PROJECT\_REPORT\_DATABASE SYSTEM

Creating a database system that captures detailed information about professors, their affiliations with universities and organizations, and their respective roles

MUHAMMAD HUSSAIN

S23BDOCS1M01147

**Background:**

The project involves managing the relationships between professors, universities, and organizations. Currently, there is a need to track which professors are associated with which universities and organizations. The challenge is to centralize this information in a well-structured system that can provide easy access to this data while maintaining data integrity and handling multiple relationships effectively. The project addresses this need by designing a normalized relational database system that organizes professors, universities, organizations, and their affiliations in a way that is both scalable and easy to query.

**Goal:**

The primary goal of this project is to create a database system that captures detailed information about professors, their affiliations with universities and organizations, and their respective roles. By the end of the project, the system should be able to:

* Store and retrieve information about professors, universities, organizations, and their relationships.
* Allow users to easily update, delete, and search for information related to professors and their affiliations.
* Ensure referential integrity between professors, universities, and organizations through well-defined relationships.
* Provide functionality for querying complex data, such as listing all professors associated with a specific university or organization.

**Requirements:**

The system must meet the following user requirements:

* **Users**: The main users of this system will be administrators and researchers who need to maintain and query the data.
* **Features**:
  + **CRUD Operations**: Ability to create, read, update, and delete professor, university, and organization data.
  + **Many-to-Many Relationships**: Support for linking professors with multiple universities and organizations via the affiliations table.
  + **Search and Reporting**: Allow users to search for professors by name, university, or organization and generate reports of affiliations.
  + **Data Integrity**: Ensure that updates and deletions to universities, organizations, or professors are handled properly, using foreign key constraints to maintain referential integrity.
  + **User Interface**: Ideally, a simple web interface could be implemented to make it easier for users to interact with the system.

**Functional Description:**

**Method of Use:**

The system will be primarily used by administrators, researchers, and staff members who manage academic data. Professors, universities, and organizations will be added and maintained by the administrators. Researchers or other users may query the system to obtain data on professor affiliations with universities or organizations. The system will allow users to perform the following actions:

* **Add a new professor, university, or organization.**
* **Link professors to universities and organizations using the affiliations table.**
* **Update or delete existing records as needed.**
* **Query the database to generate reports on professors' affiliations, such as all professors working at a specific university or organization.**

**Entity Data Model:**

**Diagram:**

The Entity-Relationship (ER) diagram included earlier depicts the following entities:

* **Professors**: Each professor has an ID, first name, last name, and a function (role).
* **Universities**: Each university has an ID, name, short name, and city.
* **Organizations**: Each organization has an ID, name, and sector.
* **Affiliations**: This is a junction table connecting professors, universities, and organizations. It stores the links between professors and the institutions they are affiliated with, with references to professor, university, and organization IDs.

**Explanation of Entities and Relationships:**

* **Professors**: A professor can be associated with multiple universities and organizations through the **affiliations** table.
* **Universities**: A university can have many professors affiliated with it, and professors can be affiliated with multiple universities.
* **Organizations**: Similar to universities, an organization can have many professors affiliated with it.
* **Affiliations**: This table creates a many-to-many relationship between professors, universities, and organizations. It connects professors to specific universities and organizations, ensuring that a professor can be linked to multiple entities simultaneously. It enforces data integrity through foreign key constraints.

# As we have a large table named as university professors.

So we will break this into small tables as given below:

**professors**: Store professor-specific information (first name, last name, function).

* **universities**: Store information about universities (name, short name, city).
* **organizations**: Store organizations and their sector information.
* **affiliations**: A many-to-many table that links professors to universities and organizations.

**Professors Table**

CREATE TABLE professors (

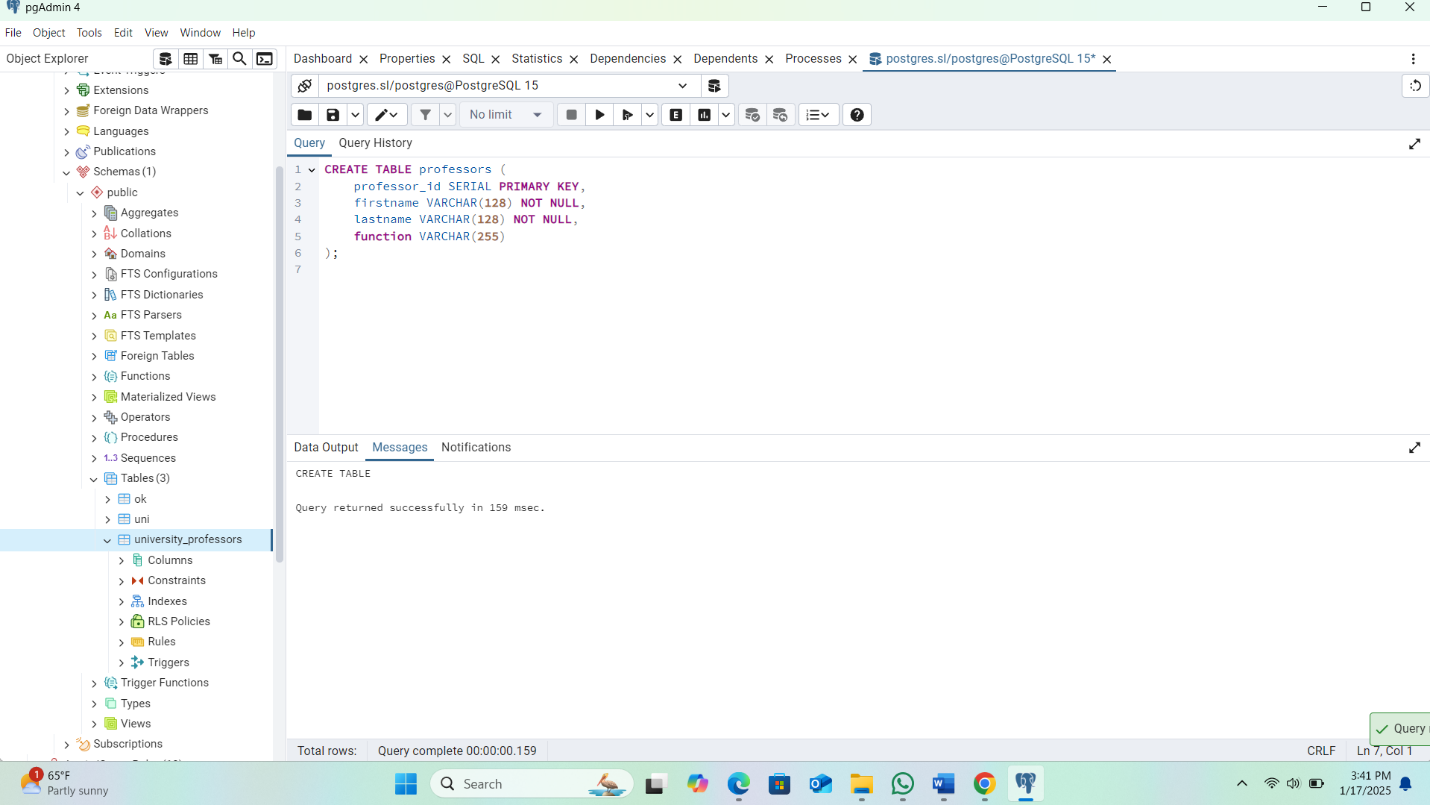
professor\_id SERIAL PRIMARY KEY,

firstname VARCHAR(128) NOT NULL,

lastname VARCHAR(128) NOT NULL,

function VARCHAR(255)

);



