**PROMPT:** give me the detailed documentation of select command with examples in PostgreSQL

## **Overview of SELECT in PostgreSQL**

The SELECT statement is used to **retrieve data from one or more tables or views** in PostgreSQL.  
 It is the most frequently used command in SQL and forms the basis of querying databases.

**Basic Syntax:**

SELECT column1, column2, ...

FROM table\_name

WHERE condition

GROUP BY column

HAVING condition

ORDER BY column

LIMIT n OFFSET m;

## **Key Points**

* **SELECT**: Specifies the columns to be retrieved.
* **\***: Selects all columns.
* **FROM**: Specifies the table or tables to query.
* **WHERE**: Filters rows based on conditions.
* **ORDER BY**: Sorts the results.
* **GROUP BY**: Groups rows for aggregation.
* **HAVING**: Filters groups.
* **LIMIT** and **OFFSET**: Restrict the number of rows returned.

## **Examples by Type**

### **1. Selecting All Columns**

SELECT \*

FROM employees;

* Retrieves all rows and columns from the employees table.

**2.** **Selecting Specific Columns**

SELECT first\_name, last\_name, salary

FROM employees;

* Only first\_name, last\_name, and salary will be shown.

### **3. Using Aliases**

SELECT first\_name AS fname, last\_name AS lname

FROM employees;

* Aliases rename columns in the result set.

### **4. Filtering Rows with WHERE**

SELECT first\_name, salary

FROM employees

WHERE salary > 50000;

* Only employees with a salary greater than 50,000 are returned.

### **5. Using Operators**

SELECT \*

FROM employees

WHERE department = 'HR' AND salary BETWEEN 30000 AND 50000;

* Uses AND and BETWEEN operators for filtering.

### **6. Sorting Results**

SELECT first\_name, salary

FROM employees

ORDER BY salary DESC;

* Sorts employees by salary in descending order.

### **7. Removing Duplicates with DISTINCT**

SELECT DISTINCT department

FROM employees;

* Returns unique department names.

### **8. Pattern Matching**

SELECT first\_name

FROM employees

WHERE first\_name LIKE 'A%';

* Returns names starting with "A".

### **9. Aggregations with GROUP BY**

SELECT department, AVG(salary) AS avg\_salary

FROM employees

GROUP BY department;

* Returns the average salary for each department.

### **10. Filtering Groups with HAVING**

SELECT department, COUNT(\*) AS emp\_count

FROM employees

GROUP BY department

HAVING COUNT(\*) > 5;

* Shows departments with more than 5 employees.

### **11. Limiting Results**

SELECT \*

FROM employees

ORDER BY salary DESC

LIMIT 5;

* Retrieves only the top 5 highest salaries.

### **12. Pagination with LIMIT & OFFSET**

SELECT \*

FROM employees

ORDER BY id

LIMIT 5 OFFSET 5;

* Skips the first 5 rows and returns the next 5.

### **13. Selecting from Multiple Tables (JOIN)**

SELECT e.first\_name, d.department\_name

FROM employees e

JOIN departments d ON e.department\_id = d.id;

* Combines data from employees and departments using an **INNER JOIN**.

### **14. Subqueries**

SELECT first\_name, salary

FROM employees

WHERE salary > (SELECT AVG(salary) FROM employees);

* Returns employees whose salary is above the company average.

### **15. Selecting Without a Table (PostgreSQL Feature)**

SELECT NOW();

* Retrieves the current timestamp without querying a table.

## **Common Operators in SELECT WHERE Clause**

| **Operator** | **Description** | **Example** |
| --- | --- | --- |
| = | Equal to | salary = 50000 |
| != or <> | Not equal to | department <> 'HR' |
| > < >= <= | Greater/Less than | salary > 60000 |
| BETWEEN | Range (inclusive) | salary BETWEEN 30000 AND 50000 |
| IN | Match against a list | department IN ('HR','IT') |
| LIKE | Pattern matching | name LIKE 'A%' |
| ILIKE | Case-insensitive LIKE (PostgreSQL) | name ILIKE 'a%' |
| IS NULL | Check for NULL | phone IS NULL |

## **Best Practices**

* Always select only the required columns instead of SELECT \* in production.
* Use aliases (AS) for better readability.
* Apply filtering early (WHERE) to improve performance.
* Combine **LIMIT** with **ORDER BY** for predictable results.
* Use **ILIKE** instead of **LIKE** for case-insensitive searches in PostgreSQL.