BÀI 7:
Thao tác dữ liệu với MySQL

Mục tiêu bài học

- Các kiểu dữ liệu trong MySQL
- Giới thiệu các lệnh join liên kết dữ liệu giữa 2 hay nhiều bảng
- Chuẩn hóa CSDL

SQL Data Types

- Các cột trong bảng sẽ chứa dữ liệu thuộc một kiểu dữ liệu. Một kiểu dữ liệu sẽ định nghĩa loại dữ liệu mà cột đó có thể chứa: dữ liệu số nguyên, số thức, ký tự, tiền tệ, ngày tháng, nhị phân v.v…
- Khi tạo bảng, người dùng cần quyết định xem loại dữ liệu mà từng cột trong bảng chứa sẽ là loại nào.
- Tùy từng cột mà người dùng sẽ quyết định kiểu dữ liệu, cũng như kích thước dữ liệu của cột đó
  - Chú ý: Kiểu dữ liệu có thể sẽ khác nhau, tùy thuộc vào từng loại CSDL.

MySQL Data Types

- Trong CSDL MySQL có nhiều kiểu dữ liệu, nhưng được phân thành 3 nhóm chính: text, number, và date.
- Các kiểu dữ liệu text:

Data type	Description
CHAR(size)	Holds a fixed length string (can contain letters, numbers, and special characters). The fixed size is specified in parenthesis. Can store up to 255 characters
VARCHAR(size)	Holds a variable length string (can contain letters, numbers, and special characters). The maximum size is specified in parenthesis. Can store up to 255 characters. **Note:** If you put a greater value than 255 it will be converted to a TEXT type
TINYTEXT	Holds a string with a maximum length of 255 characters
TEXT	Holds a string with a maximum length of 65,535 characters
BLOB	For BLOBs (Binary Large OBjects). Holds up to 65,535 bytes of data
MEDIUMTEXT	Holds a string with a maximum length of 16,777,215 characters
MEDIUMBLOB	For BLOBs (Binary Large OBjects). Holds up to 16,777,215 bytes of data
LONGTEXT	Holds a string with a maximum length of 4,294,967,295 characters
LONGBLOB	For BLOBs (Binary Large OBjects). Holds up to 4,294,967,295 bytes of data
ENUM(x,y,z,etc.)	Let you enter a list of possible values. You can list up to 65535 values in an ENUM list. If a value is inserted that is not in the list, a blank value will be inserted. **Note:** The values are sorted in the order you enter them. You enter the possible values in this format: ENUM('X','Y','Z')
SET	Similar to ENUM except that SET may contain up to 64 list items and can store more than one choice

MySQL Data Types

- Các kiểu dữ liệu number

Data type	Description
TINYINT(size)	-128 to 127 normal. 0 to 255 UNSIGNED*. The maximum number of digits may be specified in parenthesis
SMALLINT(size)	-32768 to 32767 normal. 0 to 65535 UNSIGNED*. The maximum number of digits may be specified in parenthesis
MEDIUMINT(size)	-8388608 to 8388607 normal. 0 to 16777215 UNSIGNED*. The maximum number of digits may be specified in parenthesis
INT(size)	-2147483648 to 2147483647 normal. 0 to 4294967295 UNSIGNED*. The maximum number of digits may be specified in parenthesis
BIGINT(size)	-9223372036854775808 to 9223372036854775807 normal. 0 to 18446744073709551615 UNSIGNED*. The maximum number of digits may be specified in parenthesis
FLOAT(size,d)	A small number with a floating decimal point. The maximum number of digits may be specified in the size parameter. The maximum number of digits to the right of the decimal point is specified in the d parameter
DOUBLE(size,d)	A large number with a floating decimal point. The maximum number of digits may be specified in the size parameter. The maximum number of digits to the right of the decimal point is specified in the d parameter
DECIMAL(size,d)	A DOUBLE stored as a string , allowing for a fixed decimal point. The maximum number of digits may be specified in the size parameter. The maximum number of digits to the right of the decimal point is specified in the d parameter