**Universities Table**

CREATE TABLE universities (

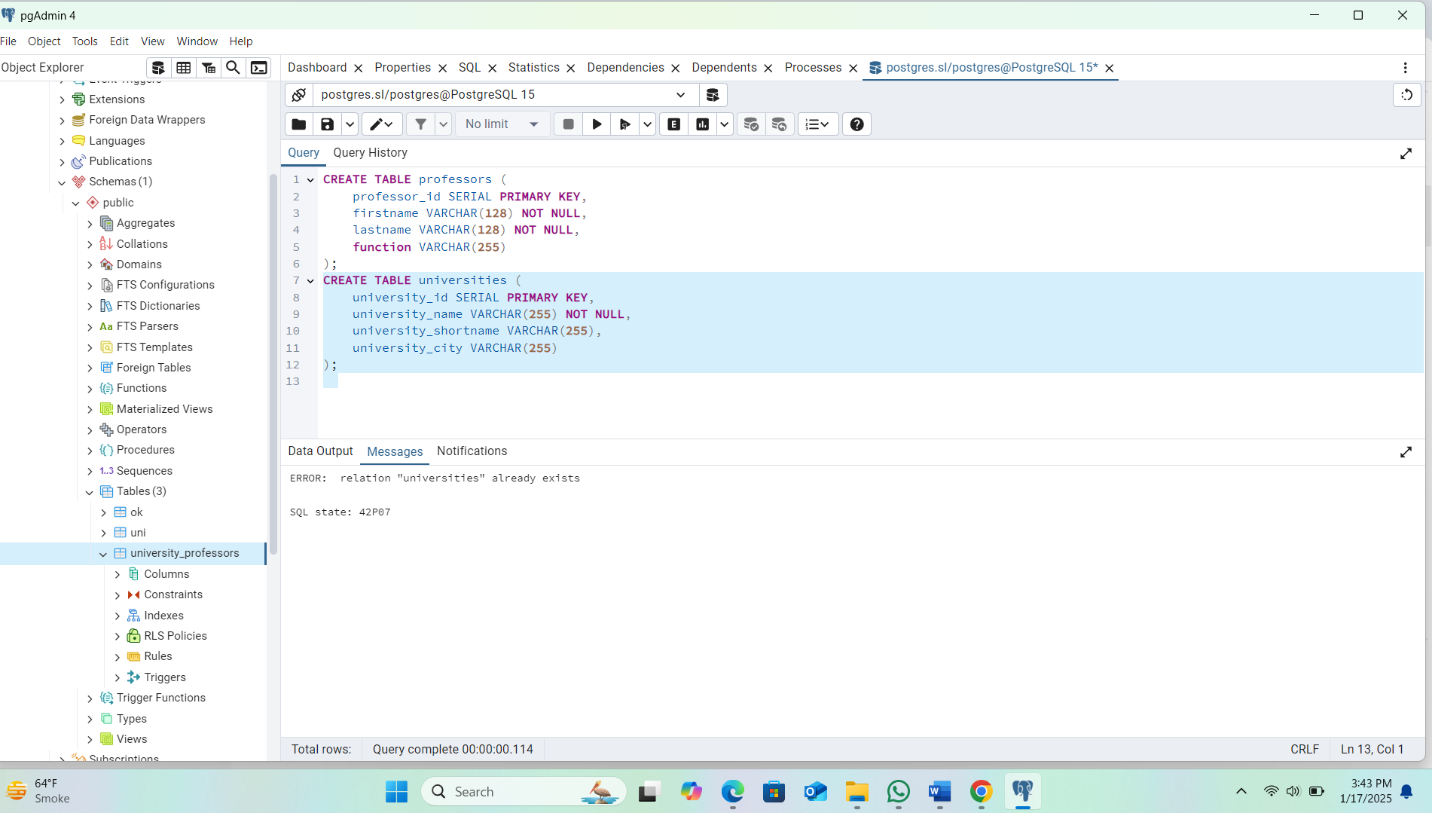
university\_id SERIAL PRIMARY KEY,

university\_name VARCHAR(255) NOT NULL,

university\_shortname VARCHAR(255),

university\_city VARCHAR(255)

);



**Organizations Table**

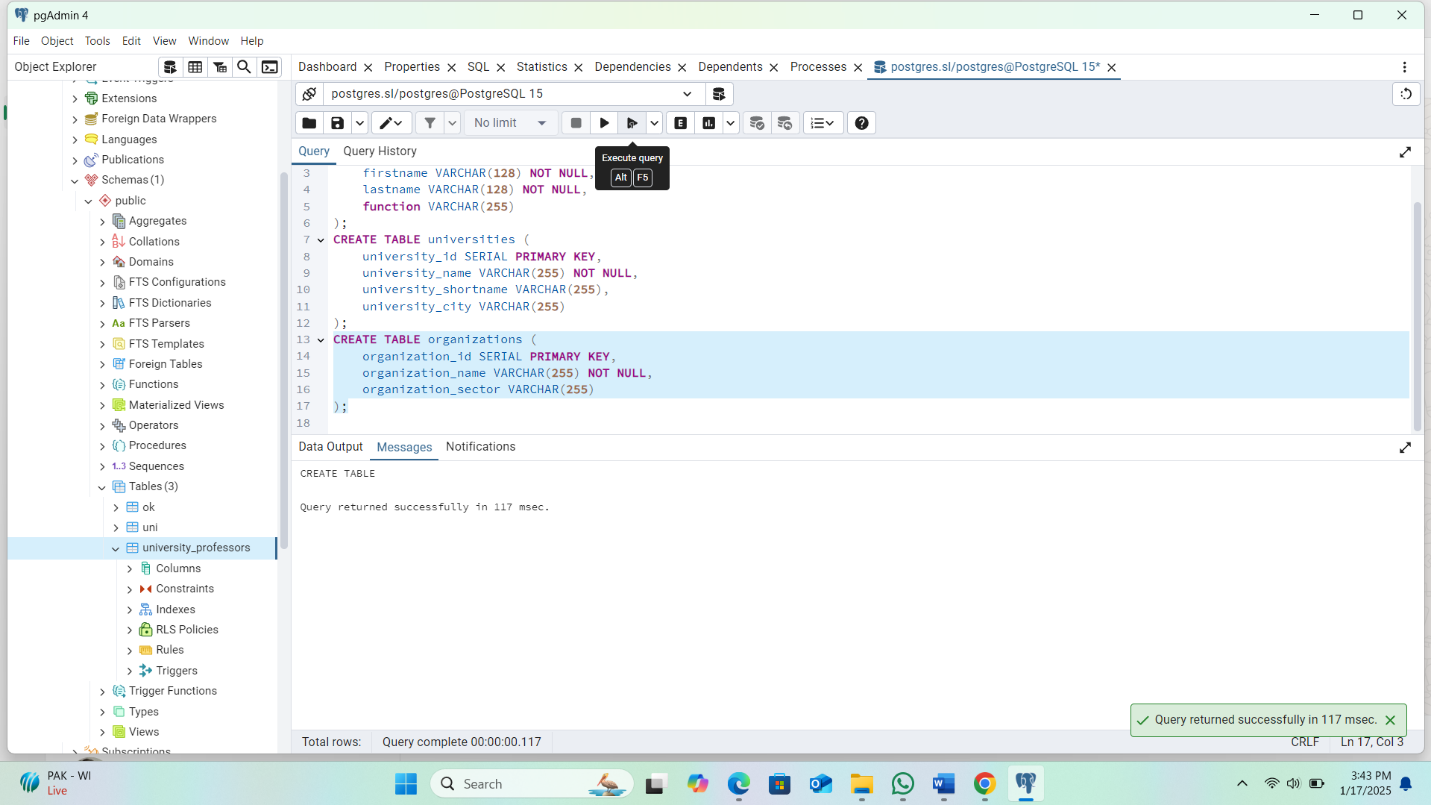
CREATE TABLE organizations (

organization\_id SERIAL PRIMARY KEY,

organization\_name VARCHAR(255) NOT NULL,

organization\_sector VARCHAR(255)

);



