Experiment No 8

Aim-To set up a PostgreSQL database, create relational tables, and perform basic CRUD operations (Create, Read, Update, Delete) using SQL queries.

Theory:Introduction to PostgreSQL: PostgreSQL is a powerful, open-source object-relational database system. It has a strong reputation for reliability, feature robustness, and performance. Originally developed at the University of California, Berkeley, PostgreSQL is now maintained by a global community. It supports a wide range of advanced features, including user-defined types, table inheritance, views, foreign keys, transactions, and multiversion concurrency control (MVCC).

Relational Databases: A relational database organizes data into one or more tables (or "relations") of columns and rows. Each table represents an entity, and each row in a table represents a record with a unique ID (usually the primary key). Relationships between tables are maintained through foreign keys, which ensure data integrity and consistency.

SQL - Structured Query Language: SQL is the standard language for interacting with relational databases.

It allows users to define, manipulate, control, and query data. PostgreSQL uses an extended version of SQL, which includes many modern programming constructs. SQL operations can be divided into several categories:

* Data Definition Language (DDL): Includes commands like CREATE, ALTER, and DROP to define and modify schema objects.
* Data Manipulation Language (DML): Includes INSERT, UPDATE, DELETE, and SELECT for modifying and querying data.
* Data Control Language (DCL): Includes GRANT and REVOKE to manage permissions.
* Transaction Control Language (TCL): Includes BEGIN, COMMIT, and ROLLBACK to manage transactions.

CRUD Operations: CRUD is an acronym for the four basic types of SQL operations that interact with a database.

1. Create (INSERT): This operation is used to add new records to a table.
2. INSERT INTO students (name, age, email) VALUES ('Alice', 22, 'alice@example.com');
3. Read (SELECT): This operation is used to retrieve data from one or more tables.
4. SELECT \* FROM students;
5. Update (UPDATE): This operation modifies existing data in a table.
6. UPDATE students SET age = 23 WHERE name = 'Alice';
7. Delete (DELETE): This operation removes data from a table.
8. DELETE FROM students WHERE name = 'Alice';

Creating a Table in PostgreSQL: To perform CRUD operations, we must first create a table in the database. Below is an example of creating a simple table to store student records. CREATE TABLE students ( id SERIAL PRIMARY KEY, name VARCHAR(100),

age INT,

email VARCHAR(100)

);

Primary Key: A unique identifier for each record in a table. Serial: Automatically generates a unique integer for the id column. Varchar: Variable-length character string. Int: Integer data type.

Relationships and Foreign Keys: In real-world applications, databases often have multiple tables that relate to each other. For example, a students table may be linked to a courses table through an enrollment table. This is implemented using foreign keys. Foreign keys establish referential integrity between the data in two tables.

Normalization: Normalization is the process of organizing data to reduce redundancy and improve data integrity. It involves dividing large tables into smaller, related tables and using keys to establish relationships. The most common normal forms are:

* 1NF (First Normal Form): Eliminate repeating groups.
* 2NF (Second Normal Form): Remove partial dependencies.
* 3NF (Third Normal Form): Remove transitive dependencies.

Indexes: Indexes improve the speed of data retrieval operations. An index is a database object that helps PostgreSQL locate data without scanning every row in a table. However, excessive indexing can slow down write operations and increase storage usage.

Constraints: PostgreSQL supports several constraints to enforce rules on data:

* NOT NULL: Ensures that a column cannot have NULL values.
* UNIQUE: Ensures all values in a column are different.
* CHECK: Ensures that values meet a specific condition.
* DEFAULT: Provides a default value for a column when no value is specified.

Transactions: A transaction is a sequence of operations performed as a single logical unit of work. PostgreSQL ensures ACID (Atomicity, Consistency, Isolation, Durability) properties, which make transactions reliable.

BEGIN;

UPDATE students SET age = 24 WHERE name = 'Alice';

COMMIT;

If any step fails, we can use ROLLBACK to undo the changes.

