**TASK 3**

**Project Title: Event Management System using PostgreSQL**

**Project Description:**

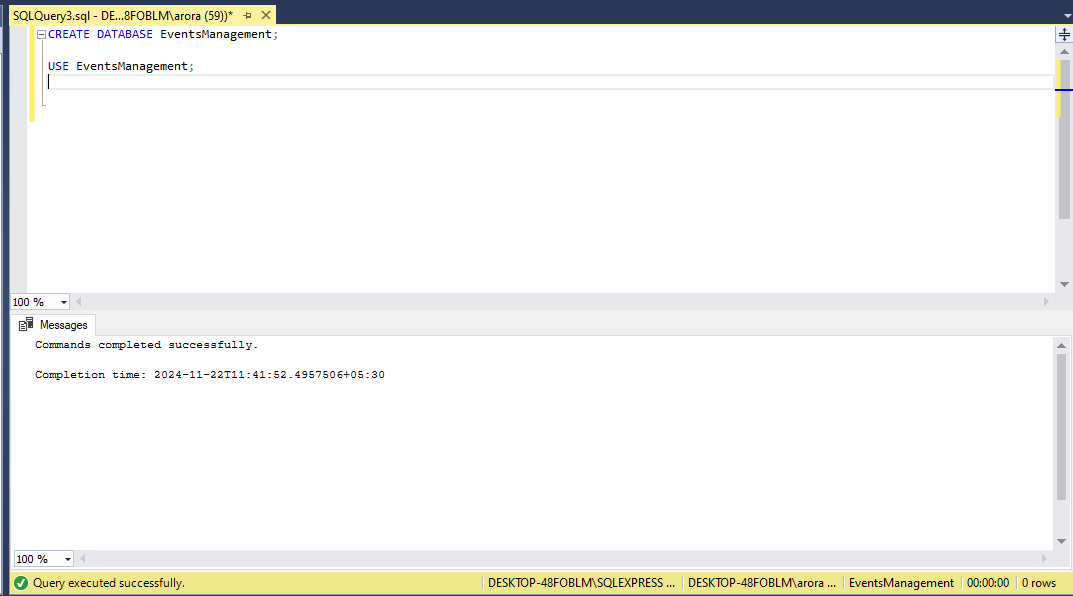
To develop the application that allows users to create and manage events, track

attendees, and handle event registrations efficiently.

1. **Database Creation**

**Code-** CREATE DATABASE EventsManagement;

USE EventsManagement;



1. **Creation of Tables**

**Code-** CREATE TABLE Events (

Event\_Id INT IDENTITY(1,1) PRIMARY KEY,

Event\_Name VARCHAR(255) NOT NULL,

Event\_Date DATE NOT NULL,

Event\_Location VARCHAR(255) NOT NULL,

Event\_Description VARCHAR(255));

CREATE TABLE Attendees (

Attendee\_Id INT IDENTITY(1,1) PRIMARY KEY,

Attendee\_Name VARCHAR(255) NOT NULL,

Attendee\_Phone INT NOT NULL,

Attendee\_Email VARCHAR(255) UNIQUE NOT NULL,

Attendee\_City VARCHAR(255) NOT NULL

);

