Event Management System using PostgreSQL.

Objective:

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

1.Database Creation

Create a database named "EventsManagement." Create tables for Events, Attendees, and Registrations. Events- Event_Id, Event_Name, Event_Date, Event_Location, Event_Description Attendees- Attendee_Id, Attendee_Name, Attendee_Phone, Attendee_Email, Attendee_City Registrations-Registration_id, Event_Id, Attendee_Id,Registration_Date,Registration_Amount. The FOREIGN KEY constraint in the Registrations table references the Event_Id column in the Events table and the Attendee_Id column in the Attendees table.

Query:

```
CREATE DATABASE EventsManagement;

\c EventsManagement

CREATE TABLE Events (Event_Id SERIAL PRIMARY KEY, Event_Name VARCHAR(100) NOT NULL, Event_Date DATE NOT NULL, Event_Location VARCHAR(200) NOT NULL, Event_Description TEXT);

CREATE TABLE Attendees (Attendee_Id SERIAL PRIMARY KEY, Attendee_Name VARCHAR(100) NOT NULL, Attendee_Phone VARCHAR(15), Attendee_Email VARCHAR(100) UNIQUE NOT NULL, Attendee_City VARCHAR(50));

CREATE TABLE Registrations (Registration_id SERIAL PRIMARY KEY, Event_Id INTEGER REFERENCES Events(Event_Id), Attendee_Id INTEGER REFERENCES Attendees(Attendee_Id), Registration_Date DATE DEFAULT CURRENT_DATE, Registration_Amount DECIMAL(10,2), UNIQUE(Event_Id, Attendee_Id));
```

2.Data Creation

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

Query:

INSERT INTO Events (Event_Name, Event_Date, Event_Location, Event_Description) VALUES ('Tech Conference 2024', '2024-03-15', 'Convention Center', 'Annual technology conference'), ('Music Festival', '2024-04-20', 'City Park', 'Summer music festival'), ('Business Summit', '2024-05-10', 'Grand Hotel', 'Leadership and innovation summit'), ('Art Exhibition', '2024-06-01', 'Art Gallery', 'Modern art showcase'), ('Food Festival', '2024-07-15', 'Downtown Square', 'International cuisine festival');

INSERT INTO Attendees (Attendee_Name, Attendee_Phone, Attendee_Email, Attendee_City) VALUES ('John Smith', '1234567890', 'john@email.com', 'New York'), ('Sarah Johnson', [REDACTED:BANK_ACCOUNT_NUMBER], 'sarah@email.com', 'Los Angeles'), ('Michael Brown', [REDACTED:BANK_ACCOUNT_NUMBER], 'michael@email.com', 'Chicago'), ('Emily Davis', [REDACTED:BANK_ACCOUNT_NUMBER], 'emily@email.com', 'Houston'), ('David Wilson', [REDACTED:BANK_ACCOUNT_NUMBER], 'david@email.com', 'Phoenix');

INSERT INTO Registrations (Event_Id, Attendee_Id, Registration_Amount) VALUES (1, 1, 100.00), (1, 2, 100.00), (2, 3, 75.50), (3, 4, 150.00), (4, 5, 50.00);

INSERT INTO Events (Event_Name, Event_Date, Event_Location, Event_Description) VALUES ('Workshop 2024', '2024-08-20', 'Training Center', 'Technical workshop');

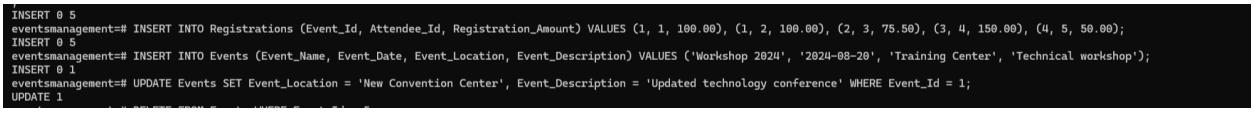
3. Manage Event Details

a) Inserting a new event.

Query:

INSERT INTO Events (Event_Name, Event_Date, Event_Location, Event_Description) VALUES ('Workshop 2024', '2024-08-20', 'Training Center', 'Technical workshop');

Output:

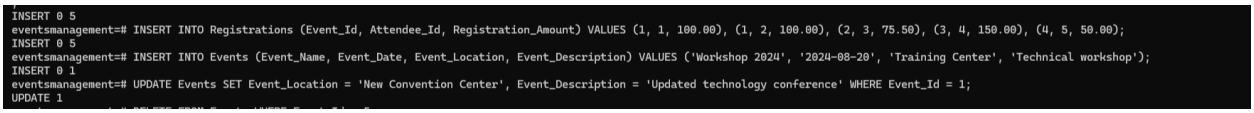


b) Updating an event's information.

Query:

UPDATE Events SET Event_Location = 'New Convention Center', Event_Description = 'Updated technology conference' WHERE Event_Id = 1;

Output:



c) Deleting an event.

Query:

DELETE FROM Events WHERE Event_Id = 5;

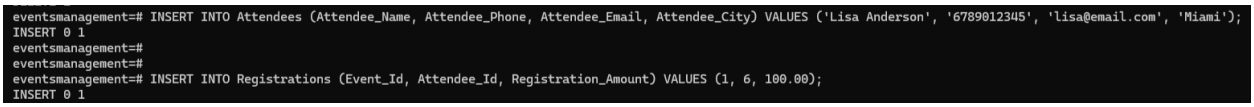
4) Manage Track Attendees & Handle Events

a)Inserting a new attendee.