**Affiliations Table (Many-to-Many Relationship)**

CREATE TABLE affiliations (

affiliation\_id SERIAL PRIMARY KEY,

professor\_id INT NOT NULL,

university\_id INT,

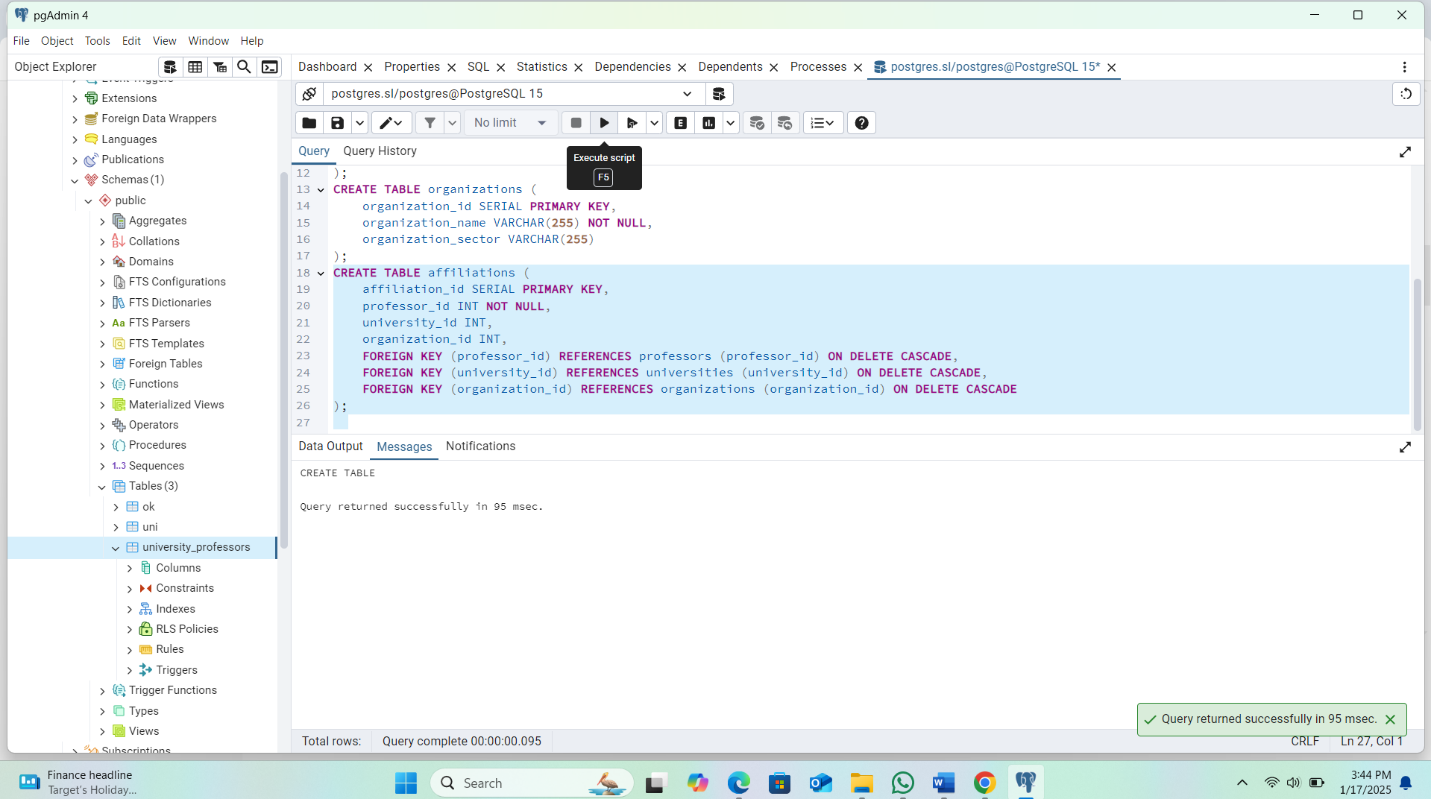
organization\_id INT,

FOREIGN KEY (professor\_id) REFERENCES professors (professor\_id) ON DELETE CASCADE,

FOREIGN KEY (university\_id) REFERENCES universities (university\_id) ON DELETE CASCADE,

FOREIGN KEY (organization\_id) REFERENCES organizations (organization\_id) ON DELETE CASCADE

);



**2. Insert Data**

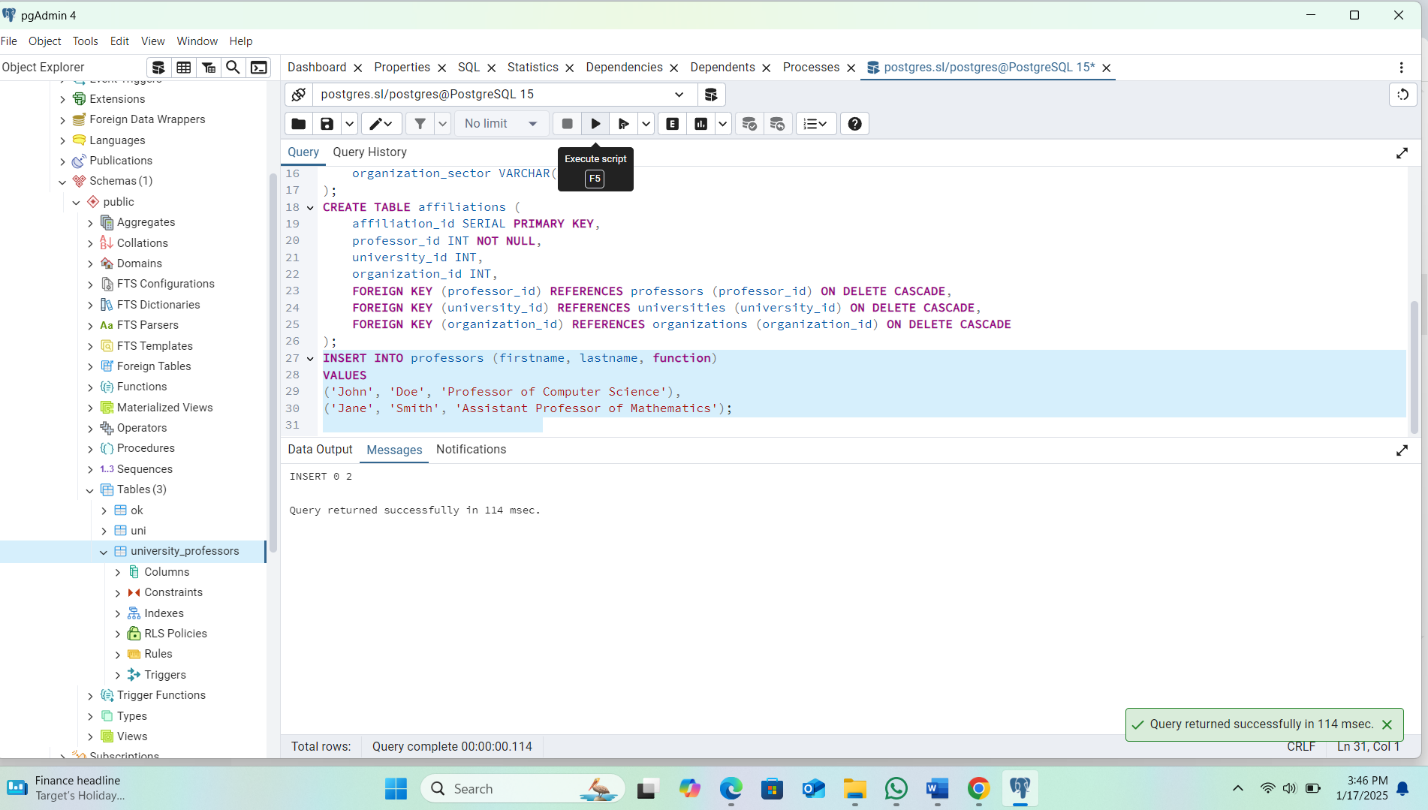
**Insert Professors**

INSERT INTO professors (firstname, lastname, function)

VALUES

('John', 'Doe', 'Professor of Computer Science'),

('Jane', 'Smith', 'Assistant Professor of Mathematics');



