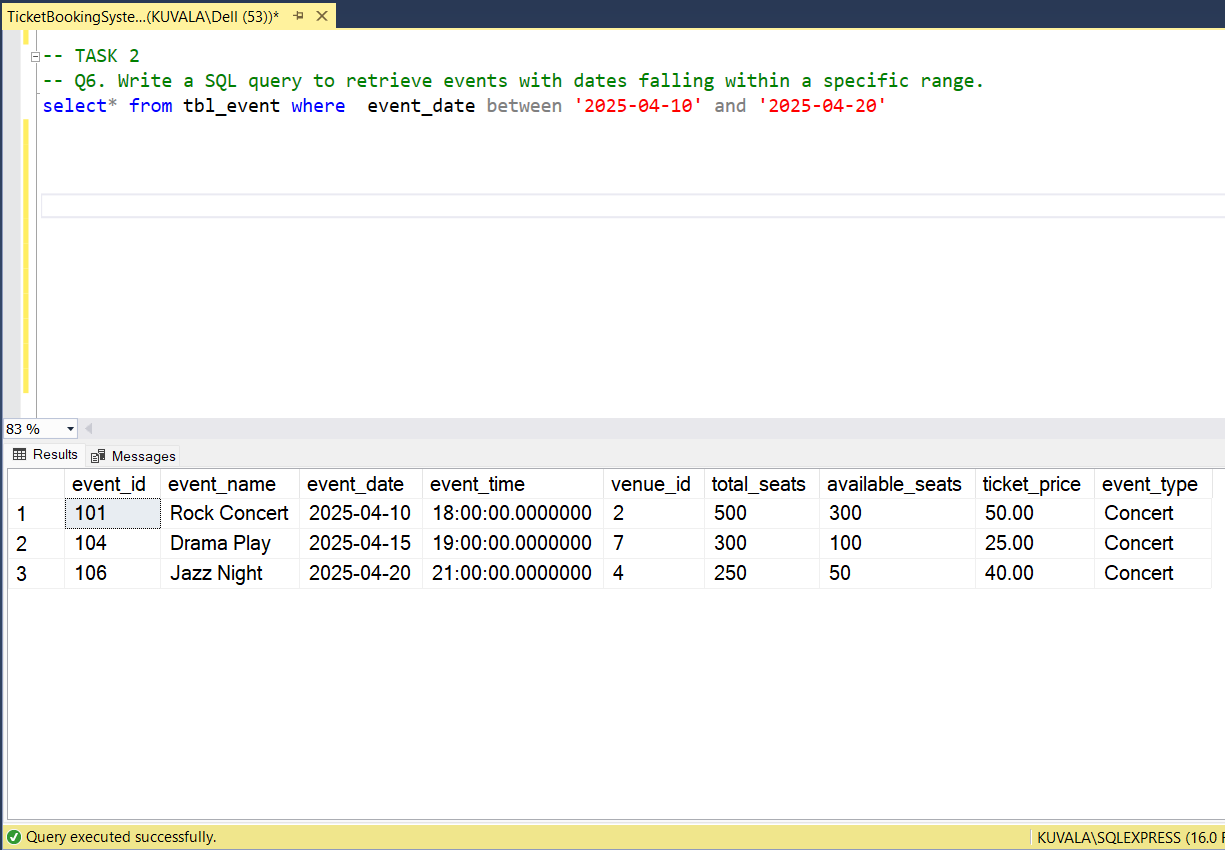
**5. Write a SQL query to select events with ticket price range is between 1000 to 2500.**

**select \* from tbl\_event where ticket\_price between 1000 and 2500**

****

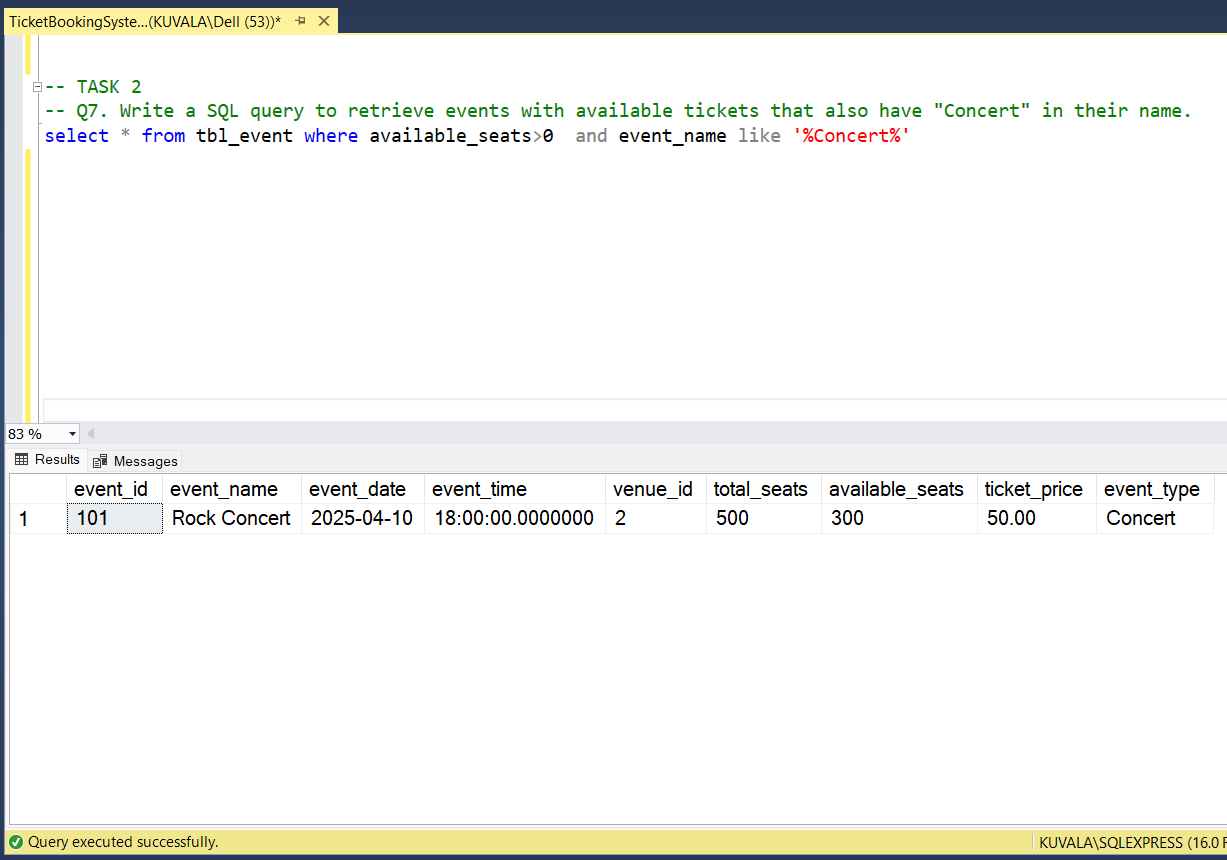
**6. Write a SQL query to retrieve events with dates falling within a specific range.**

**select\* from tbl\_event where event\_date between '2025-04-10' and '2025-04-20'**

****

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

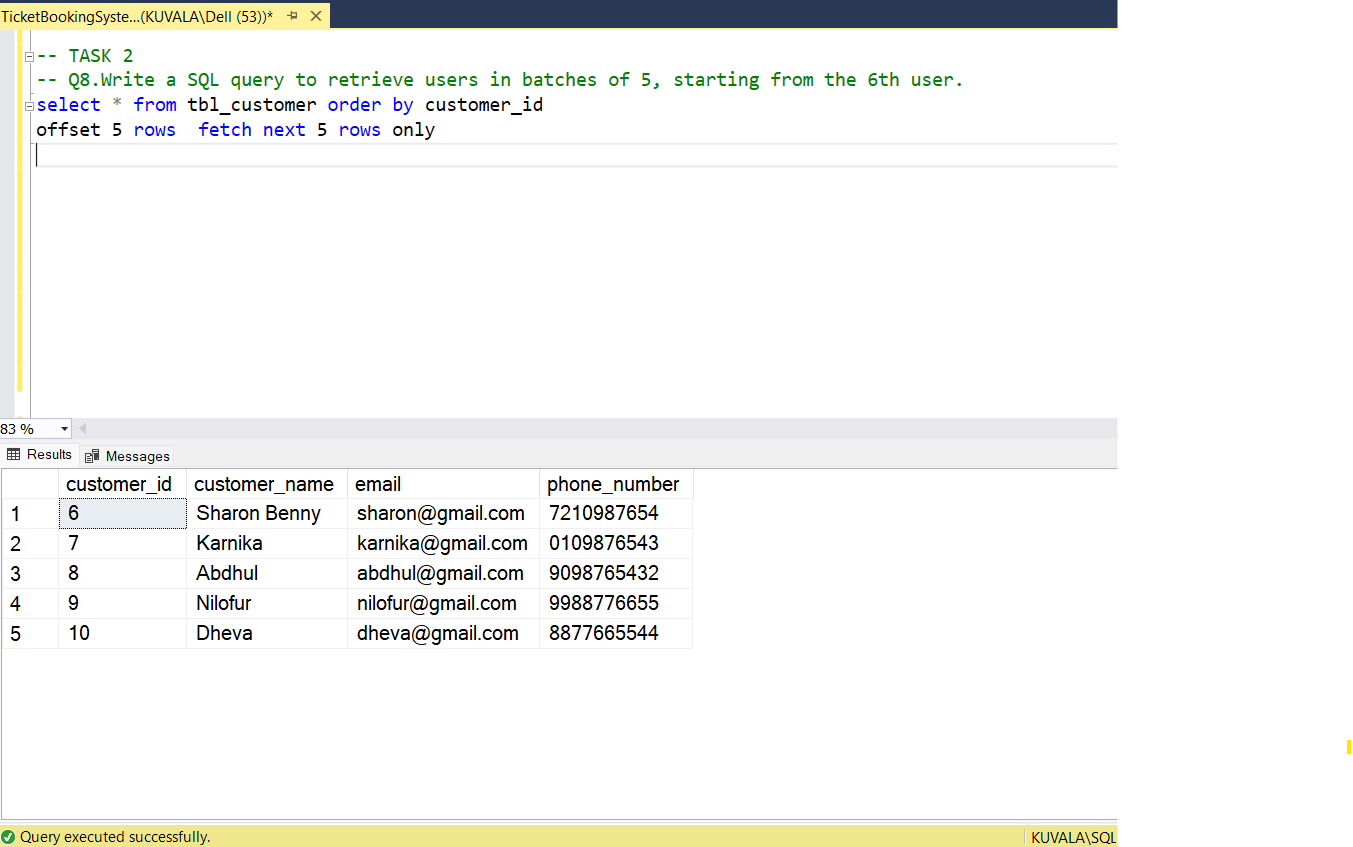
**select \* from tbl\_event where available\_seats>0 and event\_name like '%Concert%'**

****

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

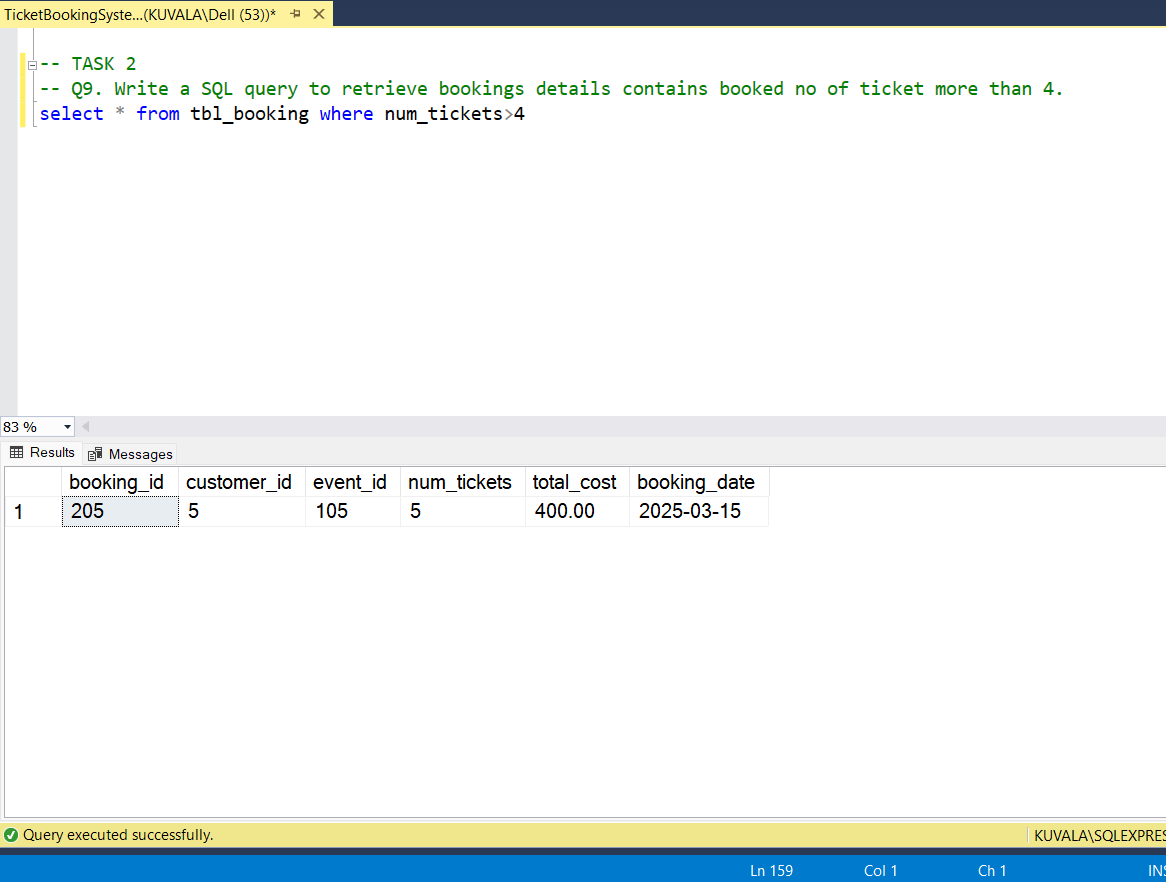
**select \* from tbl\_customer order by customer\_id**

