MySQL

1. Cơ sở dữ liệu MySQL

MySQL là một hệ thống quản trị cơ sở dữ liệu mã nguồn mở (gọi tắt là RDBMS) hoạt động theo mô hình client-server. Với RDBMS là viết tắt của Relational Database Management System. MySQL được tích hợp apache, PHP. MySQL quản lý dữ liệu thông qua các cơ sở dữ liệu. Mỗi cơ sở dữ liệu có thể có nhiều bảng quan hệ chứa dữ liệu. MySQL cũng có cùng một cách truy xuất và mã lệnh tương tự với ngôn ngữ SQL. MySQL được phát hành từ thập niên 90.

Quá trình hình thành và phát triển của MySQL được tóm tắt như sau:

* Công ty Thuy Điển MySQL AB phát triển MySQL vào năm 1994.
* Phiên bản đầu tiên của MySQL phát hành năm 1995
* Công ty Sun Microsystems mua lại MySQL AB trong năm 2008
* Năm 2010 tập đoàn Oracle thâu tóm Sun Microsystems. Ngay lúc đó, đội ngũ phát triển của MySQL tách MySQL ra thành 1 nhánh riêng gọi là MariaDB. Oracle tiếp tục phát triển MySQL lên phiên bản 5.5.
* 2013 MySQL phát hành phiên bản 5.6
* 2015 MySQL phát hành phiên bản 5.7
* MySQL đang được phát triển lên phiên bản 8.0

MySQL hiện nay có 2 phiên bản miễn phí (MySQL Community Server) và có phí (Enterprise Server).

1.1. Các kiểu dữ liệu trong MySQL

Việc xác định đúng kiểu dữ liệu của các trường trong một bảng rất quan trọng đối với việc tối ưu hóa cơ sở dữ liệu của bạn. Bạn chỉ nên sử dụng kiểu dữ liệu và kích cỡ của trường thực sự cần sử dụng. Ví dụ: không định nghĩa trường rộng 10 ký tự, nếu bạn biết rằng chỉ sử dụng 2 ký tự.

Kiểu dữ liệu trong MySQL có thể được chia thành 3 kiểu như sau:

* Numeric
* Date and Time
* Các kiểu chuỗi

1.1.1. Kiểu dữ liệu Numeric

MySQL sử dụng tất cả các loại dữ liệu số ANSI SQL tiêu chuẩn, vì vậy nếu bạn đã từng tiếp cận với một hệ thống cơ sở dữ liệu khác, kiểu dữ liệu numeric tương tự như vậy.

Danh sách sau đây cho thấy các loại dữ liệu số phổ biến và mô tả của chúng:

|  |  |
| --- | --- |
| **Kiểu dữ liệu** | **Mô tả** |
| INT | Một số nguyên với kích cỡ thông thường, có thể là signed hoặc unsigned. Nếu có dấu, thì dãy giá trị có thể là từ -2.147.483.648 tới 2.147.483.647, nếu không dấu thì dãy giá trị là từ 0 tới 4.294.967.295. Bạn có thể xác định một độ rộng lên tới 11 chữ số. |
| TINYINT | Một số nguyên với kích cỡ rất nhỏ, có thể là signed hoặc unsigned. Nếu có dấu, thì dãy giá trị có thể là từ -128 tới 127, nếu không dấu thì dãy giá trị là từ 0 tới 255. Bạn có thể xác định một độ rộng lên tới 4 chữ số. |
| MEDIUMINT | Một số nguyên với kích cỡ trung bình, có thể là signed hoặc unsigned. Nếu có dấu, thì dãy giá trị có thể là từ -8.388.608 tới 8.388.607, nếu không dấu thì dãy giá trị là từ 0 tới 16.777.215. Bạn có thể xác định một độ rộng lên tới 9 chữ số. |
| BIGINT | Một số nguyên với kích cỡ lớn, có thể là signed hoặc unsigned. Nếu có dấu, thì dãy giá trị có thể là từ -9.223.372.036.854.775.808 tới 9.223.372.036.854.775.807, nếu không dấu thì dãy giá trị là từ 0 tới 18.446.744.073.709.551.615. Bạn có thể xác định một độ rộng lên tới 20 chữ số. |
| FLOAT(M, D) | Một số thực dấu chấm động không dấu. Bạn có thể định nghĩa độ dài hiển thị (M) và số vị trí sau dấy phảy (D). Điều này là không bắt buộc và sẽ có mặc định là 10,2: với 2 là số vị trí sau dấu phảy và 10 là số chữ số (bao gồm các phần thập phân). Phần thập phân có thể lên tới 24 vị trí sau dấu phảy đối với một số FLOAT. |
| DOUBLE(M, D) | Một số thực dấu chấm động không dấu. Bạn có thể định nghĩa độ dài hiển thị (M) và số vị trí sau dấy phảy (D). Điều này là không bắt buộc và sẽ có mặc định là 16,4: với 4 là số vị trí sau dấu phảy và 16 là số chữ số (bao gồm các phần thập phân). Phần thập phân có thể lên tới 53 vị trí sau dấu phảy đối với một số DOUBLE. REAL là đồng nghĩa với DOUBLE. |
| DECIMAL(M, D) | Một kiểu khác của dấu chấm động không dấu. Mỗi chữ số thập phân chiếm 1 byte. Việc định nghĩa độ dài hiển thị (M) và số vị trí sau dấy phảy (D) là bắt buộc. NUMERIC là một từ đồng nghĩa cho DECIMAL. |

1.1.2. Kiểu dữ liệu Date và Time trong MySQL

|  |  |
| --- | --- |
| **Kiểu dữ liệu** | **Mô tả** |
| DATE | Một date trong định dạng YYYY-MM-DD, giữa 1000-01-01 và 9999-12-31. Ví dụ, ngày 25 tháng 12 năm 2015 sẽ được lưu ở dạng 2015-12-25. |
| DATETIME | Một tổ hợp Date và Time trong định dạng YYYY-MM-DD HH:MM:SS, giữa 1000-01-01 00:00:00 và 9999-12-31 23:59:59. Ví dụ, 3:30 chiều ngày 25 tháng 12, năm 2015 sẽ được lưu ở dạng 2015-12-25 15:30:00. |
| TIMESTAMP | Một Timestamp từ giữa nửa đêm ngày 1/1/1970 và 2037. Trông khá giống với định dạng DATETIME trước, khác biệt ở chỗ không có dấu gạch nối giữa các số. Ví dụ, 3:30 chiều ngày 25 tháng 12, năm 2015 sẽ được lưu dưới dạng 20151225153000 ( YYYYMMDDHHMMSS ). |
| TIME | Lưu time trong định dạng HH:MM:SS. |
| YEAR(M) | Lưu 1 năm trong định dạng 2 chữ số hoặc 4 chữ số. Nếu độ dài được xác định là 2 (ví dụ: YEAR(2)), YEAR có thể từ 1970 tới 2069 (70 tới 69). Nếu độ dài được xác định là 4, YEAR có thể từ 1901 tới 2155. Độ dài mặc định là 4. |

