Tuần 1 - Tổng hợp kiến thức Buổi học số 3

Time-Series Team

Ngày 8 tháng 6 năm 2025

Buổi học số 3 (Thứ 5, 05/06/2025) bao gồm hai nội dung chính:

- ***xxx***
- ***xxx***

1 Giới thiệu Cơ sở Dữ liệu (Database)

1.1 Định nghĩa cơ sở dữ liệu và hệ quản trị

Cơ sở dữ liệu (Database) là tập hợp thông tin hoặc dữ liệu được lưu trữ và tổ chức theo một định dạng có cấu trúc. Một số ví dụ dễ hình dung về cơ sở dữ liệu bao gồm:

- Danh bạ điện thoại.
- Danh sách mua sắm.
- Mạng lưới bạn bè trên mạng xã hội.

Cơ sở dữ liệu giúp lưu trữ thông tin một cách hệ thống, cho phép truy xuất, quản lý và xử lý hiệu quả.

Hệ quản trị cơ sở dữ liệu (Database Management System - DBMS) là phần mềm hoặc ứng dụng cho phép người dùng tương tác với cơ sở dữ liệu. Cấu trúc hoạt động của DBMS gồm:

- Người dùng gửi **câu lệnh** đến DBMS.
- DBMS thực thi câu lệnh trên **cơ sở dữ liệu**.
- Kết quả trả về cho người dùng.

Ví dụ: Khi người dùng gửi truy vấn SQL, DBMS sẽ xử lý và trả về kết quả tương ứng.

1.2 Tại sao cơ sở dữ liệu quan trọng?

Trong thời đại dữ liệu số, dữ liệu là nền tảng cho mọi quyết định kinh doanh, khoa học, và công nghệ. Hệ thống cơ sở dữ liệu giúp lưu trữ, truy cập, và quản lý khối lượng dữ liệu khổng lồ một cách hiệu quả và có tổ chức. Một số lý do chính khiến cơ sở dữ liệu trở nên quan trọng:

- **Tránh trùng lặp dữ liệu (data redundancy)** và đảm bảo **tính nhất quán (data consistency)**.
- Dễ dàng truy xuất và xử lý dữ liệu nhanh chóng.
- Hỗ trợ phân quyền, bảo mật và khôi phục dữ liệu khi có sự cố.
- Tăng hiệu quả trong phân tích dữ liệu và đưa ra quyết định dựa trên dữ liệu.
- Là nền tảng cho các hệ thống web, ứng dụng AI, và phân tích dữ liệu lớn.

1.3 Nguyên tắc ACID trong quản lý cơ sở dữ liệu

Hệ quản trị cơ sở dữ liệu (DBMS) hiện đại tuân thủ nguyên tắc **ACID** để đảm bảo độ tin cậy và toàn vẹn dữ liệu trong các giao dịch:

- **Atomicity (Tính nguyên tử)**: Mỗi giao dịch là một đơn vị không thể chia nhỏ. Hoặc tất cả các thao tác trong giao dịch đều được thực hiện, hoặc không thao tác nào được thực hiện.
- **Consistency (Tính nhất quán)**: Giao dịch đưa cơ sở dữ liệu từ trạng thái hợp lệ này sang trạng thái hợp lệ khác, tuân thủ tất cả các ràng buộc dữ liệu.
- **Isolation (Tính độc lập)**: Các giao dịch xảy ra đồng thời không ảnh hưởng đến nhau. Hệ thống đảm bảo kết quả giống như khi các giao dịch thực hiện tuần tự.
- **Durability (Tính bền vững)**: Khi một giao dịch được xác nhận (committed), thay đổi của nó được lưu vĩnh viễn, kể cả khi hệ thống bị sự cố.

2 Chuẩn hoá Cơ sở Dữ liệu: 1NF, 2NF và 3NF

2.1 Mục tiêu chuẩn hoá

Chuẩn hoá (Normalization) là quá trình tổ chức dữ liệu trong cơ sở dữ liệu nhằm:

- Loại bỏ dư thừa dữ liệu.
- Tránh các vấn đề về cập nhật (update anomalies).
- Cải thiện tính nhất quán và toàn vẹn dữ liệu.