CREATE TABLE Registrations (

Registration\_Id INT IDENTITY(1,1) PRIMARY KEY,

Event\_Id INT,

Attendee\_Id INT,

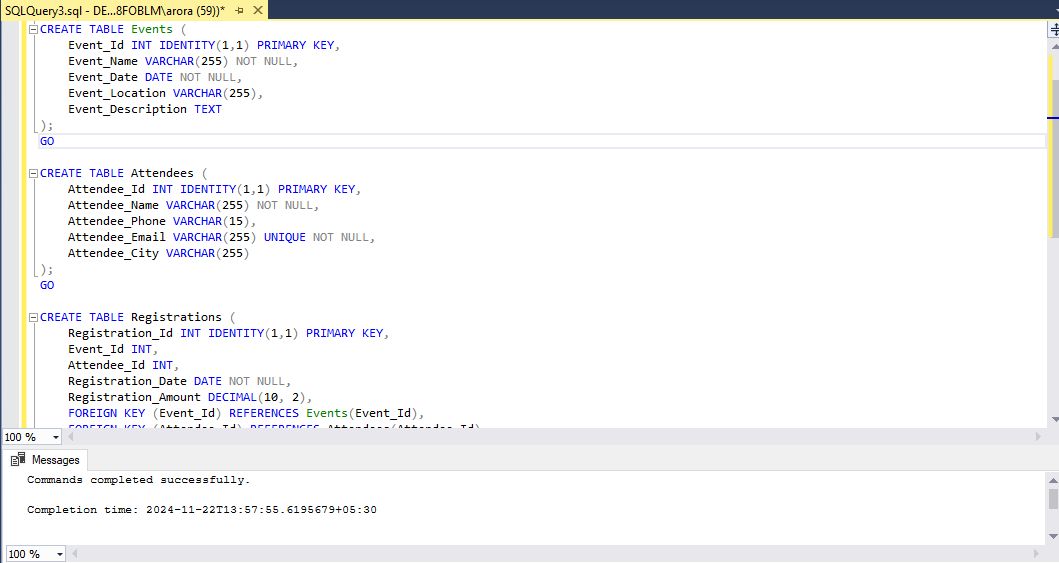
Registration\_Date DATE NOT NULL,

Registration\_Amount DECIMAL(10, 2) NOT NULL,

FOREIGN KEY (Event\_Id) REFERENCES Events(Event\_Id),

FOREIGN KEY (Attendee\_Id) REFERENCES Attendees(Attendee\_Id)

);



1. **Insertion of Values**

**Code-** INSERT INTO Events (Event\_Name, Event\_Date, Event\_Location, Event\_Description)

VALUES

('Tech Conference 2024', '2024-12-15', 'New York Convention Center', 'A conference focused on emerging technologies.'),

('Music Festival 2024', '2024-11-30', 'Central Park, New York', 'A music festival featuring local and international artists.'),

('Health & Wellness Expo', '2024-12-05', 'Los Angeles Convention Center', 'An expo focused on health, fitness, and wellness.');

INSERT INTO Attendees (Attendee\_Name, Attendee\_Phone, Attendee\_Email, Attendee\_City)

VALUES

('John Doe', '555-1234', 'johndoe@example.com', 'New York'),

('Alice Smith', '555-5678', 'alicesmith@example.com', 'Los Angeles'),

('Bob Johnson', '555-8765', 'bobjohnson@example.com', 'Chicago'),

('Emma Brown', '555-4321', 'emmabrown@example.com', 'San Francisco');

INSERT INTO Registrations (Event\_Id, Attendee\_Id, Registration\_Date, Registration\_Amount)