1.1.3. Kiểu dữ liệu chuỗi trong MySQL

Ngoài các kiểu dữ liệu số hoặc kiểu dữ liệu date và time, thì còn một kiểu dữ liệu mà bạn thường sử dụng nhất trong MySQL là kiểu dữ liệu chuỗi. Dưới đây liệt kê các kiểu dữ liệu chuỗi và phần miêu tả của chúng trong MySQL

|  |  |
| --- | --- |
| **Kiểu dữ liệu** | **Mô tả** |
| CHAR(M) | Một chuỗi có độ dài cố định có độ dài từ 1 tới 255 ký tự (ví dụ CHAR(5)). Nếu giá trị thật của một trường kiểu Char không bằng với độ dài khai báo thì phần thiếu bên phải của nó sẽ được thêm bằng các kí tự trắng một cách tự động. Định nghĩa độ dài là không bắt buộc, giá trị mặc định là 1. |
| VARCHAR(M) | Dữ liệu kiểu chuỗi có độ dài thay đổi, có độ dài từ 1 đến 255 kí tự (ví dụ Varchar(24)). Bạn phải định nghĩa độ dài khi tạo một trường VARCHAR. |
| BLOB hoặc TEXT | Trường kiểu này có độ dài tối đa 65535 kí tự. BLOBs là viết tắt của "Binary Large Objects", và được sử dụng để lưu trữ một lượng lớn dữ liệu nhị phân như các bức ảnh hoặc các loại tập tin khác. Với TEXT, trường cũng lưu trữ được một lượng lớn dữ liệu. Điểm khác nhau giữa chúng là: khi sắp xếp và so sánh dữ liệu đã lưu trữ thì với BLOBs là phân biệt kiểu chữ, còn với TEXT là không phân biệt kiểu chữ. Bạn không phải xác định độ dài với BLOBs hoặc TEXT. |
| TINYBLOB hoặc TINYTEXT | Một cột BLOB hoặc TEXT với độ dài tối đa là 255 ký tự. Bạn không cần xác định độ dài với TINYBLOB hoặc TINYTEXT. |
| MEDIUMBLOB hoặc MEDIUMTEXT | Một cột BLOB hoặc TEXT với độ dài tối đa là 16777215 ký tự. Bạn không cần xác định độ dài với MEDIUMBLOB hoặc MEDIUMTEXT. |
| LONGBLOB hoặc LONGTEXT | Một cột BLOB hoặc TEXT với độ dài tối đa là 4294967295 ký tự. Bạn không cần xác định độ dài với LONGBLOB hoặc LONGTEXT. |
| ENUM | Khi định nghĩa một trường kiểu này, tức là, ta đã chỉ ra một danh sách các đối tượng mà trường phải nhận (có thể là Null). Ví dụ, nếu ta muốn một trường nào đó chỉ nhận một trong các giá trị "A" hoặc "B" hoặc "C" thì ta phải định nghĩa kiểu ENUM cho nó như sau: ENUM ('A', 'B', 'C'). Và chỉ có các giá trị này (hoặc NULL) có thể xuất hiện trong trường đó. |

1.2. Các thuộc tính

2. Tạo và quản trị cơ sở dữ liệu SQL

2.1. Tạo Cơ sở dữ liệu (Database)

2.1.1. Tạo CSDL

Chúng ta có thể tạo CSDL bằng 2 cách:

**C1: Bằng lệnh, theo cú pháp**

**1) CREATE DATABASE** [database\_name];

*Ví dụ:* *tạo cơ sở dữ liệu sinhvien*

**CREATE DATABASE** sinhvien;

Đầy đủ hơn bằng cách chỉ định character set và collection cho phép bạn nhập kiểu ký tự vào bảng trong cơ sở dữ liệu, ví dụ bạn muốn nhập tiếng Việt có dấu thì character set bạn phải chỉ định là UTF-8. Theo cú pháp

**2) CREATE DATABASE** [database\_name] **character set** [CHAR\_SET] **collate** [COLLATION];

*Ví dụ:* *tạo cơ sở dữ liệu sinhvien có thể lữu trữ dữ liệu tiếng Việt có dấu:*

**CREATE DATABASE** sinhvien **character** **set** UTF8 **collate** utf8\_vietnamese\_ci;

MySQL hỗ trợ hai loại bộ ký tự UTF8: utf8 và utf8mb4; để sử dụng bộ ký tự utf8mb4 nên sử dụng phiên bản MySQL 5.5.3 trở lên.

Trong trường hợp CSDL muốn tạo đã tồn tại chúng ta nên dùng từ khóa IF NOT EXISTS đi theo cú pháp:

**3) CREATE DATABASE IF NOT EXISTS** [database\_name];

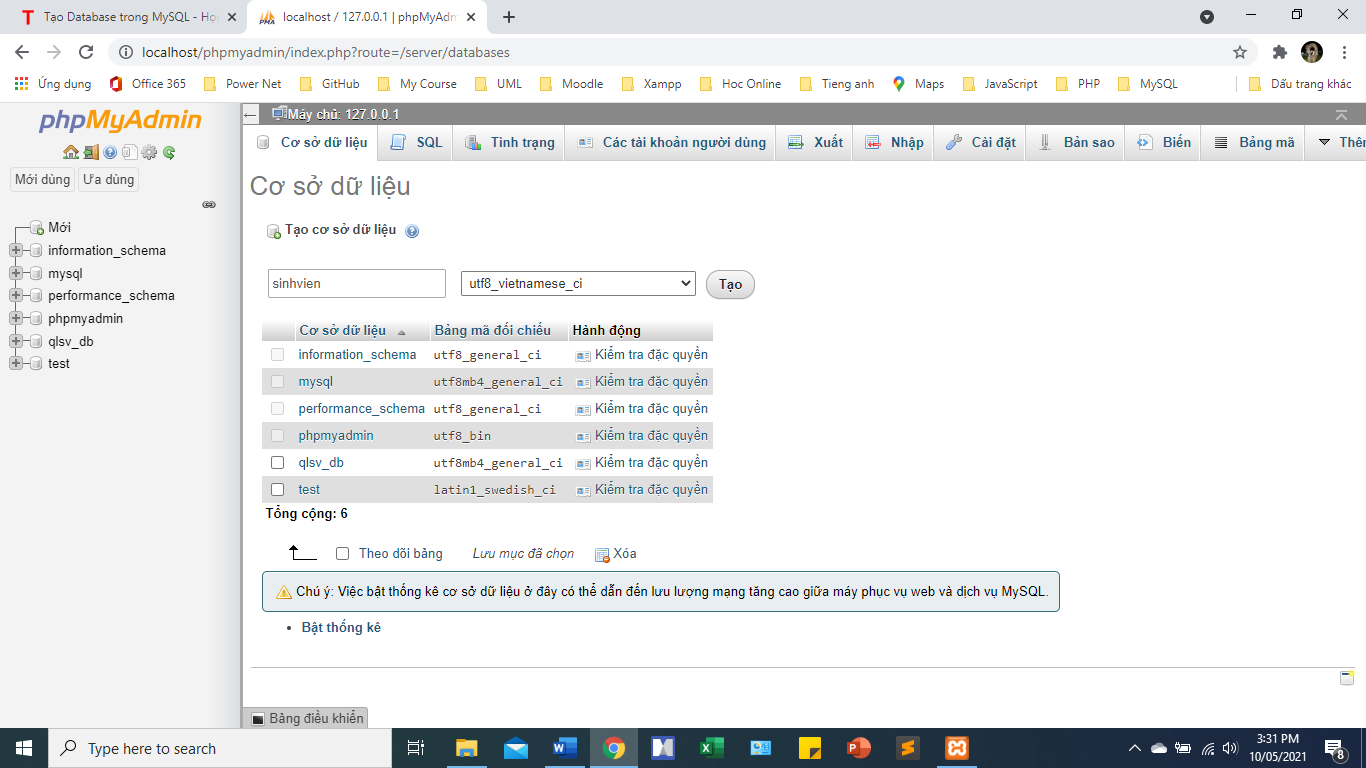
*Ví dụ*

CREATE DATABASE nhanvien;

CREATE DATABASE IF NOT EXISTS nhanvien;

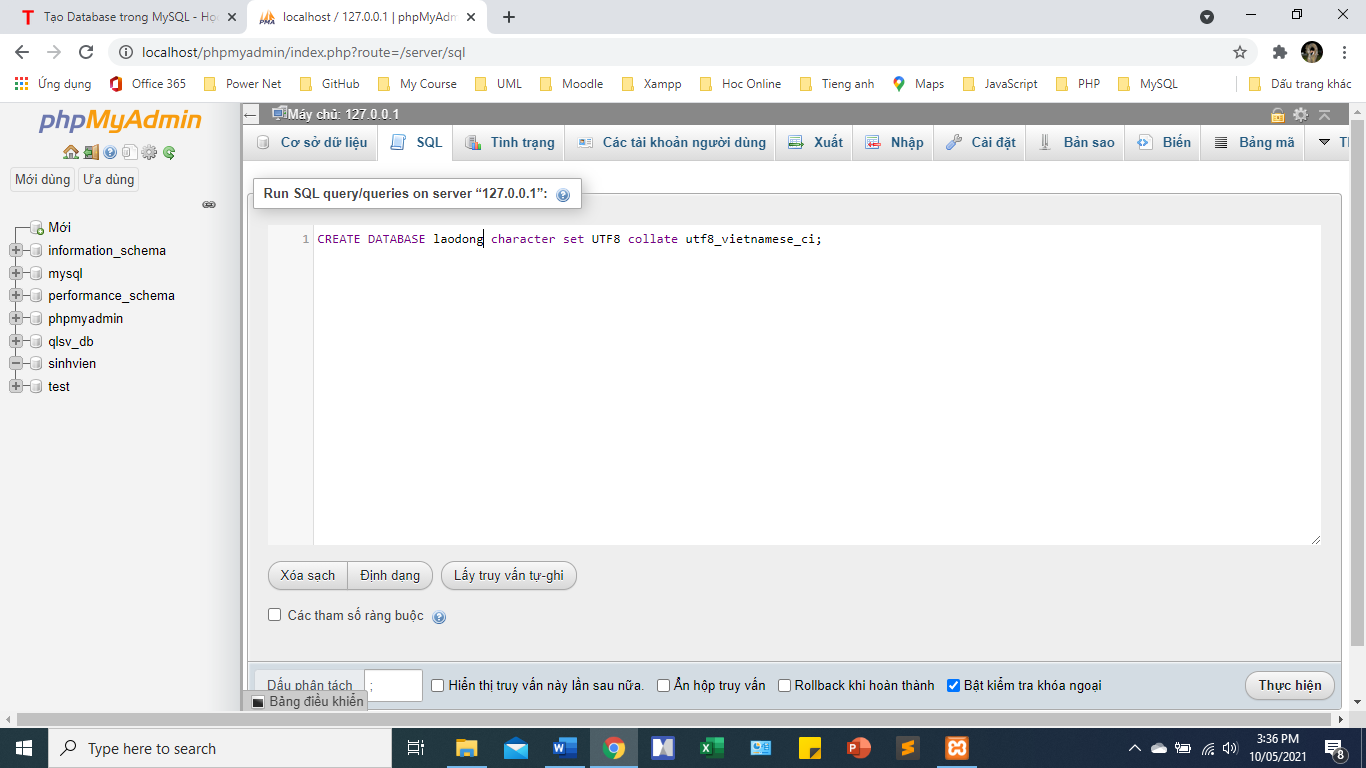
**C2: Sử dụng trình phpMyAdmin của XAMPP**

Trước tiên chúng ta khởi động phpMyAdmin, bằng cách vào địa chỉ <http://localhost/phpmyadmin/> hoặc bấm vào nút Admin (phần MySQL) của Xampp Control panel.



