Sắp xếp và nhóm dữ liệu

ORDER BY

- **ASC:** Sắp xếp tăng dần (mặc định nếu không chỉ định)
- **DESC:** Sắp xếp giảm dần

```
SELECT column1, column2, ...
FROM table_name
ORDER BY column1 [ASC|DESC], column2 [ASC|DESC], ...;
```

```
-- Sắp xếp khách hàng theo tên tăng dần
SELECT * FROM Customers ORDER BY CustomerName;
```

```
-- Sắp xếp theo quốc gia (tăng dần) và tên (giảm dần)
SELECT * FROM Customers
ORDER BY Country ASC, CustomerName DESC;
```

ORDER BY - Vı du

```sql
-- Sắp xếp theo chiều dài tên
SELECT CustomerName, City
FROM Customers
ORDER BY LENGTH(CustomerName);
```

```sql
-- Sắp xếp theo năm trong ngày đặt hàng
SELECT OrderID, CustomerID, OrderDate
FROM Orders
ORDER BY YEAR(OrderDate), MONTH(OrderDate);
```

ORDER BY - Vı du

Sắp xếp danh sách sản phẩm theo tổng giá sau khi áp dụng giảm giá

Bảng **`products`**:

- `id` (ID sản phẩm)
- `name` (Tên sản phẩm)
- `price` (Giá gốc)
- `discount` (Giảm giá theo %)

```
SELECT id, name, price, discount,
      price * (1 - discount / 100) AS final_price
FROM products
ORDER BY final_price ASC;
```

ORDER BY với CASE

Sắp xếp ưu tiên khách VIP

```sql
-- Sử dụng CustomerRank để đo độ ưu tiên
SELECT *,
       CASE
         WHEN CustomerType = 'VIP' THEN 1
         WHEN CustomerType = 'Regular' THEN 2
         ELSE 3
       END AS CustomerRank
FROM Customers
ORDER BY CustomerRank, CustomerName;
```

ORDER BY với **LIMIT**

```sql
-- Lấy 5 sản phẩm có giá cao nhất
SELECT * FROM Products
ORDER BY Price DESC
LIMIT 5;
```

```sql
-- Lấy sản phẩm có giá cao thứ 6-10
SELECT * FROM Products
ORDER BY Price DESC
LIMIT 5 OFFSET 5;
```

ORDER BY

- Sắp xếp theo **vị trí cột trong kết quả** (thay vì thuộc tính)

```
-- Sắp xếp theo cột thứ 3 trong kết quả
SELECT CustomerID, CustomerName, Country
FROM Customers
ORDER BY 3;
```

GROUP BY

- **GROUP BY** nhóm các hàng **có cùng giá trị** để sử dụng các lệnh thống kê
- Ví dụ: Đếm số lượng giảng viên của mỗi khoa
  - Phải nhóm các giảng viên trong cùng 1 khoa vào thành 1 nhóm
  - Sử dụng hàm COUNT(*) để Đếm số lượng trong mỗi nhóm

```
select dept_name, count(*) as counter
from instructor
group by dept_name;
```

#	dept_name	counter
1	Khoa Công nghệ Thông tin	3
2	Khoa Điện tử Viễn thông	2
3	Khoa Khoa học Máy tính	2
4	Khoa Kỹ thuật Máy tính	2
5	Khoa Khoa học Dữ liệu	2
6	Khoa An toàn Thông tin	2

GROUP BY

- **GROUP BY** nhóm các hàng **có cùng giá trị** để sử dụng các lệnh thống kê
- Ví dụ: Tính tổng lương giảng viên của mỗi khoa
  - Phải nhóm các giảng viên trong cùng 1 khoa vào thành 1 nhóm
  - Sử dụng hàm SUM() để tính tổng lương trong mỗi nhóm

```
select dept_name, SUM(salary) as total
from instructor
group by dept_name;
```

#	dept_name	total
1	Khoa Công nghệ Thông tin	86000000.00
2	Khoa Khoa học Dữ liệu	65050000.00
3	Khoa An toàn Thông tin	58100000.00
4	Khoa Điện tử Viễn thông	57000000.00
5	Khoa Khoa học Máy tính	54500000.00
6	Khoa Kỹ thuật Máy tính	52800000.00

