**Day 3-4 Mastering Aggregations**

**Subtopic 1: Aggregate Functions (COUNT, SUM, AVG, MIN, MAX)**

* **Definition:** An aggregate function performs a calculation on a set of rows and returns a single summary value.
  + COUNT(): Counts the number of rows.
  + SUM(): Calculates the sum of a set of values.
  + AVG(): Calculates the average of a set of values.
  + MIN(): Gets the minimum value in a set.
  + MAX(): Gets the maximum value in a set.
* **Use/Importance:** These functions are the foundation of data analysis and business intelligence. They allow you to transform raw data into meaningful insights, such as total revenue, average customer rating, or the number of items in stock.
* **Syntax/Structure:**

SELECT AGGREGATE\_FUNCTION(column\_name)

FROM table\_name;

* **Example:** To find the total sales amount and the total number of orders from an orders table.

SELECT SUM(order\_total) AS total\_revenue, COUNT(order\_id) AS number\_of\_orders

FROM orders;

**Subtopic 2: Grouping Data with GROUP BY**

* **Definition:** The GROUP BY clause is used with aggregate functions to group rows that have the same values in specified columns into summary rows.
* **Use/Importance:** This is how you move from analyzing a whole dataset to analyzing specific segments within it. Instead of asking for the total sales, you can ask for the total sales *per store* or *per product category*. It is one of the most powerful clauses in SQL for reporting.
* **Syntax/Structure:**

SELECT column\_to\_group\_by, AGGREGATE\_FUNCTION(column\_to\_aggregate)

FROM table\_name

GROUP BY column\_to\_group\_by;

* **Example:** To count the number of employees in each department.

SELECT department, COUNT(employee\_id) AS number\_of\_employees

FROM employees

GROUP BY department;

**Subtopic 3: Filtering Groups (HAVING vs. WHERE)**

* **Definition:** Both clauses filter data, but they operate at different stages of a query.
  + WHERE filters individual rows **before** they are grouped and aggregated.
  + HAVING filters entire groups **after** the data has been grouped and the aggregate functions have been calculated.
* **Use/Importance:** Understanding this difference is critical for complex analysis. Use WHERE to exclude irrelevant rows from your calculation (e.g., only consider 'Completed' orders). Use HAVING to filter the results of your calculation (e.g., only show me departments with more than 10 employees).
* **Syntax/Structure:**

SQL

SELECT column1, AGGREGATE\_FUNCTION(column2)

FROM table\_name

WHERE row\_condition -- Filters rows before grouping

GROUP BY column1

HAVING group\_condition; -- Filters groups after aggregating

* **Example:** Find all customer categories with an average purchase value over $100, but only consider purchases made after '2025-01-01'.

SQL

SELECT category, AVG(purchase\_value) AS avg\_purchase

FROM sales

WHERE sale\_date > '2025-01-01' -- Filtering rows first

GROUP BY category

HAVING AVG(purchase\_value) > 100; -- Filtering the resulting groups

**Quick Recap:**

* Aggregate functions (COUNT, SUM, AVG, MIN, MAX) summarize many rows into a single value.
* The GROUP BY clause is used to apply aggregate functions to specific subsets of data.
* **WHERE** filters rows **before** aggregation.
* **HAVING** filters groups **after** aggregation and requires a GROUP BY clause.
* You cannot use an aggregate function (like SUM() or COUNT()) in a WHERE clause; you must use it in a HAVING clause.

**Practice Tasks:**

1. **Task 1:** From a sales table with store\_id and sale\_amount columns, write a query to find the total sales for each store.
2. **Task 2:** Using an orders table with order\_date and order\_value, find the average order value for each month of the year 2025.
3. **Task 3:** From an employees table with department and employee\_id columns, find the departments that have more than 5 employees.
4. **Task 4:** Using a products table with category and price columns, find the most expensive and least expensive product price within each category.