Click vào nút **Mới**, xuất hiện cửa sổ như hình trên và nhập tên database là *sinhvien* và kiểu ký tự tiếng Việt có dấu là *utf8\_vietnamese\_ci*

Bấm nút **Tạo**, Cơ sở dữ liệu *sinhvien* sẽ được tạo ra, hoặc chúng ta sang thẻ SQL và nhập câu SQL



Và ấn nút **Thực hiện**, cách này chúng ta cũng tạo được CSDL

2.1.2. Xoá CSDL

Cú pháp để xóa database trong MySQL như sau:

**DROP DATABASE** [database\_name];

*Ví dụ:*

DROP DATABASE sinhvien;

2.1.3. Chọn CSDL

Dùng cú pháp

**USE** [database\_name];

2.2. Tạo Bảng (Table)

Bảng dữ liệu là cấu trúc có vai trò quan trọng nhất trong cơ sở dữ liệu quan hệ. Toàn bộ dữ liệu của cơ sở dữ liệu được tổ chức trong các bảng, những bảng này có thể là những bảng hệ thống được tạo ra khi tạo cơ sở dữ liệu, và cũng có thể là những bảng do người sử dụng định nghĩa.

Trong các bảng, dữ liệu được tổ chức dưới dạng các dòng và cột. Mỗi một dòng là một bản ghi duy nhất trong bảng và mỗi một cột là một trường. Các bảng trong cơ sở dữ liệu được sử dụng để biểu diễn thông tin, lưu giữ dữ liệu về các đối tượng trong thế giới thực và/hoặc mối quan hệ giữa các đối tượng.

*Cú pháp tạo bảng trong MySQL*

**CREATE TABLE** ten\_bang (

***ten\_cot1*** ***thuoc\_tinh\_cot1*** ***cac\_rang\_buoc\_cot1*** [,

...,

***ten\_cot\_n*** ***thuoc\_tinh\_cot\_n*** ***cac\_rang\_buoc\_cot\_n*** ],

[cac\_rang\_buoc\_tren\_bang]

)

*Trong đó:*

* ten\_bang : Tên của bảng cần tạo. Tên phải tuân theo qui tắc định danh và không được vượt quá 128 ký tự.
* ***ten\_cot*** : Là tên của cột (trường) cần định nghĩa, tên cột phải tuân theo qui tắc định danh và không được trùng nhau trong mỗi một bảng. Mỗi một bảng phải có ít nhất một cột. Nếu bảng có nhiều cột thì định nghĩa của các cột (tên cột, thuộc tính và các ràng buộc) phải phân cách nhau bởi dấu phẩy.
* ***thuoc\_tinh\_cot*** : Mỗi một cột trong một bảng ngoài tên cột còn có các thuộc tính bao gồm:
  + **Kiểu dữ liệu** của cột. Đây là thuộc tính bắt buộc phải có đối với mỗi cột.
  + **Giá trị mặc định** của cột: là giá trị được tự động gán cho cột nếu như người sử dụng không nhập dữ liệu cho cột một cách tường minh. Mỗi một cột chỉ có thể có nhiều nhất một giá trị mặc định.
  + Cột có tính chất IDENTITY hay không? tức là giá trị của cột có được tự động tăng mỗi khi có bản ghi mới được bổ sung hay không. Tính chất này chỉ có thể sử dụng đối với các trường kiểu số.
  + Cột có chấp nhận giá trị NULL hay không.

*Ví dụ:* *Câu lệnh dưới đây định nghĩa bảng NHANVIEN với các trường MANV (mã nhân viên), HOTEN (họ và tên), NGAYSINH (ngày sinh của nhân viên), DIENTHOAI (điện thoại) và HSLUONG (hệ số lương)*

CREATE TABLE nhanvien (

manv INT NOT NULL AUTO\_INCREMENT,

hoten VARCHAR(50) NOT NULL,

ngaysinh DATETIME NULL,

dienthoai VARCHAR(11) NULL,

hsluong DECIMAL(3, 2),

PRIMARY KEY (manv)

);

Trong câu lệnh trên, trường MANV và HOTEN của bảng NHANVIEN không được NULL (tức là bắt buộc phải có dữ liệu), trường NGAYSINH và DIENTHOAI sẽ nhận giá trị NULL.

Thuộc tính ràng buộc NOT NULL của trường được sử dụng bởi vì chúng ta không muốn trường này là NULL. Vì thế, nếu người dùng cố gắng tạo một bản ghi có giá trị NULL, thì MySQL sẽ tạo một lỗi.

Thuộc tính AUTO\_INCREMENT nói cho MySQL tự động tăng khóa chính và thêm giá trị có sẵn tiếp theo tới trường manv.

Từ khóa PRIMARY KEY được sử dụng để định nghĩa một cột là PRIMARY KEY (khóa chính). Bạn có thể sử dụng nhiều cột phân biệt nhau bởi dấu phảy để định nghĩa một PRIMARY KEY.

Chúng ta có thể thêm dữ liệu vào bảng trên bằng lệnh

INSERT INTO nhanvien (hoten, ngaysinh, dienthoai, hsluong)

VALUES('Nguyen Van Vinh', '2020/10/03', '886963', 2.14);

INSERT INTO nhanvien(hoten)

VALUES('Mai Thanh Toan');

INSERT INTO nhanvien(hoten,dienthoai)

VALUES('Tran Thi C','849290');

2.3. Sửa Bảng

- Thông thường thì việc chỉnh sửa bảng được chia làm năm trường hợp chính: sửa tên bảng, thêm một cột mới, xóa một cột, sửa tên cột, thay đổi kiểu dữ liệu của cột.