**offset 5 rows fetch next 5 rows only**

****

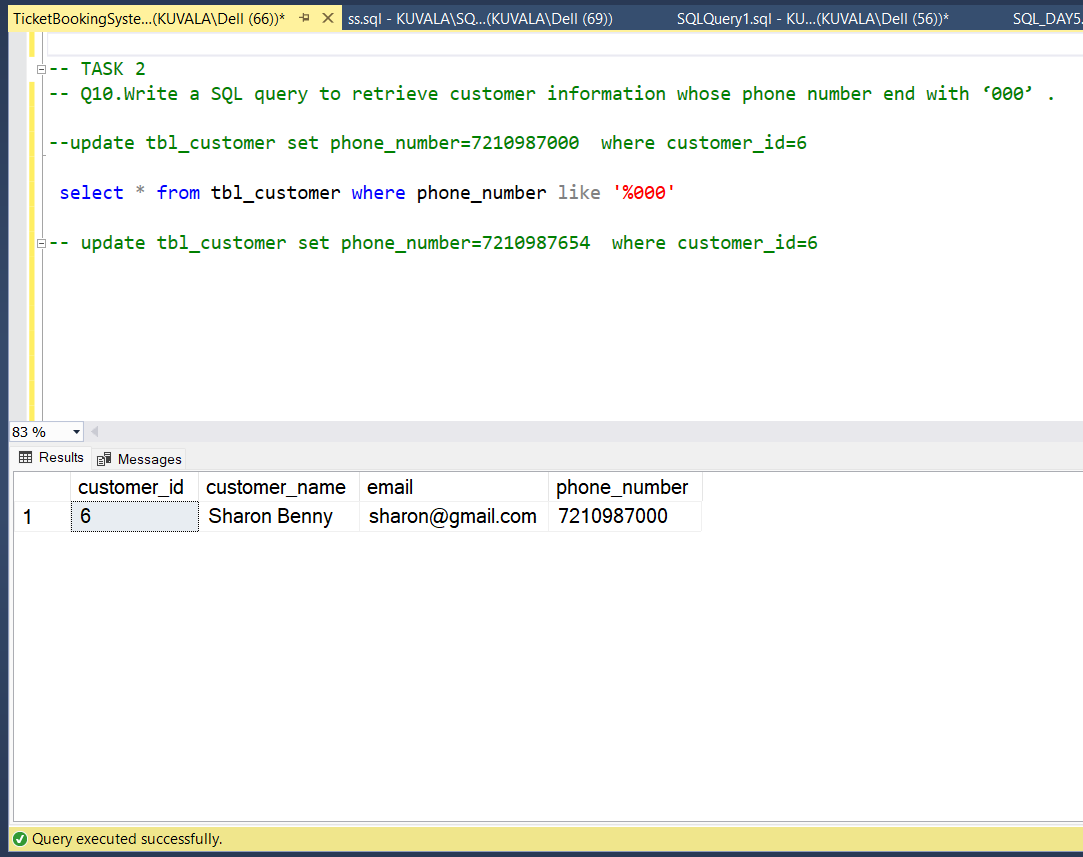
**9. Write a SQL query to retrieve bookings details contains booked no of ticket more than 4.**

**select \* from tbl\_booking where num\_tickets>4**

****

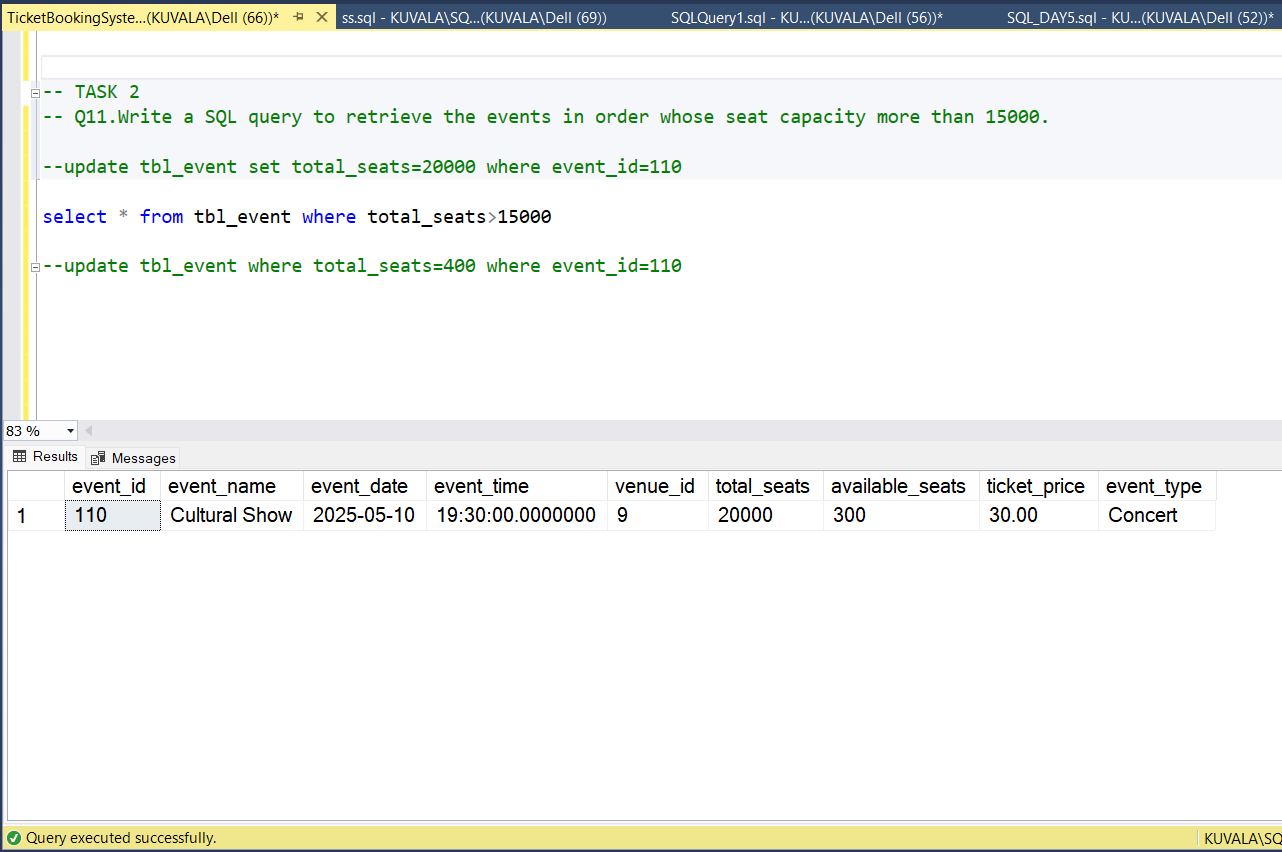
**10.Write a SQL query to retrieve customer information whose phone number end with ‘000’ .**

**select \* from tbl\_customer where phone\_number like '%000'**

****

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

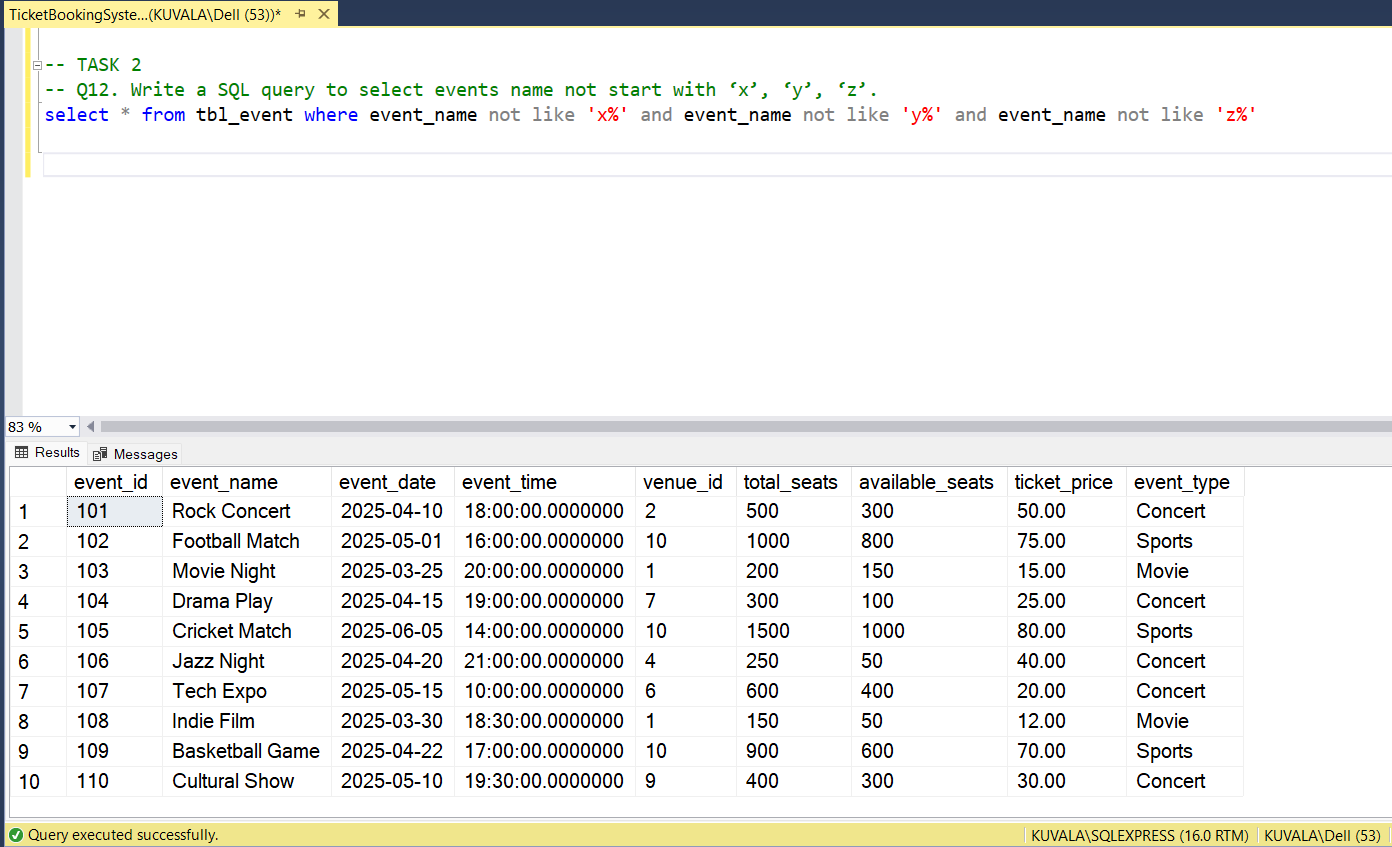
**select \* from tbl\_event where total\_seats>15000**

