**Prompt 1 :give detail documentation for fresher of insert statement with syntax in postgresql,what are different caluse to used with insert**

## **1. Introduction**

The INSERT statement in **PostgreSQL** is used to **add new rows** to a table. It is one of the most fundamental operations in SQL, essential for data entry and database population. This statement allows you to insert single or multiple rows, insert using subqueries, handle conflicts, and even return the newly inserted values.

## **2. Basic Syntax**

INSERT INTO table\_name (column1, column2, ..., columnN)

VALUES (value1, value2, ..., valueN);

### **✅ Description:**

* table\_name: The name of the table where the data is to be inserted.
* column1, column2, ..., columnN: The target columns in the table.
* value1, value2, ..., valueN: The values that correspond to each column.

## **3. Theoretical Explanation**

When you use the INSERT command, PostgreSQL allocates a new row in the table and stores the provided values in their respective columns. You must ensure that:

* The **number of columns** matches the **number of values**.
* The **data types** match (e.g., you cannot insert text into an integer column).
* You **respect constraints** like NOT NULL, UNIQUE, or PRIMARY KEY.

## **4. Examples and Use Cases**

### **🔹 4.1. Insert a Single Row**

INSERT INTO student (name, roll\_no, age, email)

VALUES ('Aarav Sharma', 'R101', 21, 'aarav.sharma@example.com');

This inserts one row into the student table.

### **🔹 4.2. Insert Multiple Rows**

INSERT INTO student (name, roll\_no, age, email)

VALUES

('Riya Mehta', 'R102', 20, 'riya@example.com'),

('Vikram Das', 'R103', 22, 'vikram@example.com');

Multiple rows can be inserted in a single query, which is more efficient and concise.

### **🔹 4.3. Insert with All Columns (Using Default)**

If your table has default values (e.g., SERIAL for auto-incrementing id), you can omit that column:

INSERT INTO student

VALUES (DEFAULT, 'Nisha Roy', 'R104', 23, 'nisha@example.com');

### **🔹 4.4. Insert Using Subquery (INSERT INTO ... SELECT)**

INSERT INTO alumni (name, roll\_no, graduation\_year)

SELECT name, roll\_no, 2023

FROM student

WHERE age > 22;

This copies filtered data from one table to another.

### **🔹 4.5. Insert and Return Values (RETURNING)**

INSERT INTO student (name, roll\_no, age, email)

VALUES ('Tanvi Kapoor', 'R105', 21, 'tanvi@example.com')

RETURNING id, name;

The RETURNING clause shows values from the inserted row, often used to retrieve the generated primary key.

### **🔹 4.6. Insert with Conflict Handling (ON CONFLICT)**

#### **➤ Ignore the insert if conflict:**

INSERT INTO student (id, name, roll\_no, age, email)

VALUES (1, 'Arjun', 'R106', 21, 'arjun@example.com')

ON CONFLICT (id) DO NOTHING;

#### **➤ Update the existing row on conflict:**

INSERT INTO student (id, name, roll\_no, age, email)

VALUES (1, 'Arjun Updated', 'R106', 22, 'arjun.upd@example.com')

ON CONFLICT (id)

DO UPDATE SET name = EXCLUDED.name, age = EXCLUDED.age;

**Note**: EXCLUDED.column\_name refers to the value you attempted to insert.

### **🔹 4.7. Insert with NULL Values**

INSERT INTO student (name, roll\_no, age, email)

VALUES ('Unknown', 'R999', NULL, NULL);

This is valid only if the columns accept NULL values.

## **5. Clauses Used with INSERT in PostgreSQL**

| **Clause** | **Purpose** | **Example** |
| --- | --- | --- |
| DEFAULT | Inserts default value for a column | DEFAULT, 'Name'... |
| RETURNING | Returns inserted data like id | RETURNING id, name |
| ON CONFLICT | Handles primary key or unique constraint violations | ON CONFLICT (id) DO NOTHING |
| SELECT | Insert result of a query into another table | INSERT INTO ... SELECT ... |

## **6. Use Case Summary**

| **Use Case** | **Syntax** |
| --- | --- |
| Insert a single record | INSERT INTO table (...) VALUES (...); |
| Insert multiple records | VALUES (...), (...), ... |
| Insert default value | DEFAULT in VALUES |
| Insert with null | Use NULL where applicable |
| Insert from another table | INSERT INTO ... SELECT ... |
| Avoid duplicate key error | ON CONFLICT DO NOTHING |
| Update on conflict | ON CONFLICT (...) DO UPDATE SET ... |
| Retrieve inserted row | Use RETURNING |

## **7. Common Errors & Fixes**

| **Error Message** | **Cause** | **Solution** |
| --- | --- | --- |
| null value in column violates not-null constraint | Inserted NULL where not allowed | Provide a valid value or use DEFAULT |
| duplicate key value violates unique constraint | Inserting duplicate value in a UNIQUE/PK column | Use ON CONFLICT clause |
| INSERT has more expressions than target columns | Mismatch between columns and values | Match the number and order of columns and values |

## **8. Conclusion**

The INSERT statement is a foundational tool for data manipulation in PostgreSQL. As a fresher, it is important to:

* Understand the core syntax and its flexibility.
* Learn to use it efficiently with DEFAULT, RETURNING, ON CONFLICT, and SELECT.
* Prevent common errors using proper constraints and checks.

Regular practice with real-world data will strengthen your confidence and proficiency in using INSERT.