**PROMPT :- brief about INSERT COMMAND in postgerSQL, syntax. Also, give two example of INSERT COMMAND  as works for beignner.**

The INSERT command in PostgreSQL is used to add new rows of data into a table. It's a fundamental Data Manipulation Language (DML) command.

**Syntax**

The basic syntax for the INSERT command is:

SQL

**INSERT INTO table\_name (column1, column2, column3, ...)**

**VALUES (value1, value2, value3, ...);**

* INSERT INTO table\_name: Specifies the table where you want to insert the data.
* (column1, column2, ...): An optional list of columns you want to populate. If you provide this, you must supply a value for each listed column. If you omit this, you must provide values for all columns in the table, in the order they were defined.
* VALUES (value1, value2, ...): A list of values to be inserted, corresponding to the columns specified. The data types of the values must match the data types of the columns.

**Examples for Beginners**

Let's imagine we have a table called employees created with the following command:

**CREATE TABLE employees (**

**id SERIAL PRIMARY KEY,**

**first\_name VARCHAR(50),**

**last\_name VARCHAR(50),**

**email VARCHAR(100),**

**hire\_date DATE**

**);**

**Example 1: Inserting a full row**

This example shows how to insert a new employee, providing values for every column except id (which is a SERIAL type and automatically generated).

SQL

INSERT INTO employees (first\_name, last\_name, email, hire\_date)

VALUES ('John', 'Doe', 'john.doe@example.com', '2023-01-15');

In this case, we specify the columns we want to fill and then provide the corresponding values. This is generally the safest and most common method, as it makes your code clear and less prone to errors if the table structure changes.

**Example 2: Inserting a partial row**

Sometimes you may only have data for a few columns. This example shows how to insert an employee with just their first and last name. The other columns (email, hire\_date) will be set to NULL (or their default value if one was specified).

SQL

**INSERT INTO employees (first\_name, last\_name)**

**VALUES ('Jane', 'Smith');**

This works because the email and hire\_date columns are not marked as NOT NULL, allowing them to be empty. This demonstrates the flexibility of the INSERT command to handle incomplete data.