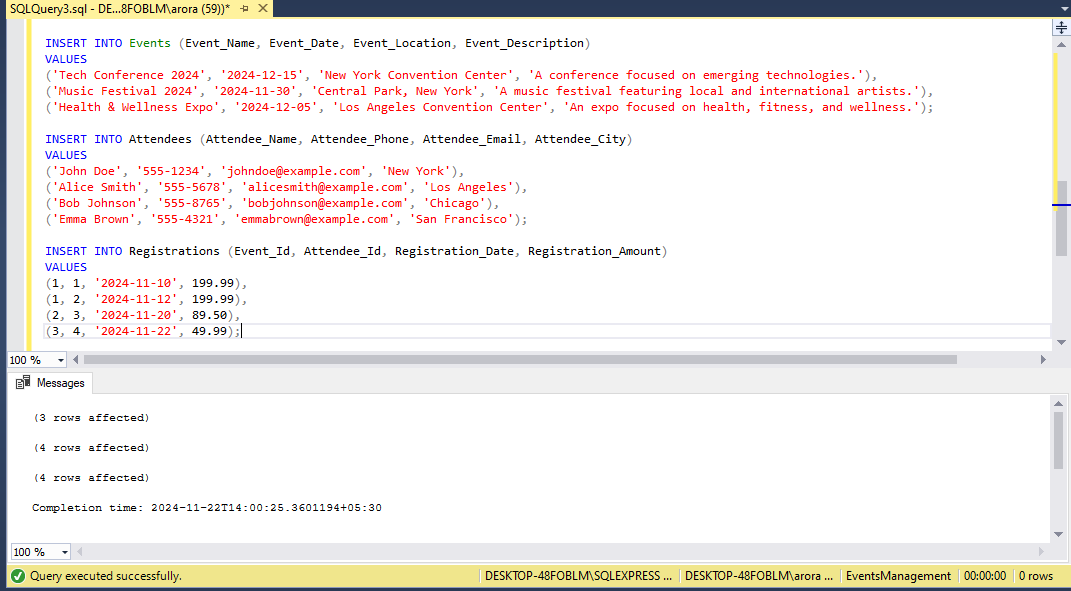
VALUES

(1, 1, '2024-11-10', 199.99),

(1, 2, '2024-11-12', 199.99),

(2, 3, '2024-11-20', 89.50),

(3, 4, '2024-11-22', 49.99);

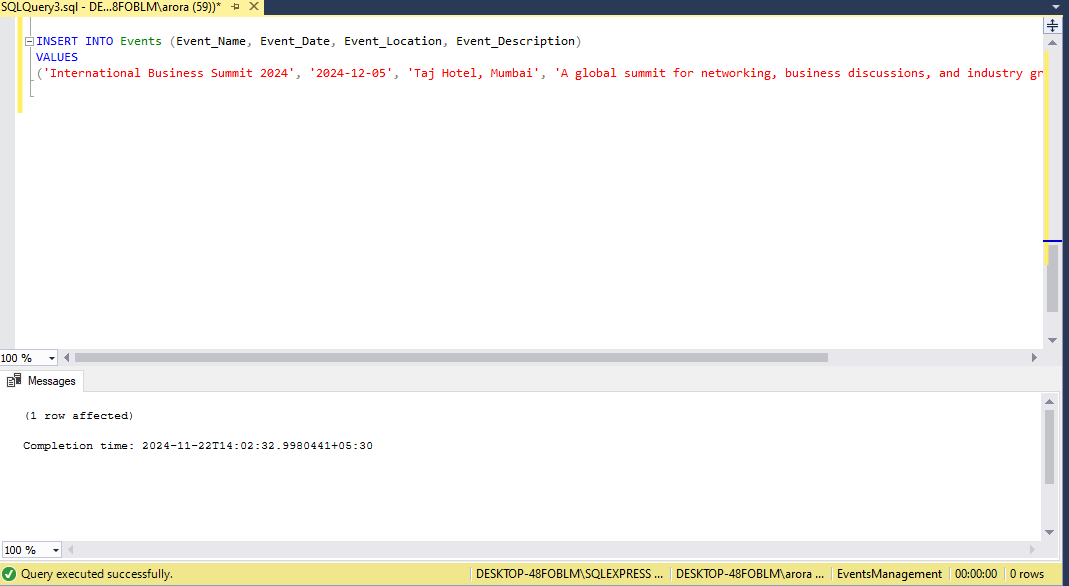


**4.a) Inserting a new event.**

**Code-** INSERT INTO Events (Event\_Name, Event\_Date, Event\_Location, Event\_Description)

VALUES

('International Business Summit 2024', '2024-12-05', 'Taj Hotel, Mumbai', 'A global summit for networking, business discussions, and industry growth.');