Advantages of Using PostgreSQL:

* Open source and free to use.
* Cross-platform and compatible with many development environments.
* ACID compliant, ensuring reliable transactions.
* Strong community support and extensive documentation.
* Support for JSON and XML data formats.
* Extensible through custom plugins and functions.
* Supports full-text search, window functions, and recursive queries.

Tools for PostgreSQL:

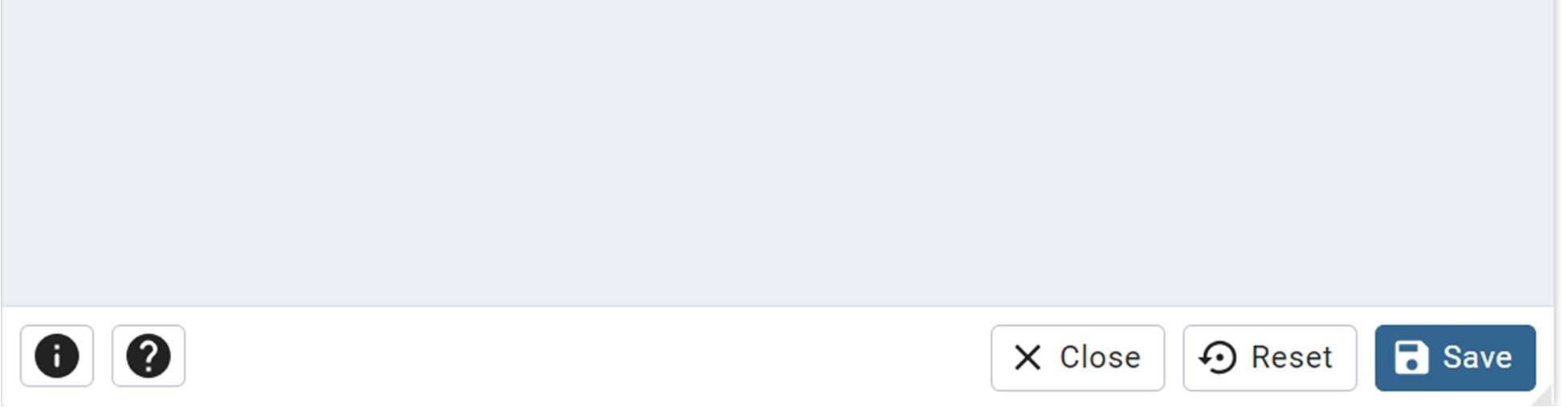
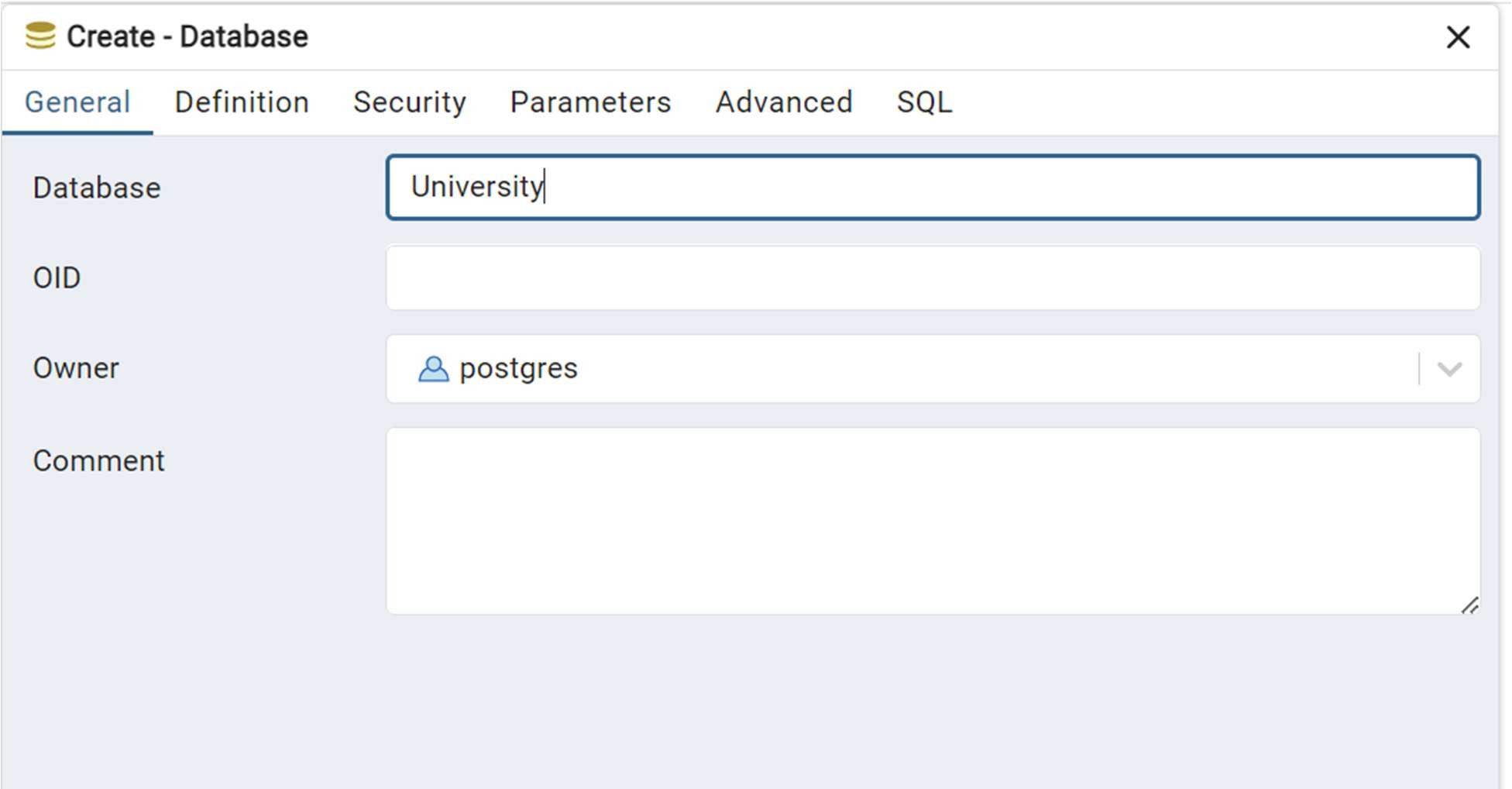
* psql: PostgreSQL command-line interface.
* pgAdmin: Graphical interface for managing PostgreSQL databases.
* DBeaver / DataGrip: Third-party database tools that support PostgreSQL.