2.2 Dạng chuẩn thứ nhất (First Normal Form - 1NF)

Định nghĩa: Một bảng được xem là đạt chuẩn 1NF nếu:

- Mỗi ô trong bảng chỉ chứa một giá trị đơn (atomic value).
- Không có nhóm lặp hay danh sách giá trị trong cùng một ô.

Ví dụ chưa đạt 1NF:

CustomerID	Name	Phones
1	John Doe	555-1234, 555-5678

Sau khi chuẩn hoá 1NF:

CustomerID	Name	Phone
1	John Doe	555-1234
1	John Doe	555-5678

2.3 Dạng chuẩn thứ hai (Second Normal Form - 2NF)

Điều kiện: Đạt chuẩn 1NF và mọi thuộc tính không khoá đều phải phụ thuộc hoàn toàn vào khoá chính (no partial dependency).

Giải thích: Nếu khoá chính là tổ hợp của nhiều cột, thì các cột khác phải phụ thuộc vào toàn bộ khoá, không phụ thuộc một phần.

Ví dụ chưa đạt 2NF:

StudentID	CourseID	StudentName	CourseName
1	CS101	Alice	Computer Science

Ở đây, `StudentName` chỉ phụ thuộc vào `StudentID`, còn `CourseName` chỉ phụ thuộc vào `CourseID` → vi phạm 2NF.

Sau chuẩn hoá 2NF:

- Bảng 1: `Student(StudentID, StudentName)`
- Bảng 2: `Course(CourseID, CourseName)`
- Bảng 3: `Enrollment(StudentID, CourseID)`

2.4 Dạng chuẩn thứ ba (Third Normal Form - 3NF)

Điều kiện: Đạt chuẩn 2NF và không có phụ thuộc bắc cầu (transitive dependency).

Giải thích: Một thuộc tính không khoá không nên phụ thuộc vào một thuộc tính không khoá khác.

Ví dụ vi phạm 3NF:

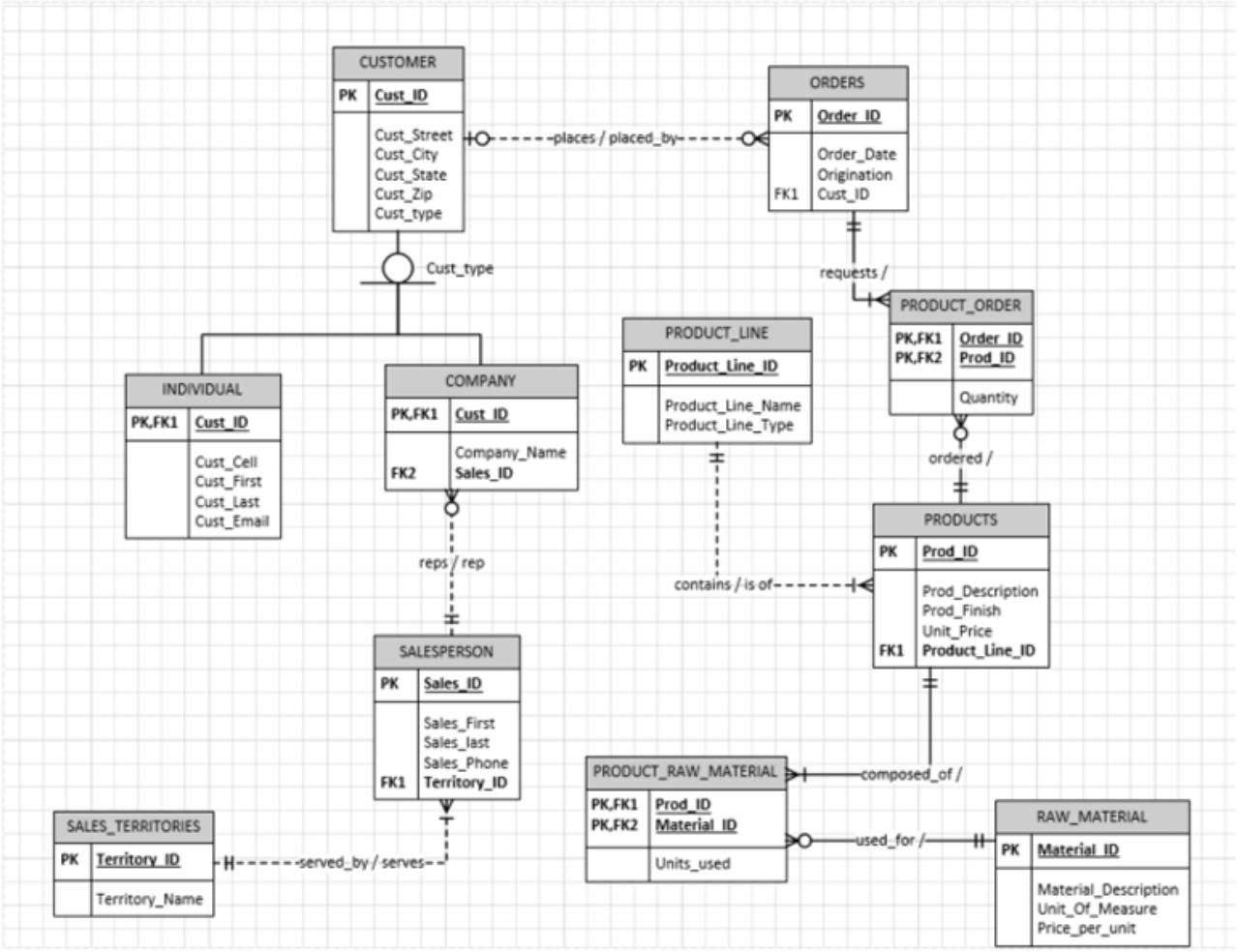
EmpID	EmpName	DeptID	DeptName
1	Bob	D01	Sales

