Chapter 5
PHP and MySQL

Lectured by:
Nguyễn Hữu Hiếu

Kết nối và đóng kết nối:
Hàm mysqli_connect() dùng để mở kết nối tới máy chủ MySQL.
Hàm mysqli_close() dùng để đóng kết nối.
Hàm mysqli_errno() trả về mã lỗi của thao tác MySQL gần nhất (nếu có).
Hàm mysqli_error() trả về thông báo lỗi chi tiết.
Dấu @ (error control operator) dùng để tắt hiển thị thông báo lỗi.
Quản lý Database:
Hàm mysqli_create_db() dùng để tạo một database mới.
Hàm mysqli_select_db() dùng để chọn một database để làm việc.
Hàm mysqli_drop_db() dùng để xóa một database.
Thực thi câu lệnh SQL:
Hàm mysqli_query() dùng để gửi các câu lệnh SQL tới MySQL.
"Result pointer" là một biến đặc biệt trỏ tới dòng dữ liệu hiện tại trong kết quả truy vấn.
Tạo và xóa bảng:
Câu lệnh CREATE TABLE (thường dùng với mysqli_query()) để tạo một bảng mới.
PRIMARY KEY clause chỉ định một hoặc nhiều trường làm khóa chính cho bảng.
AUTO_INCREMENT clause tạo một trường tự động tăng giá trị.
NOT NULL clause tạo một trường bắt buộc phải có dữ liệu.
Câu lệnh DROP TABLE (thường dùng với mysqli_query()) để xóa một bảng.
Thao tác với dữ liệu (Records):
Câu lệnh LOAD DATA (thường dùng với mysqli_query()) để thêm nhiều bản ghi từ một file text.
Câu lệnh UPDATE (thường dùng với mysqli_query()) để cập nhật các bản ghi.
Câu lệnh DELETE (thường dùng với mysqli_query()) để xóa các bản ghi.
Hàm mysqli_info() trả về thông tin về các thao tác (ví dụ: số dòng được thêm, sửa) tùy thuộc vào loại truy vấn.
Lấy dữ liệu từ kết quả truy vấn:
Hàm mysqli_fetch_row() trả về dữ liệu của dòng hiện tại dưới dạng mảng số (indexed array).
Hàm mysqli_fetch_assoc() trả về dữ liệu của dòng hiện tại dưới dạng mảng kết hợp (associative array).
Hàm mysqli_free_result() dùng để giải phóng bộ nhớ sau khi làm việc xong với kết quả truy vấn.
Thông tin về kết quả truy vấn:
Hàm mysqli_num_rows() trả về số lượng dòng trong kết quả truy vấn.
Hàm mysqli_num_fields() trả về số lượng cột trong kết quả truy vấn.
Với các truy vấn SELECT, mysqli_num_rows() cho biết số lượng bản ghi được trả về.

Objectives

In this lesson, you will:

- Connect to MySQL from PHP
- Work with MySQL databases using PHP
- Create, modify, and delete MySQL tables with PHP
- Use PHP to manipulate MySQL records
- Use PHP to retrieve database records

Connecting to MySQL with PHP

- PHP has the ability to access and manipulate any database that is ODBC (Open Database Connectivity) compliant
- PHP includes functionality that allows you to work directly with different types of databases, without going through ODBC

PHP có khả năng truy cập và thao tác bất kỳ CSDL nào tuân thủ ODBC (Open Database Connectivity).

PHP cũng có các chức năng cho phép làm việc trực tiếp với nhiều loại CSDL khác nhau mà không cần thông qua ODBC.

Which MySQL Package to Use

- The mysqli (MySQL Improved) package became available with PHP 5 and is designed to work with MySQL version 4.1.3 and later
- Earlier versions must use the mysql package
- The mysqli package is the object-oriented equivalent of the mysql package but can also be used procedurally
- Mysqli package has improved speed, security and compatibility with libraries.

Từ PHP 5 trở đi, gói mysqli (MySQL Improved) ra đời để làm việc với MySQL từ phiên bản 4.1.3 trở lên.
Các phiên bản PHP cũ hơn thì phải dùng gói mysql.
mysqli là gói hướng đối tượng, tương đương với gói mysql, nhưng cũng có thể dùng theo kiểu thủ tục.
mysqli ngon hơn mysql ở mấy khoản tốc độ, bảo mật và khả năng tương thích với các thư viện khác.