2.3.1. Sửa tên bảng

- Để sửa tên (đặt lại tên) cho bảng thì chúng ta sử dụng lệnh RENAME TABLE với cú pháp như sau:

RENAME TABLE old\_name TO new\_name;

2.3.2. Thêm một cột mới

- Để thêm một cột mới vào bên trong cái bảng thì chúng ta sử dụng lệnh ADD COLUMN với cú pháp như sau:

ALTER TABLE table\_name ADD COLUMN column\_name DATA TYPE;

*Ví dụ:*

ALTER TABLE nhanvien ADD COLUMN quequan VARCHAR(50) NOT NULL;

2.3.3. Xóa một cột

- Để xóa một cột nào đó ra khỏi bảng thì chúng ta sử dụng lệnh DROP COLUMN với cú pháp như sau:

ALTER TABLE table\_name DROP COLUMN column\_name;

*Ví dụ*

ALTER TABLE nhanvien DROP COLUMN quequan;

2.3.4. Sửa tên cột

- Để sửa tên (đặt lại tên) cho cột thì chúng ta sử dụng lệnh CHANGE COLUMN với cú pháp như sau:

ALTER TABLE table\_name CHANGE COLUMN old\_column\_name new\_column\_name DATA TYPE;

*Ví dụ*

ALTER TABLE nhanvien CHANGE COLUMN hoten hovaten VARCHAR(255);

2.4. Xoá bảng

- Để xóa một cái bảng thì chúng ta sử dụng lệnh DROP TABLE với cú pháp như sau:

DROP TABLE table\_name;

*Ví dụ*

DROP TABLE nhanvien;

2.5. Kết nối cơ sở dữ liệu bằng PHP

Kể từ phiên bản PHP5 trở đi chúng ta có thể kết nối với MySQL bằng cách sử dụng hai thư viện chính đó là:

* MySQLi (i là chữ viết tắt của improved)
* PDO (PHP Data Objects)

Với MySQLi thì chỉ có thể kết nối và xử lý với MySQL, còn PDO có thể kết nối với 12 hệ quản trị CSDL khác nhau. Như vậy nếu ứng dụng của bạn có kết nối với nhiều hệ quản trị CSDL thì bạn nên sử dụng PDO, còn chỉ một mình MySQL thôi thì có thể chọn MySQLi.

Giả sử nếu dự án của bạn đang viết ở MySQL, nhưng một ngày nào đó bạn chuyển toàn bộ data sang SQL Server thì lúc này PDO lại có lợi thế hơn bởi vì bạn chỉ cần thay chuỗi kết nối là được. Nhưng với MySQLi thì ban sẽ phải viết lại toàn bộ dự án.

2.5.1. Kết nối MySQL bằng MySQLi Procedural

Kết nối MySQL bằng MySQLi Procedural (Kết nối theo hướng thủ tục). Để kết nối PHP với MySql bằng phương thức này thì chúng ta dùng hàm mysqli\_connect() với cú pháp như sau:

**mysqli\_connect**(**hostname**, **username**, **password**, **databasename**);

*Trong đó:*

* **hostname**: Server của MySql (mặc định là localhost)
* **username**: là tên truy cập vào MySql (mặc định là root).
* **password**: Là mật khẩu truy cập vào MySql (mặc định là rỗng).
* **databasename**: Là tên database mà các bạn muốn kết nối.

*Ví dụ: Giả sử chúng ta đã tạo bằng phpAdmin một cơ sở dữ liệu có tên* ***qlsv\_db****, bây giờ để kết nối chúng ta tạo file con1.php với đoạn code sau:*

<?php

*//khai báo biến host*

$hostName = 'localhost';

*// khai báo biến username*

$userName = 'root';

*//khai báo biến password*

$passWord = '';

*// khai báo biến databaseName*

$databaseName = 'qlsv\_db';

*// khởi tạo kết nối*

$connect = mysqli\_connect($hostName, $userName, $passWord, $databaseName); *// hàm trả về TRUE nếu các tham //số truyền vào đều OK, ngược lại sẽ FALSE*

*//Kiểm tra kết nối*

if (!$connect) {

exit('Kết nối không thành công!');

}

*// thành công*

echo 'Kết nối thành công!';

?>

2.5.2. Kết nối MySQL bằng MySQLi Object-Oriented

Kết nối MySQL bằng MySQLi Object-Oriented (kết nối theo hướng đối tượng). Đối với phương thức này các bạn chỉ cần khởi tạo class mysqli với cú pháp như sau:

new mysqli(hostname, username, password, databasename);

*Ví dụ*

<?php

//khai báo biến host

$hostName = 'localhost';

// khai báo biến username

$userName = 'root';

//khai báo biến password

$passWord = '';

// khai báo biến databaseName

$databaseName = 'qlsv\_db';

// khởi tạo kết nối

$connect = new mysqli($hostName, $userName, $passWord, $databaseName);

//Kiểm tra kết nối

if ($connect->connect\_error) {

exit('Kết nối không thành công. chi tiết lỗi:' . $connect->connect\_error);

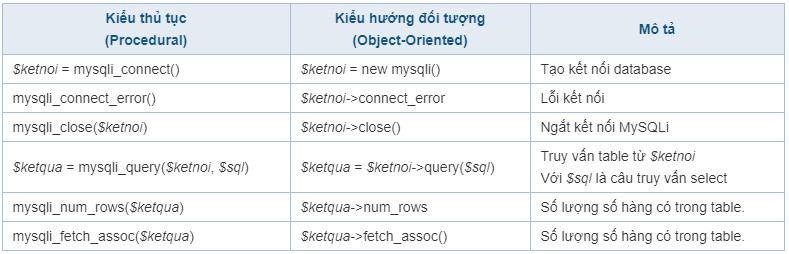
}

// thành công

echo 'Kết nối thành công!';

?>

Một số hàm cơ bản MySQLi



2.5.3. Kết nối PHP với MySql bằng PDO

Kết nối PHP với MySql bằng PDO (PHP Data Object), với cách kết nối này chúng ta có thể kết nối đến nhiều loại CSDL khác nhau: Oracel,postpreSQL,... Để kết nối PHP với MySql bằng PDO các bạn chỉ cần khởi tạo class PDO với cú pháp:

**new** **PDO** (**DSN**, **username**, **password**);

*Trong đó:*

* **DSN** (Data Source Name): DSN định nghĩa loại database (còn gọi là drive), tên database, và bất kỳ thông tin nào cần thiết khác.
* **Username**: là tên truy cập vào MySql (mặc định là root).
* **Password**: Là mật khẩu truy cập vào MySql (mặc định là rỗng).

