SQL TABLE

What is a Table?

- A **table** is a **collection of data organized in rows and columns**.
- In **DBMS terms**:
    - **Table** = **Relation**
    - **Row** = **Tuple**
    - **Column** = **Attribute**
- **Purpose**: Tables provide a simple way to store and represent relational data.

Example Table:

EMP_NAME	ADDRESS	SALARY
Ankit	Lucknow	15000
Raman	Allahabad	18000
Mike	New York	20000

SQL Table Variable (SQL Server Feature)

- Introduced in **SQL Server 2000**.
- Works like a **temporary table** but is defined as a **variable**.
- **Advantage**: No need to explicitly drop it.
- Syntax is similar to CREATE TABLE.

SQL CREATE TABLE

Purpose:

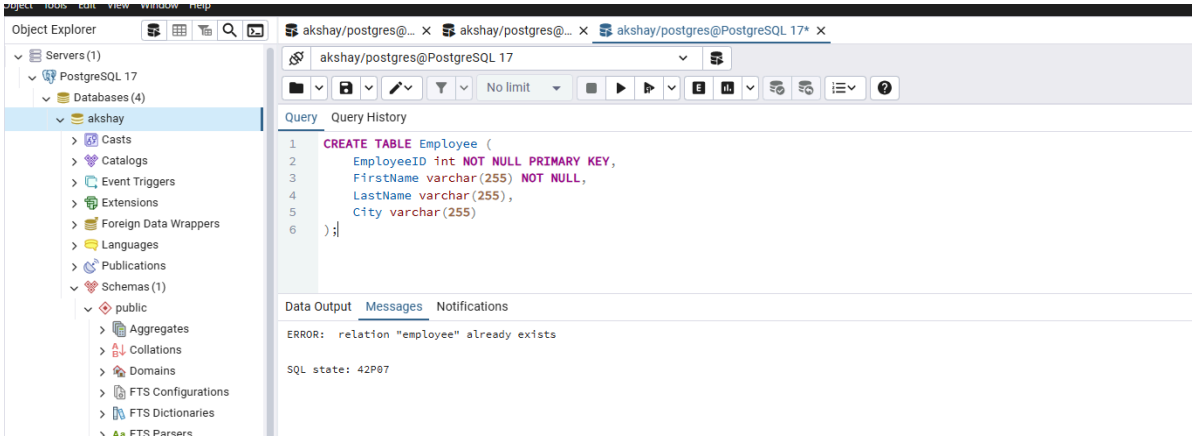
To create a new table in the database.

Syntax:

```
CREATE TABLE table_name (
    column1 data_type [constraint],
    column2 data_type [constraint],
    ...
    columnN data_type [constraint]
);
```