GROUP BY

❖ **Cú pháp GROUP BY**

```
SELECT column_name(s)
FROM table_name
WHERE condition
GROUP BY column_name(s)
ORDER BY column_name(s);
```

```
SELECT column1, column2, aggregate_function(column3)
FROM table_name
GROUP BY column1, column2;
```

GROUP BY nhieu thuoc tinh

department	position	employee_name	salary
HR	Manager	Alice	7000
HR	Staff	Bob	5000
IT	Manager	Charlie	8000
IT	Staff	David	6000
IT	Staff	Emma	6200

Đếm số nhân viên trong mỗi `department` và `position`:

department	position	total_employees
HR	Manager	1
HR	Staff	1
IT	Manager	1
IT	Staff	2

```sql
SELECT department, position, COUNT(*) AS total_employees
FROM employees
GROUP BY department, position;
-- ORDER BY department ASC, total_employees DESC;
```

Lưu y khi sư dung GROUP BY - HAVING

- Tất cả các cột trong **SELECT phải được nhóm** (GROUP BY) hoặc sử dụng trong một hàm tổng hợp (SUM, COUNT, AVG, MIN, MAX...), nếu không sẽ bị lỗi.

LỖI `salary`

```
SELECT department, salary FROM employees
GROUP BY department;
```

ĐÚNG

```
SELECT department, AVG(salary)
FROM employees GROUP BY department;
```

- **GROUP BY** chạy trước **SELECT** và trước **ORDER BY**
- Sử dụng **HAVING** thay vì **WHERE** để lọc nhóm
  - ➢ WHERE lọc từng dòng dữ liệu trước khi nhóm, còn HAVING lọc sau khi nhóm

HAVING

- **HAVING** được dùng để lọc kết quả **sau khi dữ liệu đã được nhóm bằng GROUP BY**.
- **WHERE** chỉ lọc dữ liệu trước khi nhóm
- **HAVING** lọc dữ liệu **sau khi đã tính toán các hàm tổng hợp** (COUNT(), SUM(), AVG(), MIN(), MAX())

```
SELECT column1, column2, aggregate_function(column3)
FROM table_name
GROUP BY column1, column2
HAVING condition;
```

HAVING - Ví dụ

- Tính tổng lương từng phòng ban, lọc ra những phòng ban có tổng lương > 50,000

```
SELECT department, SUM(salary) AS total_salary
FROM employees
GROUP BY department
HAVING total_salary > 50000;
```

- Đếm số nhân viên trong từng phòng ban, chỉ lấy phòng có trên 5 nhân viên

```
SELECT department, COUNT(*) AS employee_count
FROM employees
GROUP BY department
HAVING employee_count > 5;
```

HAVING ket hợp WHERE

- Tính trung bình lương của những nhân viên có lương trên 3000
- Chỉ giữ lại những phòng ban có lương trung bình trên 5000

```
SELECT department, AVG(salary) AS avg_salary
FROM employees
1 WHERE salary > 3000  -- Loại bỏ nhân viên có lương dưới 3000 trước khi nhóm
GROUP BY department
2 HAVING avg_salary > 5000;  -- Chỉ lấy nhóm có lương trung bình trên 5000
```

Thư tự thực thi cau lenh SQL

```sql
SELECT department,
     COUNT(*) AS total_employees
FROM employees
WHERE salary > 3000
GROUP BY department
HAVING total_employees > 5
ORDER BY total_employees DESC
LIMIT 3;
```

#	Câu lệnh	Mô tả
1	FROM employees	Chọn bảng employees.
2	WHERE salary > 3000	Lọc nhân viên có lương trên 3000.
3	GROUP BY department	Nhóm dữ liệu theo department.
4	COUNT(*) AS total_employees	Tính số nhân viên trong mỗi nhóm.
5	HAVING total_employees > 5	Giữ lại nhóm có hơn 5 nhân viên.
6	SELECT department, …	Chọn cột cần hiển thị.
7	ORDER BY total_employees DESC	Sắp xếp theo số nhân viên giảm dần.
8	LIMIT 3	Giới hạn kết quả trả về 3 dòng.

ORDER BY - GROUP BY - Ví dụ

Tìm lần cuối cùng mỗi xe được sử dụng (tức là lấy end_time lớn nhất của từng bike_number). Kết quả được sắp xếp theo thứ tự xe được sử dụng gần nhất.

```
select bike_number, max(end_time) end_time
from Bikes
group by bike_number
order by max(end_time) DESC
```