Security Features: PostgreSQL includes powerful access control systems using roles, permissions, SSL encryption, and audit logs to secure data. It also allows row-level security for granular access control.

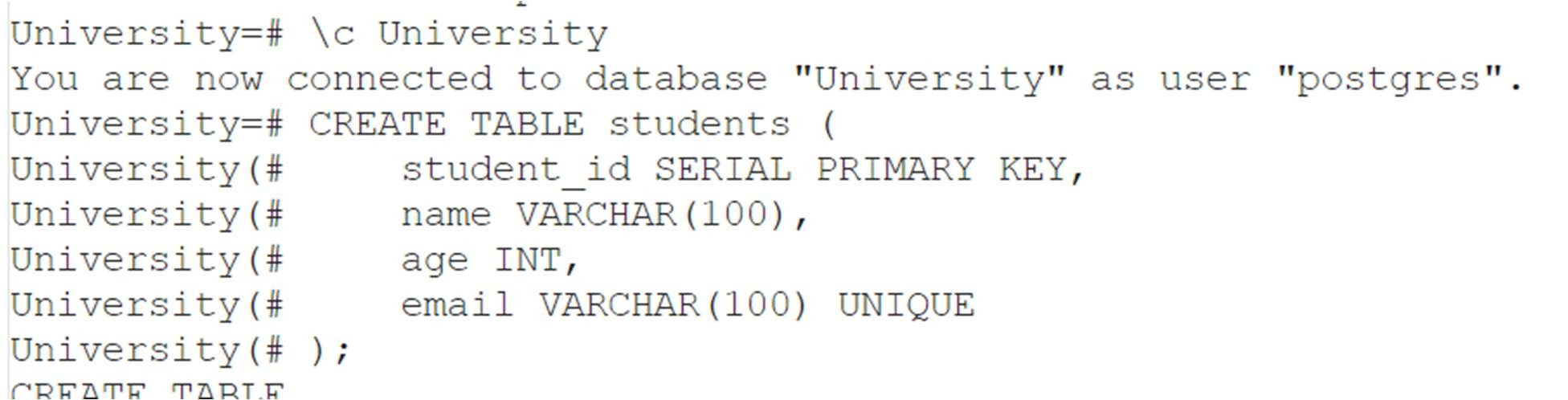
Real-World Use Cases: PostgreSQL is widely used in various sectors:

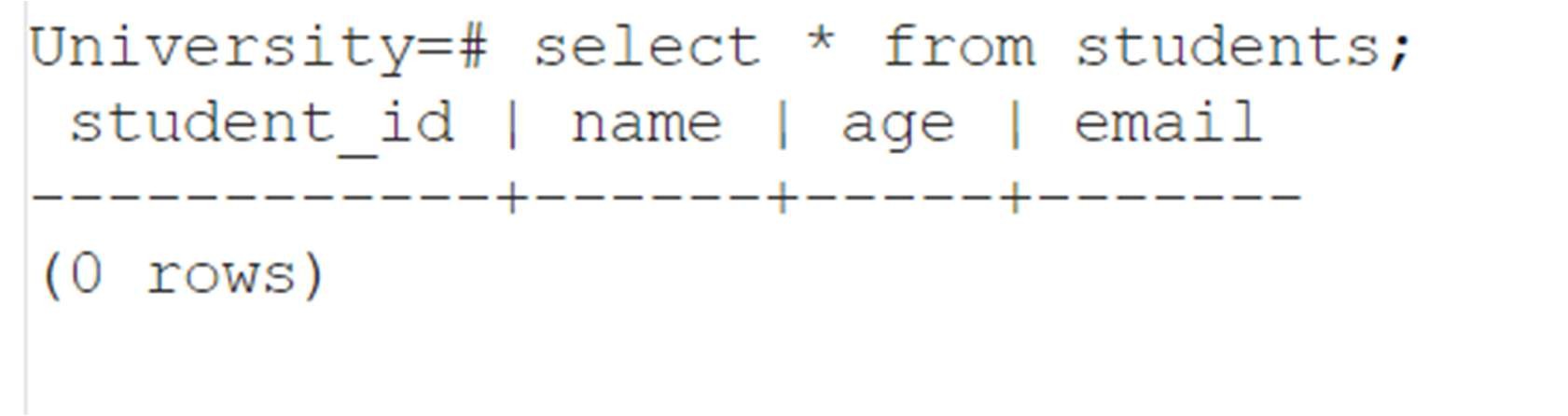
* Web applications (e.g., Django, Rails, Spring Boot)
* Data warehousing and analytics
* Financial systems
* Geographic Information Systems (GIS) with PostGIS extension
* IoT data management
* Enterprise Resource Planning (ERP) systems

