SQL Grouping Process

Let's go through the process of how a SQL database groups data, step by step, and see how the data would look right after the grouping, but before any aggregation like COUNT is applied.

Example Table: employees

department	employee_id	salary
HR	1	50000
HR	2	55000
IT	3	70000
IT	4	72000
Finance	5	60000
Finance	6	62000

Step 1: Identifying Groups Based on department

When the SQL query groups by department, it essentially clusters rows that have the same department value together. The database creates separate "buckets" or groups for each unique department.

Step 2: The Data After Grouping (Before Aggregation)

After grouping, the data is organized into these groups:

Group 1: HR

- Rows that belong to the HR department are grouped together.

| department | employee_id | salary |

|------------|-------------|--------|

| HR | 1 | 50000 |

| HR | 2 | 55000 |

Group 2: IT

- Rows that belong to the IT department are grouped together.

| department | employee_id | salary |

|------------|-------------|--------|

| IT | 3 | 70000 |

| IT | 4 | 72000 |

Group 3: Finance

- Rows that belong to the Finance department are grouped together.

| department | employee_id | salary |

|------------|-------------|--------|

| Finance | 5 | 60000 |

| Finance | 6 | 62000 |

Visualization of the Grouping Process

1. Original Table:

| department | employee_id | salary |

|------------|-------------|--------|

| HR | 1 | 50000 |

| HR | 2 | 55000 |

| IT | 3 | 70000 |

| IT | 4 | 72000 |

| Finance | 5 | 60000 |

| Finance | 6 | 62000 |

2. After Grouping by department:

- HR Group:

| department | employee_id | salary |

|---|---|---|

| HR | 1 | 50000 |

| HR | 2 | 55000 |

- IT Group:

| department | employee_id | salary |

|---|---|---|

| IT | 3 | 70000 |

| IT | 4 | 72000 |

- Finance Group:

| department | employee_id | salary |

|---|---|---|

| Finance | 5 | 60000 |

| Finance | 6 | 62000 |

Key Point:

- Grouping does not alter the rows: The rows themselves remain intact, but they are logically organized into groups based on the values in the department column. The database internally recognizes these groups as separate entities, ready for aggregation functions to be applied.

Step 3: Applying Aggregation (Optional)

After grouping, you can apply aggregate functions (like COUNT, SUM, AVG, etc.) to summarize or manipulate the data within each group. This is where the grouped data gets processed into a final result set.

For example, applying COUNT(*) would give:

department	count
HR	2
IT	2
Finance	2

But just after the grouping step, the data is still in its original form, just organized into the identified groups based on the department values.