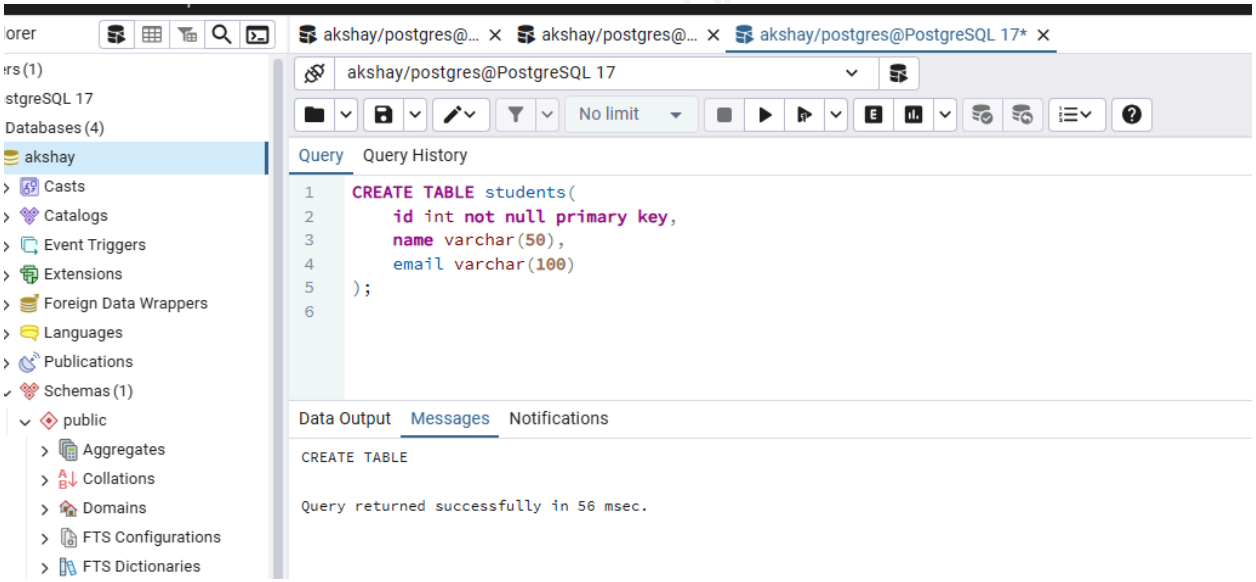

Example 1:

```
CREATE TABLE Employee (
    EmployeeID int NOT NULL PRIMARY KEY,
    FirstName varchar(255) NOT NULL,
    LastName varchar(255),
    City varchar(255)
);
```



Example 2:

```
CREATE TABLE students(
    id int not null primary key,
    name varchar(50),
    email varchar(100)
);
```



SQL INSERT INTO Statement

Purpose:

The INSERT INTO statement is used to **add new records (rows)** into an existing table.

Syntax:

Method 1: Specify Columns

```
INSERT INTO table_name (column1, column2, ..., columnN)
VALUES (value1, value2, ..., valueN);
```

Why specify columns?

- Good practice (recommended).
- Allows inserting values into specific columns only.
- Prevents errors if the table structure changes later.

Method 2: Without Column Names

```
INSERT INTO table_name
VALUES (value1, value2, ..., valueN);
```

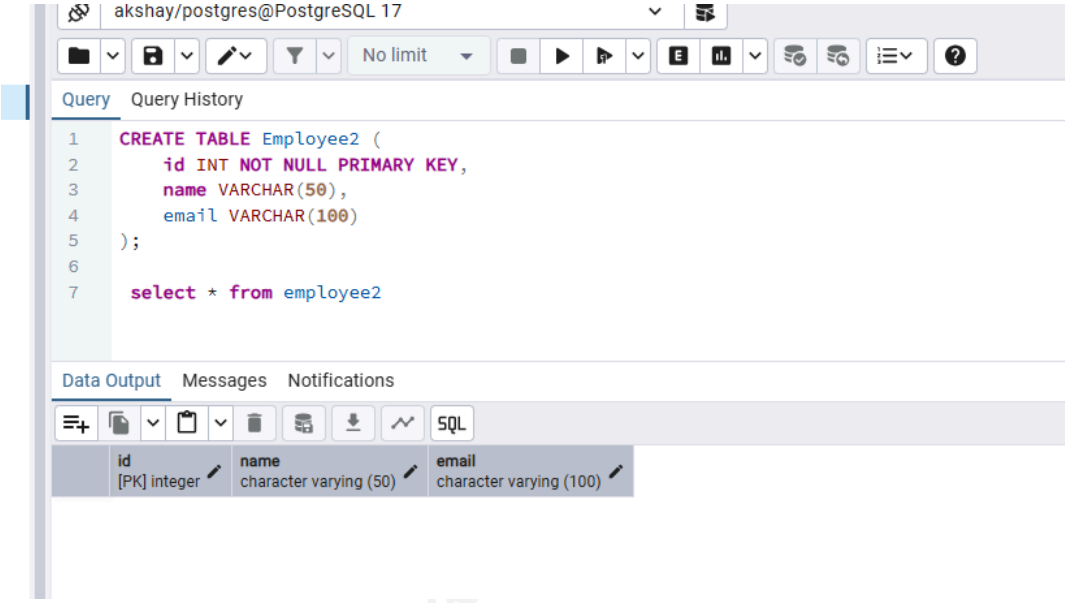
Note:

- You **must provide values for all columns in the correct order**.
- Riskier if table structure changes.