Ở đây, `DeptName` phụ thuộc vào `DeptID`, mà `DeptID` lại phụ thuộc vào `EmpID` → vi phạm 3NF.

Sau khi chuẩn hoá 3NF:

- Bảng 1: `Employee(EmpID, EmpName, DeptID)`
- Bảng 2: `Department(DeptID, DeptName)`

2.5 Ví dụ trực quan của cơ sở dữ liệu quan hệ thỏa 3NF



Hình 1: Cơ sở Dữ Liệu của Cơ sở Kinh Doanh

3 Tổng quan SQL

3.1 Cài đặt MySQL trên macOS

1. Truy cập trang web: https://www.mysql.com.
2. Nhấn vào tab **DOWNLOADS**.
3. Kéo xuống cuối trang và nhấn vào **MySQL Community (GPL) Downloads »**.
4. Chọn **MySQL Community Server** và tải về link đầu tiên.
5. Mở tệp vừa tải về.
   - Nếu hệ điều hành macOS chặn mở tệp, vào phần **System Settings**.
   - Tại tab **Privacy & Security**, cho phép mở ứng dụng từ nhà phát triển không xác định.
6. Mở lại tệp tải về và thực hiện theo các bước cài đặt.

7. Khi được yêu cầu, tạo mật khẩu cho người dùng `root`.
8. Quay lại trang tải về, kéo xuống và nhấn vào **MySQL Workbench**.
9. Tải về link đầu tiên và mở tệp.
10. Kéo biểu tượng ứng dụng vào **Applications**, sau đó mở **MySQL Workbench**.

3.2 Cài đặt MySQL trên Windows

1. Truy cập trang web: https://www.mysql.com.
2. Nhấn vào tab **DOWNLOADS**.
3. Kéo xuống cuối trang và nhấn vào **MySQL Community (GPL) Downloads »**.
4. Chọn **MySQL Community Server**, nhấn **Go to Download Page**, và tải về link đầu tiên được đề xuất.
5. Mở tệp tải về và tiến hành cài đặt:
   - Làm theo từng bước hướng dẫn cài đặt.
   - Khi được yêu cầu, tạo mật khẩu cho người dùng `root`.
6. Tiếp tục nhấn **Next** hoặc **Execute** theo thiết lập mặc định cho đến khi hoàn tất.

