# OUTPUT SHEET

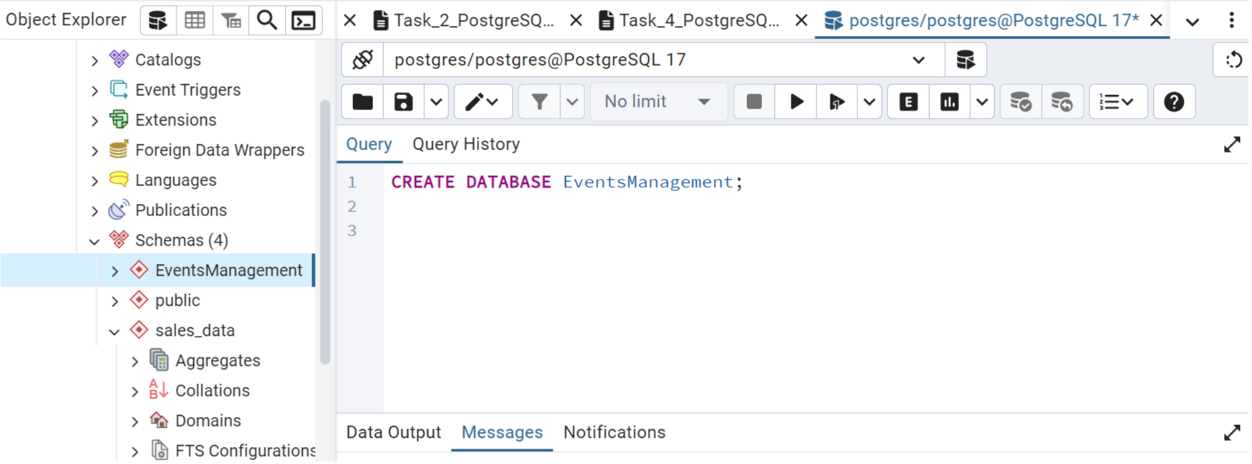
**Task 3**

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

Objective: To develop the application that allows users to create and manage events, track attendees, and handle event registrations efficiently. The project will include the following tasks:

**1. Database Creation**

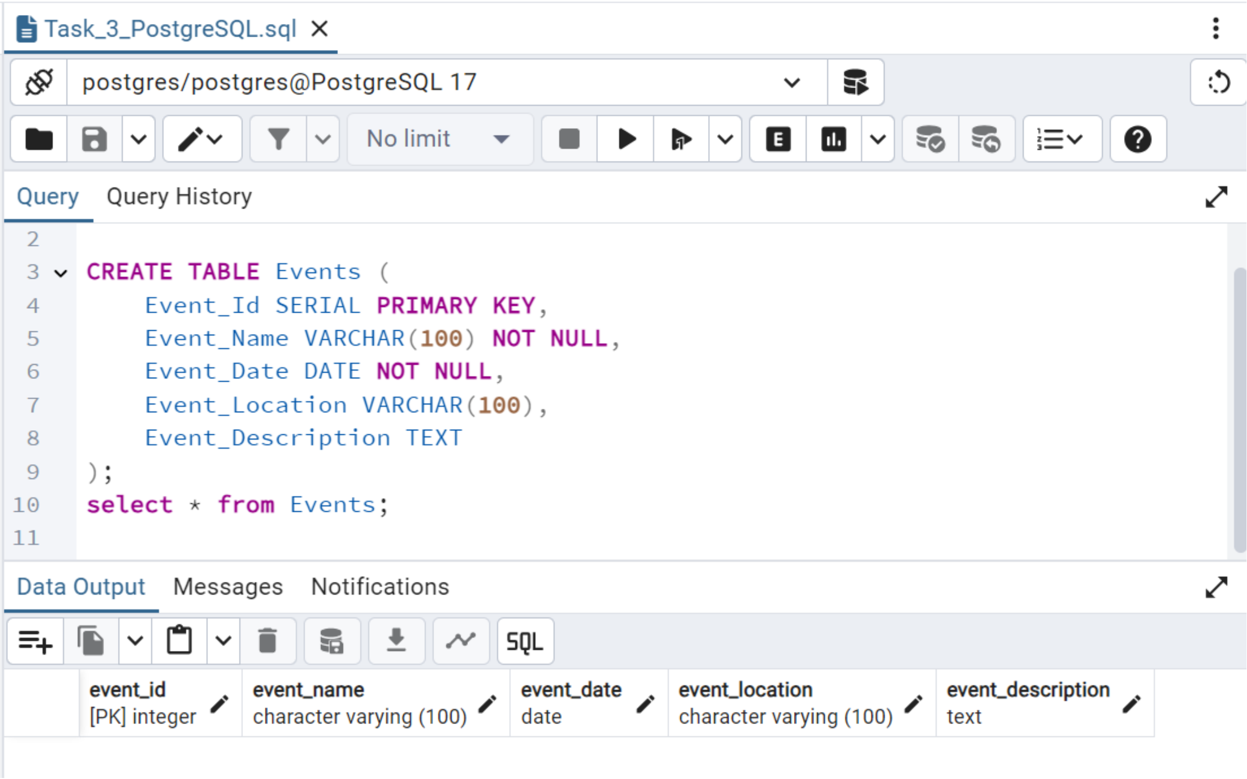
a. Create a database named "EventsManagement."



```
CREATE DATABASE EventsManagement;
```

b. Create tables for Events, Attendees, and Registrations.