Example Table: Employee

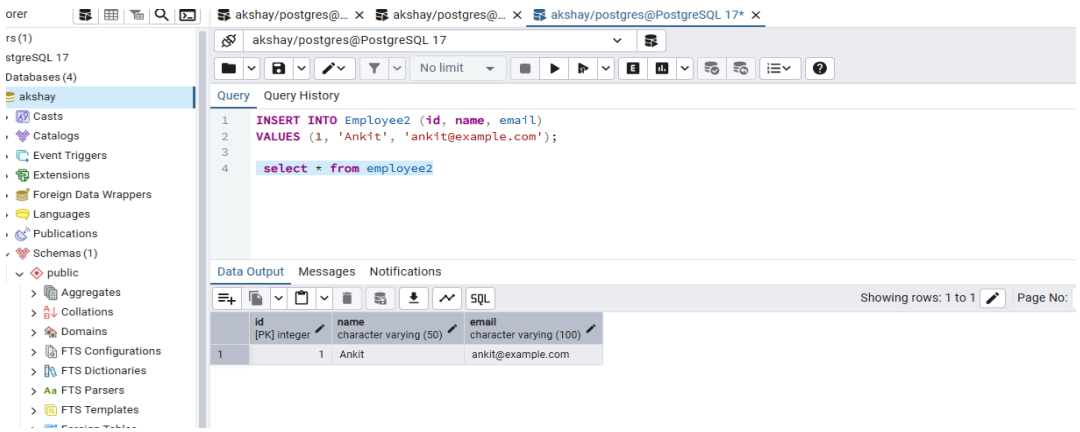
```
CREATE TABLE Employee2 (
    id INT NOT NULL PRIMARY KEY,
    name VARCHAR(50),
    email VARCHAR(100)
);
```

id	name	email



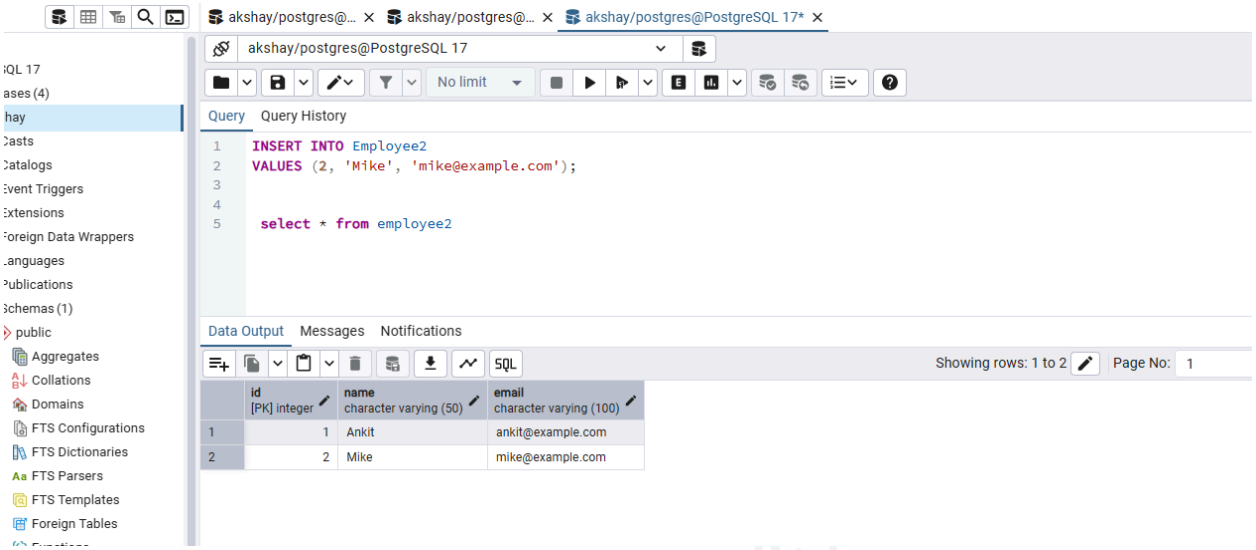
Example 1: Inserting with Column Names

```
INSERT INTO Employee2 (id, name, email)
VALUES (1, 'Ankit', 'ankit@example.com');
```