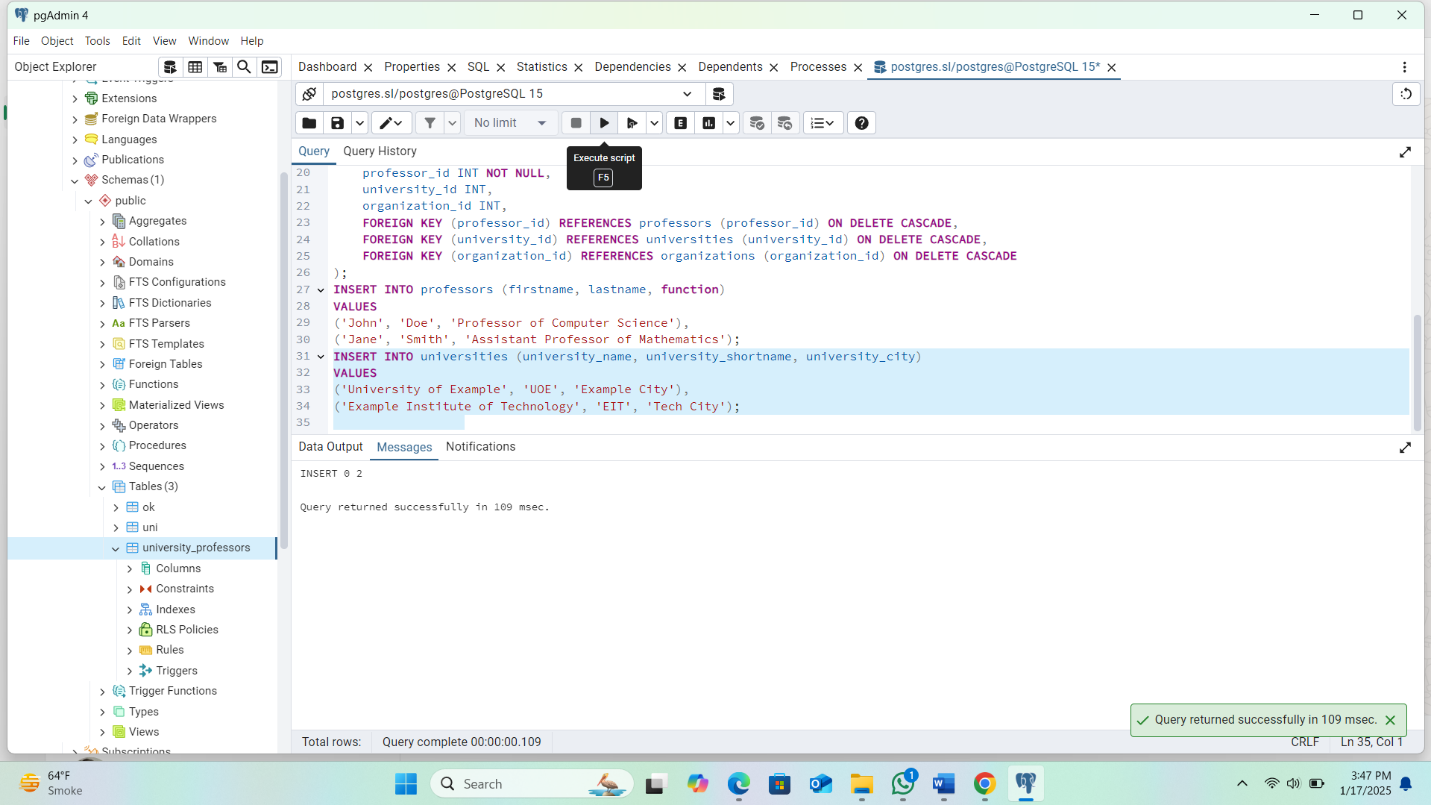
**Insert Universities**

INSERT INTO universities (university\_name, university\_shortname, university\_city)

VALUES

('University of Example', 'UOE', 'Example City'),

('Example Institute of Technology', 'EIT', 'Tech City');



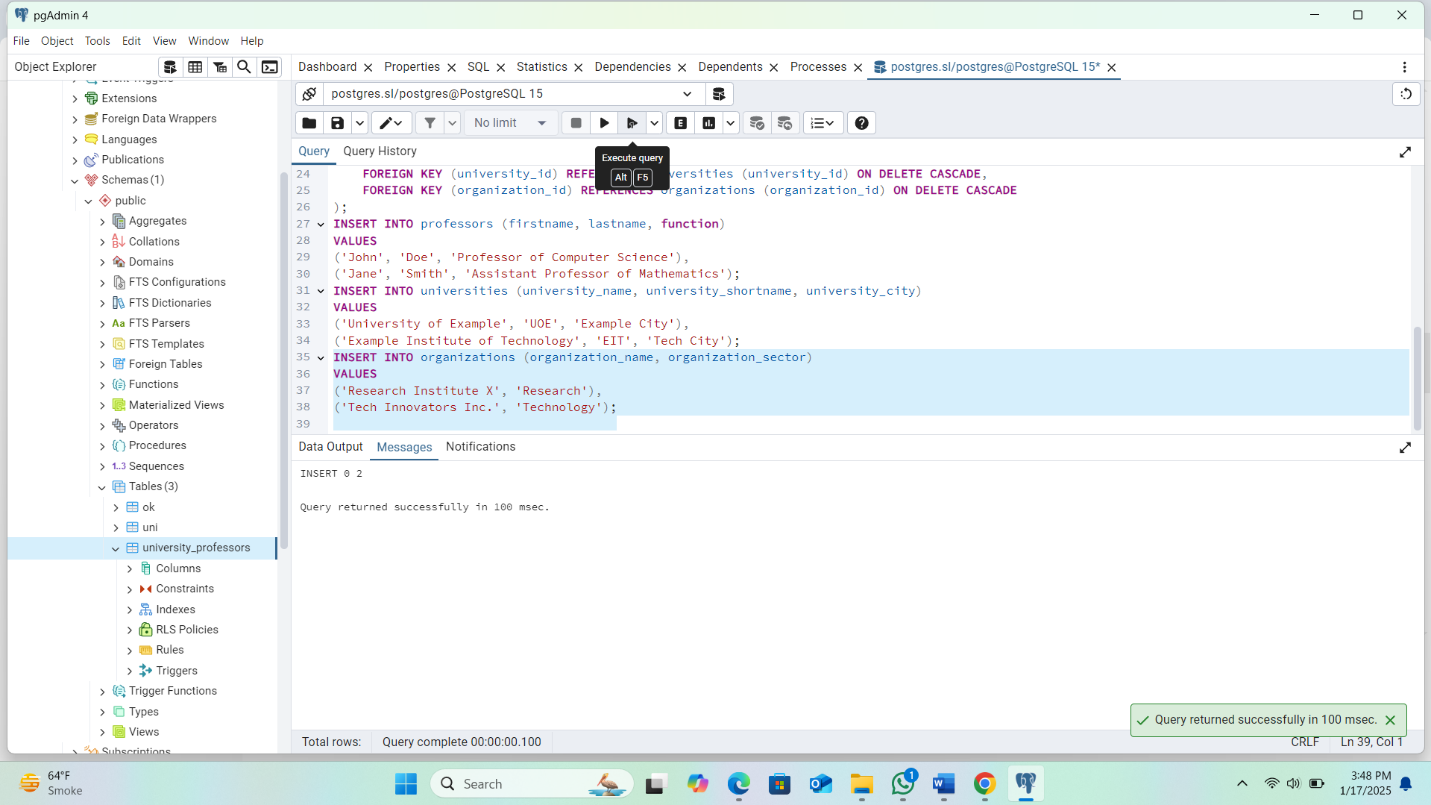
**Insert Organizations**

INSERT INTO organizations (organization\_name, organization\_sector)

VALUES

('Research Institute X', 'Research'),

('Tech Innovators Inc.', 'Technology');