*Ví dụ*

<?php

*//khai báo biến host*

$hostName = 'localhost';

*// khai báo biến username*

$userName = 'root';

*//khai báo biến password*

$passWord = '';

*// khai báo biến databaseName*

$databaseName = 'qlsv\_db';

*// khởi tạo kết nối*

try {

$connect = new PDO('mysql:host=' . $hostName . ';dbname=' . $databaseName, $userName, $passWord);

$connect->setAttribute(PDO::ATTR\_ERRMODE, PDO::ERRMODE\_EXCEPTION);

*//thành công*

echo 'thành công';

} catch (PDOException $e) {

*//thất bại*

die($e->getMessage());

}

?>

*Giải thích*

Một kết nối PDO database cần tạo ‘PDO object’ với Data Source Name (DSN), username và password.

Trong ví dụ trên ta thấy dòng:



Có thể được viết lại như sau:

$connect= new PDO("mysql:host=$hostName; dbname=$databaseName", $userName, $passWord);

Đoạn "mysql:host=$hostName; dbname=$databaseName" chính là DSN, đó là 1 chuỗi (có nháy kép bao bọc)

2.5.4. Ngắt kết nối

-Để ngắt kết nối PHP với MySql các bạn sử dụng một trong các cách sau (chú ý kết nối theo cách nào thì hủy theo cách đó).

mysqli hướng thủ tục

-Để hủy kết nối với phương thức này thì các bạn dùng hàm mysqli\_close() với cú pháp

mysqli\_close(connect);

-Với connect là biến khởi tạo kết nối.

*VD: Hủy kết nối với VD phần trên*

mysqli\_close($connect);

mysqli hướng đối tượng

-Đê hủy kết nối với cách này thì các bạn chỉ cần gọi phương thức close()

*VD: Hủy kết nối với VD phần trên*

$connect->close();

PDO

-Để hủy kết nối bằng cách này thì các bạn chỉ cần set biến khởi tạo về null.

*VD: Hủy kết nối với VD phần trên*

$connect = null

2.6. Thêm bản ghi

2.6.1. Chèn bằng MySQLi hướng thủ tục

Với cách này chúng ta dùng hàm **mysqli\_query** để thực hiện các câu truy vấn với cú pháp sử dụng như sau:

**mysqli\_query**(**connect**, **query**)

*Trong đó:*

**connect** là biến kết nối PHP với MySql, còn **query** là câu truy vấn SQL

*Ví dụ:*

1) Giải sử ta có file **con1.php** dùng để kết nối như sau:

<?php

*//khai báo biến host*

$hostName = 'hungcfi.edu.vn';

*// khai báo biến username*

$userName = 'root';

*//khai báo biến password*

$passWord = '';

*// khai báo biến databaseName*

$databaseName = 'qlsinhvien';

*// khởi tạo kết nối*

$connect = mysqli\_connect($hostName, $userName, $passWord, $databaseName);

*//Kiểm tra kết nối*

if (!$connect) {

exit('Kết nối không thành công!');

}

?>

2) Tạo file **insert.php** với nội dung

<?php

require 'con1.php';

$sql = "INSERT INTO hssv (hoten,gioitinh,ngaysinh,quequan)

VALUES ('Phan Thị Bình', 'Nữ','12-04-2020','Đà Nẵng')";

*//kiểm tra*

if (mysqli\_query($connect, $sql))

*//Thông báo nếu thành công*

echo 'Thêm thành công';

else

*//Hiện thông báo khi không thành công*

echo 'Không thành công. Lỗi' . mysqli\_error($connect);

*//ngắt kết nối*

mysqli\_close($connect);

?>

2.6.2. Chèn bằng MySQLi hướng đối tượng

Với cách này chúng ta chỉ cần gọi phương thức query của class mysqli để thực hiện truy vấn.

*Ví dụ*

<?php

require 'con1.php';

$sql = "INSERT INTO hssv (hoten,gioitinh,ngaysinh,quequan)

VALUES ('Phan Thị Bình', 'Nữ','12-04-2020','Đà Nẵng')";

*//kiểm tra*

if ($connect->query($sql) === TRUE)

*//Thông báo nếu thành công*

echo 'Thêm thành công';

else

*//Hiện thông báo khi không thành công*

echo 'Không thành công. Lỗi' . $connect->error;

*//ngắt kết nối*

$connect->close();

?>

2.7. Lấy ID của bản ghi đã chèn

Trong các mô hình CSDL có mối quan hệ khóa ngoại nên lúc Insert đôi lúc ta cần lấy giá trị khóa chính của field vừa insert để các table con sử dụng làm khóa ngoại. Lúc này ta phải sử dụng các hàm có sẵn trong PHP để giải quyết.

Mỗi thư viện database trong PHP đều cung cấp những hàm và phương thức giúp lấy ID vừa insert (tức là id cuối cùng).

Đối với thư viện MySQLi thì ta có hàm **mysqli\_insert\_id()** và thuộc tính **insert\_id**. Còn đối với PDO thì ta có phương thức **lastInsertId().**

2.7.1. MySQLi hướng thủ tục

<?php

require 'con1.php';

*//câu truy vấn*

$sql = "INSERT INTO hssv (hoten,gioitinh,ngaysinh,quequan)

VALUES ('Trịnh Nam Sơn', 'Nam','1996-10-15','An Giang')";

*//kiểm tra*

if (mysqli\_query($connect, $sql))

*//Thông báo nếu thành công*

echo 'Thêm thành công. ID=' . mysqli\_insert\_id($connect);

else

*//Hiện thông báo khi không thành công*

echo 'Không thành công. Lỗi' . mysqli\_error($connect);

*//ngắt kết nối*

mysqli\_close($connect);

?>

2.7.2. MySQLi hướng đối tượng

<?php

require 'con1.php';

*//câu truy vấn*

$sql = "INSERT INTO hssv (hoten,gioitinh,ngaysinh,quequan)

VALUES ('Trương Hoài Hương', 'Nữ','1992-12-05','Rạch Giá')";

*//kiểm tra*

if ($connect->query($sql) === TRUE)

*//Thông báo nếu thành công*

echo 'Thêm thành công. ID=' . $connect->insert\_id;

else

*//Hiện thông báo khi không thành công*

echo 'Không thành công. Lỗi' . $connect->error;

*//ngắt kết nối*

$connect->close();

?>

2.8. Chèn nhiều bảng ghi

Trong thư viện MySQLi hỗ trợ hai hàm dùng để thực hiện nhiều câu lệnh truy vấn đó là hàm **mysqli\_multi\_query()** đối với trường hợp dùng Procedural và phương thức **multi\_query()** đối với trường hợp dùng Object-oriented.