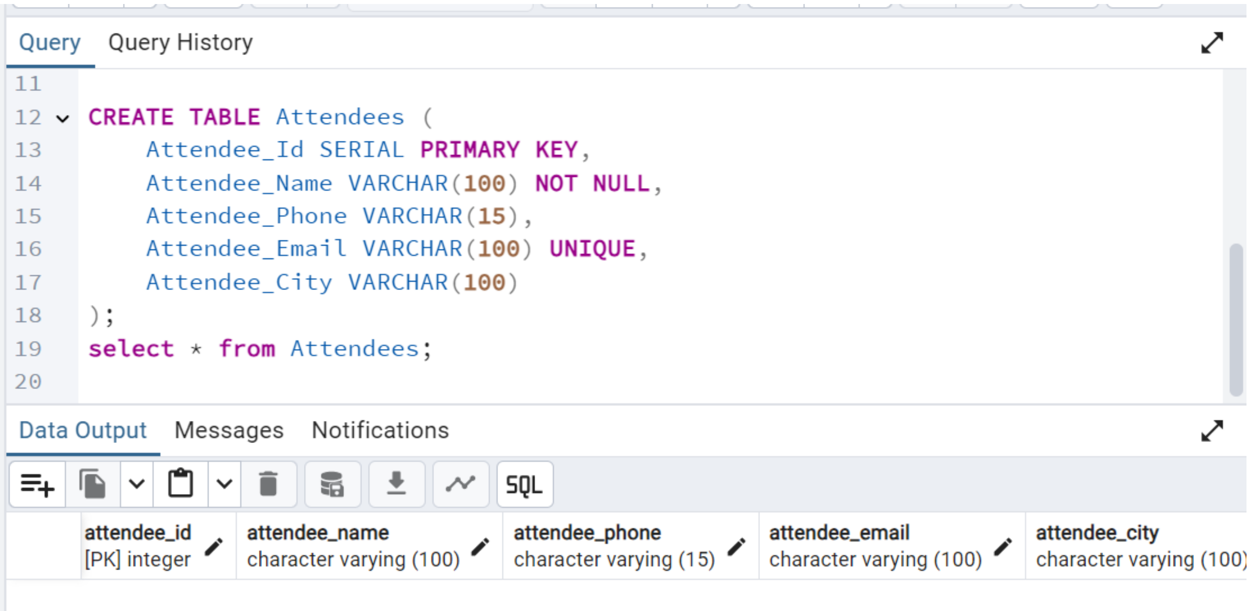
i. Create Events Table



```
CREATE TABLE Events (
    Event_Id SERIAL PRIMARY KEY,
    Event_Name VARCHAR(100) NOT NULL,
    Event_Date DATE NOT NULL,
    Event_Location VARCHAR(100),
    Event_Description TEXT
);
select * from Events;
```

| event_id<br>[PK] integer | event_name<br>character varying (100) | event_date<br>date | event_location<br>character varying (100) | event_description<br>text |
|---|---|---|---|---|

ii. Create Attendees Table



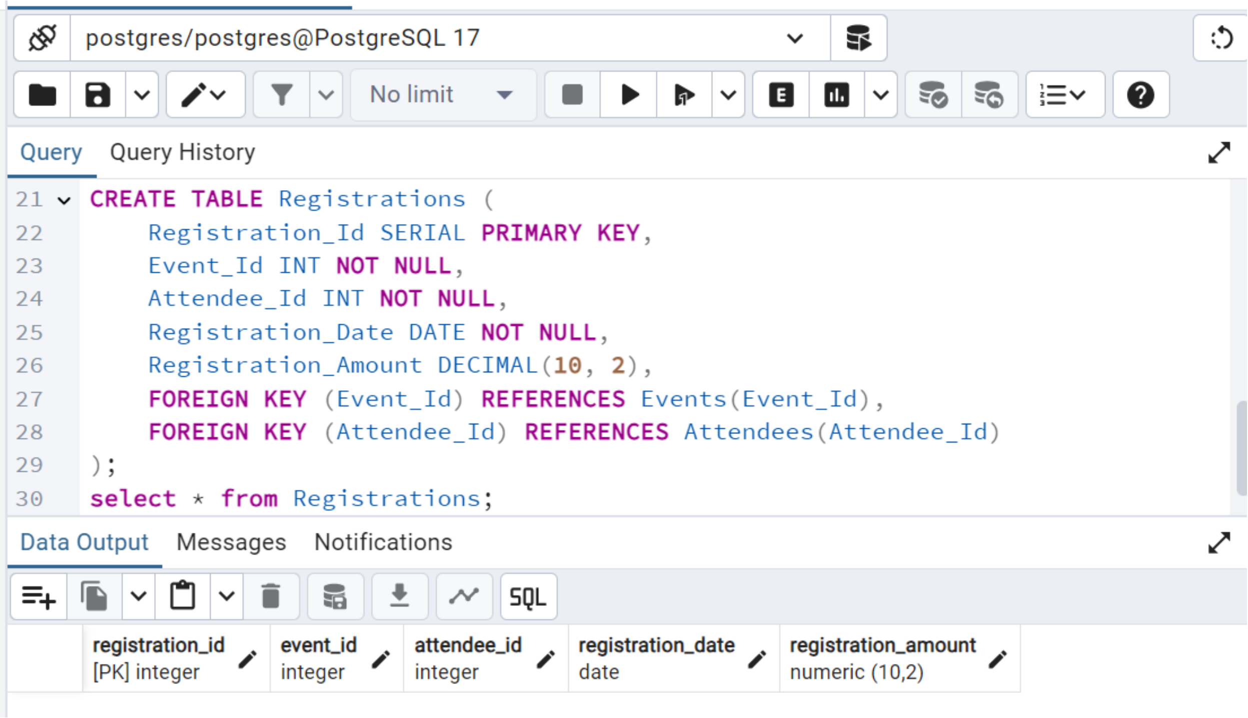
```sql

CREATE TABLE Attendees (
    Attendee_Id SERIAL PRIMARY KEY,
    Attendee_Name VARCHAR(100) NOT NULL,
    Attendee_Phone VARCHAR(15),
    Attendee_Email VARCHAR(100) UNIQUE,
    Attendee_City VARCHAR(100)
);
select * from Attendees;

```

| attendee_id<br>[PK] integer | attendee_name<br>character varying (100) | attendee_phone<br>character varying (15) | attendee_email<br>character varying (100) | attendee_city<br>character varying (100) |
|---|---|---|---|---|