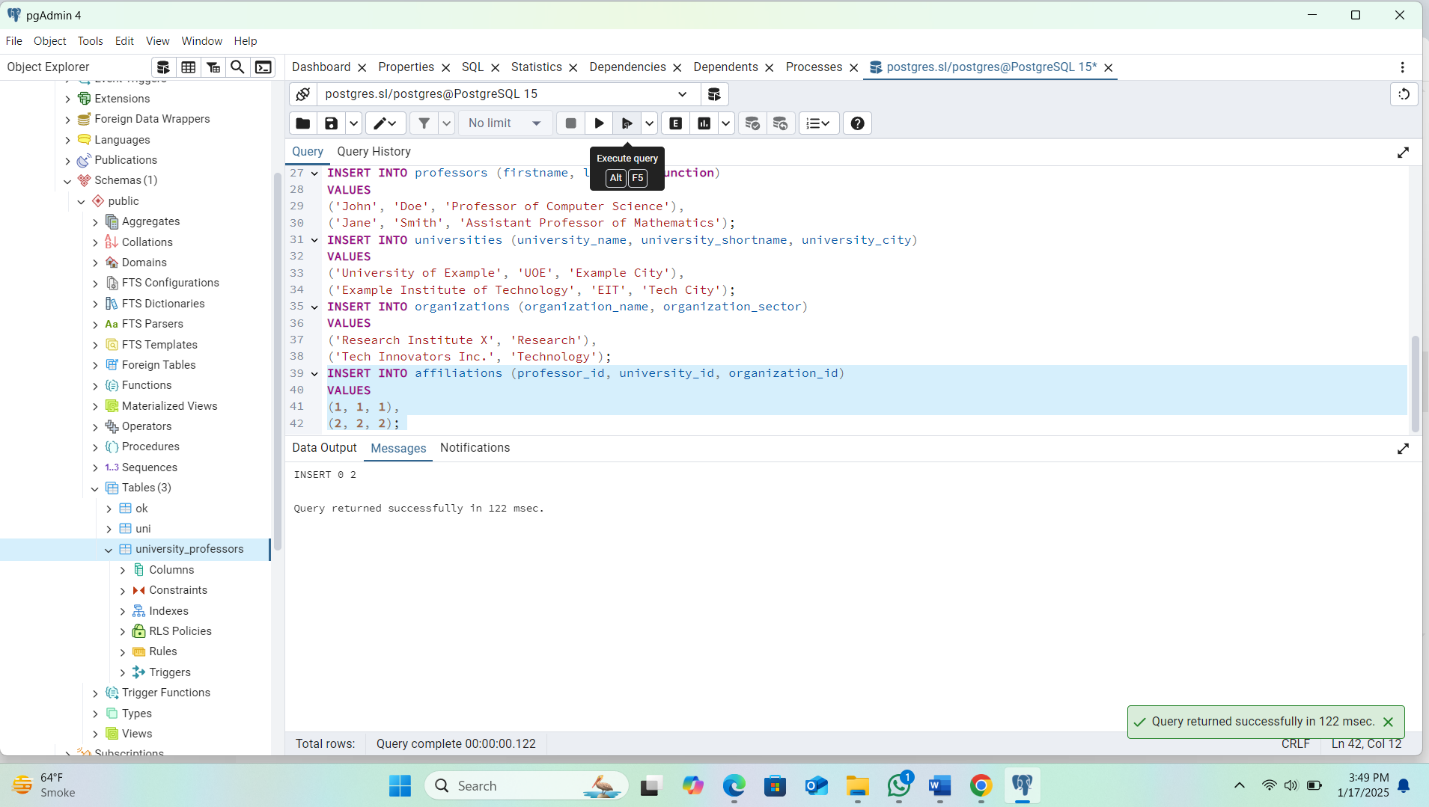
**Insert Affiliations**

INSERT INTO affiliations (professor\_id, university\_id, organization\_id)

VALUES

(1, 1, 1),

(2, 2, 2);



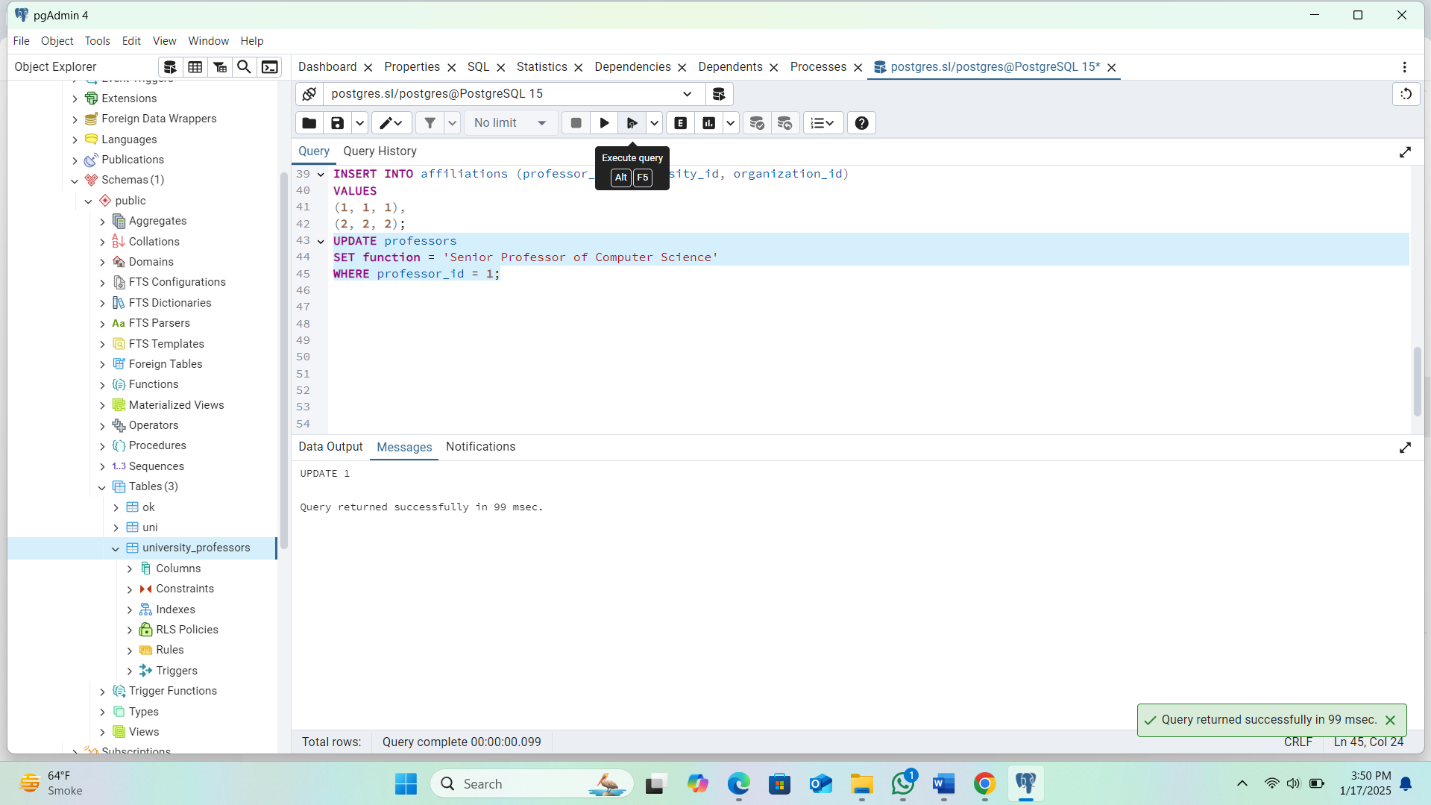
**3. Update Data**

**Update Professor Function**

UPDATE professors

SET function = 'Senior Professor of Computer Science'

WHERE professor\_id = 1;