Opening and Closing a Connection

- Open a connection to a MySQL database server with the mysqli_connect() function
- The mysqli_connect() function returns a positive integer if it connects to the database successfully or FALSE if it does not
- Assign the return value from the mysqli_connect() function to a variable that you can use to access the database in your script

Để mở một kết nối tới máy chủ MySQL, dùng hàm mysqli_connect().
Hàm mysqli_connect() trả về một số nguyên dương nếu kết nối thành công, hoặc FALSE nếu thất bại.
Gán giá trị trả về của hàm mysqli_connect() cho một biến để có thể dùng biến đó để truy cập CSDL trong script của bạn.

Opening and Closing a Connection

- The syntax for the mysqli_connect() function is:

```
$connection = mysqli_connect("host" [, "user",
"password"[,"database"]]);
```

- The host argument specifies the host name where your MySQL database server is installed
- The user and password arguments specify a MySQL account name and password
- You can optionally select the database when connecting.

$connection: Đây là biến mà bạn sẽ gán kết quả trả về của hàm mysqli_connect(). Biến này sau đó sẽ được dùng để thực hiện các thao tác với CSDL.

"host": Đối số này chỉ định tên máy chủ (hostname) nơi cài đặt MySQL. Ví dụ: " localhost" nếu MySQL chạy trên cùng máy với PHP.

"user": Đối số này chỉ định tên tài khoản MySQL để kết nối.

"password": Đối số này chỉ định mật khẩu của tài khoản MySQL.

"database" (tùy chọn): Bạn có thể chọn CSDL ngay khi kết nối bằng cách chỉ định tên CSDL ở đây.

Tóm lại, hàm mysqli_connect() cần các thông tin về máy chủ, tài khoản và mật khẩu để thiết lập kết nối tới MySQL. Bạn có thể chọn sẵn CSDL muốn làm việc ngay từ đầu hoặc chọn sau.

Nguồn và nội dung liên quan

Opening and Closing a Connection

- The database connection is assigned to the $DBConnect variable

```
$DBConnect = mysqli_connect("localhost",
"billyeakus ", "hotdog");
```

Dòng này là một ví dụ cụ thể của việc dùng hàm mysqli_connect(). Nó mở một kết nối tới máy chủ MySQL có địa chỉ là "localhost", với tài khoản người dùng là "billyeakus" và mật khẩu là "hotdog". Kết nối này được gán cho biến $DBConnect.

- Close a database connection using the mysqli_close() function

Dòng này dùng hàm mysqli_close() để đóng kết nối tới CSDL. Tham số truyền vào là biến $DBConnect, là cái "mối liên lạc" mà mình đã thiết lập trước đó.

```
mysqli_close($DBConnect);
```

Opening and Closing a Connection

mysqli_get_client_info()	Returns the MySQL client library version
mysqli_get_client_stats()	Returns statistics about client per-process
mysqli_get_client_version()	Returns the MySQL client library version as an integer
mysqli_get_connection_stats()	Returns statistics about the client connection
mysqli_get_host_info()	Returns the MySQL server hostname and the connection type
mysqli_get_proto_info()	Returns the MySQL protocol version
mysqli_get_server_info()	Returns the MySQL server version
mysqli_get_server_version()	Returns the MySQL server version as an integer

mysqli_get_client_info(): Trả về phiên bản của thư viện client MySQL đang dùng.

mysqli_get_client_stats(): Trả về thông tin thống kê về client theo từng tiến trình.

mysqli_get_client_version(): Trả về phiên bản của thư viện client MySQL dưới dạng số nguyên.

mysqli_get_connection_stats(): Trả về thông tin thống kê về kết nối client.

mysqli_get_host_info(): Trả về tên máy chủ MySQL và kiểu kết nối.

mysqli_get_proto_info(): Trả về phiên bản của giao thức MySQL.

mysqli_get_server_info(): Trả về phiên bản của máy chủ MySQL.`

mysqli_get_server_version(): Trả về phiên bản của máy chủ MySQL dưới dạng số nguyên.

Opening and Closing a Connection

version.php in a Web browser

Reporting MySQL Errors

- **Reasons for not connecting to a database server include:**
  - The database server is not running
  - Insufficient privileges to access the data source
  - Invalid username and/or password

Tài khoản MySQL mà mình dùng để kết nối không có đủ quyền để "thò tay" vào CSDL.

Reporting MySQL Errors

- The mysqli_errno() function returns the error code from the last attempted MySQL function call or 0 if no error occurred
  Hàm mysqli_errno() trả về mã lỗi của lần gọi hàm MySQL trước đó. Nếu không có lỗi thì nó trả về 0.

- The mysqli_error() — Returns the text of the error message from previous MySQL operation
  Hàm mysqli_error() thì trả về chuỗi mô tả lỗi của thao tác MySQL trước đó.