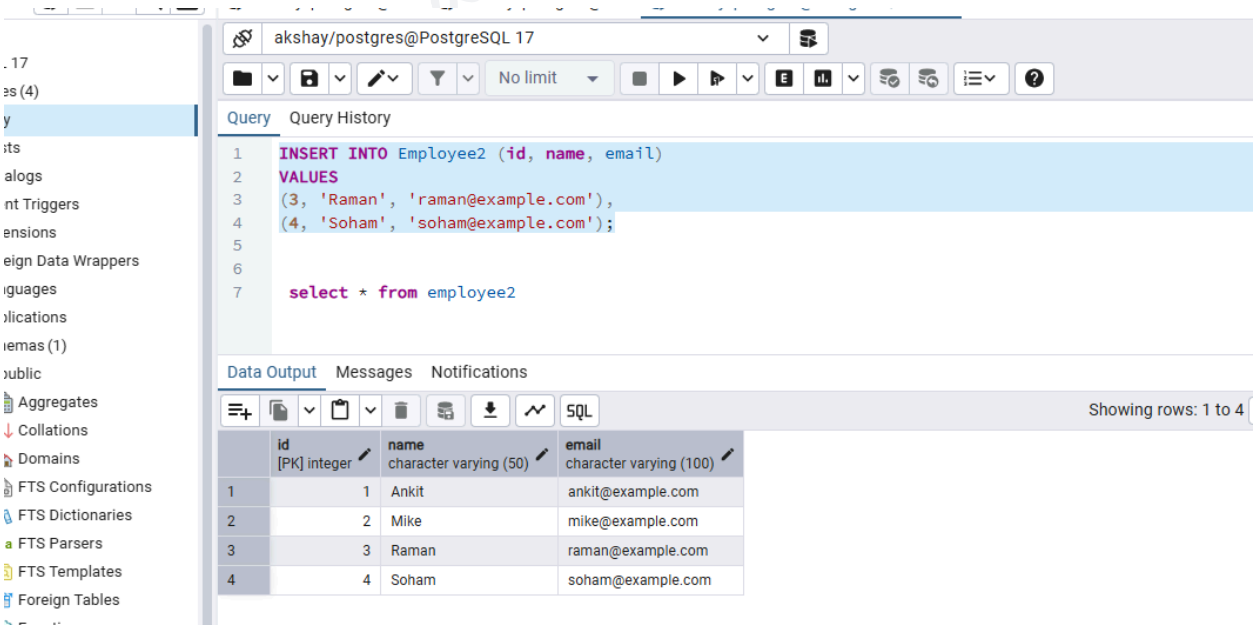
Example 2: Inserting Without Column Names

```
INSERT INTO Employee2
VALUES (2, 'Mike', 'mike@example.com');
```



Insert Multiple Rows (Bulk Insert)

```
INSERT INTO Employee2 (id, name, email)
VALUES
(3, 'Raman', 'raman@example.com'),
(4, 'Soham', 'soham@example.com');
```



INSERT INTO SELECT

Use INSERT INTO with SELECT to copy data from one table to another.

How INSERT INTO SELECT Works

Step 1: Prepare the Tables

Source Table

This is the **existing table** you want to copy data **from**.

Example:

```
CREATE TABLE Employee (
    id INT PRIMARY KEY,
    name VARCHAR(50),
    email VARCHAR(100)
);
```

Insert some data:

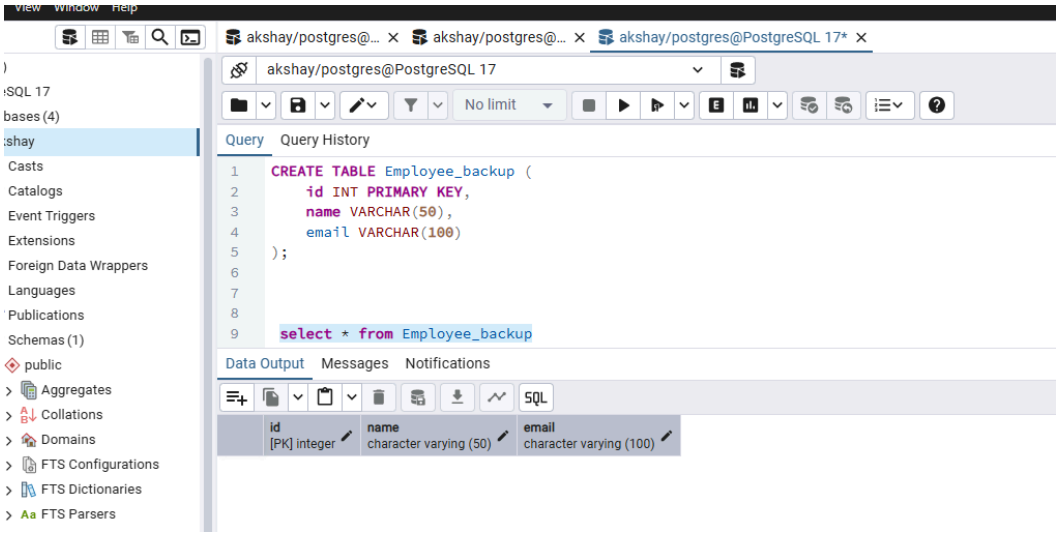
```
INSERT INTO Employee (id, name, email) VALUES
(1, 'Ankit', 'ankit@example.com'),
(2, 'Soham', 'soham@example.com'),
(3, 'Raman', 'raman@example.com');
```

Target Table

This is the **empty table** you want to copy data **into**.