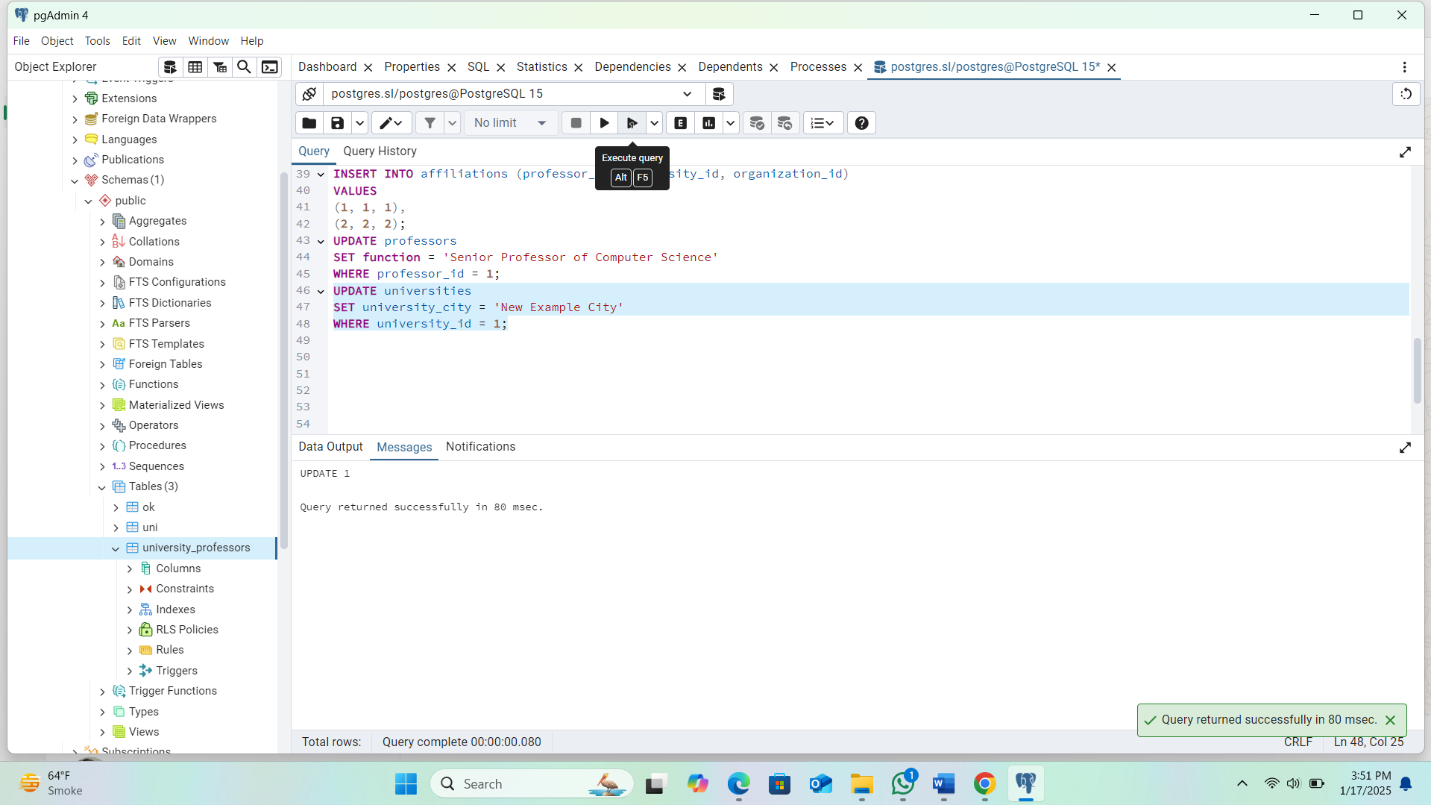


**Update University Information**

UPDATE universities

SET university\_city = 'New Example City'

WHERE university\_id = 1;