- The mysqli_errno() and mysqli_error() functions return the results of the previous mysqli*() function
  Hai hàm mysqli_errno() và mysqli_error() này cho mình biết kết quả của các hàm mysqli*() liền trước đó.

Selecting a Database

- The syntax for the mysqli_select_db() function is:

  Trong đó, connection là cái "mối liên lạc" mà mình đã thiết lập với MySQL bằng hàm mysqli_connect() lúc nãy, còn database là tên của cái database mà mình muốn xài.

  mysqli_select_db(*connection*, *database*);

- The function returns a value of TRUE if it successfully selects a database or FALSE if it does not

  Hàm này trả về TRUE nếu chọn database thành công, còn nếu không thì trả về FALSE.

- For security purposes, you may choose to use an include file to connect to the MySQL server and select a database

  Để bảo mật, có thể dùng file include để kết nối tới MySQL server và chọn database.

Sample Code

good

```
$link = mysqli_connect("cs.mvnu.edu",
"demo", "demo");

mysqli_select_db($link, "nonexistentdb");
echo mysqli_errno($link) . ": " .
mysqli_error($link). "<br>";

mysqli_select_db( $link, "demo");
mysqli_query($link,
"SELECT * FROM nonexistenttable");
echo mysqli_errno($link) . ": " .
mysqli_error($link) . "<br>";
```

bad

good

bad

Sample Code

```
$host='localhost';
$userName = 'demo';
$password = 'demo';
$database ='demo';

$link = mysqli_connect ($host, $userName, $password) ;

if (!$link) {
    die('Could not connect: ' . mysqli_error($link));
}

echo 'Connected successfully';

mysqli_close($link);
```

include('file.php'): Nhúng file file.php vào. Nếu file đó không tồn tại hoặc có lỗi, nó sẽ sinh ra cảnh báo (warning) nhưng script vẫn chạy tiếp.
include_once('file.php'): Cũng giống như include(), nhưng nó chỉ nhúng file đó một lần duy nhất. Nếu trước đó đã nhúng rồi thì thôi, không nhúng lại nữa.
require('file.php'): Nhúng file file.php vào. Nếu file đó không tồn tại hoặc có lỗi, nó sẽ sinh ra lỗi nghiêm trọng (fatal error) và script sẽ dừng lại.
require_once('file.php'): Giống như require(), nhưng cũng chỉ nhúng file đó một lần duy nhất.

include('file.php'); include_once('file.php')
require('file.php'); require_once('file.php');

index.php

```
<?php
        include_once('config/file.php');
        echo "test";
         include('config/file.php');
?>
```

File index.php dùng include_once('config/file.php ') để nhúng file config/file.php vào. Sau đó nó in ra chữ "test", rồi lại dùng include('config/file.php') để nhúng config/file.php một lần nữa. Nhưng vì đã dùng include_once() ở trên rồi nên lần này nó sẽ không nhúng lại nữa.

config/file.php

```
<?php
        include('file2.php');
?>
```

File config/file.php thì lại dùng include('file2.php') để nhúng file file2.php vào.

1. include('file.php'):

Tính chất: Thằng này nó "hiền" nhất. Nó sẽ cố gắng nhúng cái file file.php vào.
Khi file không tồn tại hoặc có lỗi: Nếu mà cái file file.php không tìm thấy hoặc trong quá trình nhúng mà có lỗi gì đó xảy ra, thì nó chỉ sinh ra một cảnh báo (warning) thôi. Script PHP vẫn sẽ cố gắng chạy tiếp những dòng code phía sau.
Khi nào dùng: Thường dùng cho mấy file không quá quan trọng, ví dụ như mấy cái template giao diện phụ, hoặc mấy file cấu hình mà nếu không có thì chương trình vẫn có thể chạy "tạm" được.
2. include_once('file.php'):

Tính chất: Giống như include() nhưng nó "khôn" hơn. Nó sẽ kiểm tra xem cái file file.php đã được nhúng vào trước đó chưa.
Khi file đã được nhúng: Nếu rồi thì nó sẽ bỏ qua, không nhúng lại nữa. Cái này giúp tránh bị trùng lặp code và mấy cái lỗi không đáng có khi một file được nhúng nhiều lần.
Khi file không tồn tại hoặc có lỗi: Vẫn giống include(), chỉ sinh ra cảnh báo và chạy tiếp.
Khi nào dùng: Mấy cái file thư viện, file hàm, hoặc mấy file cấu hình mà mình chắc chắn chỉ muốn nhúng một lần trong suốt quá trình chạy script.
3. require('file.php'):