****

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

**select \* from tbl\_event where event\_name not like 'x%' and event\_name not like 'y%' and**

**event\_name not like 'z%'**

****

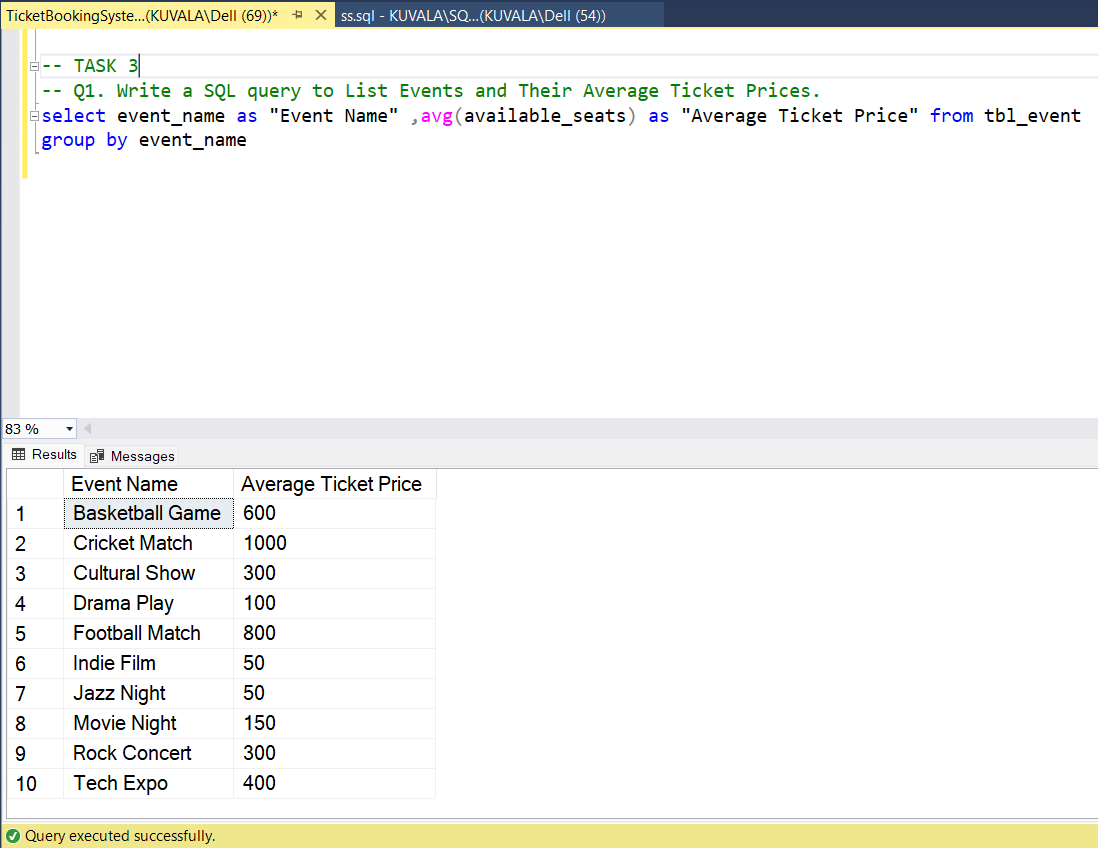
**TASK-3**

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

**select event\_name as "Event Name" ,avg(available\_seats) as "Average Ticket Price"**

**from tbl\_event**

**group by event\_name**

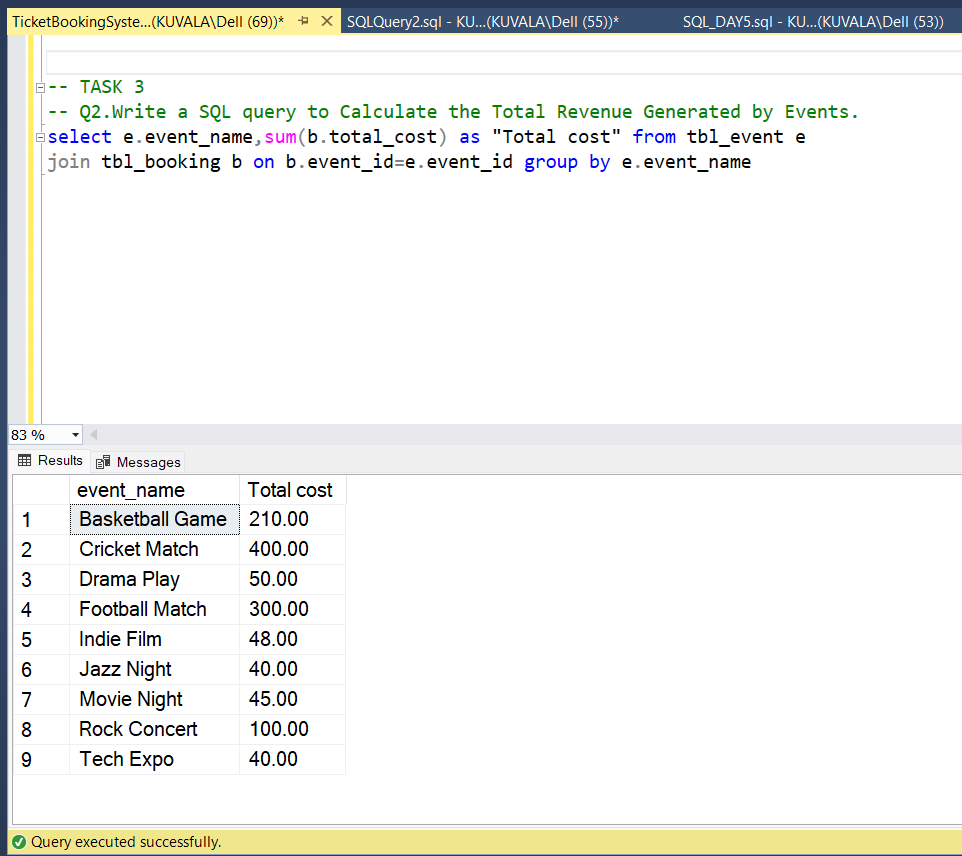
****

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

**select e.event\_name,sum(b.total\_cost) as "Total cost" from tbl\_event e**

**join tbl\_booking b on b.event\_id=e.event\_id**

**group by e.event\_name**

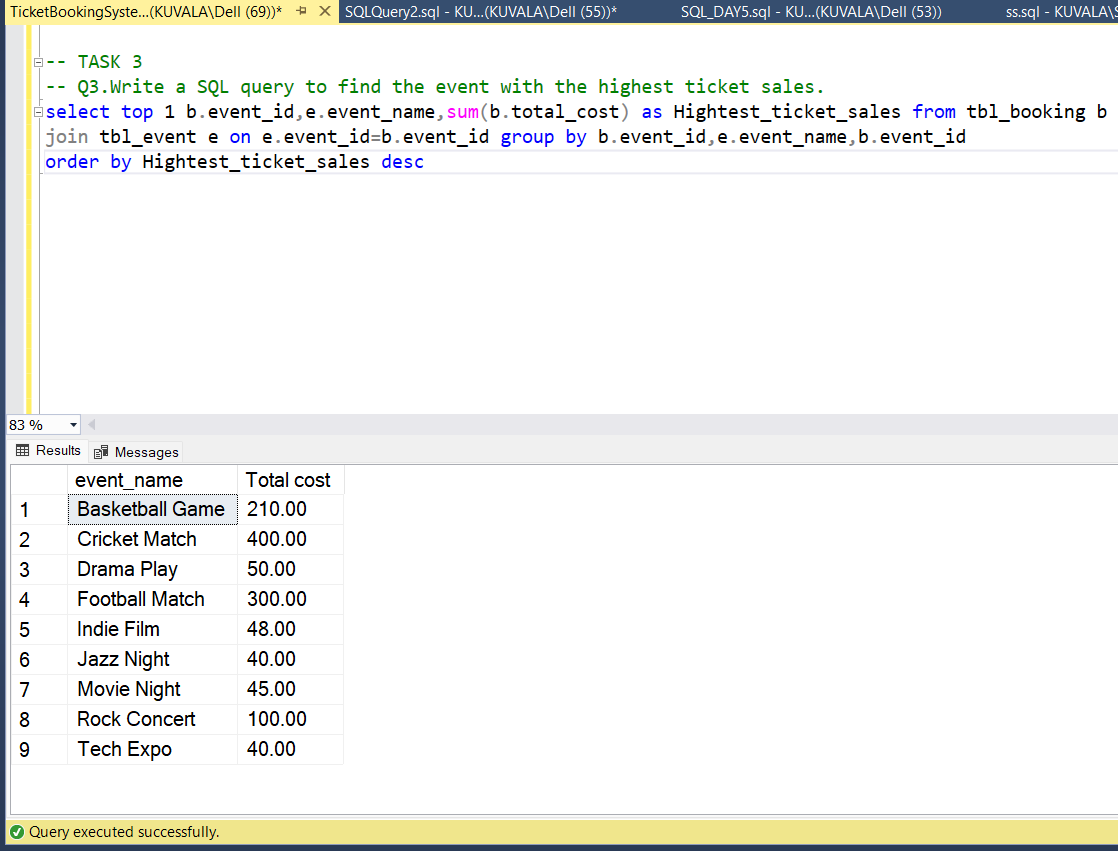
****

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

**select top 1 b.event\_id,e.event\_name,sum(b.total\_cost) as Hightest\_ticket\_sales from tbl\_booking b**

**join tbl\_event e on e.event\_id=b.event\_id group by b.event\_id,e.event\_name,b.event\_id**

**order by Hightest\_ticket\_sales desc**

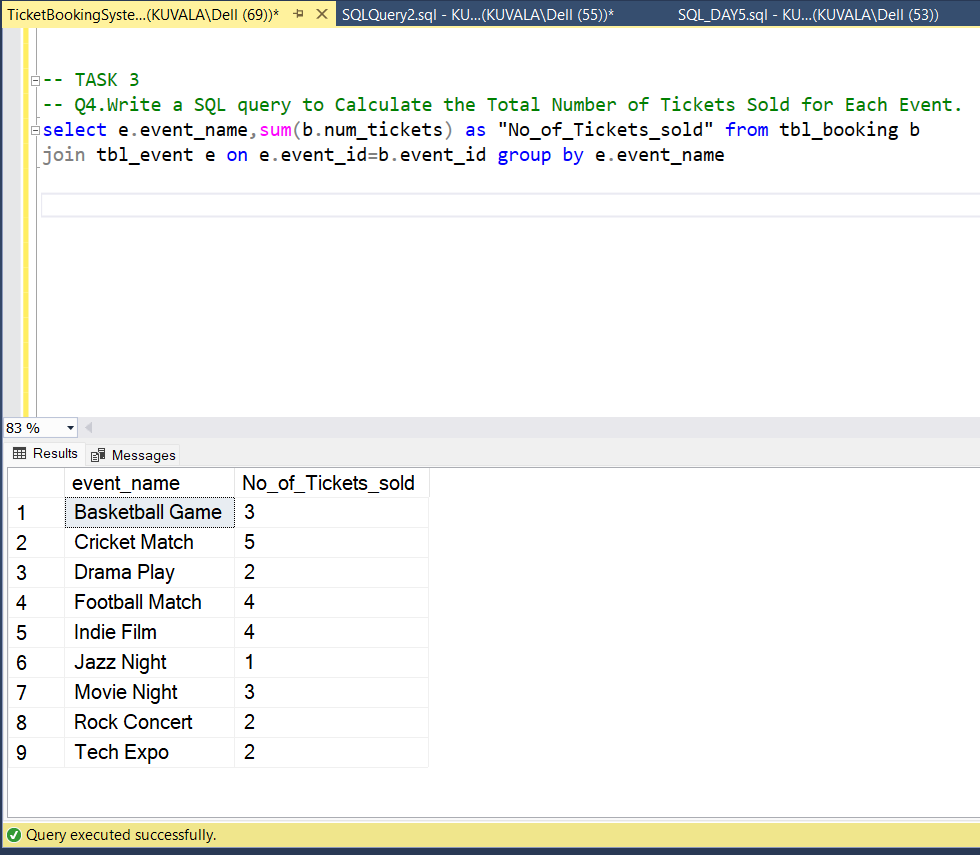
****

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

**select e.event\_name,sum(b.num\_tickets) as "No\_of\_Tickets\_sold" from tbl\_booking b**

**join tbl\_event e on e.event\_id=b.event\_id**

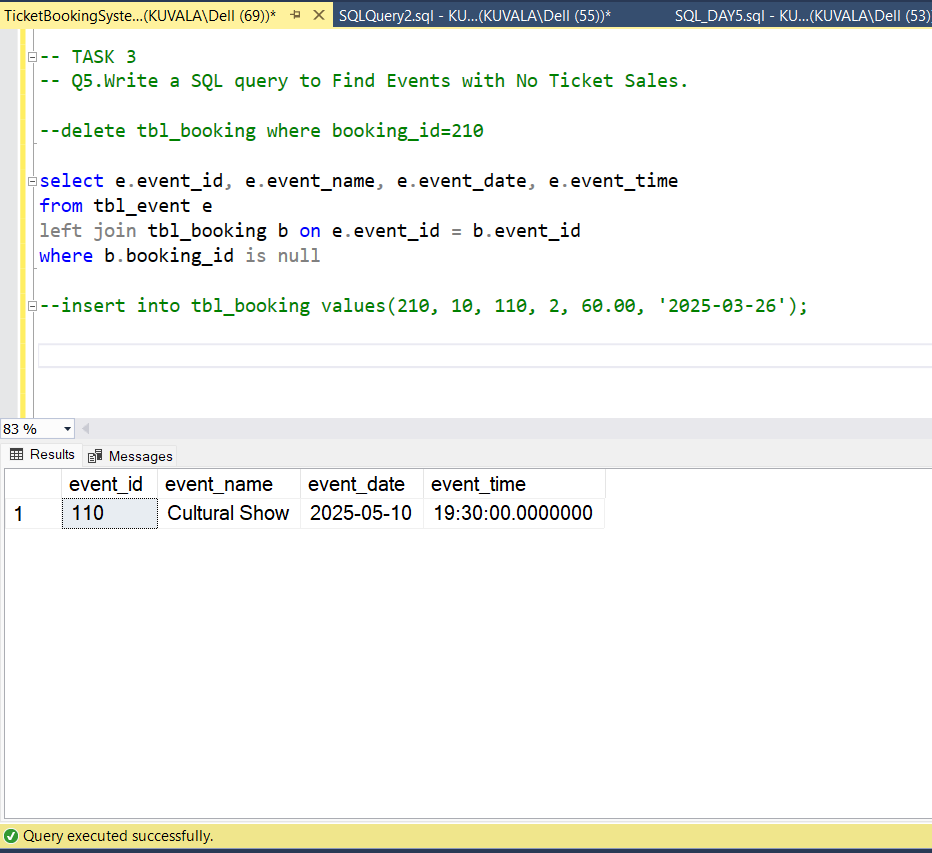
**group by e.event\_name**

****

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