MySQL Data Types

- Các kiểu dữ liệu date

Data type	Description
DATE()	A date. Format: YYYY-MM-DD **Note:** The supported range is from '1000-01-01' to '9999-12-31'
DATETIME()	*A date and time combination. Format: YYYY-MM-DD HH:MI:SS **Note:** The supported range is from '1000-01-01 00:00:00' to '9999-12-31 23:59:59'
TIMESTAMP()	*A timestamp. TIMESTAMP values are stored as the number of seconds since the Unix epoch ('1970-01-01 00:00:00' UTC). Format: YYYY-MM-DD HH:MI:SS **Note:** The supported range is from '1970-01-01 00:00:01' UTC to '2038-01-09 03:14:07' UTC
TIME()	A time. Format: HH:MI:SS **Note:** The supported range is from '-838:59:59' to '838:59:59'
YEAR()	A year in two-digit or four-digit format. **Note:** Values allowed in four-digit format: 1901 to 2155. Values allowed in two-digit format: 70 to 69, representing years from 1970 to 2069

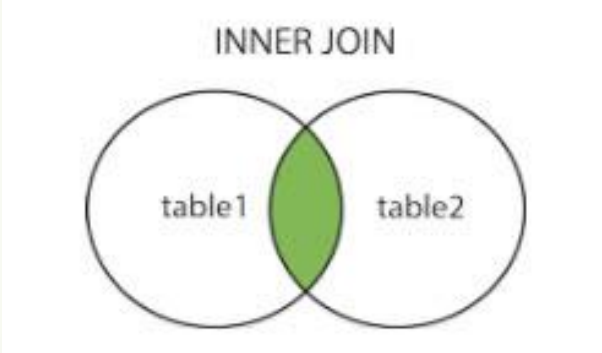
SQL JOIN

- Một mệnh đề JOIN được dùng để kết hợp và truy vấn dữ liệu từ hai hay nhiều bảng, dựa trên một cột chung giữa chúng.
- Một số loại lệnh SQL JOINs
  - **(INNER) JOIN**: Trả về các bản ghi có giá trị khớp trong cả 2 bảng.
  - **LEFT (OUTER) JOIN**: Trả về tất cả các bản ghi ở bảng bên trái, và những bản ghi thỏa mãn điều kiện (có giá trị khớp) trong bảng bên phải.
  - **RIGHT (OUTER) JOIN**: Trả về tất cả các bản ghi ở bảng bên phải, và những bản ghi thỏa mãn điều kiện (có giá trị khớp) trong bảng bên trái.
  - **FULL (OUTER) JOIN**: Trả về tất cả những bản ghi thỏa mãn điều kiện ở một trong hai bảng bên trái hoặc bên phải.

SQL INNER JOIN

- Mệnh đề INNER JOIN sẽ lựa chọn những bản ghi có giá trị khớp trong cả 2 bảng.
- Cú pháp:

```
SELECT column_name(s)
FROM table1
INNER JOIN table2 ON table1.column_name = table2.column_name;
```



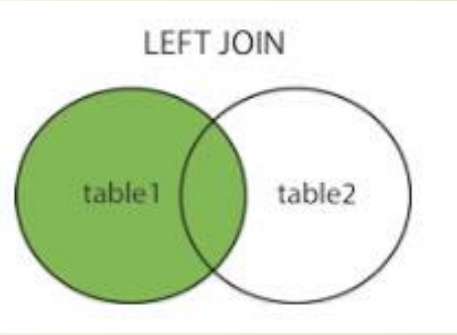
- Ví dụ:

```
SELECT Orders.OrderID, Customers.CustomerName
FROM Orders
INNER JOIN Customers ON Orders.CustomerID = Customers.CustomerID;
```

SQL LEFT JOIN

- Mệnh đề LEFT JOIN sẽ trả về tất cả các bản ghi ở bảng bên trái (table1), và những bản ghi thỏa mãn điều kiện ở bảng bên phải (table2). Nếu những dòng nào không khớp ở bảng bên phải, kết quả trả về sẽ là NULL.
- Cú pháp:

```
SELECT column_name(s)
FROM table1
LEFT JOIN table2 ON table1.column_name = table2.column_name;
```



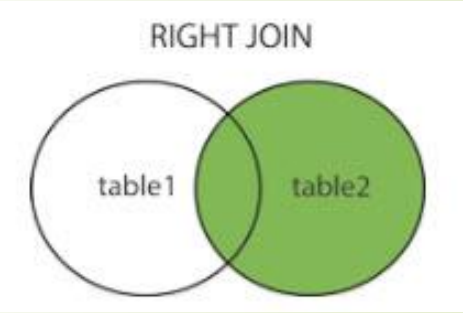
- Ví dụ:

```
SELECT Customers.CustomerName, Orders.OrderID
FROM Customers
LEFT JOIN Orders ON Customers.CustomerID = Orders.CustomerID
ORDER BY Customers.CustomerName;
```