Tính chất: Thằng này thì "gắt" hơn nhiều. Nó cũng cố gắng nhúng file file.php vào.
Khi file không tồn tại hoặc có lỗi: Nếu mà không tìm thấy file hoặc có lỗi khi nhúng, nó sẽ gây ra một lỗi nghiêm trọng (fatal error) và ngừng ngay lập tức việc chạy script.
Khi nào dùng: Dùng cho mấy file cực kỳ quan trọng, ví dụ như file chứa các hàm cốt lõi, file kết nối database. Nếu thiếu mấy file này thì chương trình chắc chắn không thể chạy được.
4. require_once('file.php'):

Tính chất: Kết hợp tính "gắt" của require() và sự "khôn ngoan" của _once. Nó sẽ kiểm tra xem file đã được nhúng chưa, nếu rồi thì thôi.
Khi file không tồn tại hoặc có lỗi: Gây ra lỗi nghiêm trọng và dừng script.
Khi nào dùng: Giống như require() nhưng dùng khi mình không chắc chắn là cái file quan trọng đó đã được nhúng ở đâu đó trước đó chưa. Đảm bảo nó chỉ được nhúng một lần duy nhất.

Sample Code

```php
<?php

   $link = mysqli_connect('localhost', 'mysql_user', 'mys
   ql_password');
   if (!$link) {
       die('Not connected : ' . mysqli_error($link));
   }

   // make foo the current db
   $db_selected = mysqli_select_db($link,'foo');
   if (!$db_selected) {
       die ('Can\'t use foo : ' . mysqli_error($link));
   }
?>
```

Executing SQL Statements

Để gửi các câu lệnh SQL tới MySQL, mình dùng hàm mysqli_query().

- Use the mysqli_query() function to send SQL statements to MySQL
- The syntax for the mysqli_query() function is:

  mysqli_query(connection, *query);*

  Trong đó, connection là cái "mối liên lạc" với MySQL, còn query là cái câu lệnh SQL mà mình muốn "sai" MySQL làm.
- The mysqli_query() function returns one of three values: Hàm mysqli_query() sẽ trả về một trong ba giá trị sau:
  - For SQL statements that do not return results (CREATE DATABASE and CREATE TABLE statements) it returns a value of TRUE if the statement executes successfully

    Đối với mấy cái câu lệnh SQL mà không trả về kết quả (ví dụ như CREATE DATABASE, CREATE TABLE), nó sẽ trả về TRUE nếu chạy thành công.

Executing SQL Statements

Đối với mấy câu lệnh SQL mà có trả về kết quả (ví dụ như SELECT, SHOW), hàm mysqli_query() sẽ trả về một cái "result pointer".

- For SQL statements that return results (`SELECT` and `SHOW` statements) the `mysqli_query()` function returns a result pointer that represents the query results

Cái "result pointer" này là một kiểu biến đặc biệt, nó trỏ tới cái dòng hiện tại trong cái tập kết quả (resultset) mà mình vừa truy vấn được.

  - A **result pointer** is a special type of variable that refers to the currently selected row in a resultset
- The `mysqli_query()` function returns a value of `FALSE` for any SQL statements that fail, regardless of whether they return results

Còn nếu mà cái câu lệnh SQL nào đó bị "toang" (dù là có trả về kết quả hay không), thì hàm mysqli_query () sẽ trả về FALSE.

Cái hàm mysqli_query() nó sẽ trả về cho mày một cái "result pointer". Mày có thể hiểu nó như là một cái "con trỏ" hay một cái "thẻ đánh dấu" đang chỉ vào vị trí đầu tiên của cái bảng kết quả đó (thường là cái dòng đầu tiên).

Để lấy được dữ liệu thực tế từ cái bảng kết quả này, mày cần phải dùng thêm mấy cái hàm khác, ví dụ như:

mysqli_fetch_assoc(): Lấy dữ liệu của cái dòng mà "result pointer" đang chỉ tới dưới dạng một mảng kết hợp (associative array), tức là mình có thể truy cập dữ liệu bằng tên cột. Sau khi lấy xong, cái "result pointer" sẽ tự động được di chuyển xuống dòng tiếp theo.
mysqli_fetch_row(): Tương tự, nhưng nó lấy dữ liệu dưới dạng một mảng số (indexed array), tức là mình truy cập dữ liệu bằng số thứ tự của cột. "Result pointer" cũng sẽ tự động di chuyển.
Mày cứ tưởng tượng cái "result pointer" giống như là cái ngón tay của mày đang chỉ vào một dòng trong cái bảng kết quả. Mỗi lần mày dùng mấy cái hàm mysqli_fetch_*() là mày sẽ "đọc" được cái dòng mà ngón tay mày đang chỉ, rồi sau đó mày lại di chuyển ngón tay xuống dòng kế tiếp để đọc tiếp.