**4.b) Updating an event's information.**

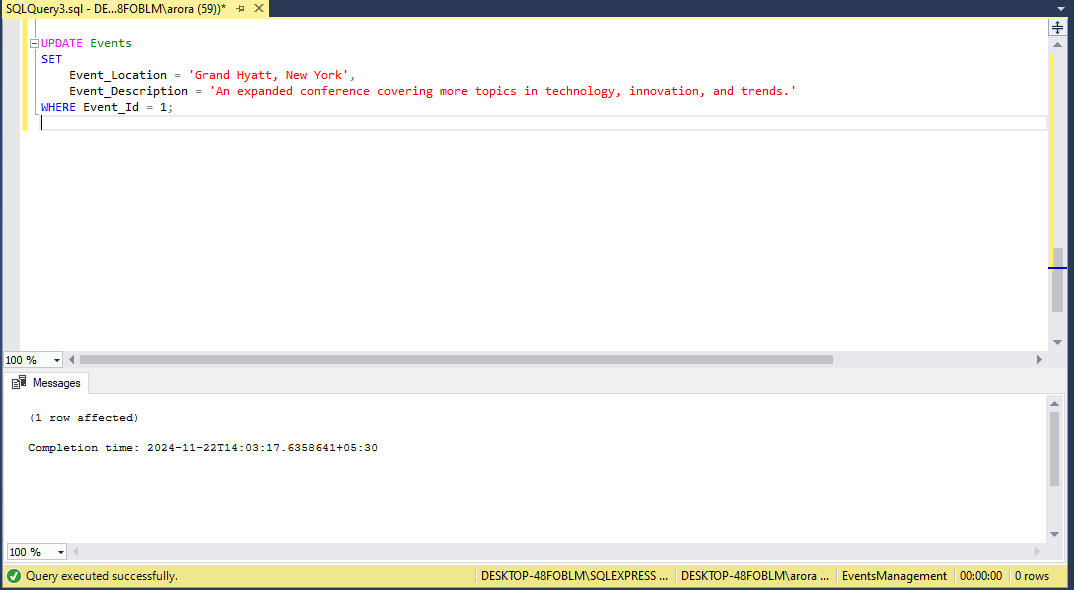
**Code-** UPDATE Events

SET

Event\_Location = 'Grand Hyatt, New York',

Event\_Description = 'An expanded conference covering more topics in technology, innovation, and trends.'

WHERE Event\_Id = 1;



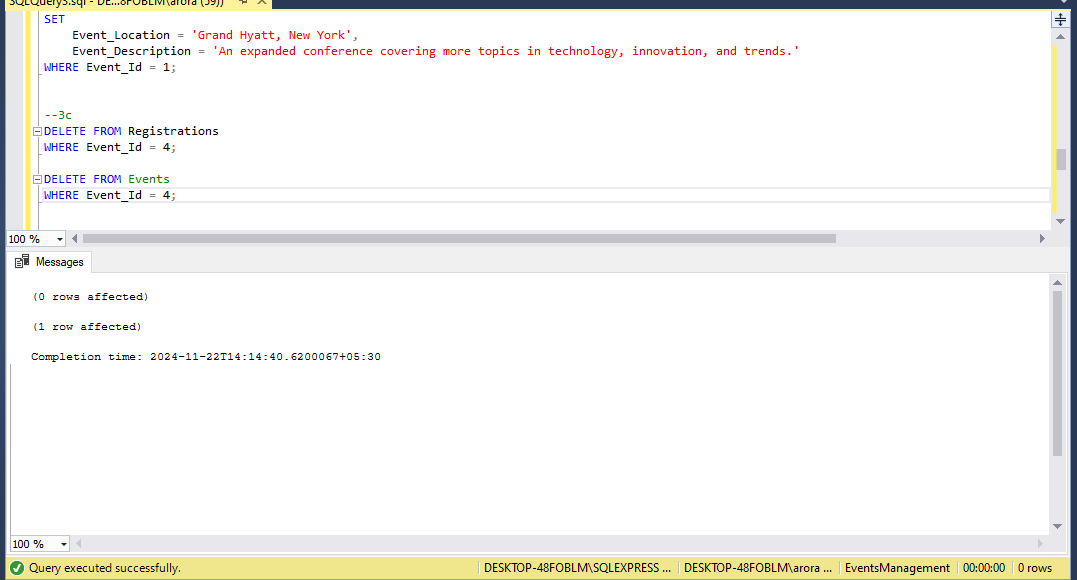
**4.c) Deleting an event.**

**Code-** DELETE FROM Registrations

WHERE Event\_Id = 4;

DELETE FROM Events

WHERE Event\_Id = 4;