iii. Create Registrations Table



postgres/postgres@PostgreSQL 17

No limit

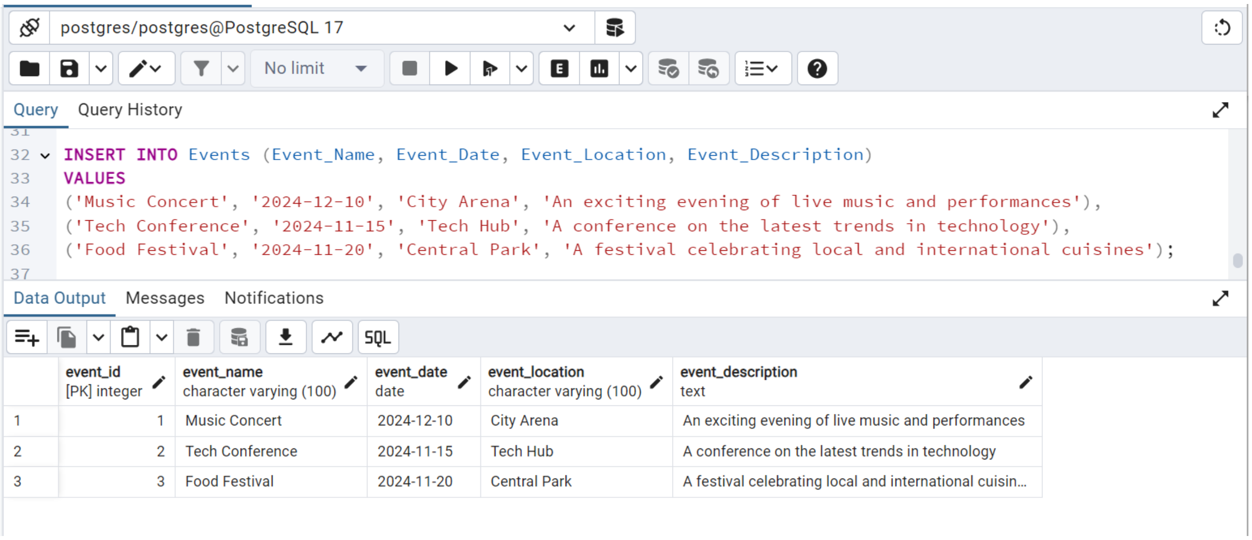
```sql
CREATE TABLE Registrations (
    Registration_Id SERIAL PRIMARY KEY,
    Event_Id INT NOT NULL,
    Attendee_Id INT NOT NULL,
    Registration_Date DATE NOT NULL,
    Registration_Amount DECIMAL(10, 2),
    FOREIGN KEY (Event_Id) REFERENCES Events(Event_Id),
    FOREIGN KEY (Attendee_Id) REFERENCES Attendees(Attendee_Id)
);
select * from Registrations;
```

| registration_id<br>[PK] integer | event_id<br>integer | attendee_id<br>integer | registration_date<br>date | registration_amount<br>numeric (10,2) |
|---|---|---|---|---|

## 2. Data Creation

Insert some sample data for Events, Attendees, and Registrations tables with respective fields.

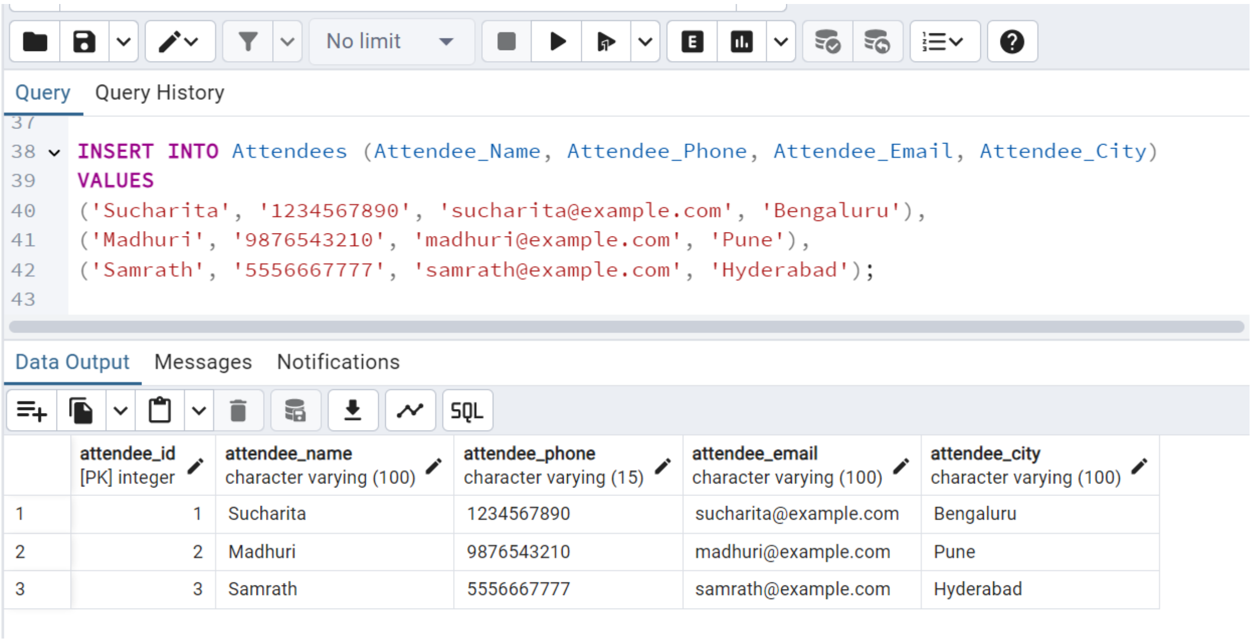
a. Insert data in Events Table



```
INSERT INTO Events (Event_Name, Event_Date, Event_Location, Event_Description)
VALUES
('Music Concert', '2024-12-10', 'City Arena', 'An exciting evening of live music and performances'),
('Tech Conference', '2024-11-15', 'Tech Hub', 'A conference on the latest trends in technology'),
('Food Festival', '2024-11-20', 'Central Park', 'A festival celebrating local and international cuisines');
```

| | event_id<br>[PK] integer | event_name<br>character varying (100) | event_date<br>date | event_location<br>character varying (100) | event_description<br>text |
|---|---|---|---|---|---|
| 1 | 1 | Music Concert | 2024-12-10 | City Arena | An exciting evening of live music and performances |
| 2 | 2 | Tech Conference | 2024-11-15 | Tech Hub | A conference on the latest trends in technology |
| 3 | 3 | Food Festival | 2024-11-20 | Central Park | A festival celebrating local and international cuisin... |