Sample Code

```php
<?php
  // This could be supplied by a user, for example
  $firstname = 'fred';
  $lastname  = 'fox';

  //never trust user data
 $firstname= mysqli_real_escape_string($firstname);
 $lastname= mysqli_real_escape_string($lastname);

 // Formulate Query
 // For more examples, see mysqli_real_escape_string()
 $query = "SELECT firstname, lastname, address, age FROM friends WHERE firstname='$firstname '
 AND lastname= '$lastname'";


 // Perform Query
 $result = mysqli_query($link, $query);

 // Check result
 // This shows the actual query sent to MySQL, and the error. Useful for debugging.
 if (!$result) {
     $message  = 'Invalid query: ' . mysqli_error() . "<br>";
     $message .= 'Whole query: ' . $query;
     die($message);
 }

 // Use result
 // Attempting to print $result won't allow access to information in the resource
 // One of the mysql result functions must be used
 // See also mysqli_fetch_array(), mysqli_fetch_row(), etc.
 while ($row = mysqli_fetch_assoc($result)) {
     echo $row['firstname'];
     echo $row['lastname'];
     echo $row['address'];
     echo $row['age'];
 }

 // Free the resources associated with the result set
 // This is done automatically at the end of the script
 mysqli_free_result($result);
?>
```

Adding, Deleting, and Updating Records

Để thêm dữ liệu vào bảng, dùng mấy cái từ khóa INSERT và VALUES kết hợp với hàm mysqli_query().

- To add records to a table, use the INSERT and VALUES keywords with the mysqli_query() function
- To add multiple records to a database, use the LOAD DATA statement with the name of the local text file containing the records you want to add

Để thêm nhiều dữ liệu cùng lúc, dùng câu lệnh LOAD DATA với tên của cái file text chứa dữ liệu đó.

- To update records in a table, use the UPDATE statement

Để sửa dữ liệu trong bảng, dùng câu lệnh UPDATE.

Adding, Deleting, and Updating Records

```
<?php
$con = mysqli_connect("localhost","demo","demo");
if (!$con)
  {
    die('Could not connect: ' . mysqli_error($con));
  }
mysqli_select_db($con, "demo");
mysqli_query($con, "INSERT INTO friends (FirstName,
LastName, Age) VALUES ('Lester', 'Longbottom', '35')");

mysqli_query($con,"INSERT INTO friends (FirstName,
LastName, Age) VALUES ('Carly', 'Sampson', '33')");

mysqli_close($con);
?>
```

Adding, Deleting, and Updating Records

- The UPDATE keyword specifies the name of the table to update
- The SET keyword specifies the value to assign to the fields in the records that match the condition in the WHERE clause
- To delete records in a table, use the DELETE statement with the mysqli_query() function
- Omit the WHERE clause to delete all records in a table

UPDATE: Cho biết cái bảng nào cần sửa.

SET: Cho biết là sửa cái cột nào và sửa thành giá trị nào.

WHERE: Cho biết là chỉ sửa những dòng nào thỏa mãn điều kiện gì đó. Nếu không có WHERE thì nó sẽ sửa hết tất cả các dòng trong bảng luôn đó.

Adding, Deleting, and Updating Records

```
?php
$con =
mysqli_connect("localhost","demo","demo");
if (!$con)
  {
    die('Could not connect: ' .
mysqli_error($con));
  }
mysqli_select_db($con,"demo");
mysqli_query($con,"UPDATE friends SET Age =
'61'
WHERE FirstName = 'Bill' AND LastName =
'Yeakus'");
mysqli_close($con);
?>
```

Retrieving Records into an Indexed Array

The mysqli_fetch_row() function returns the fields in the current row of a resultset into an indexed array and moves the result pointer to the next row

```
echo "<table border=1>";
echo "<tr><th>First</th><th>Last</th>
      <th>Address</th><th>age</th></tr>";
$Row = mysqli_fetch_row($result);
do {
    echo "<tr><td>{$Row[0]}</td>";
    echo "<td>{$Row[1]}</td>";
    echo "<td>{$Row[2]}</td>";
    echo "<td>{$Row[3]}</td></tr>";
    $Row = mysqli_fetch_row($result);
} while ($Row);
echo "</table>";
mysqli_close($con);
?>
```

Lấy cột 1, cột 2, cột 3... của hàng đang chọn
(mỗi lần mysqli_fetch_row() trong do_while sẽ là một dòng của bảng