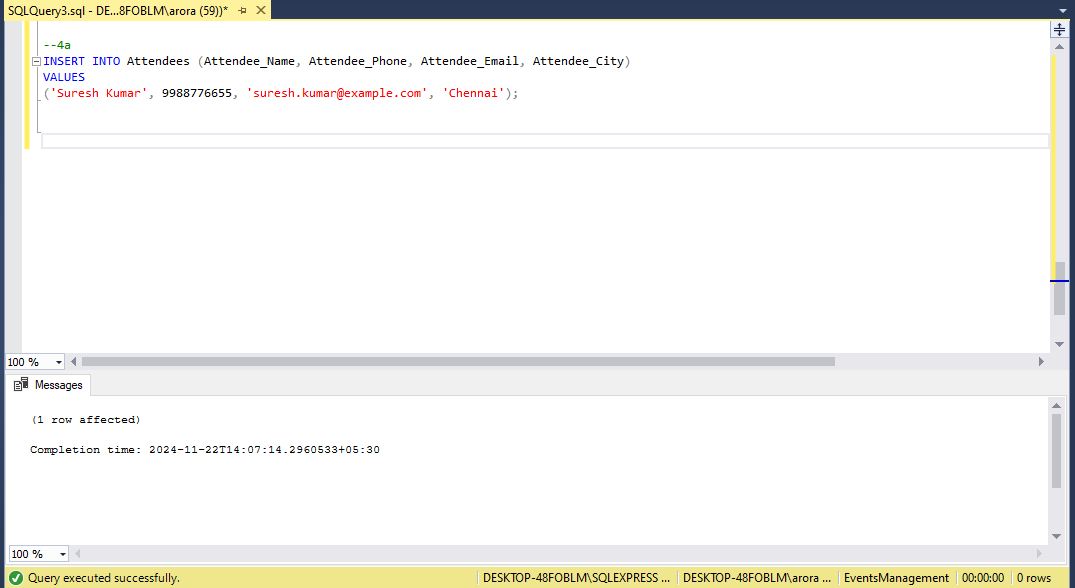
**5.a) Inserting a new attendee.**

**Code-** Inserting a new attendee.

INSERT INTO Attendees (Attendee\_Name, Attendee\_Phone, Attendee\_Email, Attendee\_City)

VALUES

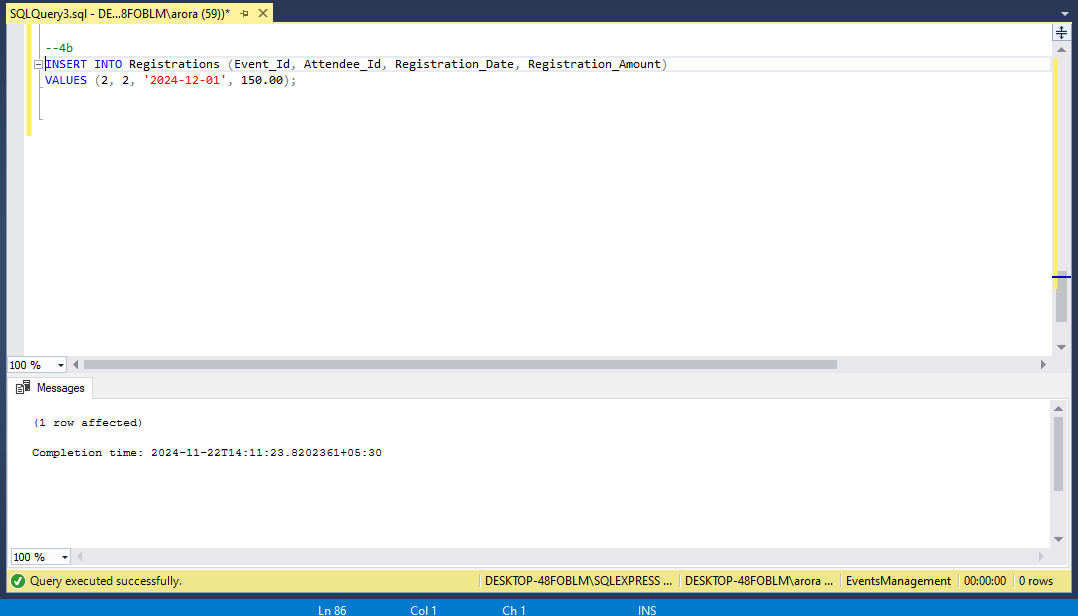
('Suresh Kumar', 9988776655, 'suresh.kumar@example.com', 'Chennai');



**5.b) Registering an attendee for an event.**

**Code-** INSERT INTO Registrations (Event\_Id, Attendee\_Id, Registration\_Date, Registration\_Amount)

VALUES (2, 2, '2024-12-01', 150.00);