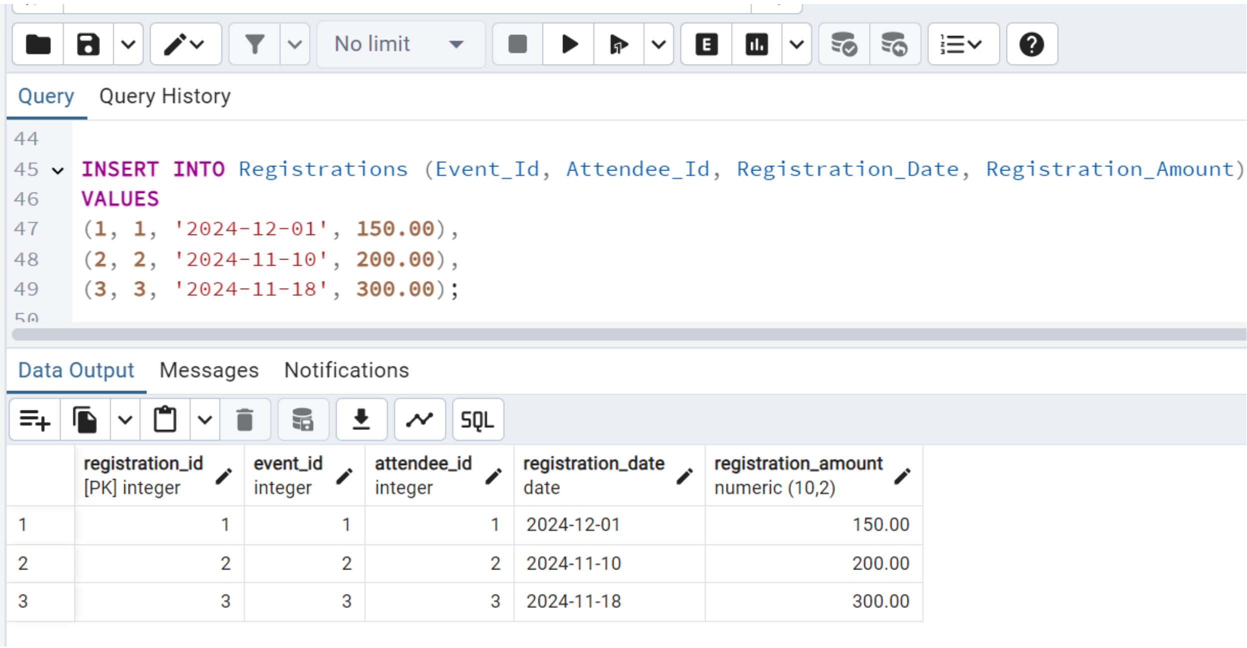
b. Insert data in Attendees Table



```
INSERT INTO Attendees (Attendee_Name, Attendee_Phone, Attendee_Email, Attendee_City)
VALUES
('Sucharita', '1234567890', 'sucharita@example.com', 'Bengaluru'),
('Madhuri', '9876543210', 'madhuri@example.com', 'Pune'),
('Samrath', '5556667777', 'samrath@example.com', 'Hyderabad');
```

| | attendee_id<br>[PK] integer | attendee_name<br>character varying (100) | attendee_phone<br>character varying (15) | attendee_email<br>character varying (100) | attendee_city<br>character varying (100) |
|---|---|---|---|---|---|
| 1 | 1 | Sucharita | 1234567890 | sucharita@example.com | Bengaluru |
| 2 | 2 | Madhuri | 9876543210 | madhuri@example.com | Pune |
| 3 | 3 | Samrath | 5556667777 | samrath@example.com | Hyderabad |

c. Insert data in Registrations Table



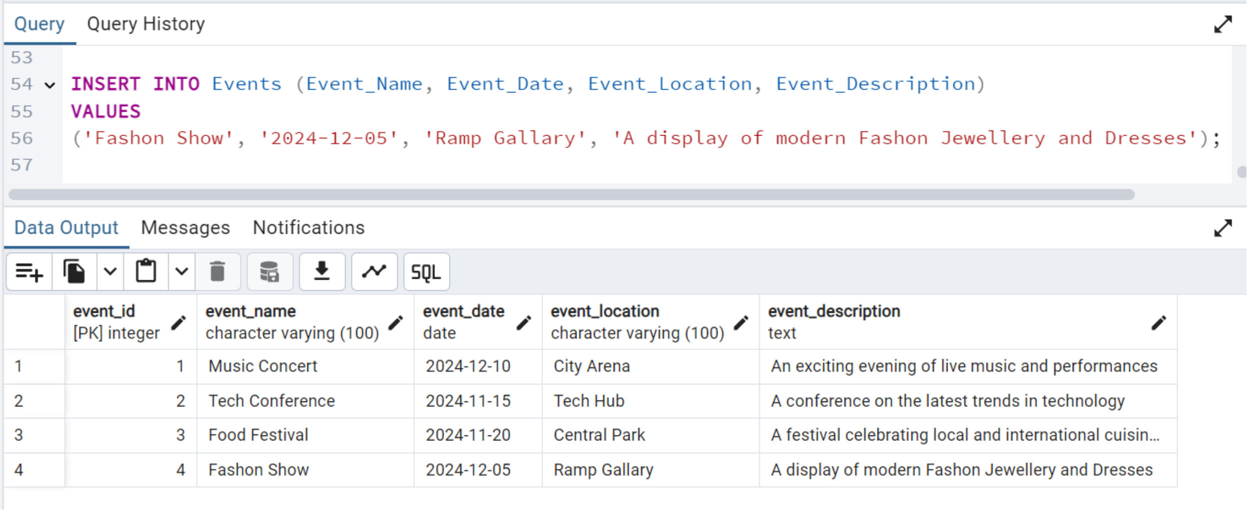
```sql

INSERT INTO Registrations (Event_Id, Attendee_Id, Registration_Date, Registration_Amount)
VALUES
(1, 1, '2024-12-01', 150.00),
(2, 2, '2024-11-10', 200.00),
(3, 3, '2024-11-18', 300.00);
```

| | registration_id<br>[PK] integer | event_id<br>integer | attendee_id<br>integer | registration_date<br>date | registration_amount<br>numeric (10,2) |
|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 2024-12-01 | 150.00 |
| 2 | 2 | 2 | 2 | 2024-11-10 | 200.00 |
| 3 | 3 | 3 | 3 | 2024-11-18 | 300.00 |

## 3. Manage Event Details

a) Inserting a new event.