Example:

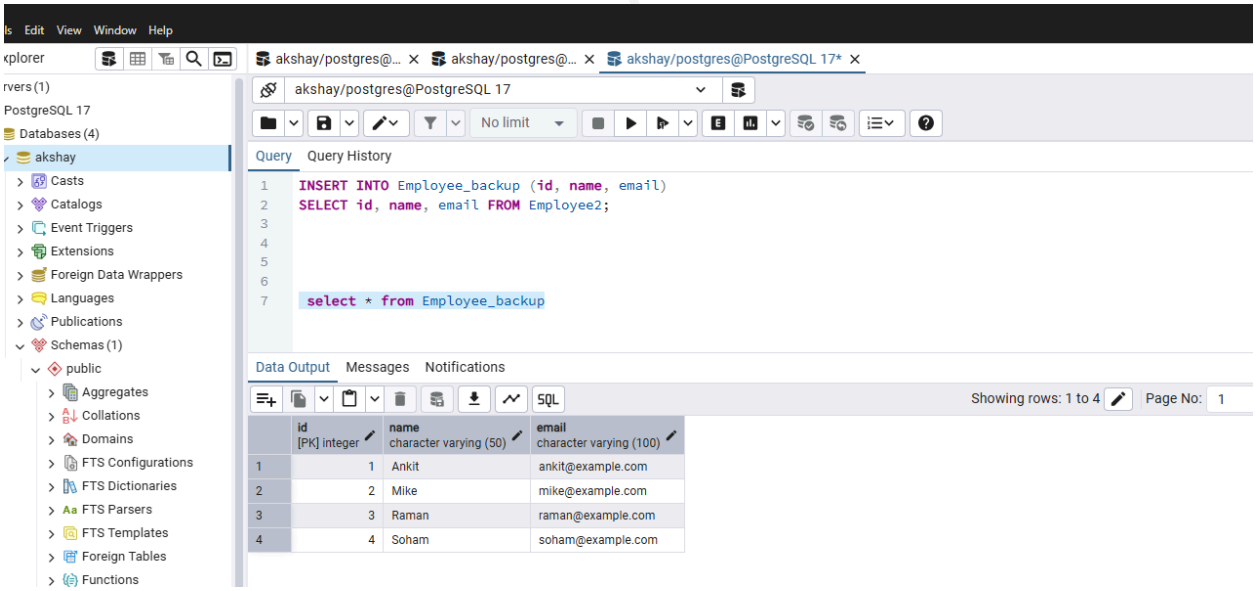
```
CREATE TABLE Employee_backup (
    id INT PRIMARY KEY,
    name VARCHAR(50),
    email VARCHAR(100)
);
```



Step 2: Use INSERT INTO SELECT

Run this query to **copy data**:

```
INSERT INTO Employee_backup (id, name, email)
SELECT id, name, email FROM Employee2;
```



Common Errors:

Error	Reason
Column count doesn't match	Missing or extra values
Data type mismatch	Wrong value type for column
NULL constraint violation	Not providing value for NOT NULL column
Duplicate primary key	Trying to insert same id twice

SQL DROP TABLE

Purpose:

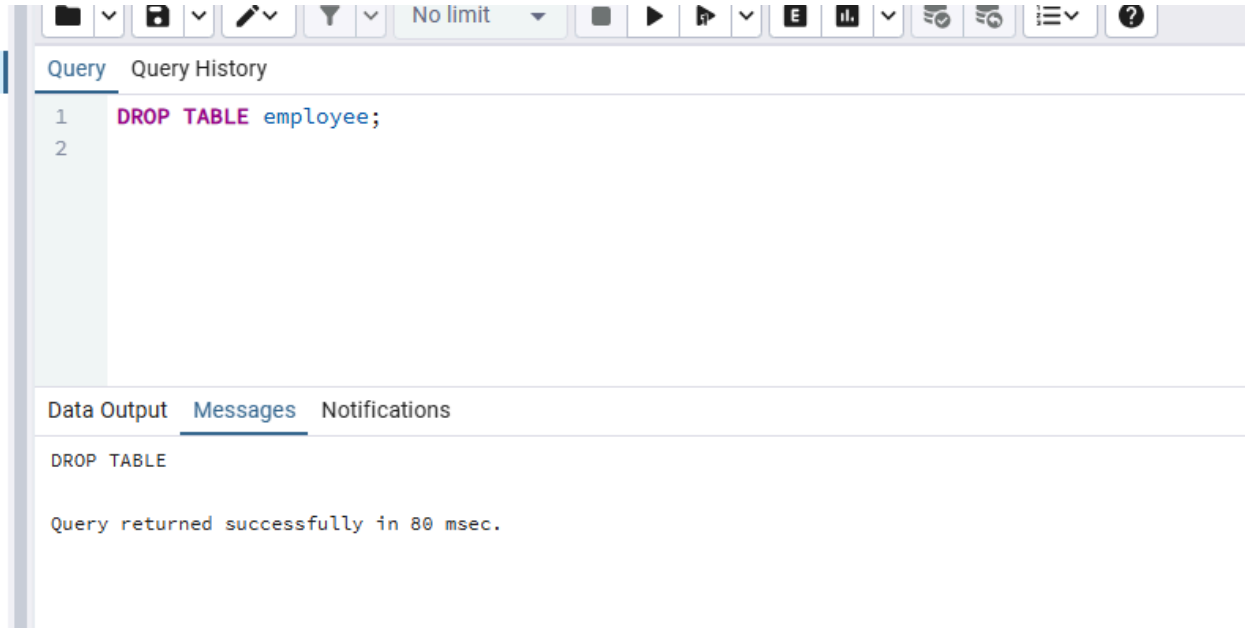
- Deletes the **table structure** and **all its data permanently**.