# CREATING DATABASE THROUGH POSTGRESQL

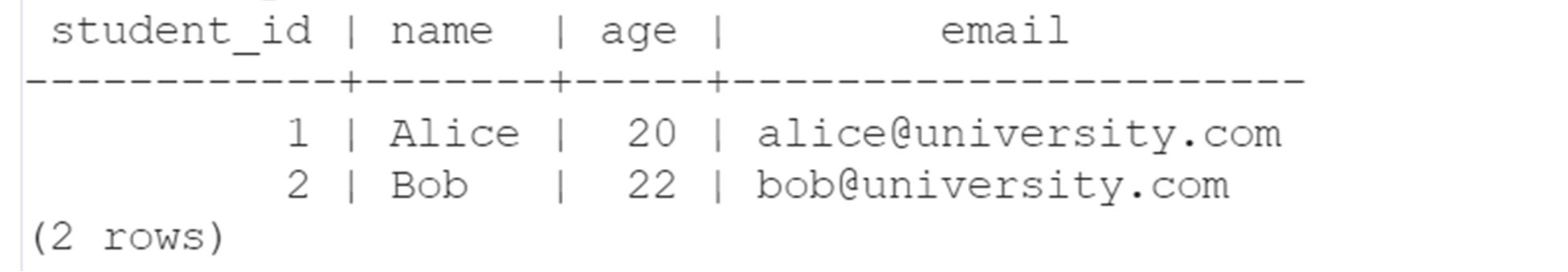
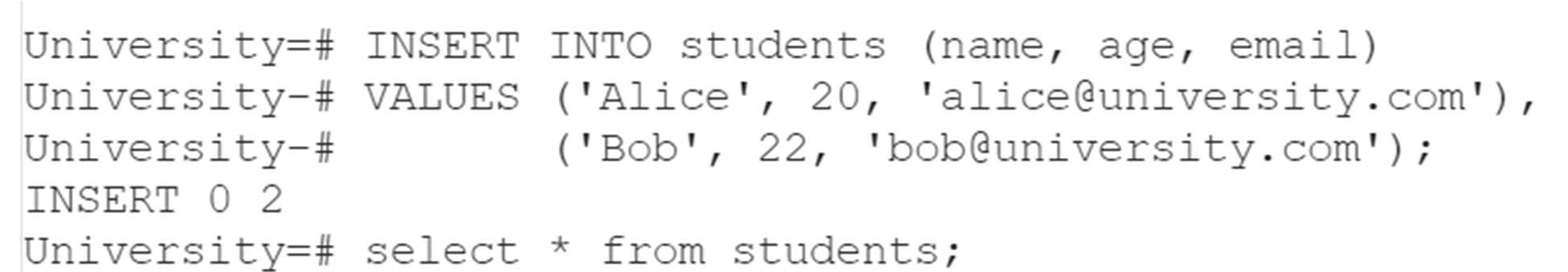