4 Tạo và Quản lý Cơ sở Dữ liệu trong MySQL

4.1 Tổng quan các bước

Để lưu trữ thông tin khách hàng và sản phẩm trong cơ sở dữ liệu MySQL, ta thực hiện theo trình tự các bước sau:

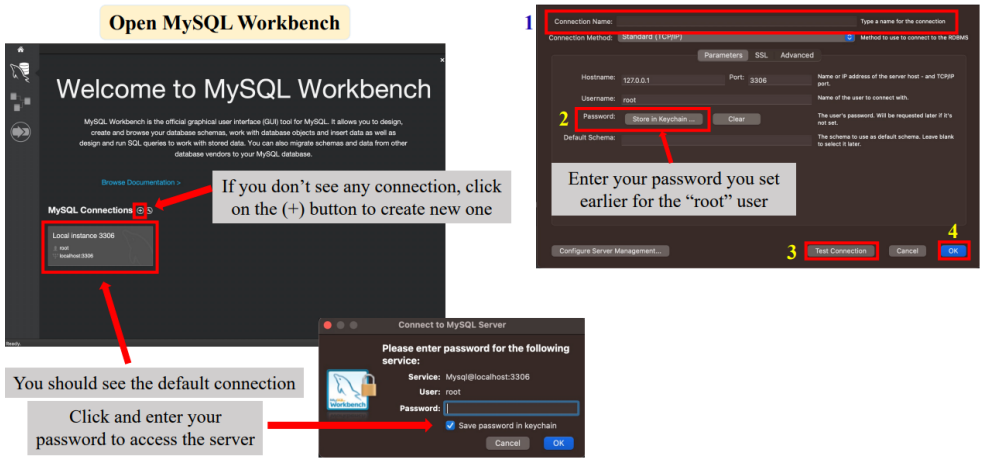
1. **Thiết kế sơ đồ cơ sở dữ liệu (Database Schema)**: Xác định các bảng và mối quan hệ giữa các bảng.
2. **Tạo bảng (Create Table)**: Sử dụng câu lệnh `CREATE TABLE` để xây dựng cấu trúc dữ liệu.
3. **Chèn dữ liệu (Insert Data)**: Dùng `INSERT INTO` để thêm dữ liệu vào bảng.
4. **Truy vấn dữ liệu (Retrieve Data)**: Dùng `SELECT` để đọc dữ liệu từ bảng.
5. **Cập nhật hoặc xoá dữ liệu (Update/Delete)**: Sử dụng `UPDATE` và `DELETE` để thay đổi dữ liệu.

4.2 Ví dụ mô tả tổng quan các bước tạo và quản lý cơ sở dữ liệu thông tin khách hàng

CustomerID	FirstName	LastName	Email	Phone
1	John	Doe	john.doe@example.com	555-1234
2	Jane	Smith	jane.smith@example.com	555-5678
3	Alice	Johnson	alice.johnson@example.com	555-8765

4.2.1 Mở MySQL Workbench

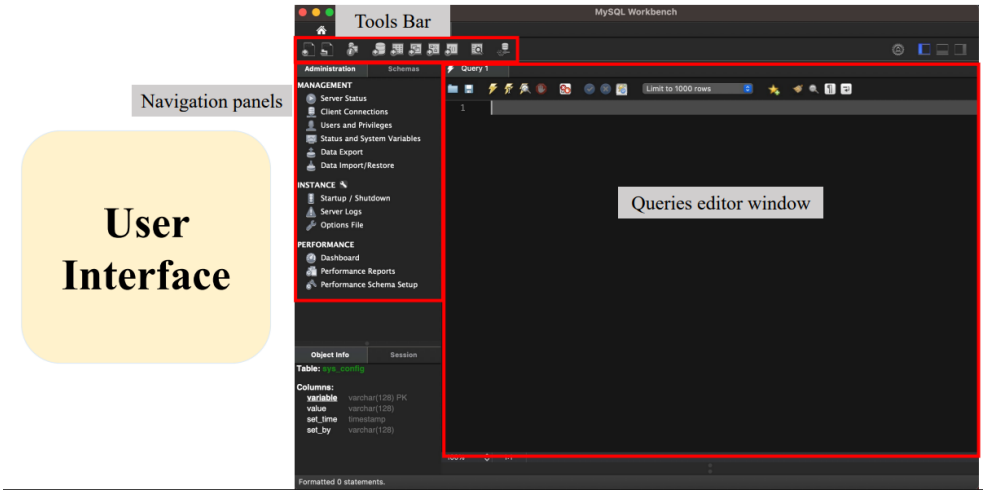
1. Mở ứng dụng **MySQL Workbench**.
2. Nếu không thấy kết nối mặc định, nhấn vào nút dấu + để tạo kết nối mới.
3. Nhập mật khẩu `root` đã cài đặt trước đó.
4. Nhấn vào kết nối để truy cập máy chủ cơ sở dữ liệu.