Syntax:

DROP TABLE table_name;

Example:

DROP TABLE employee;



Important:

After DROP TABLE, all **data, structure, relationships, and privileges** are lost.

SQL DELETE Statement

Purpose:

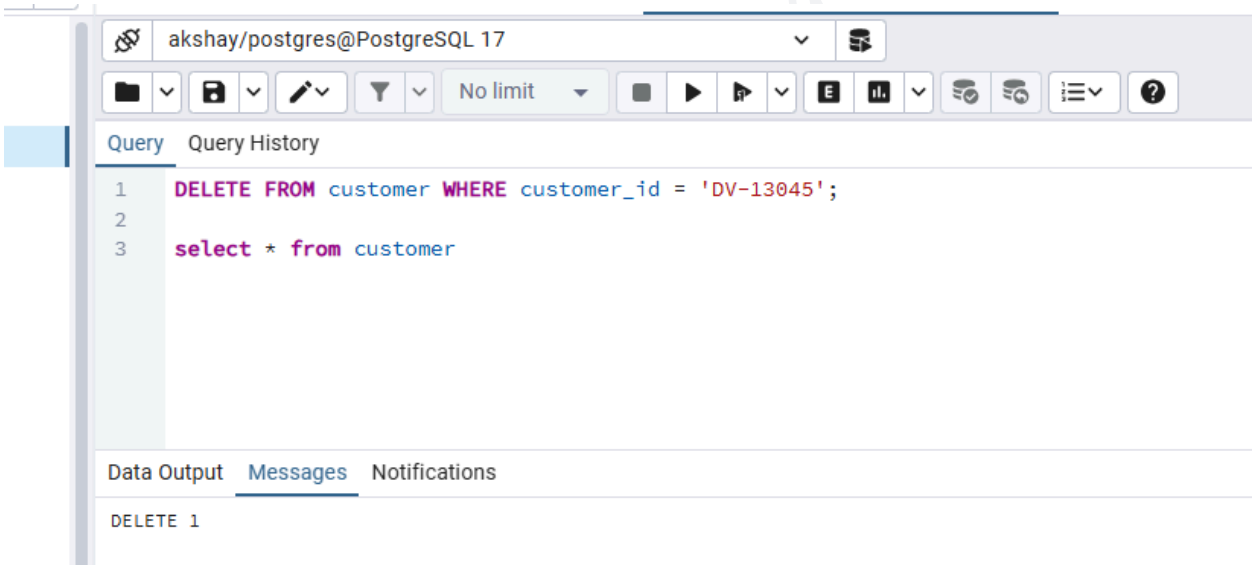
- Deletes **specific rows** from a table using WHERE condition.
- If WHERE is not used, **all rows** are deleted but table structure remains.

Syntax:

DELETE FROM table_name WHERE condition;

Example:

DELETE FROM customer WHERE customer_id = 'DV-13045';



SQL TRUNCATE TABLE

Purpose:

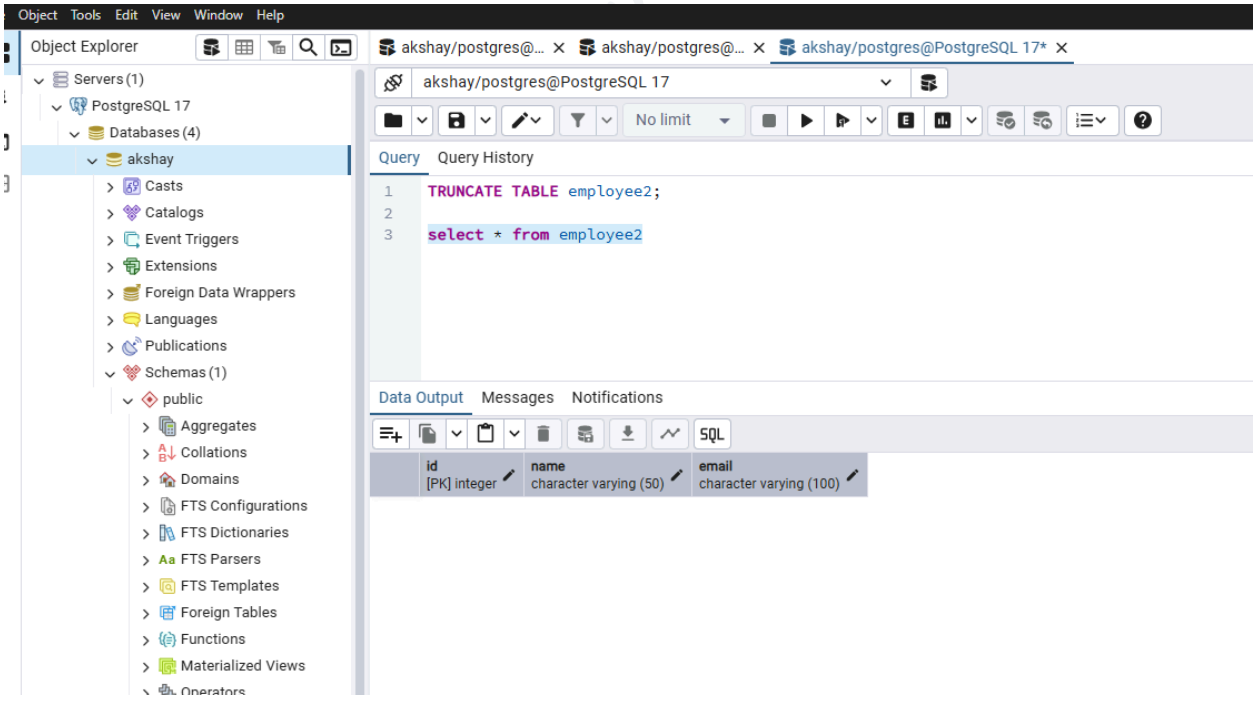
- Deletes **all rows** from a table.
- **Faster** than DELETE.
- Frees the space occupied by the data.
- **Keeps the table structure** intact.

Syntax:

TRUNCATE TABLE table_name;

Example:

TRUNCATE TABLE employee;



Difference Between DELETE and TRUNCATE

DELETE	TRUNCATE
Deletes **specific rows** (with WHERE).	Deletes **all rows** without condition.
Slower due to row-by-row deletion.	**Faster**, works like bulk deletion.
Table structure **remains**.	Table structure **remains**.
Log maintained for each row.	**Minimal log usage**.

Difference Between DROP and TRUNCATE

DROP TABLE	TRUNCATE TABLE
Removes **table structure and data**.	Removes **only data**.
Deletes **relationships and constraints**.	Keeps **structure, constraints** intact.
Cannot recover table easily.	Table remains for future use.

SQL RENAME TABLE

Purpose:

- Rename an existing table for better clarity or maintenance.

Syntax:

ALTER TABLE old_table_name RENAME TO new_table_name;

Example:

ALTER TABLE customer RENAME TO customers;

SQL ALTER TABLE

Purpose:

Modify an existing table to:

- Add columns
- Modify columns
- Delete columns
- Add or remove constraints
- Rename the table

Add Column Syntax:

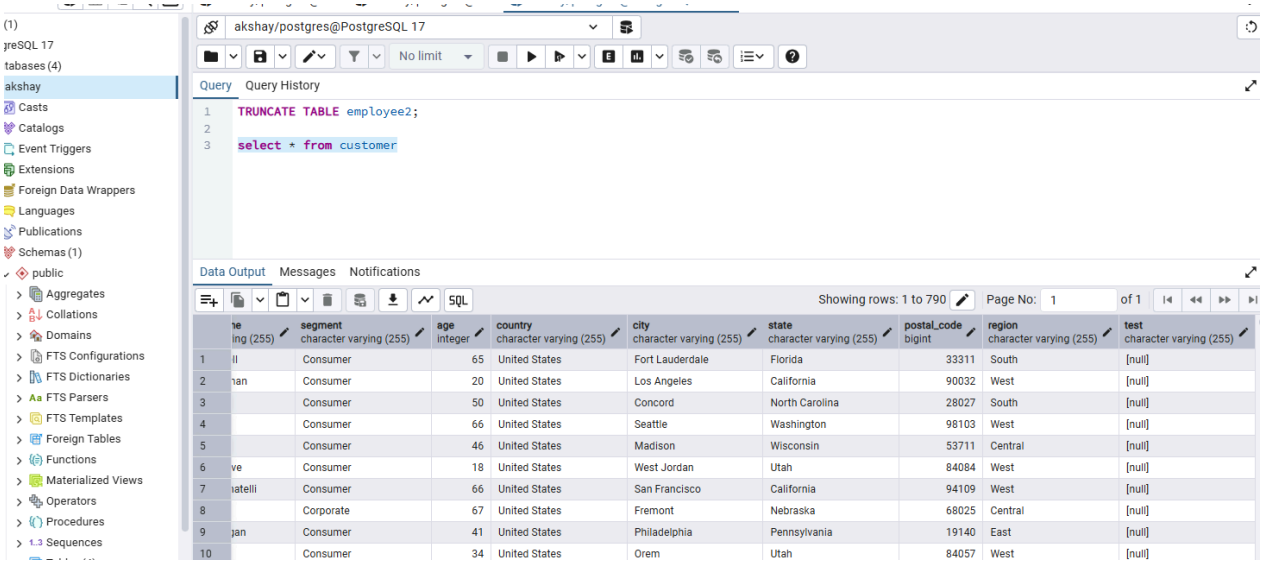
```
ALTER TABLE table_name
ADD column_name data_type;
```

Add Multiple Columns:

```
ALTER TABLE table_name
ADD (
   column1 data_type,
   column2 data_type,
   ...
);
```

Example:

```
ALTER TABLE customers
ADD COLUMN test varchar(255);
```



segment character varying (255)	age integer	country character varying (255)	city character varying (255)	state character varying (255)	postal_code bigint	region character varying (255)	test character varying (255)
Consumer	65	United States	Fort Lauderdale	Florida	33311	South	[null]
Consumer	20	United States	Los Angeles	California	90032	West	[null]
Consumer	50	United States	Concord	North Carolina	28027	South	[null]
Consumer	66	United States	Seattle	Washington	98103	West	[null]
Consumer	46	United States	Madison	Wisconsin	53711	Central	[null]
Consumer	18	United States	West Jordan	Utah	84084	West	[null]
Consumer	66	United States	San Francisco	California	94109	West	[null]
Corporate	67	United States	Fremont	Nebraska	68025	Central	[null]
Consumer	41	United States	Philadelphia	Pennsylvania	19140	East	[null]
Consumer	34	United States	Orem	Utah	84057	West	[null]

Summary of SQL Table Operations

Operation	SQL Command
Create Table	CREATE TABLE
Delete Table (Structure + Data)	DROP TABLE
Delete Specific Rows	DELETE
Delete All Rows (Keep Structure)	TRUNCATE TABLE
Rename Table	ALTER TABLE ... RENAME TO
Add Columns	ALTER TABLE ... ADD

Practice Questions: SQL Table Operations

1. CREATE TABLE

Q1. Write an SQL query to create a table Products with the following fields:

- ProductID (Primary Key, Integer)
- ProductName (Varchar 100)
- Price (Decimal(10,2))
- Stock (Integer)

Q2. Create a table Students with the following columns:

- RollNo (Primary Key, Integer)
- Name (Varchar 50)
- Class (Varchar 10)
- Marks (Float)

2. INSERT INTO

Insert the following record into Students:

RollNo	Name	Class	Marks
101	Rahul	10th	85.5

3. ALTER TABLE

Write a query to add a new column Email (varchar 100) to the Students table.

Q5. Add two columns to Products:

- Category (Varchar 50)
- Brand (Varchar 50)

4. DELETE & TRUNCATE

Q1. Delete all records from Students where Marks is less than 40.

Q2. Remove all data from Products but keep the table structure intact.

5. DROP TABLE

Write an SQL command to completely remove the Products table from the database.

6. RENAME TABLE

Rename the Students table to School_Students.

7.Difference Based Questions

What is the difference between the DELETE and TRUNCATE commands?
Give one example where TRUNCATE is preferred over DELETE.

What happens if you DROP a table that is referenced by a **foreign key** in another table?

You can download the complete set of SQL notes and practice files from this GitHub repository:
👉 **SQL-resources-and-tutorials by akshay-dhage**