**select e.event\_id, e.event\_name, e.event\_date, e.event\_time from tbl\_event e**

**left join tbl\_booking b on e.event\_id = b.event\_id where b.booking\_id is null**

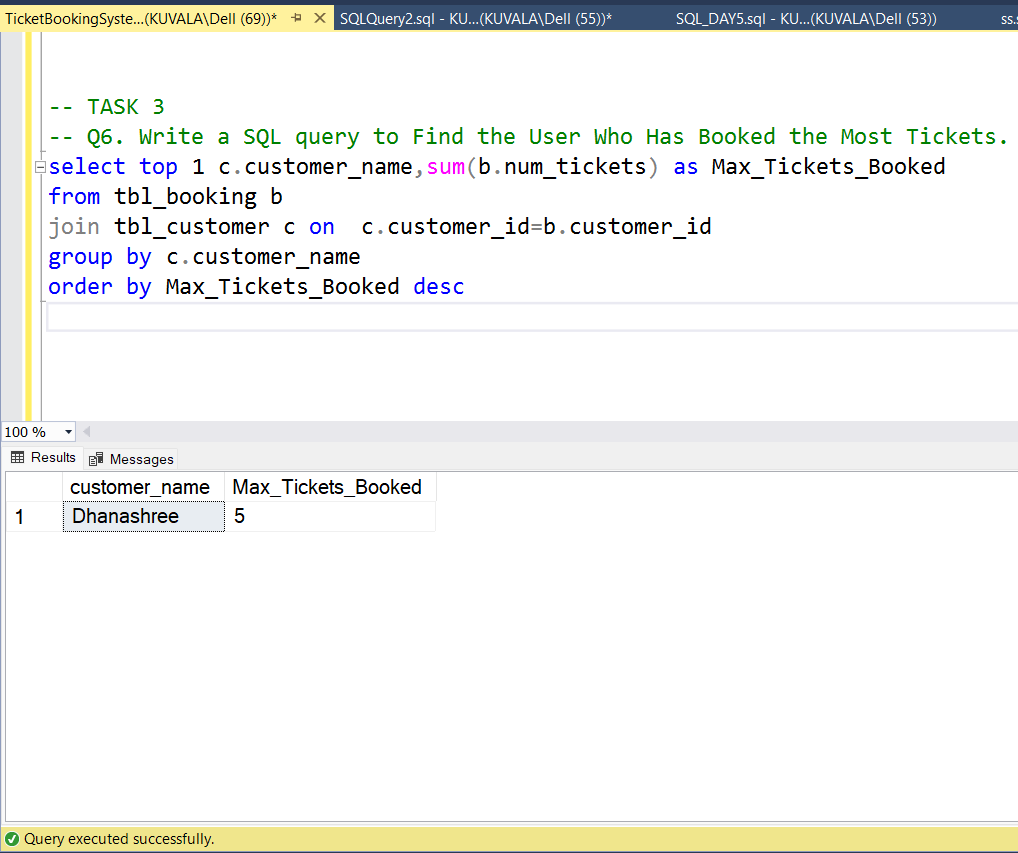
****

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

**select top 1 c.customer\_name,sum(b.num\_tickets) as Max\_Tickets\_Booked from tbl\_booking b**

**join tbl\_customer c on c.customer\_id=b.customer\_id group by c.customer\_name**

**order by Max\_Tickets\_Booked desc**

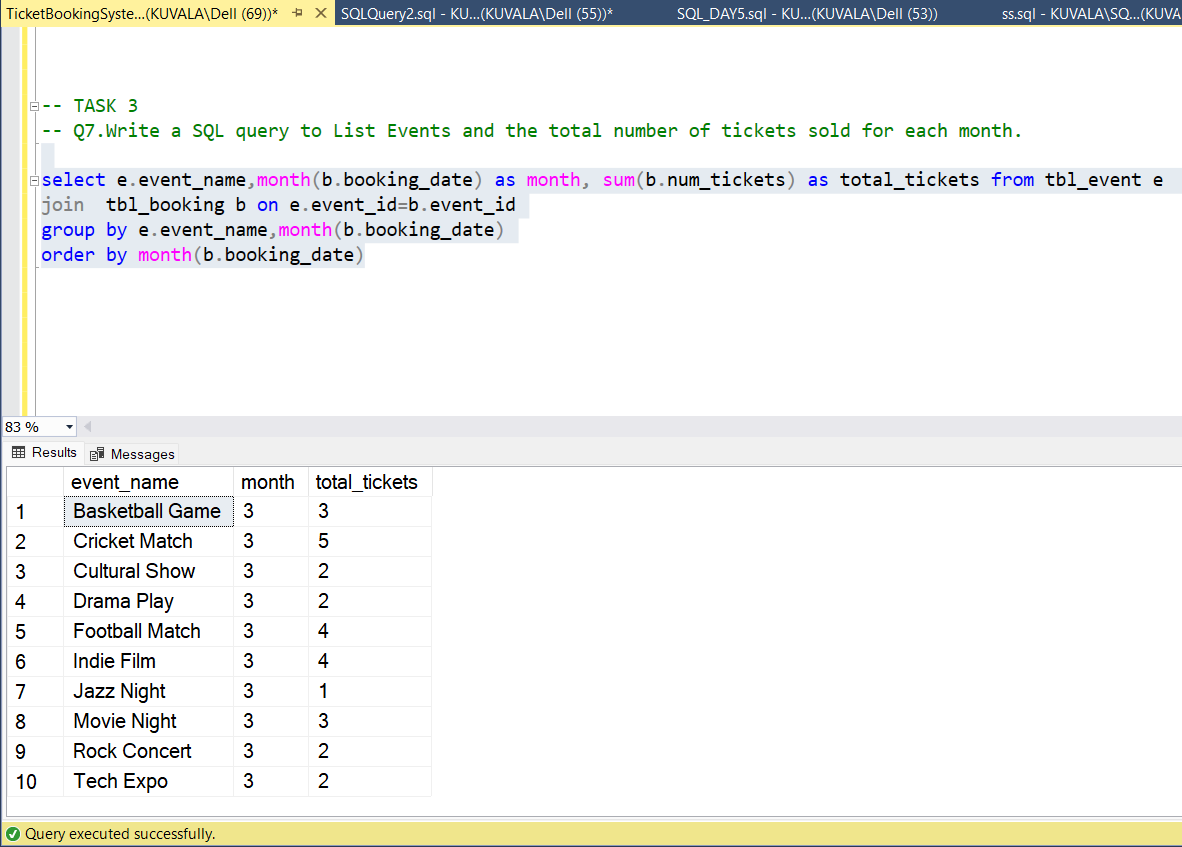
****

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

**select e.event\_name,month(b.booking\_date) as month, sum(b.num\_tickets) as total\_tickets from tbl\_event e**

**join tbl\_booking b on e.event\_id=b.event\_id group by e.event\_name,month(b.booking\_date)**

**order by month(b.booking\_date)**

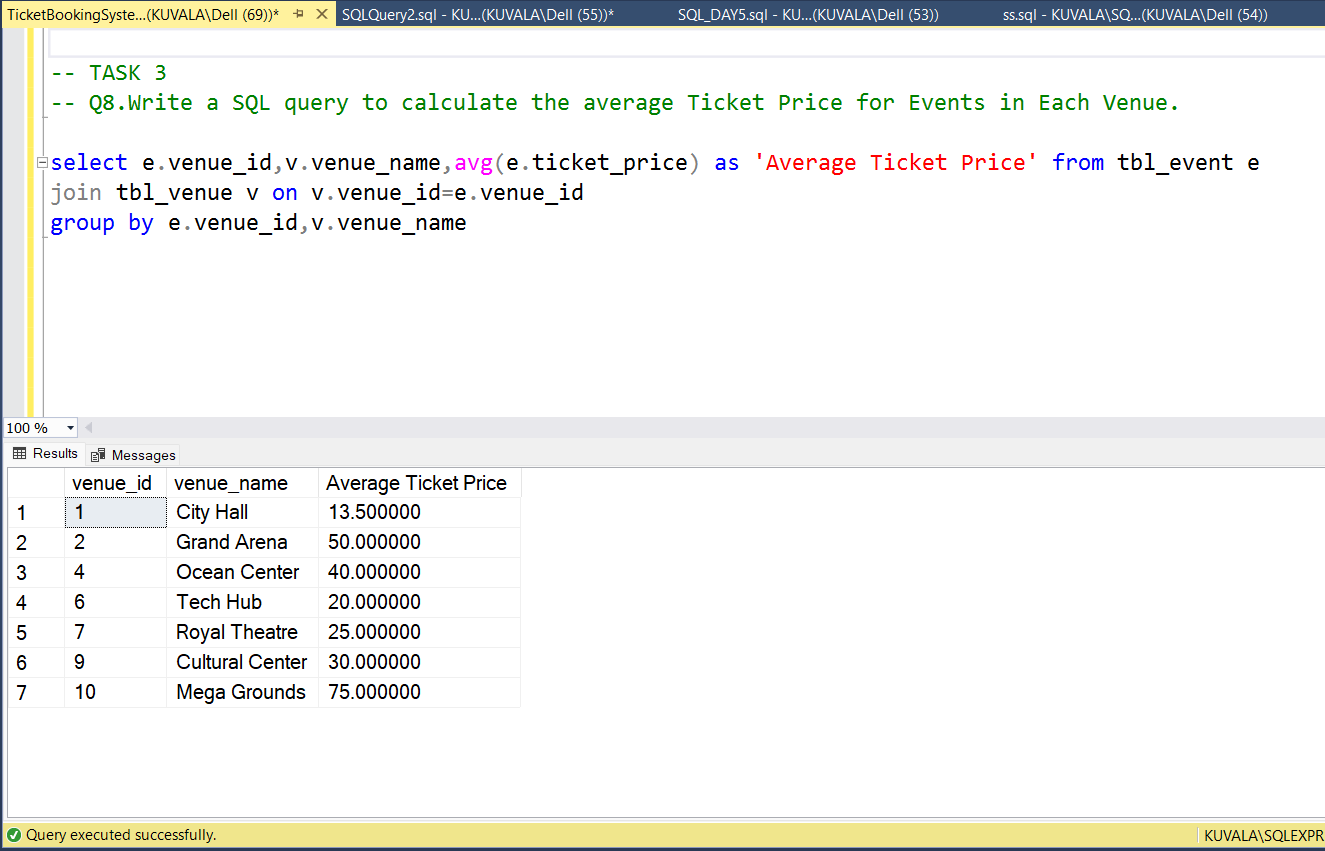
****

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

**select e.venue\_id,v.venue\_name,avg(e.ticket\_price) as 'Average Ticket Price' from tbl\_event e**

**join tbl\_venue v on v.venue\_id=e.venue\_id**

**group by e.venue\_id,v.venue\_name**

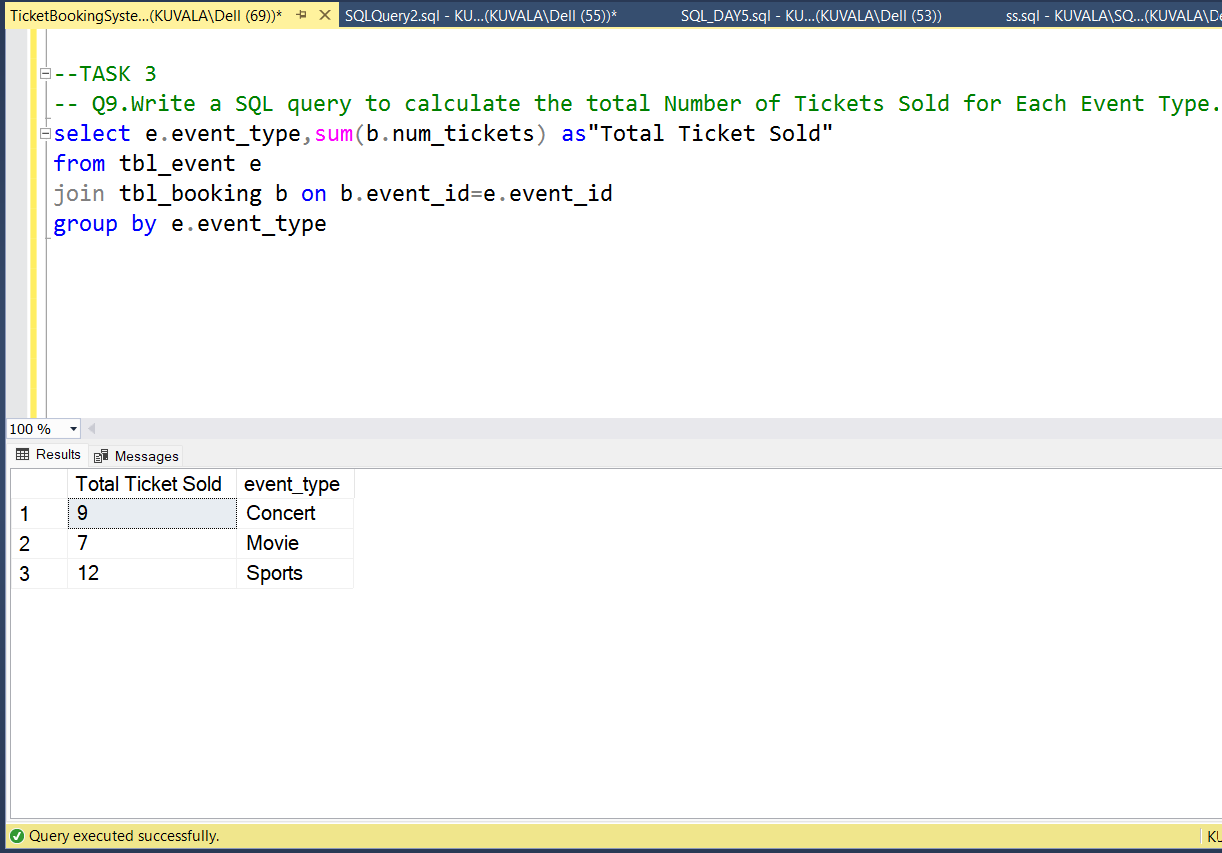
****

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

**select e.event\_type,sum(b.num\_tickets) as"Total Ticket Sold"**

**from tbl\_event e join tbl\_booking b on b.event\_id=e.event\_id**

**group by e.event\_type**

****

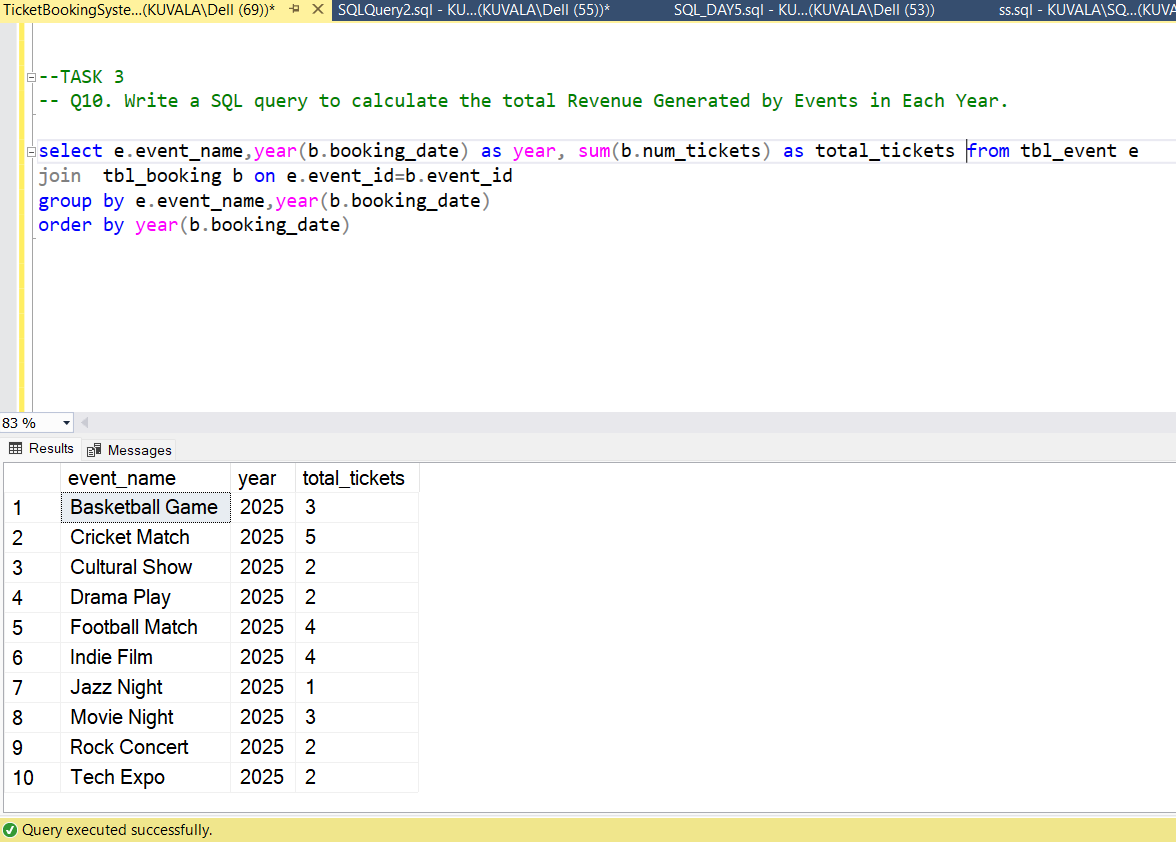
**10. Write a SQL query to calculate the total Revenue Generated by Events in Each Year.**