Mỗi lần gọi mysqli_fetch_row() nó sẽ lấy dữ liệu của một dòng , rồi mình truy cập các cột bằng $Row[0], $Row[1], $Row[2], $Row[3] (tương ứng với First, Last, Address, age)

Lấy dữ liệu dưới dạng mảng số (Indexed Array):

Cái hàm mysqli_fetch_row() nó làm cái vụ này.
Nó sẽ trả về dữ liệu của cái dòng hiện tại trong kết quả truy vấn (resultset) dưới dạng một cái mảng mà mình truy cập bằng số thứ tự của cột (bắt đầu từ 0).
Sau khi lấy xong một dòng, nó sẽ tự động "nhảy" tới dòng tiếp theo.

Ví dụ trên:

Đoạn này nó truy vấn tất cả dữ liệu từ bảng "friends" (SELECT * FROM friends).
Rồi nó dùng vòng lặp do...while để lấy từng dòng dữ liệu bằng mysqli_fetch_row() và in ra thành một cái bảng HTML.
Mỗi lần gọi mysqli_fetch_row() nó sẽ lấy dữ liệu của một dòng, rồi mình truy cập các cột bằng $Row[0], $Row[1], $Row[2], $Row[3] (tương ứng với First, Last, Address, age).

Lấy dữ liệu dưới dạng mảng kết hợp (Associative Array):

Cái hàm mysqli_fetch_assoc() nó làm cái vụ này.
Thay vì trả về mảng số, nó trả về mảng kết hợp, tức là mình truy cập dữ liệu bằng tên cột.
Ví dụ: $row['firstname'], $row['lastname'],...
Đoạn code ví dụ thứ hai nó dùng mysqli_fetch_assoc() nè.

```
<?php
$con = mysqli_connect("localhost","demo","demo","demo");
if (!$con)
{
die('Could not connect: ' . mysqli_error($con));
}
$q = "SELECT * FROM friends";
$result = mysqli_query($con,$q);
echo "<table border=1>";
echo "<tr><th>First</th><th>Last</th> <th>Address</th><th>age</th></tr>";
while ($Row=mysqli_fetch_assoc($result)) {
echo "<tr><td>{$Row['firstname']}</td>";
echo "<td>{$Row['lastname']}</td>";
echo "<td>{$Row['address']}</td>";
echo "<td>{$Row['age']}</td></tr>";
}
echo "</table>";
mysqli_close($con);
?>
```

Sample Code

```
<?php
$con = mysqli_connect("localhost","demo","demo","demo");
if (!$con)
  {
    die('Could not connect: ' . mysqli_error($con));
  }
$q = "SELECT * FROM friends";
$result = mysqli_query($con,$q);
echo "<table border=1>";
echo "<tr><th>First</th><th>Last</th>
      <th>Address</th><th>age</th></tr>";
while ($Row=mysqli_fetch_assoc($result)) {
   echo "<tr><td>{$Row['firstname']}</td>";
   echo "<td>{$Row['lastname']}</td>";
   echo "<td>{$Row['address']}</td>";
   echo "<td>{$Row['age']}</td></tr>";
}
echo "</table>";
mysqli_close($con);
?>
```

Using the `mysqli_affected_rows()` Function

Khi mình chạy mấy cái truy vấn mà nó trả về kết quả (ví dụ như câu lệnh SELECT để lấy dữ liệu), thì mình dùng hàm mysqli_num_rows() để biết là có bao nhiêu dòng dữ liệu được trả về.

- With queries that return results (SELECT queries), use the mysqli_num_rows() function to find the number of records returned from the query
- With queries that modify tables but do not return results (INSERT, UPDATE, and DELETE queries), use the mysqli_affected_rows() function to determine the number of affected rows

Còn khi mình chạy mấy cái truy vấn mà nó thay đổi dữ liệu trong bảng nhưng không trả về kết quả (ví dụ như INSERT để thêm dữ liệu, UPDATE để sửa, DELETE để xóa), thì mình dùng hàm mysqli_affected_rows() để biết là có bao nhiêu dòng dữ liệu bị ảnh hưởng bởi cái truy vấn đó.