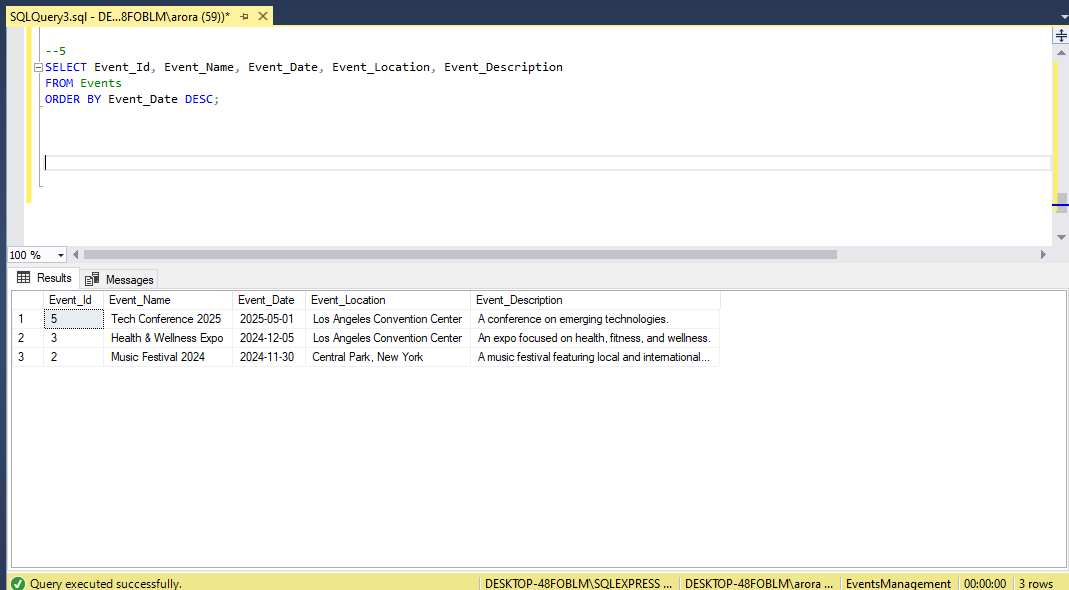


**6. Develop queries to retrieve event information, generate attendee lists, and calculate event attendance statistics.**

**Code-** SELECT Event\_Id, Event\_Name, Event\_Date, Event\_Location, Event\_Description

FROM Events

ORDER BY Event\_Date DESC;



SELECT A.Attendee\_Name, A.Attendee\_City

FROM Attendees A

JOIN Registrations R ON A.Attendee\_Id = R.Attendee\_Id

WHERE R.Event\_Id = 2

ORDER BY A.Attendee\_Name;

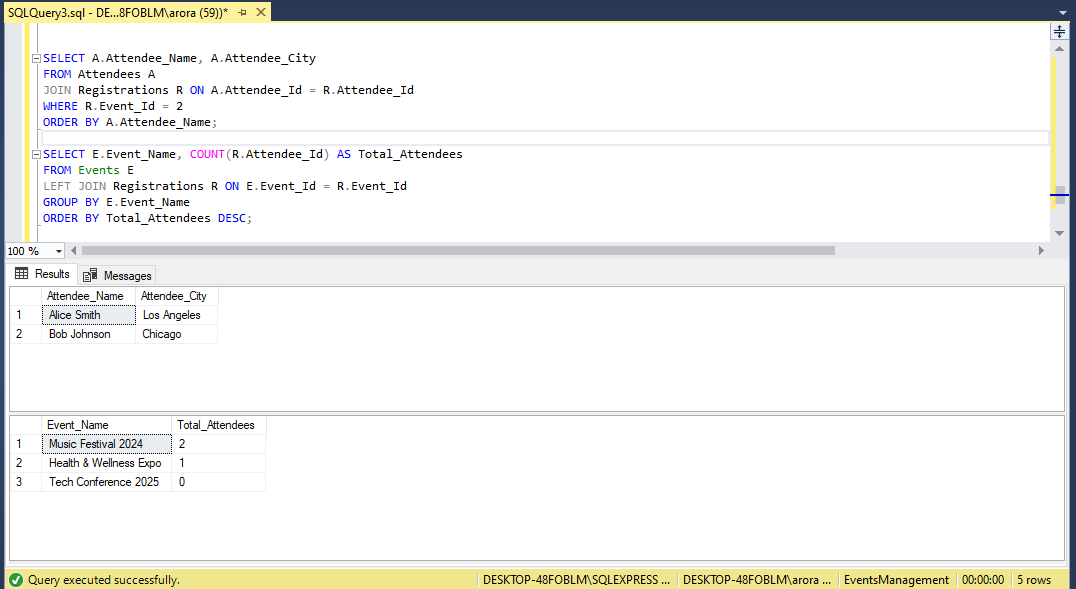
SELECT E.Event\_Name, COUNT(R.Attendee\_Id) AS Total\_Attendees