**select e.event\_name,year(b.booking\_date) as year, sum(b.num\_tickets) as total\_tickets from tbl\_event e**

**join tbl\_booking b on e.event\_id=b.event\_id**

**group by e.event\_name,year(b.booking\_date)**

**order by year(b.booking\_date)**

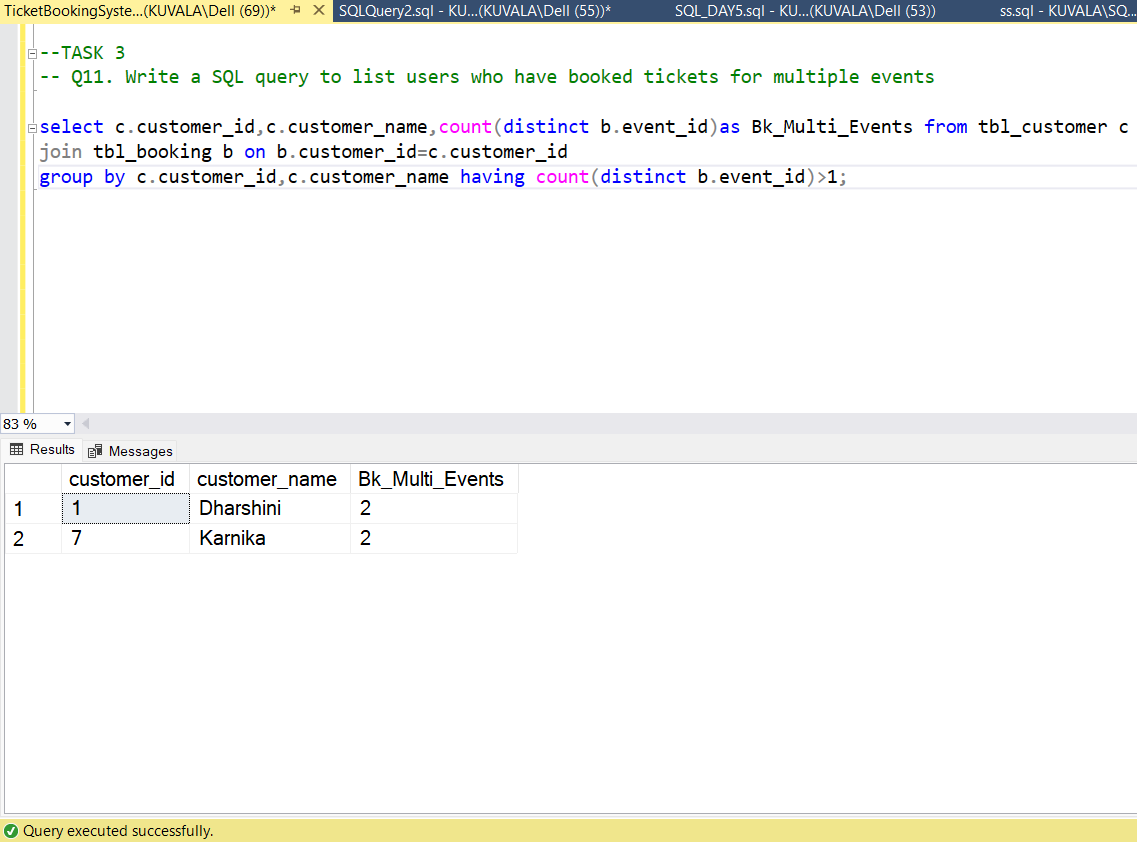
****

**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 Bk\_Multi\_Events from tbl\_customer c**

**join tbl\_booking b on b.customer\_id=c.customer\_id**

**group by c.customer\_id,c.customer\_name having count(distinct b.event\_id)>1;**

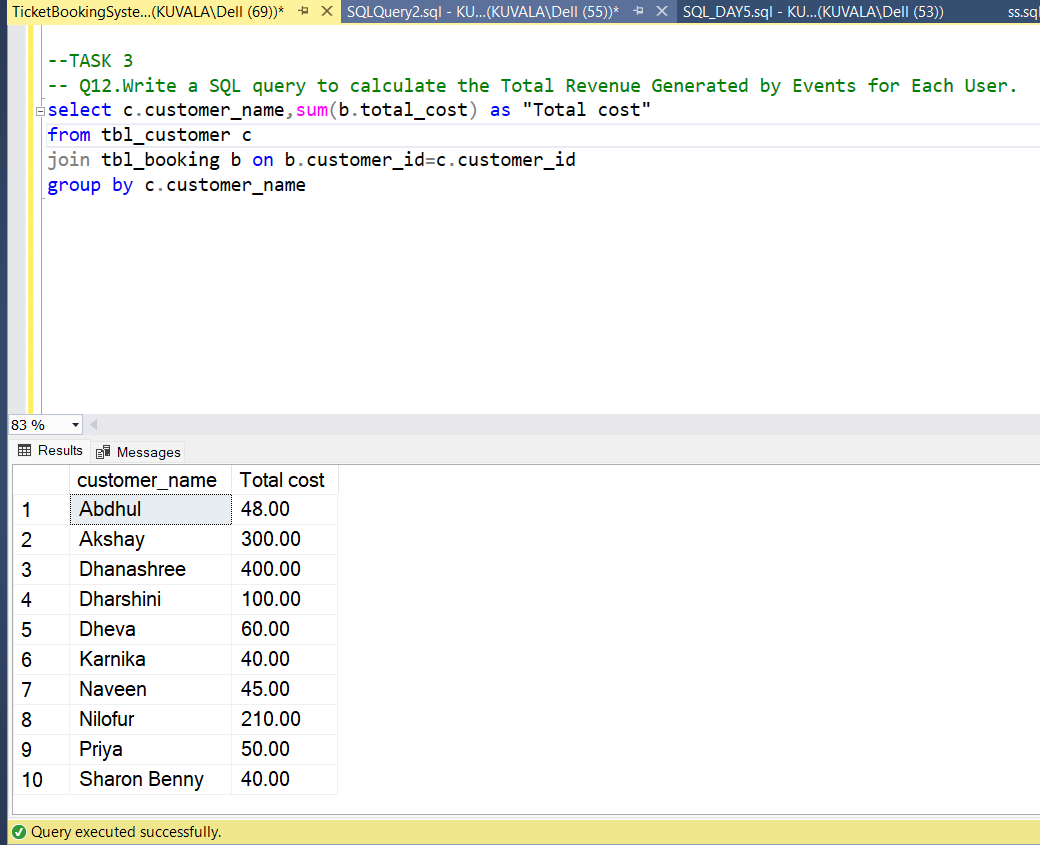
****

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

**select c.customer\_name,sum(b.total\_cost) as "Total cost"**

**from tbl\_customer c join tbl\_booking b on b.customer\_id=c.customer\_id**

**group by c.customer\_name**

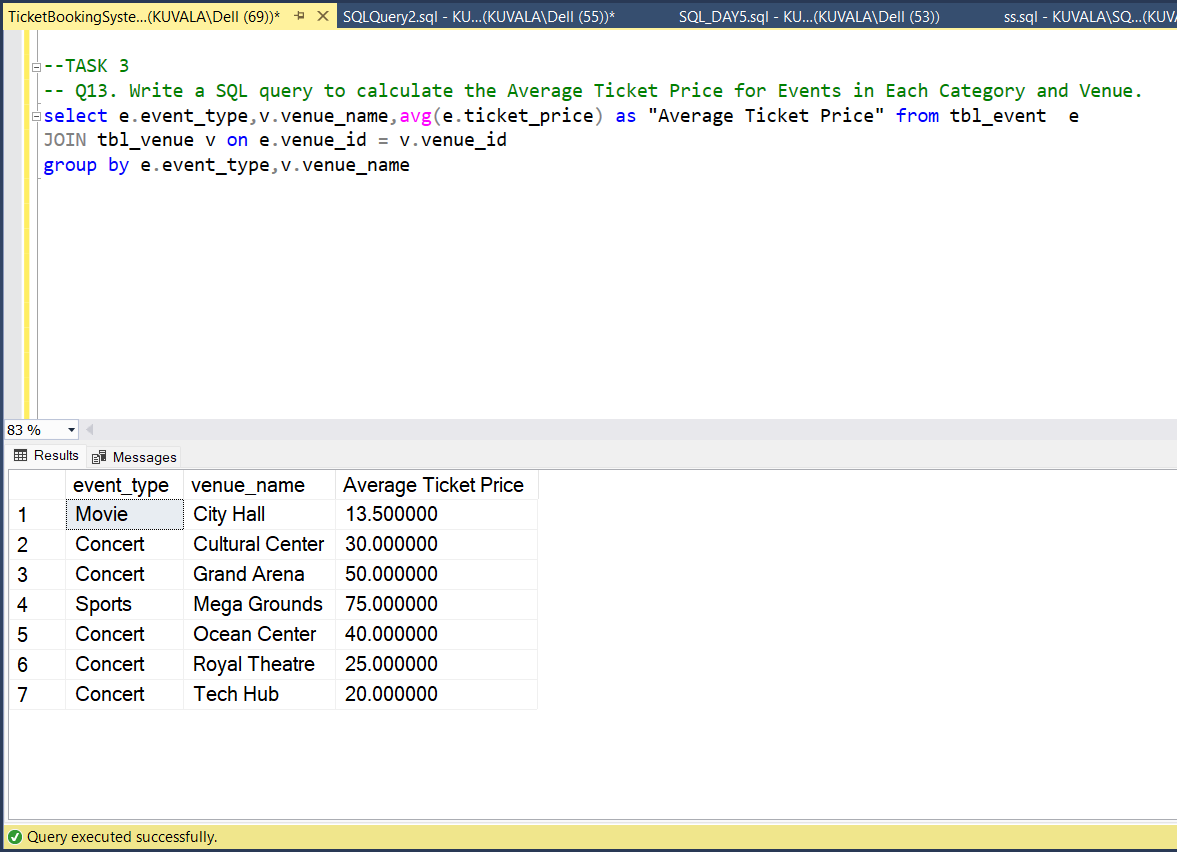
****

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

**select e.event\_type,v.venue\_name,avg(e.ticket\_price) as "Average Ticket Price" from tbl\_event e**

**JOIN tbl\_venue v on e.venue\_id = v.venue\_id**

**group by e.event\_type,v.venue\_name**

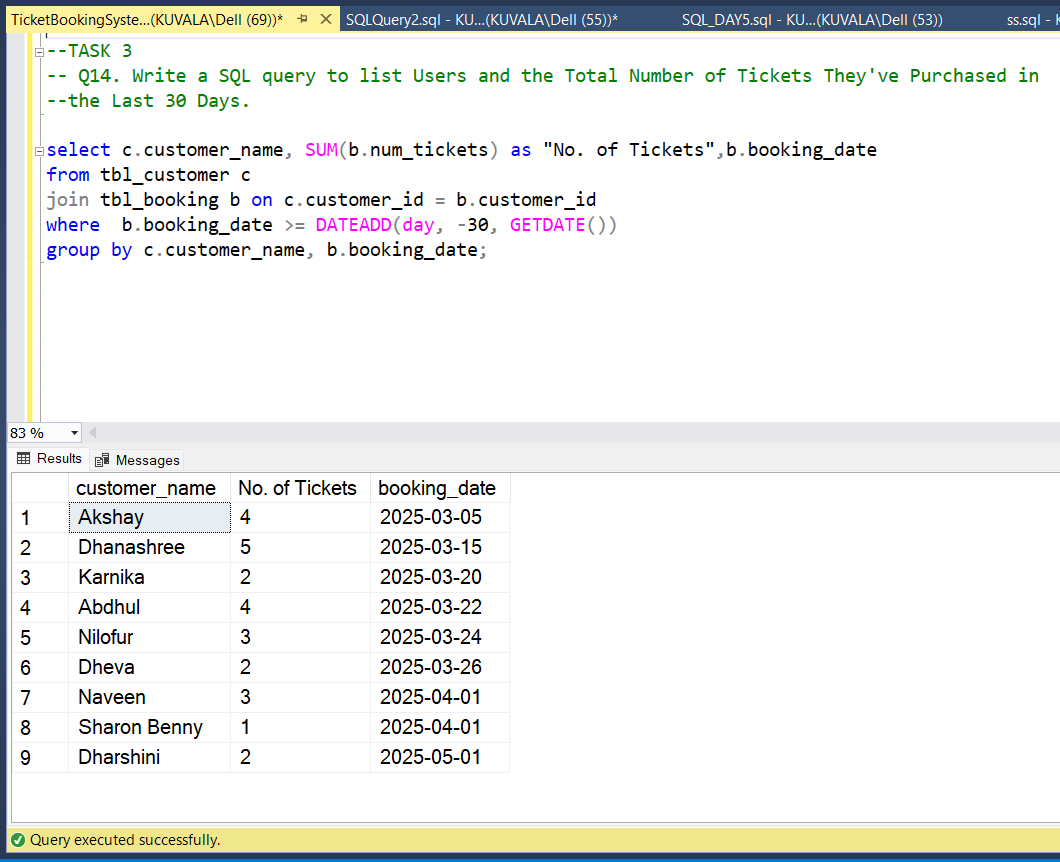
****

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

**select c.customer\_name, SUM(b.num\_tickets) as "No. of Tickets",b.booking\_date**

**from tbl\_customer c join tbl\_booking b on c.customer\_id = b.customer\_id**

**where b.booking\_date >= DATEADD(day, -30, GETDATE()) group by c.customer\_name, b.booking\_date;**

****

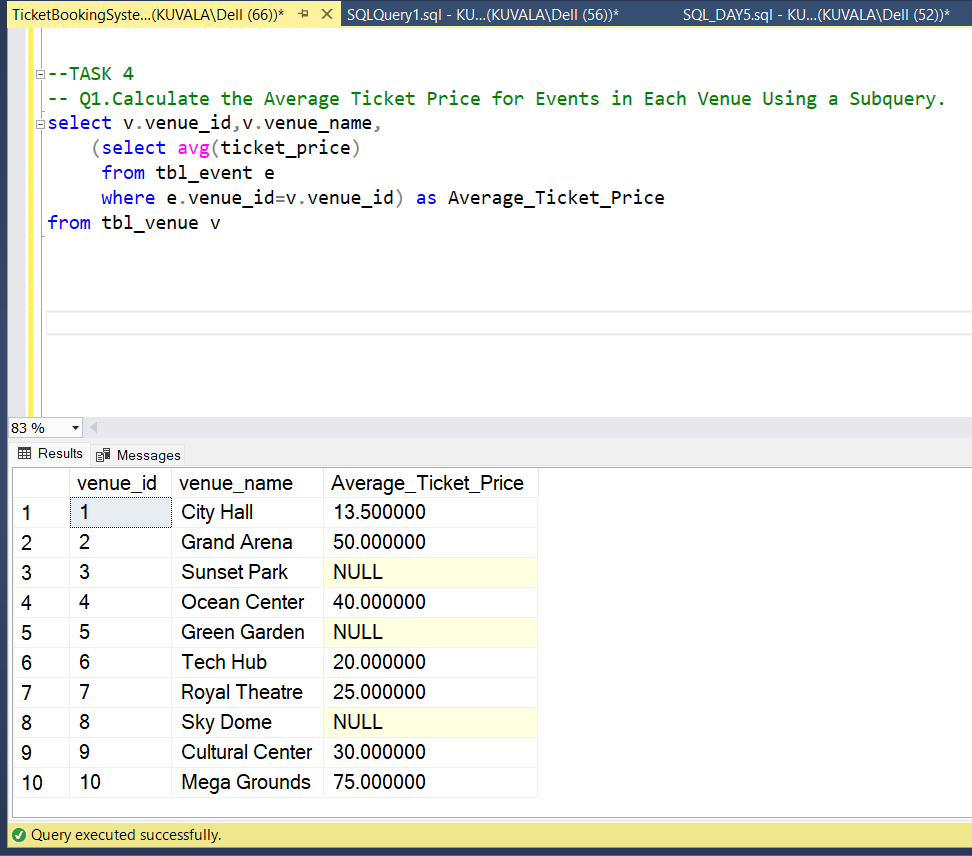