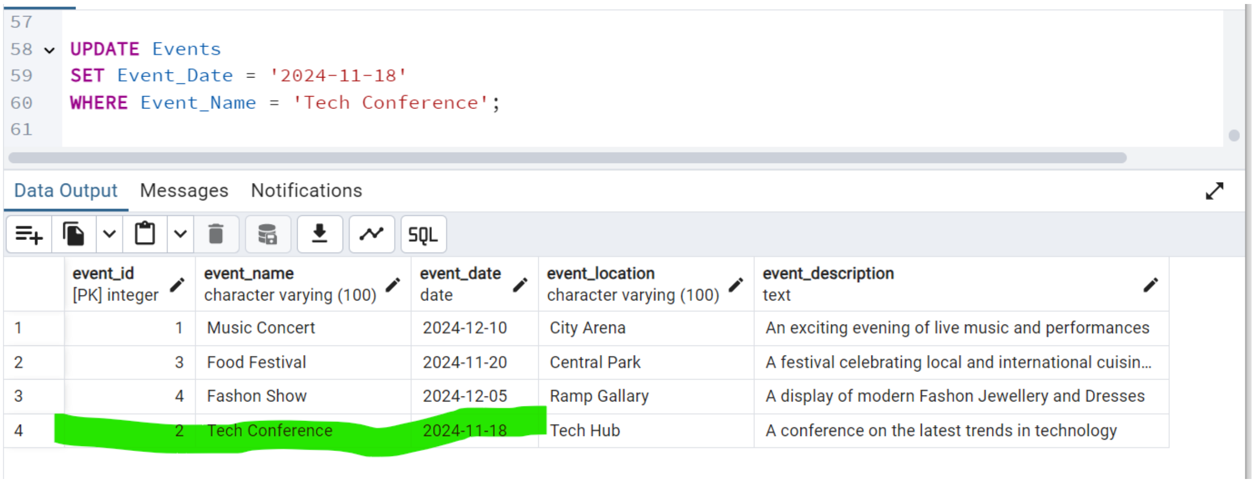
```sql

INSERT INTO Events (Event_Name, Event_Date, Event_Location, Event_Description)
VALUES
('Fashon Show', '2024-12-05', 'Ramp Gallary', 'A display of modern Fashon Jewellery and Dresses');

```

| | event_id<br>[PK] integer | event_name<br>character varying (100) | event_date<br>date | event_location<br>character varying (100) | event_description<br>text |
|---|---|---|---|---|---|
| 1 | 1 | Music Concert | 2024-12-10 | City Arena | An exciting evening of live music and performances |
| 2 | 2 | Tech Conference | 2024-11-15 | Tech Hub | A conference on the latest trends in technology |
| 3 | 3 | Food Festival | 2024-11-20 | Central Park | A festival celebrating local and international cuisin... |
| 4 | 4 | Fashon Show | 2024-12-05 | Ramp Gallary | A display of modern Fashon Jewellery and Dresses |

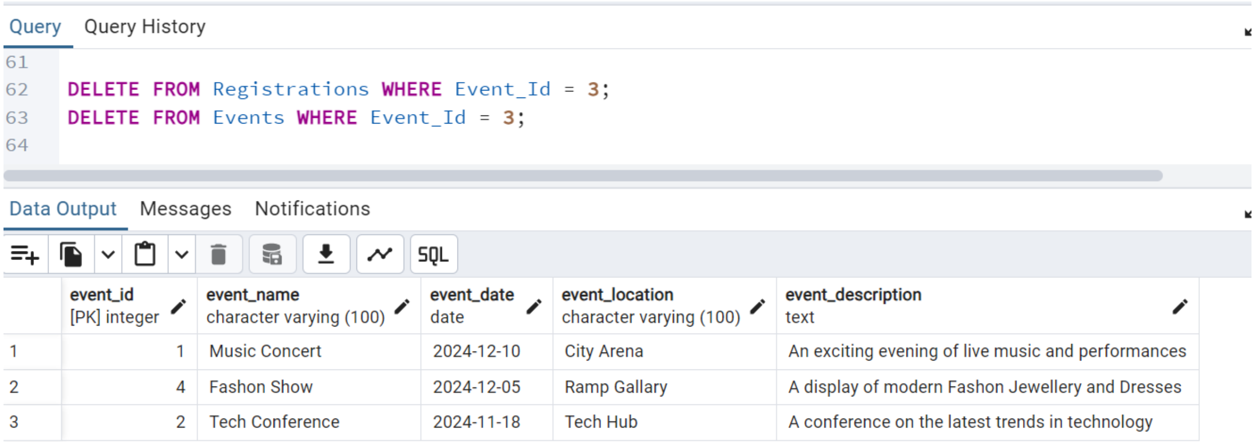
b) Updating an event's information.



```sql
UPDATE Events
SET Event_Date = '2024-11-18'
WHERE Event_Name = 'Tech Conference';
```

| | event_id<br>[PK] integer | event_name<br>character varying (100) | event_date<br>date | event_location<br>character varying (100) | event_description<br>text |
|---|---|---|---|---|---|
| 1 | 1 | Music Concert | 2024-12-10 | City Arena | An exciting evening of live music and performances |
| 2 | 3 | Food Festival | 2024-11-20 | Central Park | A festival celebrating local and international cuisin... |
| 3 | 4 | Fashon Show | 2024-12-05 | Ramp Gallary | A display of modern Fashon Jewellery and Dresses |
| 4 | 2 | Tech Conference | 2024-11-18 | Tech Hub | A conference on the latest trends in technology |

c) Deleting an event.

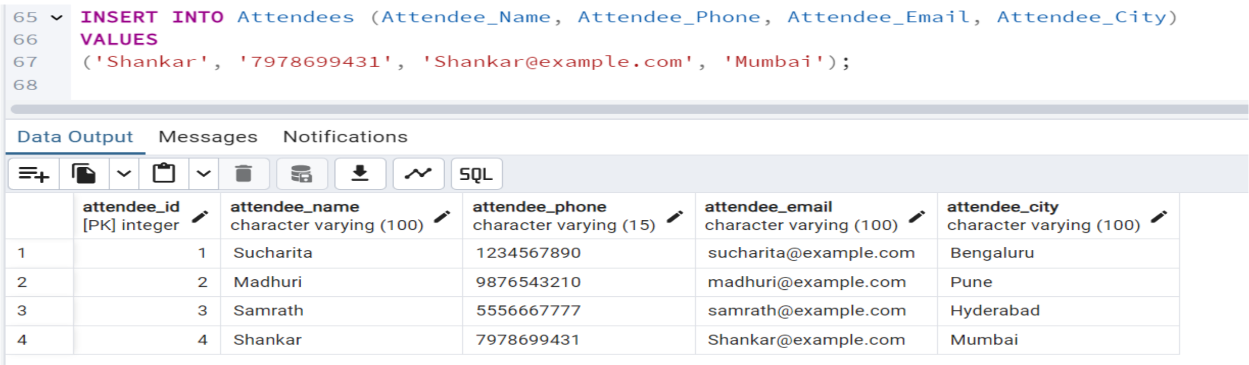


```sql
DELETE FROM Registrations WHERE Event_Id = 3;
DELETE FROM Events WHERE Event_Id = 3;
```

| | event_id<br>[PK] integer | event_name<br>character varying (100) | event_date<br>date | event_location<br>character varying (100) | event_description<br>text |
|---|---|---|---|---|---|
| 1 | 1 | Music Concert | 2024-12-10 | City Arena | An exciting evening of live music and performances |
| 2 | 4 | Fashon Show | 2024-12-05 | Ramp Gallary | A display of modern Fashon Jewellery and Dresses |
| 3 | 2 | Tech Conference | 2024-11-18 | Tech Hub | A conference on the latest trends in technology |