Hình 2: Mô tả quy trình mở MySQL Workbench

4.2.2 Giao diện người dùng Workbench

- **Tools Bar**: Thanh công cụ chứa các lệnh chính.
- **Navigation Panels**: Các bảng điều hướng.
- **Query Editor Window**: Cửa sổ soạn và thực thi lệnh SQL.



Hình 3: Giao diện người dùng Workbench

4.2.3 Tạo cơ sở dữ liệu và bảng

Ví dụ: Tạo cơ sở dữ liệu ShoppingDB và bảng Customers

```
CREATE DATABASE ShoppingDB;
USE ShoppingDB;

CREATE TABLE Customers (
    CustomerID INT PRIMARY KEY,
    FirstName VARCHAR(50),
    LastName VARCHAR(50),
    Email VARCHAR(100),
    Phone VARCHAR(20)
);
```

4.2.4 Chèn và truy vấn dữ liệu

Chèn dữ liệu:

```
INSERT INTO Customers VALUES
    (1, 'John', 'Doe', 'john.doe@example.com', '555-1234');
```

Truy vấn dữ liệu:

```
SELECT * FROM Customers;
```

Lưu ý quan trọng khi sử dụng INSERT:

- Dữ liệu chèn vào bảng cần tuân thủ **ràng buộc khóa chính (Primary Key)**:
  - Không được trùng lặp.
  - Không được để trống (NULL).
- Nếu bảng có **khóa ngoại (Foreign Key)**, dữ liệu phải được tham chiếu đúng đến bảng cha:
  - Phải chèn dữ liệu vào bảng cha trước.
  - Không được chèn giá trị không tồn tại trong bảng cha.

Ví dụ lỗi thường gặp:

```
INSERT INTO Orders(OrderID, CustomerID)
VALUES (101, 999); -- ỗLi vì CustomerID 999 ưcha ồtn ạti trong ảbng Customers
```

Cách làm đúng:

```
INSERT INTO Customers VALUES (999, 'Alice', 'Nguyen', 'alice@example.com', '
   555-7890');
INSERT INTO Orders VALUES (101, 999, '2024-06-01');
```

4.3 Tạo bảng sản phẩm (Products)

Dữ liệu mẫu:

ProductID	ProductName	Price	Stock
1	Laptop	999.99	50
2	Smartphone	499.99	100
3	Tablet	299.99	200

Tạo bảng và thêm dữ liệu:

```sql
CREATE TABLE Products (
    ProductID INT PRIMARY KEY,
    ProductName VARCHAR(100),
    Price DECIMAL(10,2),
    Stock INT
);

INSERT INTO Products VALUES (1, 'Laptop', 999.99, 50);
```

4.3.1 Thiết lập quan hệ đặt hàng

Khách hàng có thể đặt nhiều sản phẩm. Ta thiết kế các bảng bổ sung như sau:

- **Orders**: Lưu thông tin đơn hàng.
- **OrderItems**: Lưu chi tiết sản phẩm trong từng đơn hàng.