**TASK-4**

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

**select v.venue\_id,v.venue\_name,**

**(select avg(ticket\_price) from tbl\_event e where e.venue\_id=v.venue\_id) as Average\_Ticket\_Price**

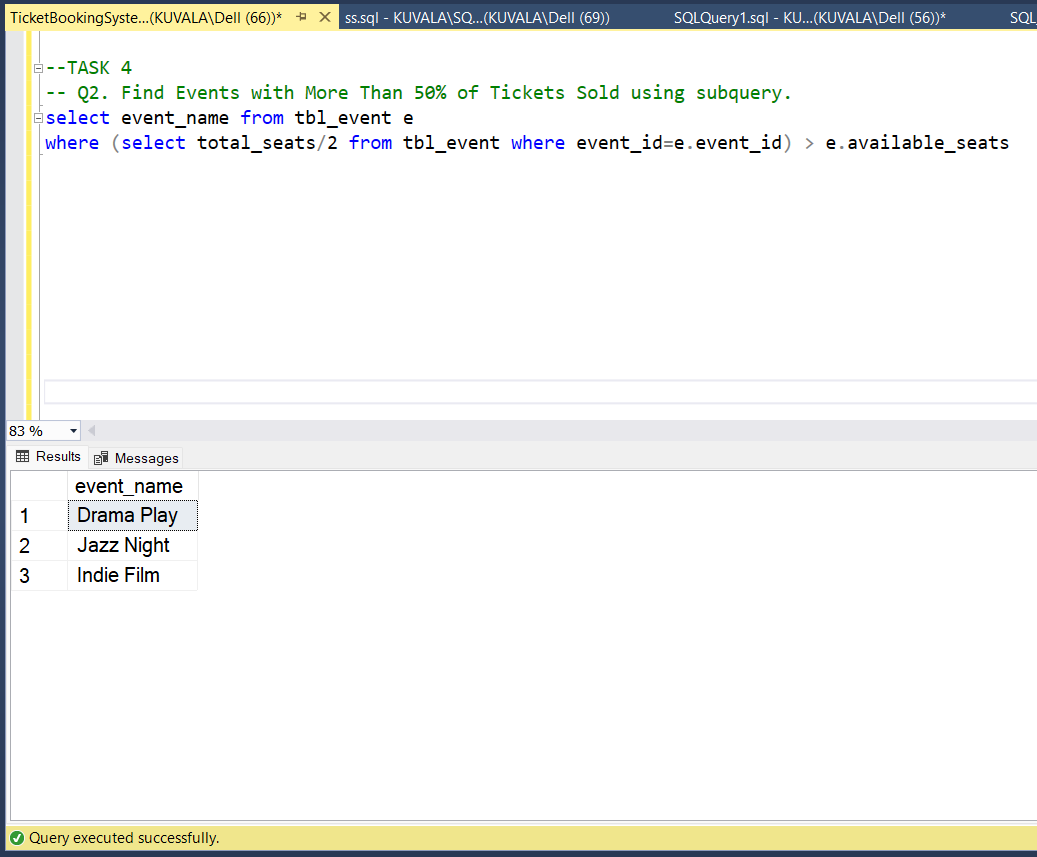
**from tbl\_venue v**

****

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

**select event\_name from tbl\_event e**

**where (select total\_seats/2 from tbl\_event where event\_id=e.event\_id) > e.available\_seats**

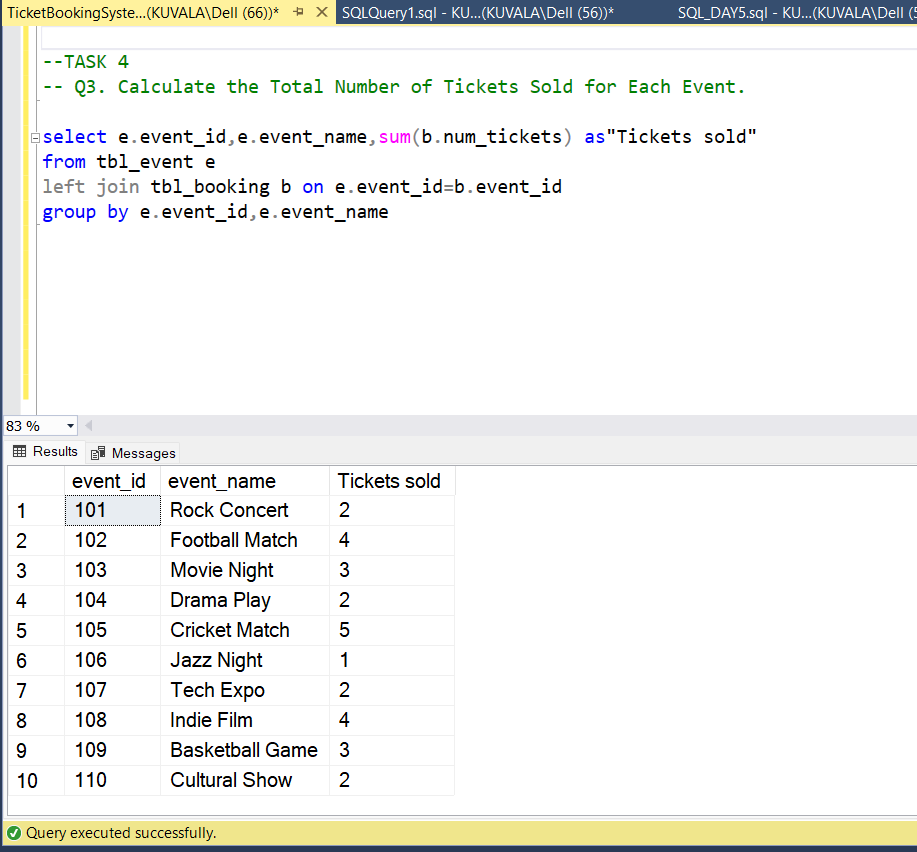
****

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

**select e.event\_id,e.event\_name,sum(b.num\_tickets) as"Tickets sold"**

**from tbl\_event e left join tbl\_booking b on e.event\_id=b.event\_id**

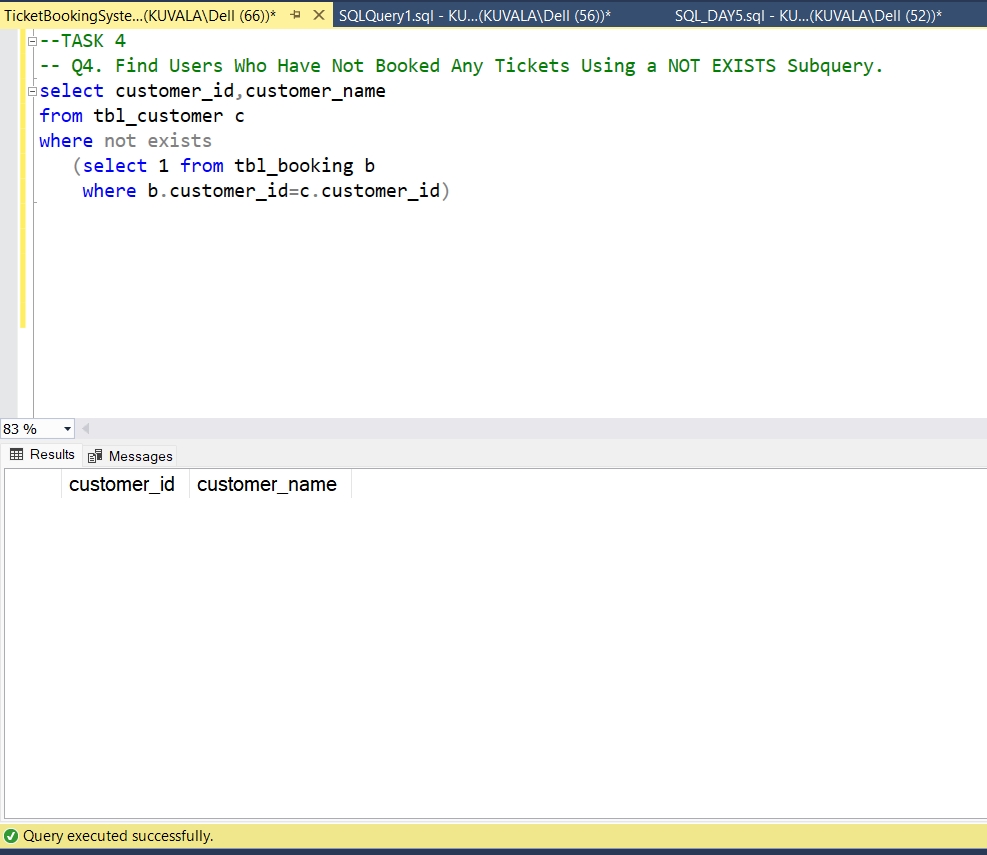
**group by e.event\_id,e.event\_name**

****

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

**select customer\_id,customer\_name from tbl\_customer c**

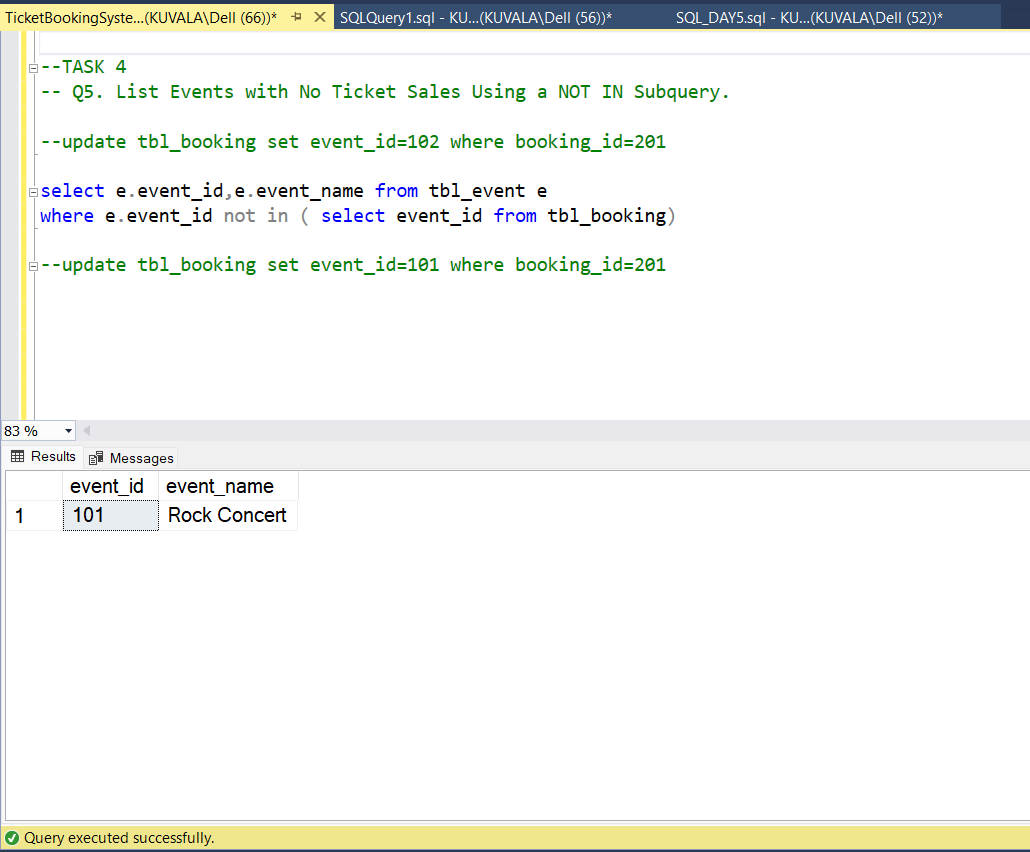
**where not exists (select 1 from tbl\_booking b where b.customer\_id=c.customer\_id)**

****

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

**select e.event\_id,e.event\_name from tbl\_event e**

**where e.event\_id not in ( select event\_id from tbl\_booking)**

****

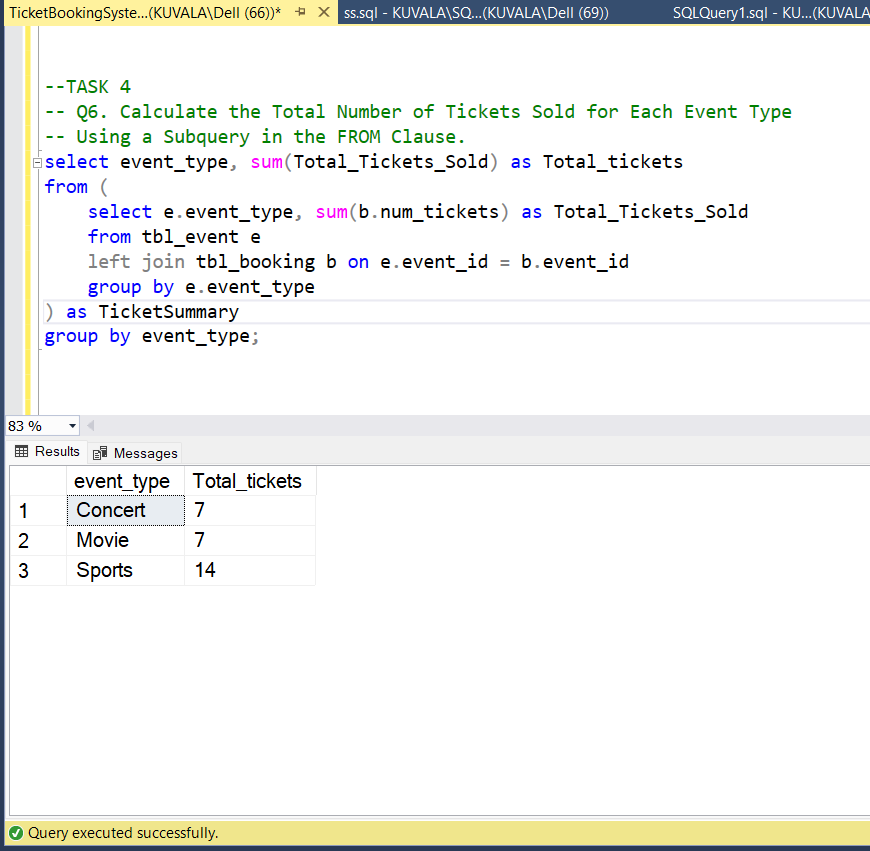
**6.** **Calculate the Total Number of Tickets Sold for Each Event Type Using a Subquery in the FROM Clause.**