SQL RIGHT JOIN

- Mệnh đề RIGHT JOIN sẽ trả về tất cả các bản ghi ở bảng bên phải (table2), và những bản ghi thỏa mãn điều kiện ở bảng bên trái (table1). Nếu những dòng nào không khớp ở bảng bên trái, kết quả trả về sẽ là NULL.
- Cú pháp:

```
SELECT column_name(s)
FROM table1
RIGHT JOIN table2 ON table1.column_name = table2.column_name;
```



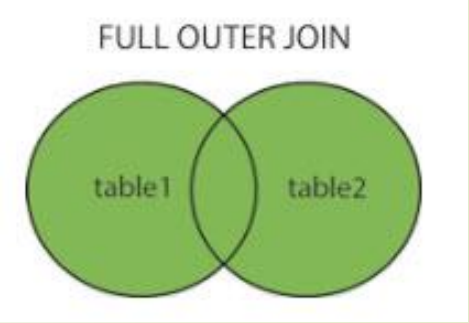
- Ví dụ:

```
SELECT Orders.OrderID, Employees.LastName, Employees.FirstName
FROM Orders
RIGHT JOIN Employees ON Orders.EmployeeID = Employees.EmployeeID
ORDER BY Orders.OrderID;
```

FULL OUTER JOIN

- Mệnh đề FULL OUTER JOIN trả về tất cả những bản ghi mà chỉ cần khớp với một trong hai bảng bên trái (table1) hoặc bên phải (table2).
  - **Chú ý:** Việc sử dụng mệnh đề FULL OUTER JOIN có thể trả về một tập bản ghi rất lớn !
- Cú pháp:

```
SELECT column_name(s)
FROM table1
FULL OUTER JOIN table2 ON table1.column_name = table2.column_name;
```



- Ví dụ:

```
SELECT Customers.CustomerName, Orders.OrderID
FROM Customers
FULL OUTER JOIN Orders ON Customers.CustomerID=Orders.CustomerID
ORDER BY Customers.CustomerName;
```

Demo

- Giáo viên demo việc sử dụng câu lệnh join cho sinh viên.

Dư thừa dữ liệu (Data Redundancy)

- Dư thừa (Redundancy):
  - Làm tăng thời gian cập nhật (thêm sửa xóa) dữ liệu trong bảng.
  - Làm tăng dung lượng đĩa cứng để lưu trữ dữ liệu, do đó sẽ làm tốn tài nguyên.
- Dư thừa có thể dẫn đến những bất lợi sau:
  - Thêm sửa xóa dữ liệu, có thể dẫn đến tình trạng không nhất quán về dữ liệu.
  - Có thể có những lỗi xảy ra khi dữ liệu bị dư thừa hoặc trùng lặp.
  - Tốn kém dung lượng đĩa cứng để lưu trữ dữ liệu thừa.
  - Làm tốn thời gian truy vấn dữ liệu trong bảng.

Dư thừa dữ liệu (Data Redundancy)

- Bảng STUDENT chứa dữ liệu như hình bên dưới:

STUDENT ID	STUDENT NAME	....	STUDENT SEMESTER	STUDENT TEST1	STUDENT TEST2
001	Mary	....	SEM-1	40	65
001	Mary	....	SEM-2	56	48
002	Jake	....	SEM-1	93	84
002	Jake	....	SEM-2	85	90

Trong bảng Student, các thông tin của sinh viên như STUDENTID và STUDENTNAME bị lặp lại khi lưu lại điểm trong các học kỳ khác nhau.

Chuẩn hóa dữ liệu

- Chuẩn hóa (Normalization):
  - Là quá trình tách các bảng có cấu trúc phức tạp trở thành những bảng có cấu trúc đơn giản hơn bằng cách sử dụng một số quy luật nhất định.
  - Lợi ích của quá trình chuẩn hóa:
    - Giúp duy trì tính toàn vẹn của dữ liệu.
    - Giúp làm đơn giản hóa cấu trúc bảng, nhờ đó giúp làm cho CSDL trở nên tối ưu hơn.
    - Giúp loại bỏ và hạn chế các giá trị null, vì vậy làm giảm sự phức tạo của các thao tác dữ liệu.