## 4) Manage Track Attendees & Handle Events

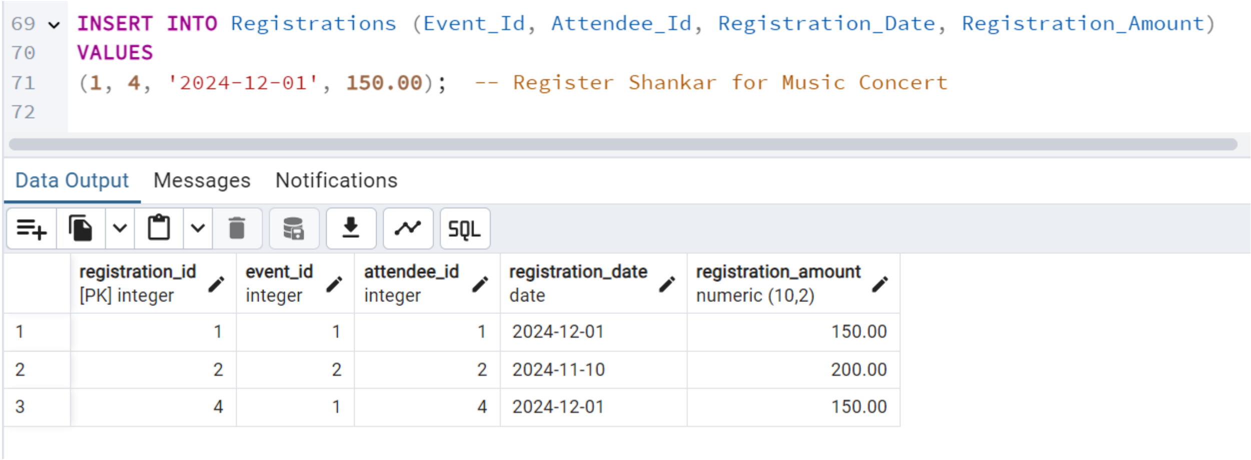
a) Inserting a new attendee.



```sql
INSERT INTO Attendees (Attendee_Name, Attendee_Phone, Attendee_Email, Attendee_City)
VALUES
('Shankar', '7978699431', 'Shankar@example.com', 'Mumbai');
```

| | attendee_id<br>[PK] integer | attendee_name<br>character varying (100) | attendee_phone<br>character varying (15) | attendee_email<br>character varying (100) | attendee_city<br>character varying (100) |
|---|---|---|---|---|---|
| 1 | 1 | Sucharita | 1234567890 | sucharita@example.com | Bengaluru |
| 2 | 2 | Madhuri | 9876543210 | madhuri@example.com | Pune |
| 3 | 3 | Samrath | 5556667777 | samrath@example.com | Hyderabad |
| 4 | 4 | Shankar | 7978699431 | Shankar@example.com | Mumbai |

b) Registering an attendee for an event.

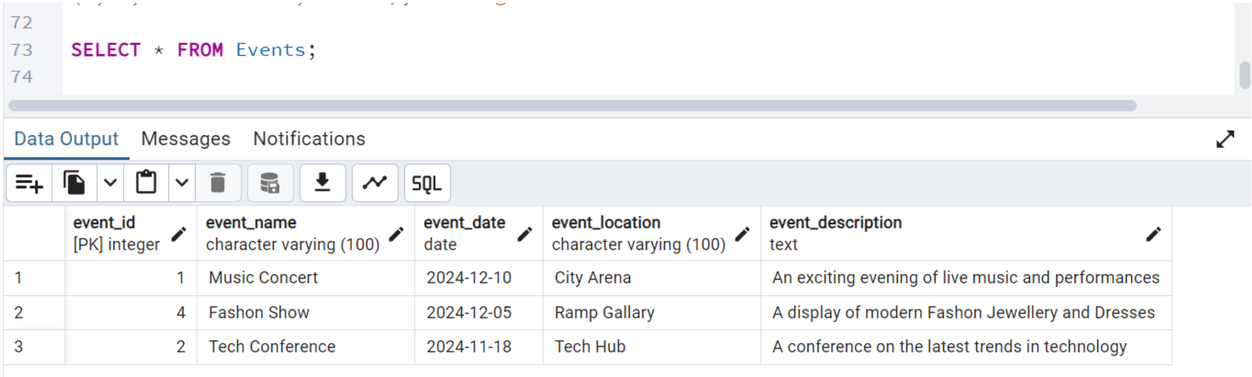


```sql
INSERT INTO Registrations (Event_Id, Attendee_Id, Registration_Date, Registration_Amount)
VALUES
(1, 4, '2024-12-01', 150.00);  -- Register Shankar for Music Concert
```

| | registration_id<br>[PK] integer | event_id<br>integer | attendee_id<br>integer | registration_date<br>date | registration_amount<br>numeric (10,2) |
|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 2024-12-01 | 150.00 |
| 2 | 2 | 2 | 2 | 2024-11-10 | 200.00 |
| 3 | 4 | 1 | 4 | 2024-12-01 | 150.00 |