**select event\_type, sum(Total\_Tickets\_Sold) as Total\_tickets**

**from (select e.event\_type, sum(b.num\_tickets) as Total\_Tickets\_Sold**

**from tbl\_event e left join tbl\_booking b on e.event\_id = b.event\_id group by e.event\_type) as TicketSummary**

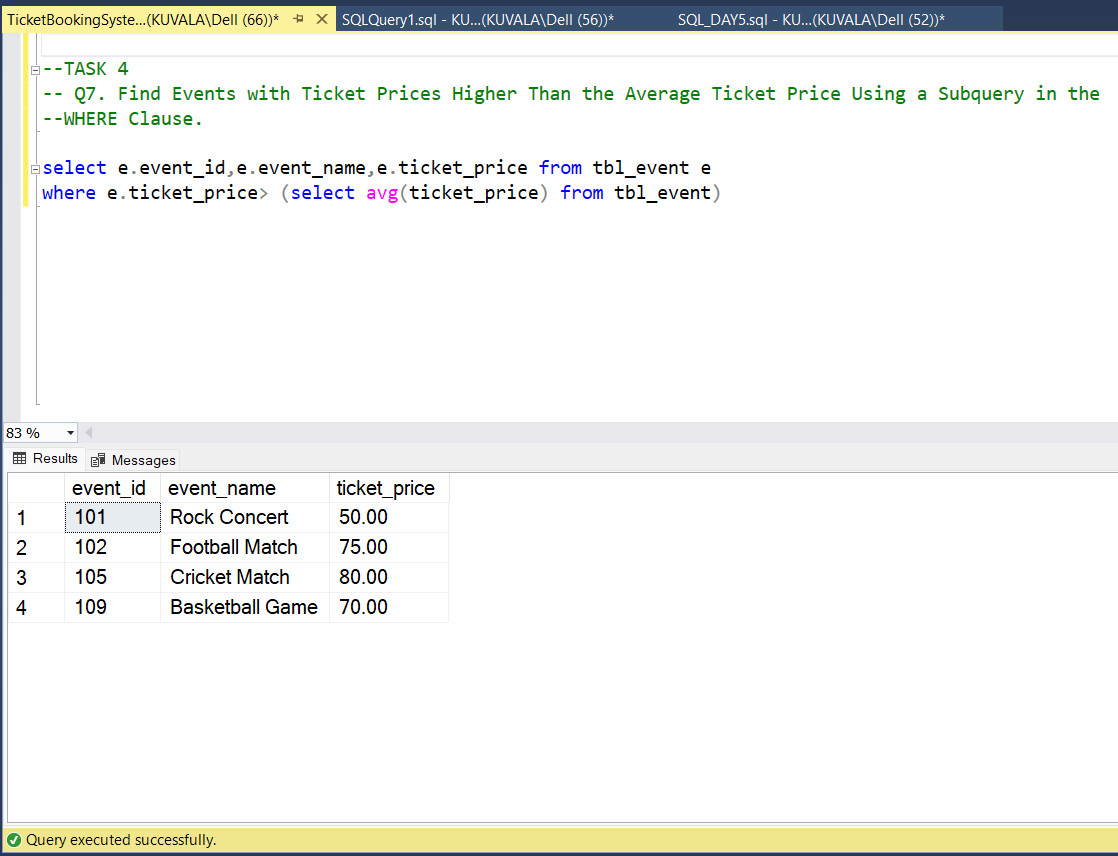
**group by event\_type**

****

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

**select e.event\_id,e.event\_name,e.ticket\_price from tbl\_event e**

**where e.ticket\_price> (select avg(ticket\_price) from tbl\_event)**

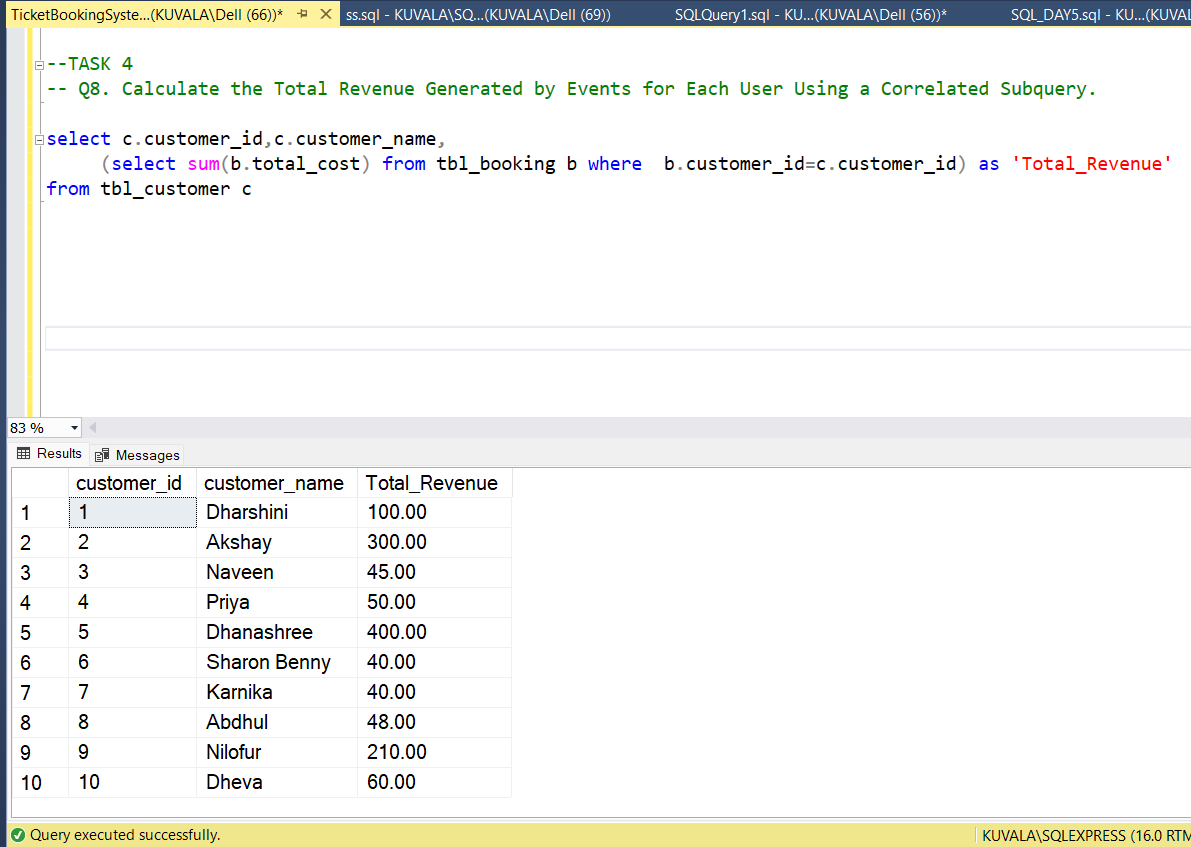
****

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

**select c.customer\_id,c.customer\_name,**

**(select sum(b.total\_cost) from tbl\_booking b where b.customer\_id=c.customer\_id) as 'Total\_Revenue'**

**from tbl\_customer c**

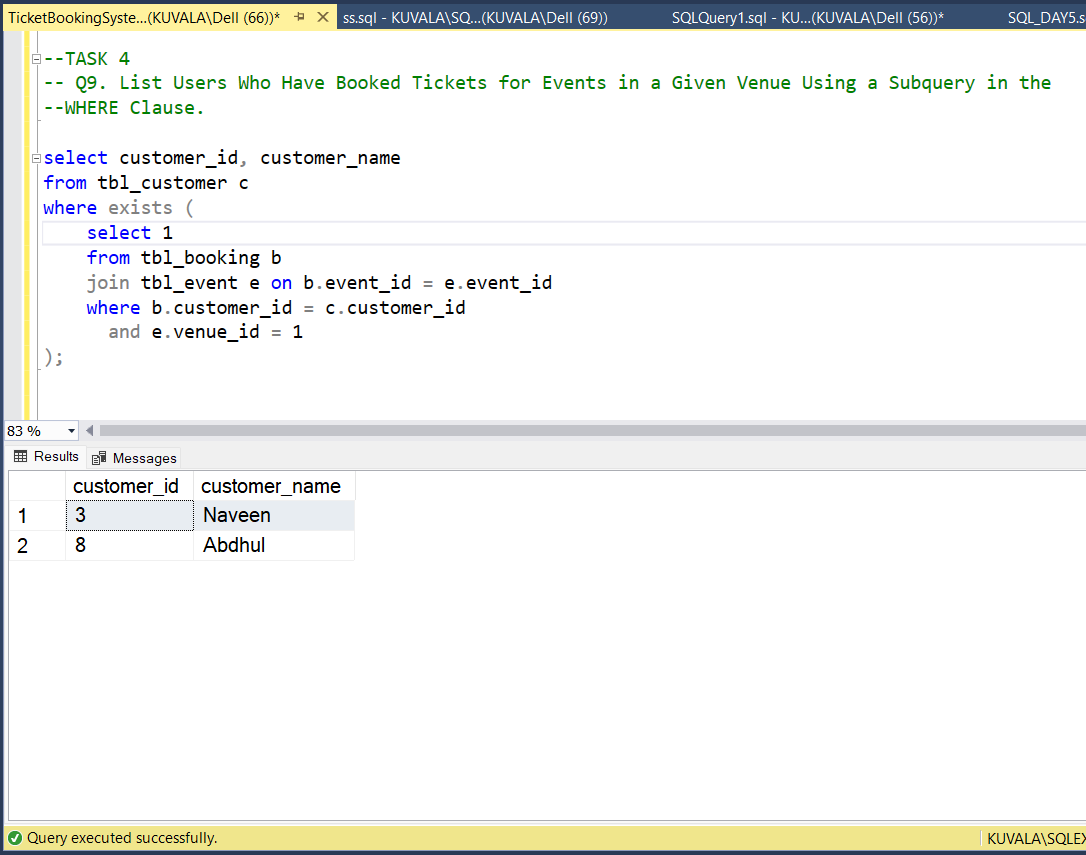
****

**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 tbl\_customer c**

**where exists ( select 1 from tbl\_booking b join tbl\_event e on b.event\_id = e.event\_id**

**where b.customer\_id = c.customer\_id and e.venue\_id = 1)**

****

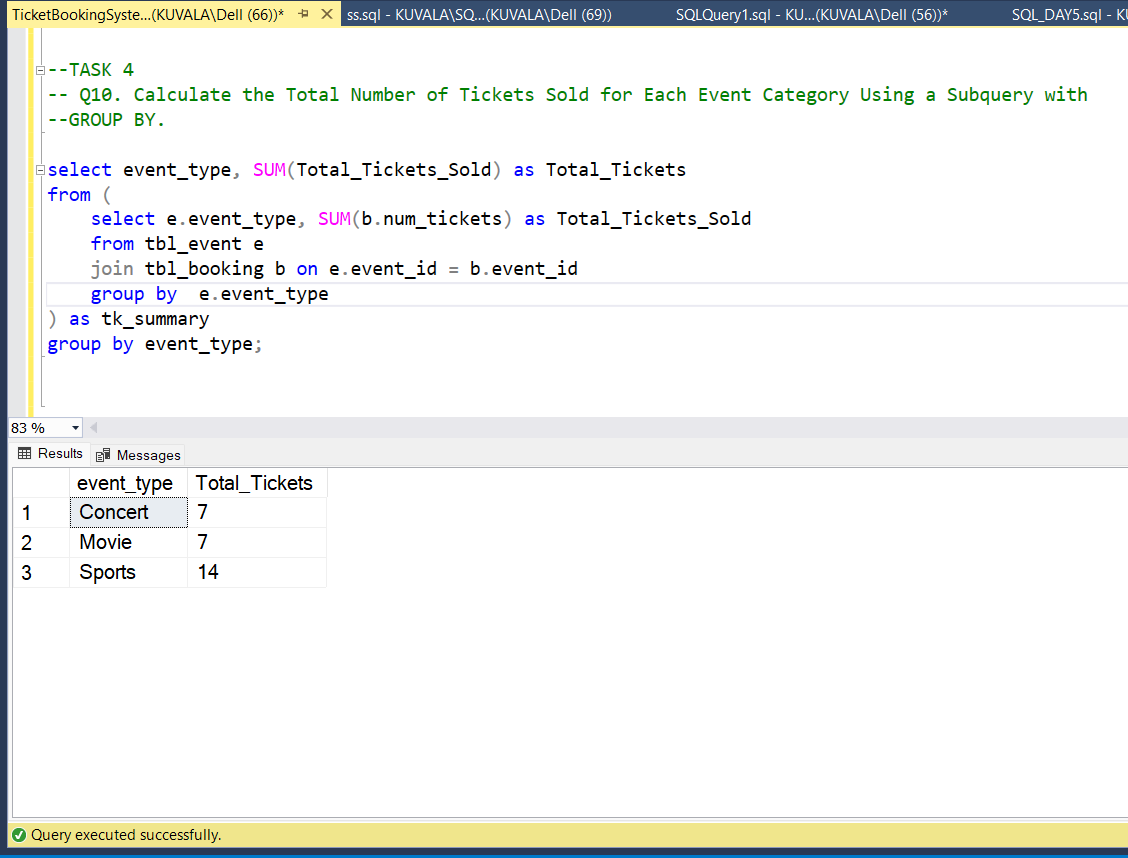