Chuẩn hóa dữ liệu

- Một số quy tắc cần tuân thủ để có được một CSDL tốt:
  - Mỗi bảng cần có một khóa chính.
  - Mỗi bảng nên lưu trữ dữ liệu về một kiểu thực thể đơn..
  - Cần tránh các cột chấp nhận giá trị NULL.
  - Cần tránh lưu trữ các giá trị trùng lặp.

Chuẩn hóa dữ liệu

- Quá trình chuẩn hóa sẽ dẫn đến việc cấu trúc các bảng trong CSDL thỏa mãn một số dạng nhất định.
- Những dạng chuẩn này sẽ giúp loại bỏ đi những bất thường và không nhất quán về dữ liệu trong các bảng thuộc CSDL.
- Dưới đây là một số dạng chuẩn phổ biến:
  - Dạng chuẩn 1 (1NF)
  - Dạng chuẩn 2 (2NF)
  - Dạng chuẩn 3 (3NF)
  - Dạng chuẩn Boyce-Codd (BCNF)

Chuẩn hóa dữ liệu

- Dạng chuẩn 1 (1NF):
  - Một bảng được coi là đạt dạng chuẩn 1 khi mỗi ô trong bảng chỉ chứa một giá trị nguyên tố.
  - Một số hướng dẫn để chuyển đổi một bảng thành dạng chuẩn 1:
    - Đặt những giá trị dữ liệu có liên quan đến nhau vào trong một bảng.
    - Không đặt những nhóm dữ liệu trùng lặp trong bảng.
    - Mỗi bảng sẽ phải có một khóa chính.

Chuẩn hóa dữ liệu

- Dạng chuẩn 2 (2NF):
  - Một bảng được coi là đạt dạng chuẩn 2NF khi:
    - Bảng đó đã đạt dạng chuẩn 1NF
    - Không tồn tại sự phụ thuộc một phần giữa các thuộc tính không khóa (non-key attributes) với các thuộc tính khóa (key attributes).
  - Một số hướng dẫn để chuyển một bảng thành dạng chuẩn 2NF:
    - Tìm và loại bỏ những thuộc tính chỉ phụ thuộc hàm một phần vào khóa, tách chúng ra một bảng khác.
    - Nhóm những thuộc tính còn lại.

Chuẩn hóa dữ liệu

- Dạng chuẩn 3 (3NF):
  - Một bảng được coi là đạt dạng chuẩn 3 nếu và chỉ nếu:
    - Nó đạt dạng chuẩn 2
    - Không có sự phụ thuộc bắc cầu giữa các thuộc tính không khóa (non-key attributes) và các thuộc tính khóa (key attributes).
  - Những hướng dẫn để chuyển một bảng thành dạng chuẩn 3NF:
    - Tìm và loại bỏ những thuộc tính không khóa (non-key attributes) phụ thuộc hàm vào những thuộc tính không phải là khóa chính. Đặt chúng vào một bảng khác.
    - Nhóm những thuộc tính còn lại.

Chuẩn hóa dữ liệu

- Dạng chuẩn Boyce-Codd (BCNF):
  - Ba dạng chuẩn đôi khi không đủ để chuẩn hóa toàn bộ CSDL trong một số tình huống. Ba dạng chuẩn sẽ không thỏa mãn đối với các bảng có những đặc điểm sau:
    - Có nhiều candidate keys.
    - Có nhiều candidate keys là tổ hợp
    - Có nhiều candidate keys chồng lắp nhau (có ít nhất một thuộc tính chung).
  - Một số gợi ý để chuyển đổi một bảng thành dạng chuẩn BCNF gồm:
    - Tìm và loại bỏ những candidate keys chồng lắp nhau. Đặt các candidate key và những thuộc tính mà phụ thuộc hàm vào khóa ra một bảng khác.
    - Gom những thuộc tính còn lại vào một bảng.

Tổng kết

- Các kiểu dữ liệu trong MySQL
- Giới thiệu các lệnh join liên kết dữ liệu giữa 2 hay nhiều bảng
- Chuẩn hóa CSDL