Ví dụ dữ liệu bảng OrderItems:

OrderItemID	OrderID	ProductID	Quantity
1	1	1	1
2	1	3	2
3	2	2	1

4.3.2 Tạo bảng và thêm dữ liệu đơn hàng

Tạo bảng:

- `CREATE TABLE Orders (`

```sql
CREATE TABLE Orders (
    OrderID INT PRIMARY KEY,
    CustomerID INT,
    OrderDate DATE,
    FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID)
);

```

- `CREATE TABLE OrderItems (`

```sql
CREATE TABLE OrderItems (
    OrderItemID INT PRIMARY KEY,
    OrderID INT,
    ProductID INT,
    Quantity INT,
    FOREIGN KEY (OrderID) REFERENCES Orders(OrderID),
    FOREIGN KEY (ProductID) REFERENCES Products(ProductID)
);

```

Lưu ý: Cần chèn dữ liệu theo thứ tự hợp lý để tránh lỗi:

- Đầu tiên, chèn dữ liệu vào bảng `Customers` và `Products`.
- Tiếp theo, chèn dữ liệu vào bảng `Orders`.
- Cuối cùng, chèn dữ liệu vào bảng `OrderItems`.

4.3.3 Sử dụng script có sẵn

1. Tải tập tin SQL mẫu từ nguồn cung cấp.
2. Mở tập tin trong MySQL Workbench.
3. Nhấn biểu tượng tia sét để thực thi toàn bộ script.
4. Vào tab **Schemas** trong bảng điều hướng, sau đó nhấn biểu tượng làm mới để hiển thị cơ sở dữ liệu vừa tạo.

5 Truy vấn dữ liệu với SQL

SQL (Structured Query Language) là ngôn ngữ truy vấn và thao tác dữ liệu trong hệ quản trị cơ sở dữ liệu quan hệ (DBMS). Phần này trình bày các lệnh SQL cơ bản phục vụ cho việc truy xuất và xử lý dữ liệu.

5.1 Lệnh SELECT – Truy vấn cơ bản

- **Chọn tất cả dữ liệu từ bảng:**

```sql
SELECT * FROM customers;

```

Lưu ý: Chỉ nên dùng `SELECT *` khi cần lấy toàn bộ cột. Nếu không, hãy chỉ định cụ thể các cột cần lấy để tối ưu hiệu suất.

- **Chọn cột cụ thể:**

```sql
SELECT first_name, last_name FROM customers;

```

Lưu ý: Việc chỉ định rõ các cột giúp kết quả dễ đọc và truy vấn nhanh hơn.

- **Chọn và tính toán giá trị:**

```sql
SELECT points, points * 1.1 AS VAT FROM customers;

```

Lưu ý: Có thể thực hiện phép toán trực tiếp trên các cột.

- **Sử dụng bí danh (alias):**

```sql
SELECT product_name AS Name FROM products;

```

Lưu ý: Dễ dàng đổi tên cột hiển thị trong kết quả, dùng từ khóa `AS` để rõ ràng.

- **Loại bỏ các giá trị trùng lặp:**

```sql
SELECT DISTINCT state FROM customers;

```

Lưu ý: Giúp lấy danh sách các giá trị duy nhất, tránh trùng lặp.

5.2 WHERE – Điều kiện lọc dữ liệu

- **So sánh:**

```
SELECT * FROM customers WHERE points > 3000;
SELECT * FROM customers WHERE state = 'VA';
SELECT * FROM customers WHERE state != 'VA';

```

Lưu ý: Luôn dùng dấu nháy đơn `'` cho dữ liệu kiểu chuỗi (VARCHAR, TEXT).

- **Toán tử logic AND, OR, NOT:**

```
SELECT * FROM customers WHERE state = 'VA' AND points > 1000;
SELECT * FROM customers WHERE NOT state = 'VA';

```

Lưu ý: Dùng dấu ngoặc `()` khi kết hợp nhiều điều kiện để tránh nhầm lẫn.

- **So sánh biểu thức:**

```
SELECT *, unit_price * quantity AS total
FROM order_items
WHERE order_id = 6 AND unit_price * quantity < 30;

```

Lưu ý: Có thể sử dụng phép tính trong mệnh đề `WHERE`.

5.3 IN – BETWEEN – Truy vấn tập hợp giá trị

- **Dùng IN:**

```
SELECT * FROM customers
WHERE state IN ('VA', 'GA', 'FL');

```

Lưu ý: Tương đương nhiều `OR`, giúp câu truy vấn gọn hơn khi kiểm tra nhiều giá trị.