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

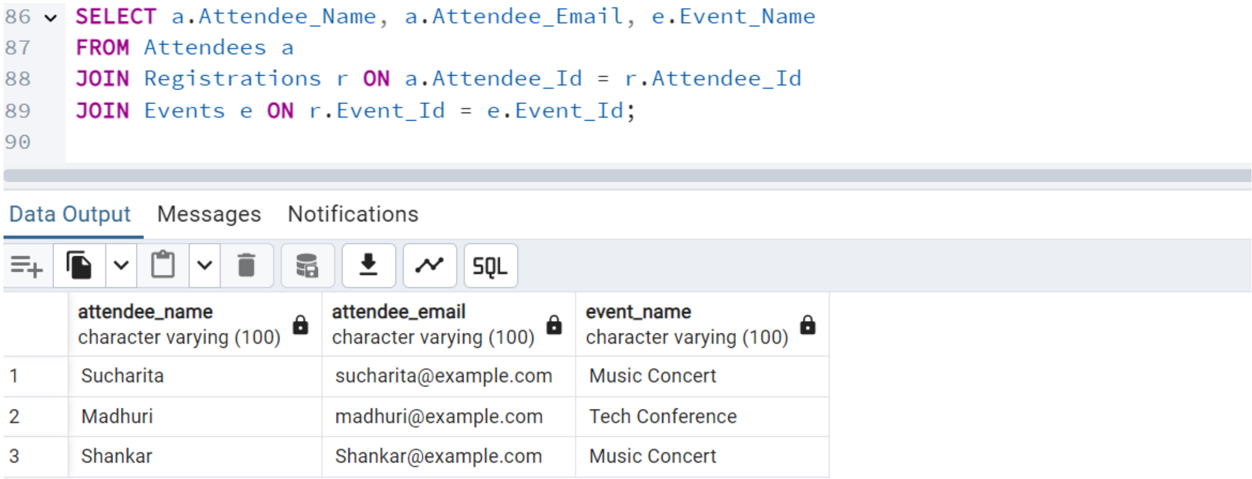
a) To retrieve event information



```sql
SELECT * FROM Events;
```

| | event_id<br>[PK] integer | event_name<br>character varying (100) | event_date<br>date | event_location<br>character varying (100) | event_description<br>text |
|---|---|---|---|---|---|
| 1 | 1 | Music Concert | 2024-12-10 | City Arena | An exciting evening of live music and performances |
| 2 | 4 | Fashon Show | 2024-12-05 | Ramp Gallary | A display of modern Fashon Jewellery and Dresses |
| 3 | 2 | Tech Conference | 2024-11-18 | Tech Hub | A conference on the latest trends in technology |

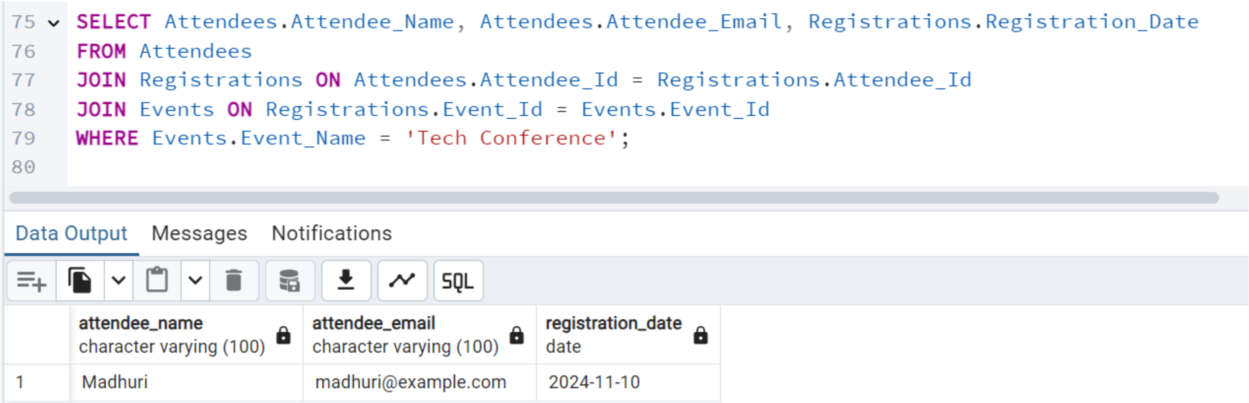
b) To generate attendee lists



```sql
SELECT a.Attendee_Name, a.Attendee_Email, e.Event_Name
FROM Attendees a
JOIN Registrations r ON a.Attendee_Id = r.Attendee_Id
JOIN Events e ON r.Event_Id = e.Event_Id;
```

| | attendee_name<br>character varying (100) | attendee_email<br>character varying (100) | event_name<br>character varying (100) |
|---|---|---|---|
| 1 | Sucharita | sucharita@example.com | Music Concert |
| 2 | Madhuri | madhuri@example.com | Tech Conference |
| 3 | Shankar | Shankar@example.com | Music Concert |

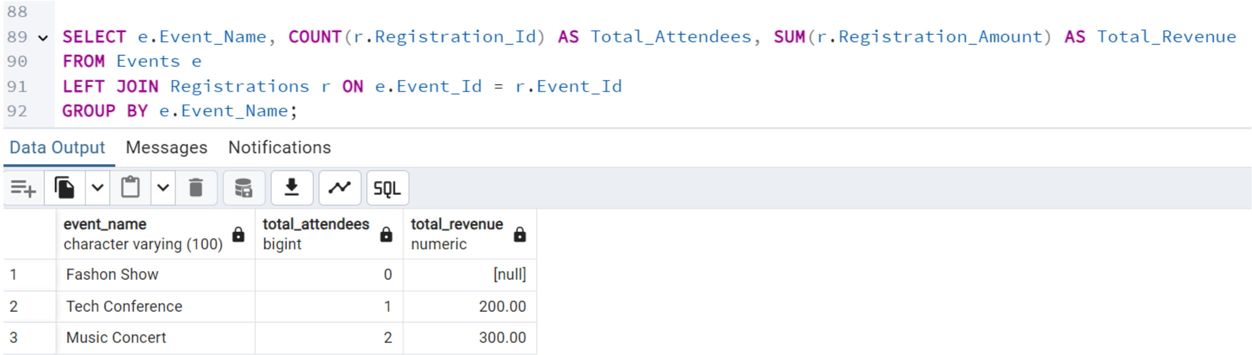
c) To generate attendee list for a specific Event (Tech Conference)



```sql
SELECT Attendees.Attendee_Name, Attendees.Attendee_Email, Registrations.Registration_Date
FROM Attendees
JOIN Registrations ON Attendees.Attendee_Id = Registrations.Attendee_Id
JOIN Events ON Registrations.Event_Id = Events.Event_Id
WHERE Events.Event_Name = 'Tech Conference';
```

| | attendee_name<br>character varying (100) | attendee_email<br>character varying (100) | registration_date<br>date |
|---|---|---|---|
| 1 | Madhuri | madhuri@example.com | 2024-11-10 |

d) To calculate event attendance statistics



```sql
SELECT e.Event_Name, COUNT(r.Registration_Id) AS Total_Attendees, SUM(r.Registration_Amount) AS Total_Revenue
FROM Events e
LEFT JOIN Registrations r ON e.Event_Id = r.Event_Id
GROUP BY e.Event_Name;
```

| | event_name<br>character varying (100) | total_attendees<br>bigint | total_revenue<br>numeric |
|---|---|---|---|
| 1 | Fashon Show | 0 | [null] |
| 2 | Tech Conference | 1 | 200.00 |
| 3 | Music Concert | 2 | 300.00 |

