## Prompt: Tell Me In Brief About INSERT Command In Postgresql, What It Do, How It Do, Where We Use It And What Is Structure Of It With 2 Examples, Explain Me Like A Teacher And I Am New In This Field Explain Step By Step In Ascending Order Of Difficulty Level

The **INSERT** command in PostgreSQL is used to add new rows of data into a table. If the CREATE command builds the house, the INSERT command moves the furniture in. 🛋️

### What it does and Where we use it

Imagine you have a spreadsheet for your products. The INSERT command is the action of filling out a new, empty row with the details of a new product.

You use it **every time you need to add new records to your database**. This could be when:

* A new user signs up on your website.
* You add a new product to your online store.
* A customer places a new order.

The basic structure is to specify the **table**, the **columns** you want to fill, and the **values** for those columns.

INSERT INTO table\_name (column\_list) VALUES (value\_list);

Let's use the products table we created earlier and see two examples.

-- This is the structure of our table for reference  
CREATE TABLE products (  
 product\_id SERIAL PRIMARY KEY,  
 product\_name VARCHAR(100) NOT NULL,  
 price NUMERIC(8, 2)  
);

### Level 1: Inserting a Single Row

This is the most common and straightforward use of the INSERT command. You provide the values for one new row.

#### Structure

You list the specific columns you are providing data for, then provide the corresponding values. The order of the values must match the order of the columns.

INSERT INTO table\_name (column1, column2, column3)  
VALUES (value1, value2, value3);

#### Example

Let's add a new product, a 'Gaming Mouse', to our products table.

1. **The Command**:  
     
   INSERT INTO products (product\_name, price)  
   VALUES ('Gaming Mouse', 75.50);
2. **What Happens?**:
   * INSERT INTO products: We tell PostgreSQL we want to add data to the products table.
   * (product\_name, price): We specify that we are only providing values for these two columns. Notice we've omitted product\_id. Because product\_id is a SERIAL type, the database will automatically generate a unique ID for it (it will start with 1).
   * VALUES ('Gaming Mouse', 75.50): We provide the actual data. 'Gaming Mouse' corresponds to product\_name, and 75.50 corresponds to price. The text is enclosed in single quotes.

After running this command, your products table will have one row in it.

### Level 2: Inserting Multiple Rows at Once

To be more efficient, you can add several rows with a single INSERT command. This is much faster than running separate commands for each row.

#### Structure

The structure is the same as before, but you provide a comma-separated list of value sets.

INSERT INTO table\_name (column1, column2)  
VALUES  
 (row1\_value1, row1\_value2),  
 (row2\_value1, row2\_value2),  
 (row3\_value1, row3\_value2);

#### Example

Let's add three more products to our store in one go.

1. **The Command**:  
     
   INSERT INTO products (product\_name, price)  
   VALUES  
    ('Mechanical Keyboard', 120.00),  
    ('4K Webcam', 89.99),  
    ('USB-C Hub', 25.00);

1. **What Happens?**:
   * PostgreSQL takes each parenthesis-enclosed set of values as a new row to be inserted.
   * ('Mechanical Keyboard', 120.00) becomes the first new row.
   * ('4K Webcam', 89.99) becomes the second new row.
   * ('USB-C Hub', 25.00) becomes the third.
   * The database will automatically assign the next available product\_id values (2, 3, and 4) to these new rows.

Now, your products table is populated with four different items, and you did it with just two commands!