FROM Events E

LEFT JOIN Registrations R ON E.Event\_Id = R.Event\_Id

GROUP BY E.Event\_Name

ORDER BY Total\_Attendees DESC;



SELECT E.Event\_Name, SUM(R.Registration\_Amount) AS Total\_Revenue

FROM Events E

JOIN Registrations R ON E.Event\_Id = R.Event\_Id

GROUP BY E.Event\_Name

ORDER BY Total\_Revenue DESC;

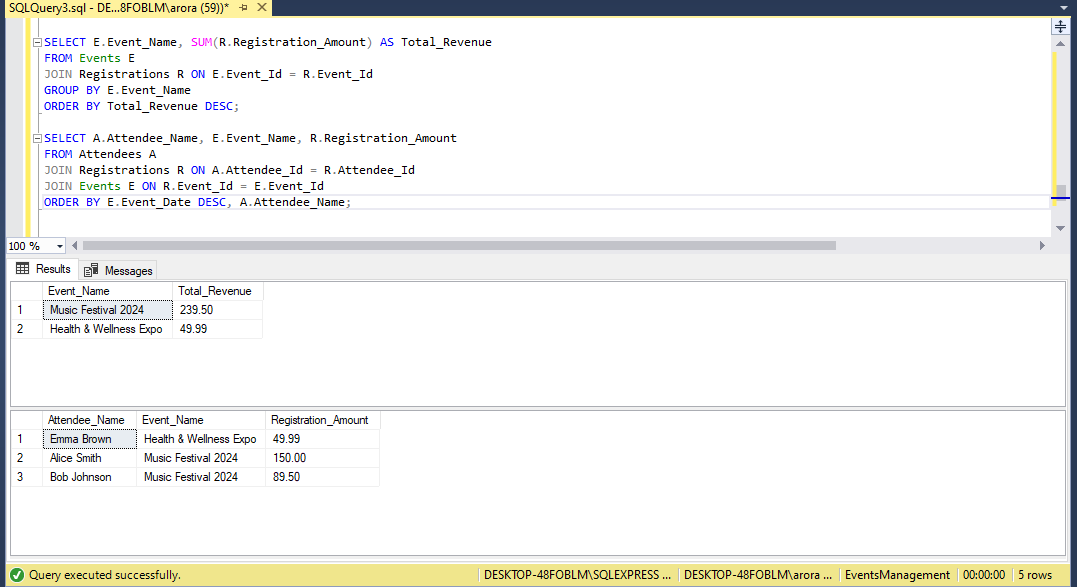
SELECT A.Attendee\_Name, E.Event\_Name, R.Registration\_Amount

FROM Attendees A

JOIN Registrations R ON A.Attendee\_Id = R.Attendee\_Id

JOIN Events E ON R.Event\_Id = E.Event\_Id

ORDER BY E.Event\_Date DESC, A.Attendee\_Name;



SELECT AVG(Attendees\_Count) AS Average\_Attendees\_Per\_Event

FROM (

SELECT COUNT(Attendee\_Id) AS Attendees\_Count

FROM Registrations

GROUP BY Event\_Id) AS Event\_Attendees;

SELECT E.Event\_Name

FROM Events E

LEFT JOIN Registrations R ON E.Event\_Id = R.Event\_Id

WHERE R.Attendee\_Id IS NULL;