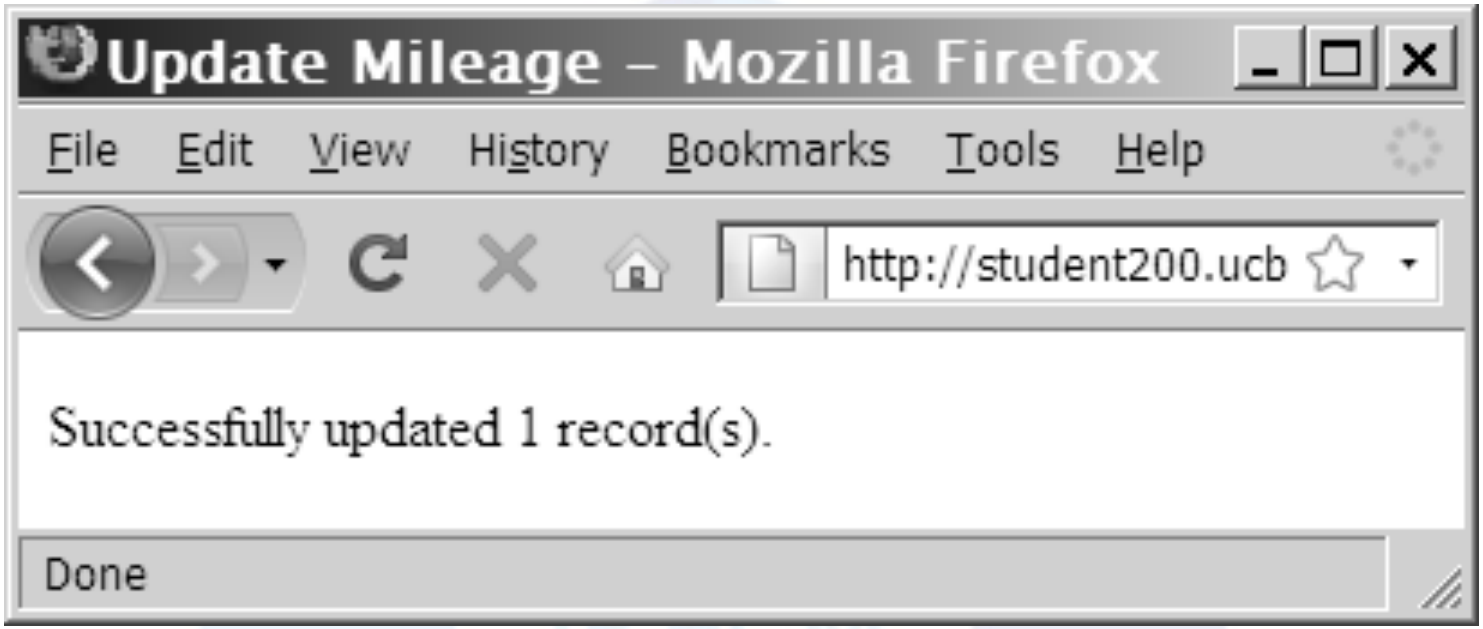
Using the `mysqli_affected_rows()` Function

```
$QueryResult = mysqli_query($con,"UPDATE friends SET
  Age = '67'
WHERE FirstName = 'Bill' AND LastName = 'Yeakus'");

if ($QueryResult === FALSE)
      echo "<p>Unable to execute the query.</p>"
     . "<p>Error code " . mysqli_errno($con)
     . ": " . mysqli_error($con) . "</p>";
else
    echo "<p>Successfully updated "
          . mysqli_affected_rows($con) . "
  record(s).</p>";
mysqli_close($con);
?>
```

Đoạn này nó thực hiện câu lệnh UPDATE để sửa tuổi của một người trong bảng "friends". Sau đó, nó dùng mysqli_affected_rows() để in ra số dòng đã được sửa. Nếu câu lệnh UPDATE thành công và có 1 dòng được sửa thì nó sẽ in ra "Successfully updated 1 record(s).", còn nếu không có dòng nào thỏa mãn điều kiện WHERE thì nó sẽ in ra "Successfully updated 0 record(s).".

Using the `mysqli_affected_rows()` Function



Output of `mysqli_affected_rows()` function for an `UPDATE` query

Using the `mysqli_info()` Function

Cái hàm này nó cũng cho mình biết thông tin về mấy cái truy vấn, nhưng mà nó hơi khác với mysqli_affected_rows() một chút.

- For queries that add or update records, or alter a table's structure, use the mysqli_info() function to return information about the query
- The mysqli_info() function returns the number of operations for various types of actions, depending on the type of query
- The mysqli_info() function returns information about the last query that was executed on the database connection

mysqli_info() chủ yếu dùng cho mấy cái truy vấn mà nó thêm, sửa dữ liệu, hoặc thay đổi cấu trúc của bảng. Nó trả về thông tin chi tiết hơn về các thao tác đã được thực hiện.