Query:

INSERT INTO Attendees (Attendee_Name, Attendee_Phone, Attendee_Email, Attendee_City) VALUES ('Lisa Anderson', '6789012345', 'lisa@email.com', 'Miami');

Output:

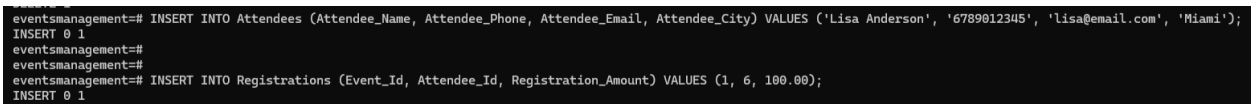


b)Registering an attendee for an event.

Query:

INSERT INTO Registrations (Event_Id, Attendee_Id, Registration_Amount) VALUES (1, 6, 100.00);

Output:



Query:

SELECT e.Event_Id, e.Event_Name, e.Event_Date, COUNT(r.Attendee_Id) as Attendee_Count FROM Events e LEFT JOIN Registrations r ON e.Event_Id = r.Event_Id GROUP BY e.Event_Id, e.Event_Name, e.Event_Date ORDER BY e.Event_Date;

Output:

```
eventsmanagement=# SELECT e.Event_Id, e.Event_Name, e.Event_Date, COUNT(r.Attendee_Id) as Attendee_Count FROM Events e LEFT JOIN Registrations r ON e.Event_Id = r.Event_Id GROUP BY e.Event_Id, e.Event_Name, e
.Event_Date ORDER BY e.Event_Date;
 event_id |      event_name      | event_date | attendee_count
----------+----------------------+------------+----------------
        1 | Tech Conference 2024 | 2024-03-15 |              3
        2 | Music Festival       | 2024-04-20 |              1
        3 | Business Summit      | 2024-05-10 |              1
        4 | Art Exhibition       | 2024-06-01 |              1
        6 | Workshop 2024        | 2024-08-20 |              0
(5 rows)
```

5.Develop queries:

To retrieve event information,

-- Calculate total registration amount by event

Query:

SELECT Event_Name, Event_Date FROM Events e LEFT JOIN Registrations r ON e.Event_Id = r.Event_Id WHERE r.Registration_id IS NULL;

Output:

```
eventsmanagement=#
eventsmanagement=#
eventsmanagement=# SELECT Event_Name, Event_Date FROM Events e LEFT JOIN Registrations r ON e.Event_Id = r.Event_Id WHERE r.Registration_id IS NULL;
  event_name   | event_date
---------------+------------
 Workshop 2024 | 2024-08-20
(1 row)
```

-- Find events with no registrations

Query:

SELECT a.Attendee_Name, COUNT(r.Event_Id) as Event_Count FROM Attendees a JOIN Registrations r ON a.Attendee_Id = r.Attendee_Id GROUP BY a.Attendee_Name HAVING COUNT(r.Event_Id) > 1;

Output:

```
eventsmanagement=#
eventsmanagement=#
eventsmanagement=# SELECT a.Attendee_Name, COUNT(r.Event_Id) as Event_Count FROM Attendees a JOIN Registrations r ON a.Attendee_Id = r.Attendee_Id GROUP BY a.Attendee_Name HAVING COUNT(r.Event_Id) > 1;
 attendee_name | event_count
---------------+-------------
(0 rows)
```

-- Get upcoming events

Query:

SELECT Event_Name, Event_Date, Event_Location FROM EventsWHERE Event_Date > CURRENT_DATEORDER BY Event_Date;

Output:

```
eventsmanagement=#
eventsmanagement=# SELECT Event_Name, Event_Date, Event_Location FROM Events WHERE Event_Date > CURRENT_DATE ORDER BY Event_Date;
 event_name | event_date | event_location
------------+------------+----------------
(0 rows)
```

generate attendee lists calculate event attendance statistics.

-- List all events with their attendee count

Query:

SELECT a.Attendee_Name, a.Attendee_Email, a.Attendee_City, r.Registration_Date FROM Attendees a JOIN Registrations r ON a.Attendee_Id = r.Attendee_Id WHERE r.Event_Id = 1;

Output:

```
eventsmanagement=#
eventsmanagement=# SELECT a.Attendee_Name, a.Attendee_Email, a.Attendee_City, r.Registration_Date FROM Attendees a JOIN Registrations r ON a.Attendee_Id = r.Attendee_Id WHERE r.Event_Id = 1;
 attendee_name | attendee_email  | attendee_city | registration_date
---------------+-----------------+---------------+-------------------
 John Smith    | john@email.com  | New York      | 2025-08-25
 Sarah Johnson | sarah@email.com | Los Angeles   | 2025-08-25
 Lisa Anderson | lisa@email.com  | Miami         | 2025-08-25
(3 rows)
```

-- List attendees for a specific event

Query:

SELECT e.Event_Name, SUM(r.Registration_Amount) as Total_Amount, COUNT(r.Attendee_Id) as Attendee_Count FROM Events e LEFT JOIN Registrations r ON e.Event_Id = r.Event_Id GROUP BY e.Event_Name;

Output:

```
eventsmanagement=#
eventsmanagement=#
eventsmanagement=#
eventsmanagement=# SELECT e.Event_Name, SUM(r.Registration_Amount) as Total_Amount, COUNT(r.Attendee_Id) as Attendee_Count FROM Events e LEFT JOIN Registrations r ON e.Event_Id = r.Event_Id GROUP BY e.Event_N
ame;
      event_name      | total_amount | attendee_count
----------------------+--------------+----------------
 Music Festival       |        75.50 |              1
 Workshop 2024        |              |              0
 Tech Conference 2024 |       300.00 |              3
 Business Summit      |       150.00 |              1
 Art Exhibition       |        50.00 |              1
(5 rows)
```