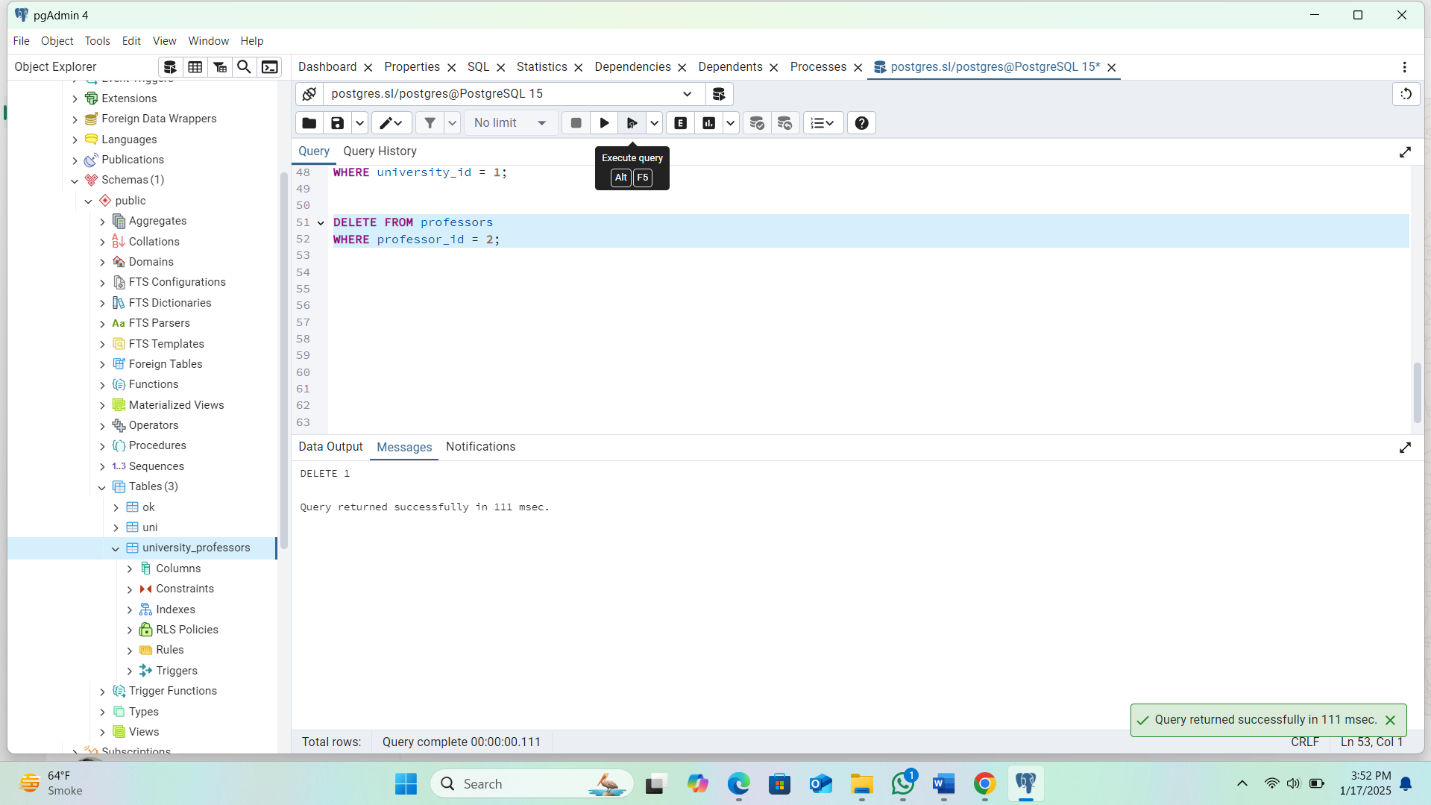


**4. Delete Data**

**Delete Professor**

DELETE FROM professors

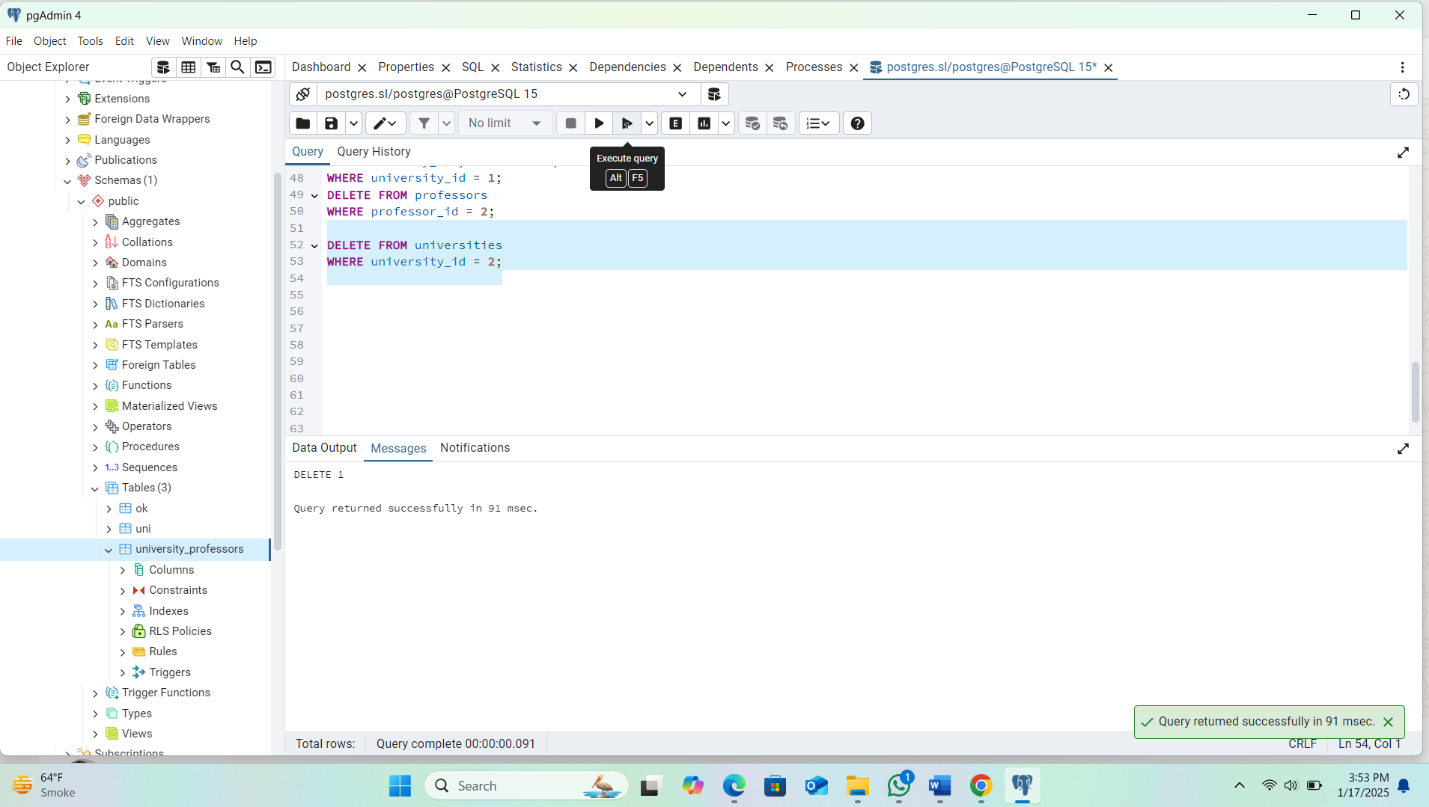
WHERE professor\_id = 2;



**Delete University**

DELETE FROM universities

WHERE university\_id = 2;



**5. Joins**

**Get All Professors with Their Affiliations**

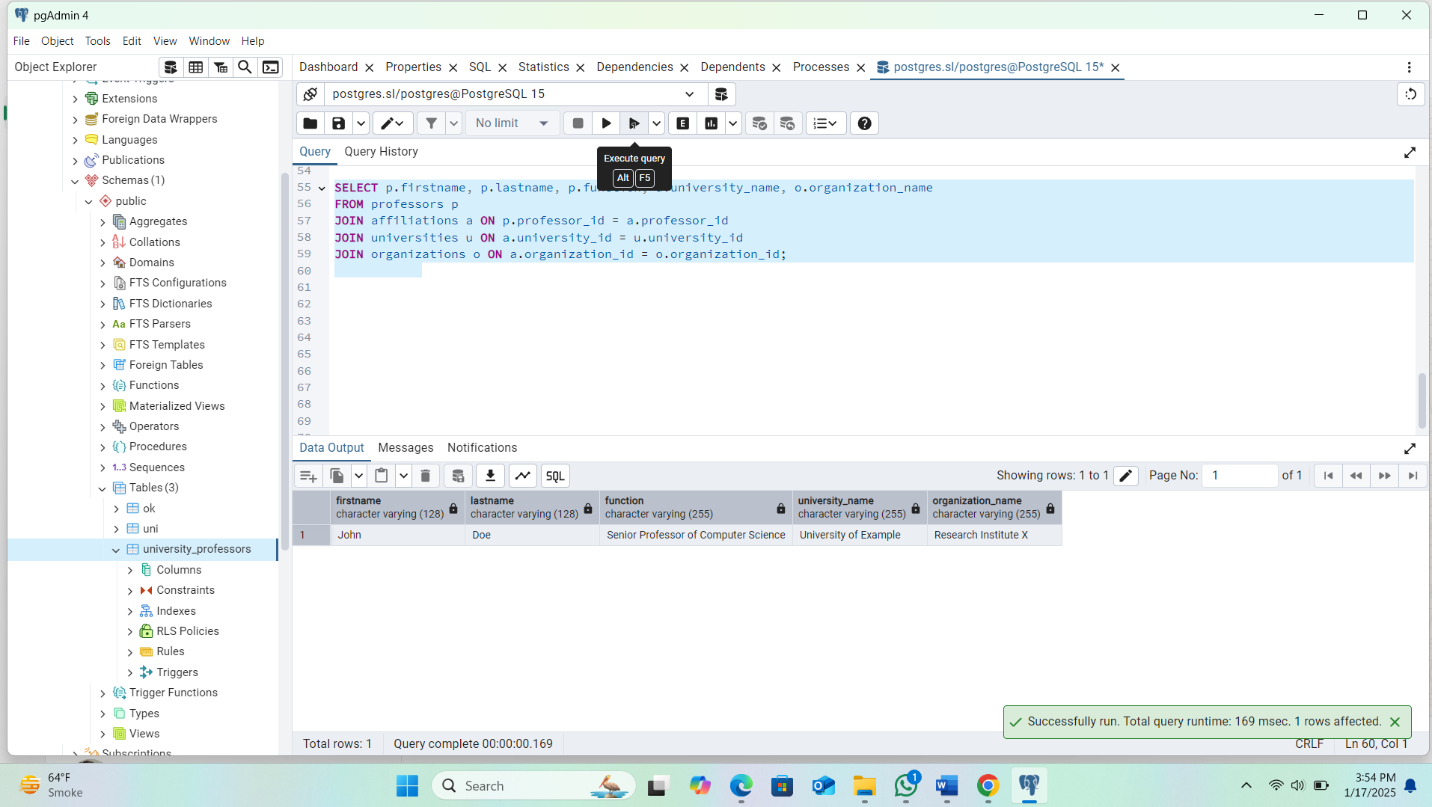
SELECT p.firstname, p.lastname, p.function, u.university\_name, o.organization\_name

FROM professors p

JOIN affiliations a ON p.professor\_id = a.professor\_id

JOIN universities u ON a.university\_id = u.university\_id

JOIN organizations o ON a.organization\_id = o.organization\_id;