Riêng đối với PDO thì ta phải thực hiện khác chút xíu đó là phải hợp transaction và mỗi lần mỗi câu truy vấn chứ không phải là chạy cùng lúc.

2.8.1. MySQLi hướng thủ tục

<?php

require 'con1.php';

*//câu truy vấn*

$sql = "INSERT INTO hssv (hoten,gioitinh,ngaysinh,quequan)

VALUES ('Trịnh Nam Sơn', 'Nam','1996-10-15','An Giang'),

INSERT INTO hssv (hoten,gioitinh,ngaysinh,quequan)

VALUES ('Trương Hoài Hương', 'Nữ','1992-12-05','Rạch Giá')";

*//kiểm tra*

if (mysqli\_multi\_query($connect, $sql))

*//Thông báo nếu thành công*

echo 'Thêm thành công.';

else

*//Hiện thông báo khi không thành công*

echo 'Không thành công. Lỗi' . mysqli\_error($connect);

*//ngắt kết nối*

mysqli\_close($connect);

?>

2.8.2. MySQLi hướng đối tượng

<?php

require 'con1.php';

*//câu truy vấn*

$sql = "INSERT INTO hssv (hoten,gioitinh,ngaysinh,quequan)

VALUES ('Trịnh Nam Sơn', 'Nam','1996-10-15','An Giang'),

INSERT INTO hssv (hoten,gioitinh,ngaysinh,quequan)

VALUES ('Trương Hoài Hương', 'Nữ','1992-12-05','Rạch Giá')";

*//kiểm tra*

if ($connect->multi\_query($sql) === TRUE)

*//Thông báo nếu thành công*

echo 'Thêm thành công.';

else

*//Hiện thông báo khi không thành công*

echo 'Không thành công. Lỗi' . $connect->error;

*//ngắt kết nối*

$connect->close();

?>

2.9. Select dữ liệu

Câu lệnh select thường dùng để lấy dữ liệu và hiển thị lên website ở cả frontend lẫn backend, đây là câu lệnh thông dụng hay sử dụng nhất trong một website.

Trong MySQL để lấy danh sách records trong một bảng dữ liệu thì ta sử dụng lệnh select với cú pháp:

SELECT field1, field2 FROM table\_name

Bên cạnh đó ta cũng có sử dụng thêm điều kiện where và limit để giới hạn kết quả trả về. Như vậy việc lấy danh sách dữ liệu trong MySQL bằng PHP thực chất là ta sử dụng PHP để chạy câu truy vấn SQL.

Bây giờ chúng ta cùng tìm hiểu các thao tác lấy dữ liệu với MySQLi.

2.9.1. MySQLi hướng thủ tục

<?php

require 'con1.php';

*//khởi tạo biến $i để đếm;*

$i = 1;

*//câu truy vấn*

$sql = "SELECT \* FROM hssv";

*//kiểm tra*

if ($result = mysqli\_query($connect, $sql)) {

while ($row = mysqli\_fetch\_assoc($result)) {

*//hiển thị dữ liệu*

echo 'Dữ liệu thứ ' . $i . ' gồm: ' . $row['masv'] . '-' . $row['hoten'] . '-' . $row['gioitinh'] . '-'.$row['ngaysinh'].'-'.$row['quequan'].'<br/>';

*//tăng $i lên 1*

$i++;

}

} else

*//Hiện thông báo khi không thành công*

echo 'Không thành công. Lỗi' . mysqli\_error($connect);

*//ngắt kết nối*

mysqli\_close($connect);

?>

*Trong đó:*

Hàm **mysqli\_fetch\_assoc()** sẽ tìm và trả về một dòng kết quả của một truy vấn MySQL nào đó dưới dạng một mảng kết hợp. (có thể dùng hàm **mysqli\_fetch\_array()** tương tự)

Cú pháp: mysqli\_fetch\_assoc( $result);

Trong đó:

$result là kết quả của truy vấn, là kết quả trả về của các hàm: mysqli\_query(), mysqli\_store\_result() hoặc mysqli\_use\_result().

2.9.2. MySQLi hướng đối tượng

<?php

require 'con2.php';

*//câu truy vấn*

$sql = "SELECT \* FROM hssv";

*//run câu truy vấn*

$result = $connect->query($sql);

*//kiểm tra xem có dữ liệu trả vê không*

if ($result && $result->num\_rows > 0) {

*// nếu có thì tiến hành lặp để in ra dữ liệu*

$i = 1;

while ($row = $result->fetch\_assoc()) {

*//hiển thị dữ liệu*

echo 'Dữ liệu thứ ' . $i . ' gồm: ' . $row['masv'] . '-' . $row['hoten'] . '-' . $row['gioitinh'] . '-'.$row['ngaysinh'].'-'.$row['quequan'].'<br/>';

*//tăng $i lên 1*

$i++;

}

} else

*//Hiện thông báo khi không thành công*

echo 'Không thành công. Lỗi' . $connect->error;

*//ngắt kết nối*

$connect->close();

?>

2.7. Cập nhật bản ghi (update record)

Trong phần này chúng ta sẽ tìm hiểu đến thao tác cuối cùng thường hay sử dụng trong MySQL lẫn PHP đó là cập nhật dữ liệu, đây là thao tác được dùng trong backend dùng để sửa thông tin sản phẩm, tin tức, chuyên mục, ...

Cú pháp của lệnh update như sau:

UPDATE table\_name

SET column1 = value1, column2 = value2,...

WHERE some\_column = some\_value;

Tương tự như trường hợp delete bạn phải truyền điều kiện update vào nếu không nó sẽ cập nhật cho toàn bộ records có trong table. Thông thường thì chúng ta lấy điều kiện theo khóa chính.