## Query Explanation:-

## 1. Database Creation

```sql
CREATE DATABASE EventsManagement;
```

## 2. Create Tables

### a. Create the Events table

```sql
CREATE TABLE Events (

    Event_Id SERIAL PRIMARY KEY,

    Event_Name VARCHAR(100) NOT NULL,

    Event_Date DATE NOT NULL,

    Event_Location VARCHAR(100),

    Event_Description TEXT
);

select * from Events;
```

### Explanation:-

- The Event_Id and Attendee_Id are being auto-generated as sequential integers because the SERIAL data type was used for those columns. This ensures that each ID is unique and incremented automatically when a new record is inserted, simplifying the database design and management.
- Using SERIAL, prevents duplicate entry error. If we try to insert a row with a duplicate Event_Id or Attendee_Id, and the column is a primary key, the insertion will fail.
- Similarly, in MySQL we can use AUTO_INCREMENT in place of SERIAL.

### b. Create the Attendees table

```
CREATE TABLE Attendees (

  Attendee_Id SERIAL PRIMARY KEY,

  Attendee_Name VARCHAR(100) NOT NULL,

  Attendee_Phone VARCHAR(15),

  Attendee_Email VARCHAR(100) UNIQUE,

  Attendee_City VARCHAR(100)

);

select * from Attendees;
```

### c. Create the Registrations table

```
CREATE TABLE Registrations (

  Registration_Id SERIAL PRIMARY KEY,

  Event_Id INT NOT NULL,

  Attendee_Id INT NOT NULL,

  Registration_Date DATE NOT NULL,

  Registration_Amount DECIMAL(10, 2),

  FOREIGN KEY (Event_Id) REFERENCES Events(Event_Id),

  FOREIGN KEY (Attendee_Id) REFERENCES Attendees(Attendee_Id)

);

select * from Registrations;
```

## 3. Data Creation

### a. Inserting sample data for Events

```
INSERT INTO Events (Event_Name, Event_Date, Event_Location, Event_Description)
VALUES
('Music Concert', '2024-12-10', 'City Arena', 'An exciting evening of live music and performances'),
('Tech Conference', '2024-11-15', 'Tech Hub', 'A conference on the latest trends in technology'),
('Food Festival', '2024-11-20', 'Central Park', 'A festival celebrating local and international cuisines');
```

### b. Inserting sample data for Attendees

```
INSERT INTO Attendees (Attendee_Name, Attendee_Phone, Attendee_Email, Attendee_City)
VALUES
('Sucharita', '1234567890', 'sucharita@example.com', 'Bengaluru'),
('Madhuri', '9876543210', 'madhuri@example.com', 'Pune'),
('Samrath', '5556667777', 'samrath@example.com', 'Hyderabad');
```

**Explanation:-**

- Sucharita registered for Music Concert
- Madhuri registered for Tech Conference
- Samrath registered for Food Festival

### b. Inserting sample data for Registrations

```
INSERT INTO Registrations (Event_Id, Attendee_Id, Registration_Date, Registration_Amount)
VALUES
(1, 1, '2024-12-01', 150.00),
(2, 2, '2024-11-10', 200.00),
(3, 3, '2024-11-18', 300.00);
```

## 4. Manage Event Details

### a) **Inserting a New Event**

To insert a new event into the Events table:

```
INSERT INTO Events (Event_Name, Event_Date, Event_Location, Event_Description)
VALUES
```

```
('Fashon Show', '2024-12-05', 'Ramp Gallary', 'A display of modern Fashon Jewellery and Dresses');
```

### b) Updating an Event's Information

To update an existing event's details (e.g., changing the date of the "Tech Conference"):

```
UPDATE Events
SET Event_Date = '2024-11-18'
WHERE Event_Name = 'Tech Conference';
```

### c) Deleting an Event

To delete an event (e.g., delete the "Food Festival")

```
DELETE FROM Registrations WHERE Event_Id = 3;
DELETE FROM Events WHERE Event_Id = 3;
```

## 5. Manage Track Attendees & Handle Events

### a) Inserting a New Attendee

To insert a new attendee Shankar

```
INSERT INTO Attendees (Attendee_Name, Attendee_Phone, Attendee_Email, Attendee_City)
VALUES
('Shankar', '7978699431', 'Shankar@example.com', 'Mumbai');
```

### b) Registering an Attendee for an Event

```
INSERT INTO Registrations (Event_Id, Attendee_Id, Registration_Date, Registration_Amount)
VALUES
(1, 4, '2024-12-01', 150.00);  -- Register Shankar for Music Concert
```

## 6. Develop Queries to Retrieve Event Information and Attendance Statistics

### a) Retrieve All Event Information

```
SELECT * FROM Events;
```

### b) Retrieve List of Attendees for a Specific Event

To generate attendee list for a specific Event for ex:- Tech Conference

```
SELECT Attendees.Attendee_Name, Attendees.Attendee_Email, Registrations.Registration_Date
FROM Attendees
```

```
JOIN Registrations ON Attendees.Attendee_Id = Registrations.Attendee_Id

JOIN Events ON Registrations.Event_Id = Events.Event_Id

WHERE Events.Event_Name = 'Tech Conference';
```

### c) To generate attendee lists

```
SELECT a.Attendee_Name, a.Attendee_Email, e.Event_Name

FROM Attendees a

JOIN Registrations r ON a.Attendee_Id = r.Attendee_Id

JOIN Events e ON r.Event_Id = e.Event_Id;
```

### d) To calculate event attendance statistics

```
SELECT e.Event_Name, COUNT(r.Registration_Id) AS Total_Attendees, SUM(r.Registration_Amount) AS Total_Revenue

FROM Events e

LEFT JOIN Registrations r ON e.Event_Id = r.Event_Id

GROUP BY e.Event_Name;
```