## CREATING TABLE

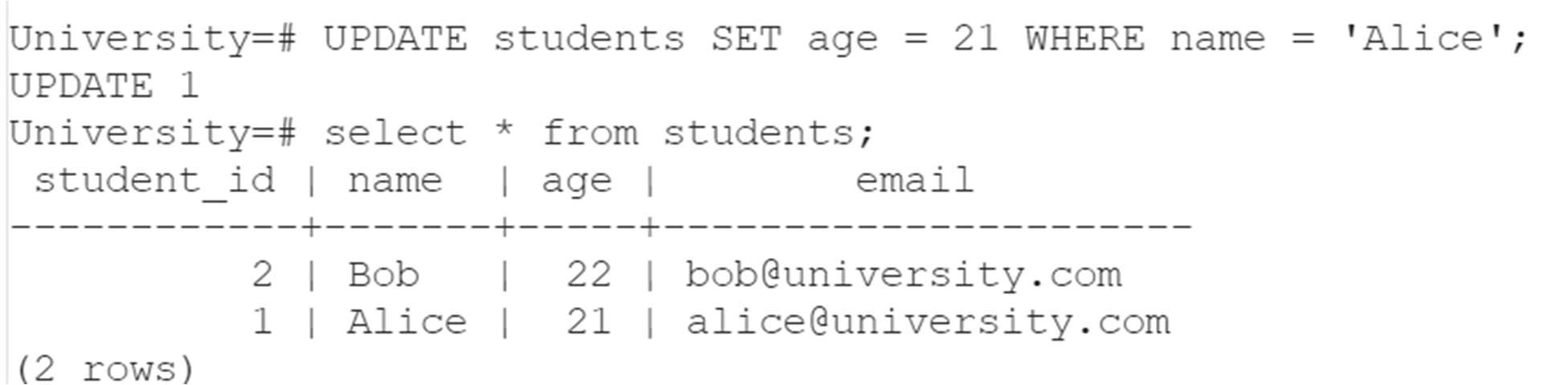




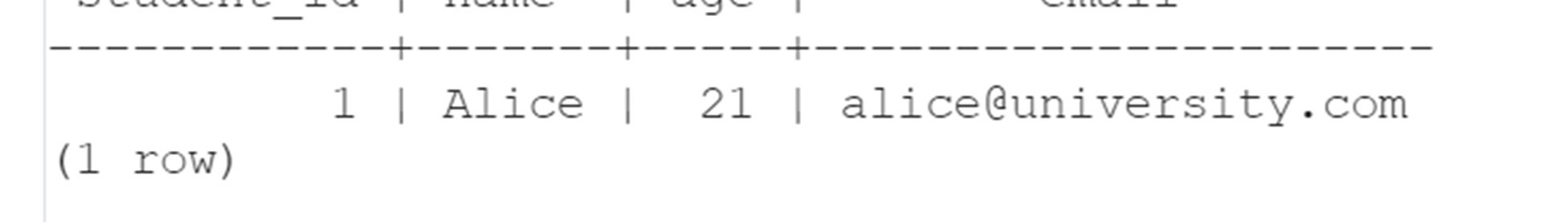
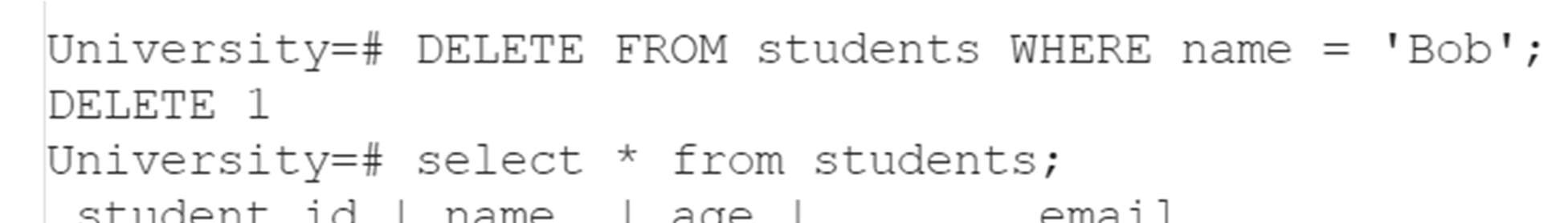
## INSERTING VALUES IN TABLE



## UPDATING VALUES IN TABLE



## DELETING VALUES FROM TABLE



Conclusion:

In this experiment, we successfully learned the core concepts of setting up and interacting with a PostgreSQL database. We created relational tables to organize structured data efficiently and applied the four primary CRUD operations using SQL queries. These operations form the backbone of any application that requires data storage and management. Understanding CRUD in PostgreSQL lays the foundation for more advanced database concepts such as indexing, joins, transactions, triggers, and stored procedures.

By mastering PostgreSQL and SQL operations, students and developers are equipped with essential tools for building scalable and data-driven applications in real-world environments. This hands-on practice also strengthens the ability to design efficient schemas, enforce data integrity, and perform robust data manipulation tasks in production-level software systems.