**PROMPT :** **give me detail documentation of how the create table in postgresql with 2 example and i am a fresher.**

**📘 PostgreSQL CREATE TABLE – Complete Beginner's Guide**

**🔹 What is CREATE TABLE ?**

The CREATE TABLE command in PostgreSQL is used to define a new table in the database. A table is a collection of rows and columns, where each column has a data type, and each row is a record.

**🔹 Why is CREATE TABLE Important in PostgreSQL?**

✅ **a. It Defines the Blueprint of Data**

The CREATE TABLE command is like defining the schema or blueprint of how your data will be organized.

Just like a house cannot be built without a blueprint, a database cannot function without tables.

✅ **b. Ensures Data Consistency**

By using data types and constraints (like NOT NULL, UNIQUE, CHECK, etc.), you ensure that only valid and clean data is stored.

✅ **c. Supports Relationships**

PostgreSQL supports foreign keys, so tables can be linked together to model real-world relationships like:

* A student enrolled in many courses
* An employee working in a department

**✅ d. Enables Efficient Querying**

Well-designed tables allow fast searching, filtering, and report generation, which are essential for apps, analytics, and data-driven decisions.

**🔹 Basic Syntax**

sql

CREATE TABLE table\_name (

column1 data\_type [constraint],

column2 data\_type [constraint],

...

);

**🔸 Components:**

| **Part** | **Description** |
| --- | --- |
| **table\_name** | Name of the table you want to create. |
| **column1, column2, ...** | Names of the columns (fields) in the table. |
| **data\_type** | Type of data the column will store, e.g., INT, VARCHAR, DATE. |
| **constraint** | Optional rules like PRIMARY KEY, NOT NULL, UNIQUE, etc. |

**🔹 Common PostgreSQL Data Types**

| **Data Type** | **Description** |
| --- | --- |
| **INT or INTEGER** | Whole numbers |
| **SERIAL** | Auto-incrementing integers |
| **VARCHAR(n)** | Variable-length string (max n characters) |
| **TEXT** | Unlimited text |
| **DATE** | Date (YYYY-MM-DD) |
| **BOOLEAN** | True or False |
| **DECIMAL(p, s)** | Decimal numbers (precision p, scale s) |

**PostgreSQL Data Types Overview**

**Character Types:**

* VARCHAR(n): Variable-length string (limit n)
* CHAR(n): Fixed-length string (padded)
* TEXT: Unlimited-length string

**Numeric Types:**

* INTEGER: Whole numbers
* SMALLINT: Smaller whole numbers
* BIGINT: Large whole numbers
* SERIAL: Auto-increment integer
* NUMERIC(p,s): Fixed-point decimal

**Date/Time:**

* DATE: Calendar dates
* TIME: Time of day
* TIMESTAMP: Date and time

**Boolean:**

* BOOLEAN: TRUE, FALSE, or NULL

**🔹 Common Constraints**

| **Constraint** | **Meaning** |
| --- | --- |
| **PRIMARY KEY** | Uniquely identifies each record |
| **NOT NULL** | Column cannot be left empty |
| **UNIQUE** | All values must be different |
| **FOREIGN KEY** | Links one table to another |
| **DEFAULT** | Sets a default value |

**📘 The Role of CREATE Beyond Tables in PostgreSQL**

**🔹 1. CREATE TABLE**

Creates a new table to store data.

sql

CREATE TABLE students (

id SERIAL PRIMARY KEY,

name VARCHAR(100)

);

**🔹 2. CREATE DATABASE**

Creates a new database.

sql

CREATE DATABASE mydb;

**🔹 3. CREATE SCHEMA**

Organizes tables into a namespace.

sql

CREATE SCHEMA school;

**🔹 4. CREATE INDEX**

Improves search performance.

sql

CREATE INDEX idx\_name ON students(name);

**🔹 5. CREATE VIEW**

Creates a virtual table from a query.

sql

CREATE VIEW active\_students AS

SELECT \* FROM students WHERE is\_active = TRUE;

**🔹 6. CREATE FUNCTION**

Defines a custom function.

sql

CREATE FUNCTION add(a INT, b INT) RETURNS INT AS $$

BEGIN RETURN a + b; END;

$$ LANGUAGE plpgsql;

**🔹 7. CREATE TRIGGER**

Performs an automatic action on data changes (like insert/update).

**🔹 8. CREATE ROLE**

Adds a user or role to manage access.

sql

CREATE ROLE readonly\_user WITH LOGIN PASSWORD 'pass';

**🔹 9. CREATE EXTENSION**

Adds extra features to PostgreSQL.

sql

CREATE EXTENSION "uuid-ossp";

**EXAMPLE : -**

**✅ Example 1: Creating a students Table**

sql

CREATE TABLE students (

student\_id SERIAL PRIMARY KEY,

name VARCHAR(100) NOT NULL,

email VARCHAR(100) UNIQUE,

age INT,

enrollment\_date DATE DEFAULT CURRENT\_DATE

);

**🔍 Explanation:**

* student\_id is the primary key and auto-increments (SERIAL).
* name is required (NOT NULL).
* email must be unique.
* enrollment\_date will use the current date if no value is given.

**✅ Example 2: Creating a courses Table with a Foreign Key**

sql

CREATE TABLE courses (

course\_id SERIAL PRIMARY KEY,

course\_name VARCHAR(100) NOT NULL,

credits INT CHECK (credits > 0),

instructor VARCHAR(100),

student\_id INT REFERENCES students(student\_id)

);

**🔍 Explanation:**

* course\_id is the primary key.
* credits must be more than 0 (CHECK constraint).
* student\_id is a foreign key that references the students table.

**🔹 How to Run These Commands?**

1. Install PostgreSQL and open pgAdmin or use psql CLI.
2. Connect to your database.
3. Paste and run the SQL statements.