- **Dùng BETWEEN:**

```
SELECT * FROM customers
WHERE points BETWEEN 300 AND 2000;

```

Lưu ý: `BETWEEN` bao gồm cả giá trị đầu và cuối (300 và 2000).

5.4 IS NULL – ORDER BY – LIMIT

- **Lọc giá trị NULL:**

```
SELECT * FROM customers WHERE phone IS NULL;
SELECT * FROM customers WHERE phone IS NOT NULL;

```

Lưu ý: Dùng `IS NULL` thay vì `= NULL` để kiểm tra NULL.

- **Sắp xếp dữ liệu:**

```
SELECT * FROM customers ORDER BY points;
SELECT * FROM customers WHERE points < 1000 ORDER BY points DESC;

```

Lưu ý: Mặc định là `ASC` (tăng dần). Dùng `DESC` để giảm dần.

- **Giới hạn số dòng kết quả:**

```
SELECT * FROM customers ORDER BY points DESC LIMIT 3;
SELECT * FROM customers ORDER BY points DESC LIMIT 3, 4;

```

Lưu ý: `LIMIT offset, count` – bỏ qua `offset` dòng đầu và lấy `count` dòng tiếp theo.

5.5 LIKE – REGEXP – Truy vấn theo mẫu chuỗi

5.5.1 LIKE – ký tự đại diện:

```
SELECT * FROM customers WHERE last_name LIKE 'B%';
SELECT * FROM customers WHERE last_name LIKE '%b%';
SELECT * FROM customers WHERE last_name LIKE '_____y';
SELECT * FROM customers WHERE address LIKE '%trail%' OR address LIKE '%avenue%';
SELECT * FROM customers WHERE phone LIKE '%9___';
```

Lưu ý:

- `%` đại diện cho chuỗi ký tự bất kỳ (0 hoặc nhiều ký tự).
- `_` đại diện cho đúng 1 ký tự.
- LIKE phân biệt chữ hoa/thường tùy hệ quản trị (MySQL mặc định không phân biệt).

5.5.2 REGEXP – biểu thức chính quy:

```
SELECT * FROM customers WHERE first_name REGEXP 'ELKA|AMBUR';
SELECT * FROM customers WHERE last_name REGEXP 'EY$|ON$';
SELECT * FROM customers WHERE last_name REGEXP '^MY|SE';
SELECT * FROM customers WHERE last_name REGEXP 'B[RU]';
```

REGEX SYNTAX	MEANING	EXAMPLE	MATCHES	DOES NOT MATCH
.	Any single character	go.gle	google, goggle	gogle
[abc]	Any of these character	analy[zs]e	analyse, analyze	analyxe
[a-z]	Any character in this range	demo[2-4]	demo2, demo3	demo1, demo5
[^abc]	None of these characters	analy[^zs]e	analyxe	analyse, analyze
[^a-z]	Not a character in this range	demo[^2-4]	demo1, demo5	demo2, demo3
|	Or	demo|example	demo, demos, example	test
^	Starts with	^demo	demos, demonstration	my demo
$	Ends with	demo$	my demo	demonstration
?	Zero or one times (greedy)	demos?123	demo123, demos123	demoA123
??	Zero or one times (lazy)			
*	Zero or more times (greedy)	goo*gle	gogle, goooogle	goggle
*?	Zero or more times (lazy)			
+	One or more times (greedy)	goo+gle	google, goooogle	gogle, goggle
+?	One or more times (lazy)			
{n}	n times exactly	w{3}	www	w, ww
{n,m}	from n to m times	a{4, 7}	aaaa, aaaaa, aaaaaa, aaaaaaa	aaaaaaaa, aaa, a
{n,}	at least n times	go{2,}gle	google, goooogle, goooogle	ggle, gogle
()	Group	^(demo|example)[0-9]+	demo1, example4	demoexample2
(?:)	Passive group (Useful for filters)			
\	Escape	AU\$10	AU$10, AU$100	AU10, 10
\s	White space			
\S	Non-white space			
\d	Digit character			
\D	Non-digit character			
\w	Word			
\W	Non-word (e.g. punctuation, spaces)			

Hình 4: Các syntax cơ bản của REGEX