**10. Calculate the Total Number of Tickets Sold for Each Event Category Using a Subquery with GROUP BY.**

**select event\_type, SUM(Total\_Tickets\_Sold) as Total\_Tickets**

**from (select e.event\_type, SUM(b.num\_tickets) as Total\_Tickets\_Sold from tbl\_event e**

**join tbl\_booking b on e.event\_id = b.event\_id group by e.event\_type) as tk\_summary**

**group by event\_type**

****

**11. Find Users Who Have Booked Tickets for Events in each Month Using a Subquery with DATE\_FORMAT.**

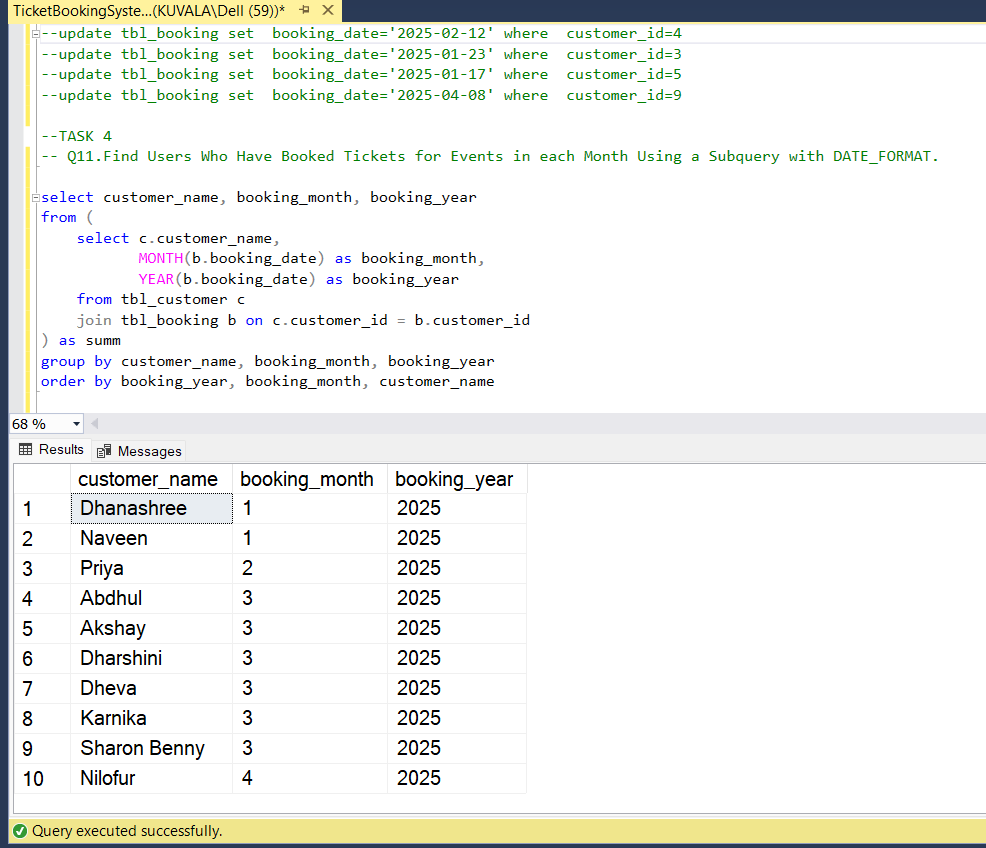
**select customer\_name, booking\_month, booking\_year from**

**(select c.customer\_name, MONTH(b.booking\_date) as booking\_month, YEAR(b.booking\_date) as booking\_year**

**from tbl\_customer c join tbl\_booking b on c.customer\_id = b.customer\_id) as summ**

**group by customer\_name, booking\_month, booking\_year**

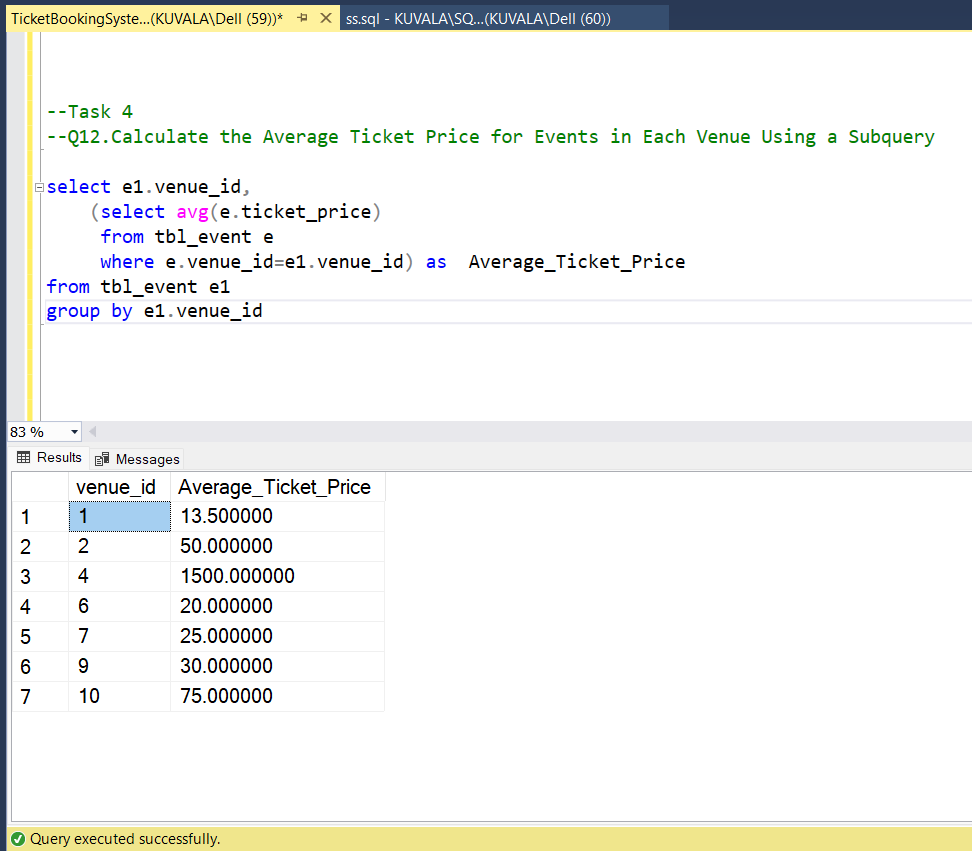
**order by booking\_year, booking\_month, customer\_name**

****

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

**select e1.venue\_id, (select avg(e.ticket\_price) from tbl\_event e where e.venue\_id=e1.venue\_id) as**

**Average\_Ticket\_Price from tbl\_event e1 group by e1.venue\_id**

****

**End.**