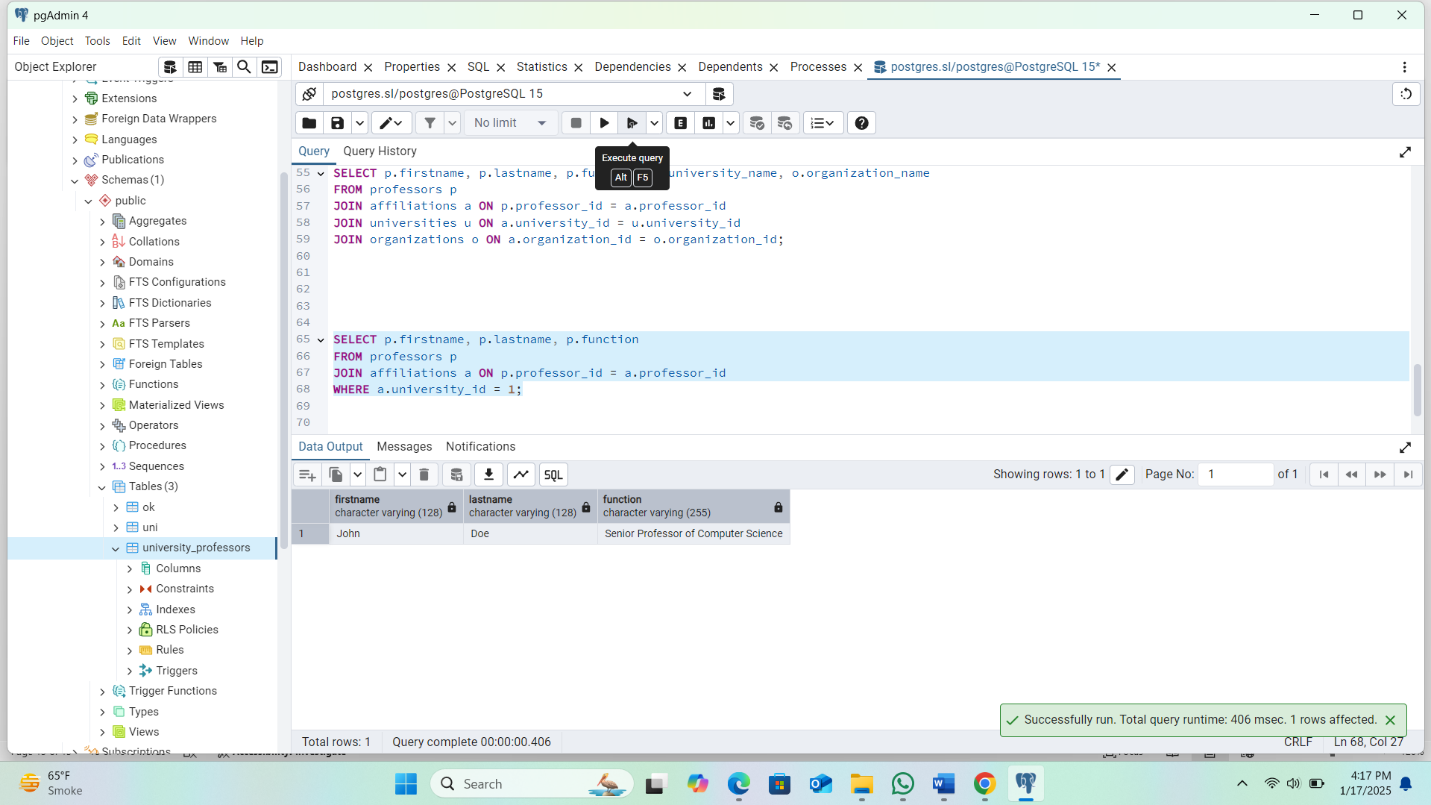
**Get All Professors in a Specific University**

SELECT p.firstname, p.lastname, p.function

FROM professors p

JOIN affiliations a ON p.professor\_id = a.professor\_id

WHERE a.university\_id = 1;



**6. Referential Integrity**

The foreign key constraints used in the affiliations table ensure referential integrity. Specifically:

* If a professor is deleted, their corresponding records in the affiliations table are automatically deleted due to ON DELETE CASCADE.
* If a university or organization is deleted, all associated affiliations will also be deleted automatically.

**7.ER DIAGRAM**

**1. Entities:**

**The main entities involved in the ER diagram are:**

* **Professors: This represents the individuals who are faculty members at universities and organizations.**
* **Universities: This represents academic institutions where professors are affiliated.**
* **Organizations: This represents organizations that professors can be associated with.**
* **Affiliations: This represents the relationship between professors, universities, and organizations. It links the professors to the universities and organizations.**

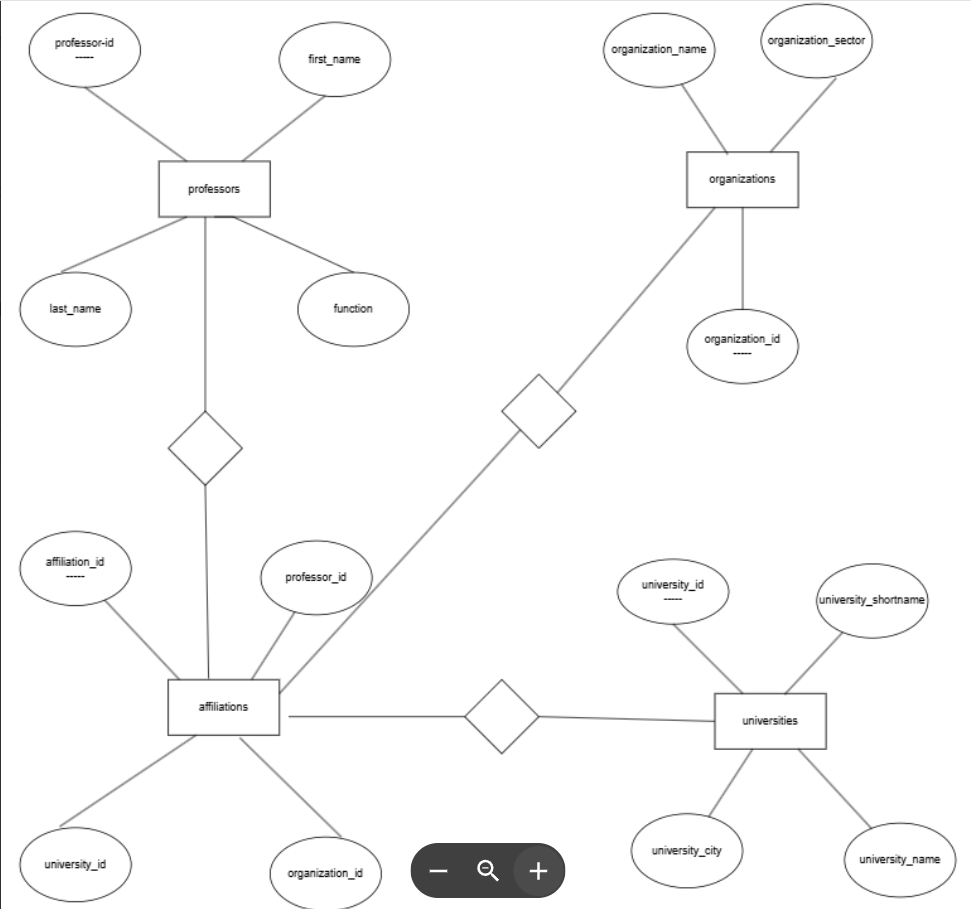
**2. Attributes:**

**Each entity has its own set of attributes (fields) that describe its properties:**

* **Professors: professor\_id (primary key), firstname, lastname, function**
* **Universities: university\_id (primary key), university\_name, university\_shortname, university\_city**
* **Organizations: organization\_id (primary key), organization\_name, organization\_sector**
* **Affiliations: affiliation\_id (primary key), professor\_id (foreign key), university\_id (foreign key), organization\_id (foreign key)**

**3. Relationships:**

* **Professors to Affiliations: A professor can be affiliated with multiple universities and organizations, so there is a one-to-many relationship from Professors to Affiliations (one professor can have many affiliations).**
* **Universities to Affiliations: A university can have many professors associated with it through affiliations. This is a one-to-many relationship from Universities to Affiliations (one university can have many professors).**
* **Organizations to Affiliations: An organization can have many professors affiliated with it through affiliations. This is a one-to-many relationship from Organizations to Affiliations (one organization can have many professors).**

****

**4. Foreign Keys:**

* **professor\_id in the Affiliations table references the professor\_id in the Professors table.**
* **university\_id in the Affiliations table references the university\_id in the Universities table.**
* **organization\_id in the Affiliations table references the organization\_id in the Organizations table.**

**ER Diagram Creation:**

**To represent the above information in an ER diagram:**

1. **Entities are represented as rectangles.**
2. **Attributes are shown as ovals connected to their respective entities.**
3. **Primary Keys (unique identifiers) are often underlined in the ER diagram.**
4. **Relationships are represented as diamonds between entities, labeled with verbs describing the relationship (e.g., "has", "affiliated with").**
5. **Foreign Keys are represented as lines connecting the primary keys in the related entities.**

**Relationship Description:**

* **Professors can be affiliated with many Universities and Organizations through the Affiliations table.**
* **Affiliations links professors to universities and organizations.**
* **Universities and Organizations are linked to many professors via the Affiliations table.**