2.7.1. Update bằng MySQLi hướng thủ tục

*Ví dụ: trong CSDL qlsinhvien, chúng ta cần sửa lại nội dung của bản ghi có ID=2*

<?php

require 'con1.php';

$sql = "UPDATE hssv SET hoten='Trần Minh Cường',gioitinh='Nam' WHERE masv=2";

*//kiểm tra*

if (mysqli\_query($connect, $sql))

*//Thông báo nếu thành công*

echo 'Sửa thành công';

else

*//Hiện thông báo khi không thành công*

echo 'Không thành công. Lỗi' . mysqli\_error($connect);

*//ngắt kết nối*

mysqli\_close($connect);

?>

2.7.2. Update bằng MySQLi hướng đối tượng

*Ví dụ: Sửa bản ghi có masv=3*

<?php

require 'con2.php';

$sql = "UPDATE hssv SET hoten='Phạm Nhật Ánh',gioitinh='Nam', ngaysinh='1992-10-06' WHERE masv=3";

*//kiểm tra*

if ($connect->query($sql) === TRUE)

*//Thông báo nếu thành công*

echo 'Sửa thành công';

else

*//Hiện thông báo khi không thành công*

echo 'Không thành công. Lỗi' . $connect->error;

*//ngắt kết nối*

$connect->close();

?>

2.8. Xóa bản ghi

Lệnh delete dùng để xóa các records trong MySQL, lệnh này thường dùng trong backend đẻ xóa các tin tức, sản phẩm, chuyên mục. Cú pháp của lệnh này như sau:

DELETE FROM table\_name WHERE some\_columns = some\_values;

Thường trong ứng dụng website thì ta sẽ delete theo khóa chính bởi vì khóa chính tăng tự động nên truy vấn tìm kiếm sẽ nhanh hơn. Nếu bạn không truyền vào điều kiện where thì nó sẽ xóa hết tất cả data trong table ,vì vậy hãy kiểm tra kỹ điều kiện trước khi thực hiện lệnh này.

2.8.1. Delete bằng MySQLi hướng thủ tục

<?php

require 'con1.php';

$sql = "DELETE FROM hssv WHERE masv=2";

*//kiểm tra*

if (mysqli\_query($connect, $sql))

*//Thông báo nếu thành công*

echo 'Xoá thành công';

else

*//Hiện thông báo khi không thành công*

echo 'Xoá không thành công. Lỗi' . mysqli\_error($connect);

*//ngắt kết nối*

mysqli\_close($connect);

?>

2.8.2. Delete bằng MySQLi hướng đối tượng

<?php

require 'con2.php';

$sql = "DELETE FROM hssv WHERE masv=2";

*//kiểm tra*

if ($connect->query($sql) === TRUE)

*//Thông báo nếu thành công*

echo 'Xoá thành công';

else

*//Hiện thông báo khi không thành công*

echo 'Xoá không thành công. Lỗi' . $connect->error;

*//ngắt kết nối*

$connect->close();

?>

[1. Cơ sở dữ liệu MySQL 1](#_Toc72047977)

[1.1. Các kiểu dữ liệu trong MySQL 1](#_Toc72047978)

[1.1.1. Kiểu dữ liệu Numeric 1](#_Toc72047979)

[1.1.2. Kiểu dữ liệu Date và Time trong MySQL 2](#_Toc72047980)

[1.1.3. Kiểu dữ liệu chuỗi trong MySQL 2](#_Toc72047981)

[1.2. Các thuộc tính 3](#_Toc72047982)

[2. Tạo và quản trị cơ sở dữ liệu SQL 3](#_Toc72047983)

[2.1. Tạo Cơ sở dữ liệu (Database) 3](#_Toc72047984)

[2.1.1. Tạo CSDL 3](#_Toc72047985)

[2.1.2. Xoá CSDL 5](#_Toc72047986)

[2.1.3. Chọn CSDL 5](#_Toc72047987)

[2.2. Tạo Bảng (Table) 5](#_Toc72047988)

[2.3. Sửa Bảng 6](#_Toc72047989)

[2.3.1. Sửa tên bảng 6](#_Toc72047990)

[2.3.2. Thêm một cột mới 7](#_Toc72047991)

[2.3.3. Xóa một cột 7](#_Toc72047992)

[2.3.4. Sửa tên cột 7](#_Toc72047993)

[2.4. Xoá bảng 7](#_Toc72047994)

[2.5. Kết nối cơ sở dữ liệu bằng PHP 7](#_Toc72047995)

[2.5.1. Kết nối MySQL bằng MySQLi Procedural 7](#_Toc72047996)

[2.5.2. Kết nối MySQL bằng MySQLi Object-Oriented 8](#_Toc72047997)

[Một số hàm cơ bản MySQLi 9](#_Toc72047998)

[2.5.3. Kết nối PHP với MySql bằng PDO 9](#_Toc72047999)

[2.5.4. Ngắt kết nối 10](#_Toc72048000)

[mysqli hướng thủ tục 10](#_Toc72048001)

[mysqli hướng đối tượng 10](#_Toc72048002)

[PDO 10](#_Toc72048003)

[2.6. Thêm bản ghi 10](#_Toc72048004)

[2.6.1. Chèn bằng MySQLi hướng thủ tục 10](#_Toc72048005)

[2.6.2. Chèn bằng MySQLi hướng đối tượng 11](#_Toc72048006)

[2.7. Lấy ID của bản ghi đã chèn 12](#_Toc72048007)

[2.7.1. MySQLi hướng thủ tục 12](#_Toc72048008)

[2.7.2. MySQLi hướng đối tượng 12](#_Toc72048009)

[2.8. Chèn nhiều bảng ghi 13](#_Toc72048010)

[2.8.1. MySQLi hướng thủ tục 13](#_Toc72048011)

[2.8.2. MySQLi hướng đối tượng 13](#_Toc72048012)

[2.9. Select dữ liệu 14](#_Toc72048013)

[2.9.1. MySQLi hướng thủ tục 14](#_Toc72048014)

[2.9.2. MySQLi hướng đối tượng 14](#_Toc72048015)

[2.7. Cập nhật bản ghi (update record) 15](#_Toc72048016)

[2.7.1. Update bằng MySQLi hướng thủ tục 15](#_Toc72048017)

[2.7.2. Update bằng MySQLi hướng đối tượng 16](#_Toc72048018)

[2.8. Xóa bản ghi 16](#_Toc72048019)

[2.8.1. Delete bằng MySQLi hướng thủ tục 16](#_Toc72048020)

[2.8.2. Delete bằng MySQLi hướng đối tượng 16](#_Toc72048021)