Hàm này sẽ trả về số lượng các hành động khác nhau, tùy thuộc vào loại truy vấn.

Nó chỉ trả về thông tin về cái truy vấn cuối cùng đã được thực hiện trên kết nối database.

Using the `mysqli_info()` Function

- **The mysqli_info() function returns information about queries that match one of the following formats:**
  - `INSERT INTO...SELECT...`
  - `INSERT INTO...VALUES (...),(...),(...)`
  - `LOAD DATA INFILE ...`
  - `ALTER TABLE ...`
  - `UPDATE`
- **For any queries that do not match one of these formats, the mysqli_info() function returns an empty string**

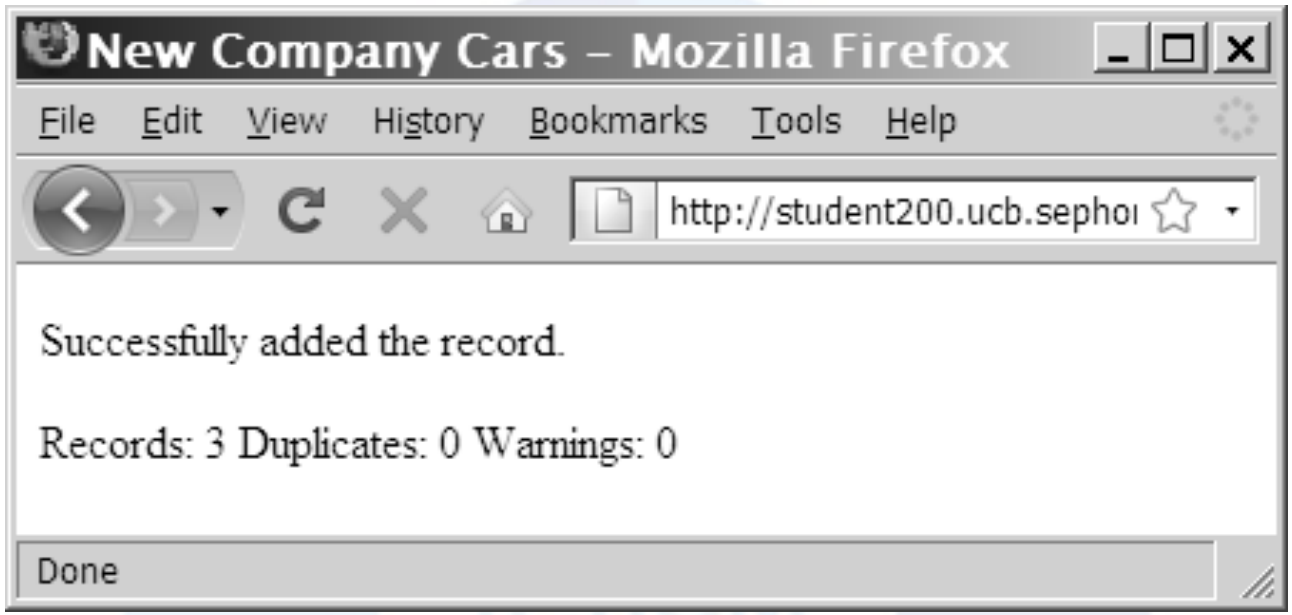
Nếu mà cái truy vấn của mày không thuộc mấy cái dạng trên thì mysqli_info() sẽ trả về một chuỗi rỗng.

Using the `mysqli_info()` Function

```
$SQLstring = "INSERT INTO company_cars " .
     " (license, model_year, make, model, mileage) " .
     " VALUES " .
     " ('CPQ-894', 2011, 'Honda', 'Insight', 49.2), " .
     " ('CPQ-895', 2011, 'Honda', 'Insight', 17.9), " .
     " ('CPQ-896', 2011, 'Honda', 'Insight', 22.6)";
$QueryResult = @mysqli_query($DBConnect,$SQLstring);
if ($QueryResult === FALSE)
      echo "<p>Unable to execute the query.</p>"
     . "<p>Error code " . mysqli_errno($DBConnect)
     . ": " . mysqli_error($DBConnect) . "</p>";
else {
     echo "<p>Successfully added the record.</p>";
     echo "<p>" . mysqli_info($DBConnect) . "</p>";
}
```

Đoạn này nó thêm nhiều dòng dữ liệu vào bảng "company_cars" bằng một câu lệnh INSERT. Sau đó, nó dùng mysqli_info() để lấy thông tin về cái truy vấn này, ví dụ như số dòng đã được thêm vào.

Using the `mysqli_info()` Function



Output of `mysqli_info()` function for an `INSERT` query that adds multiple records

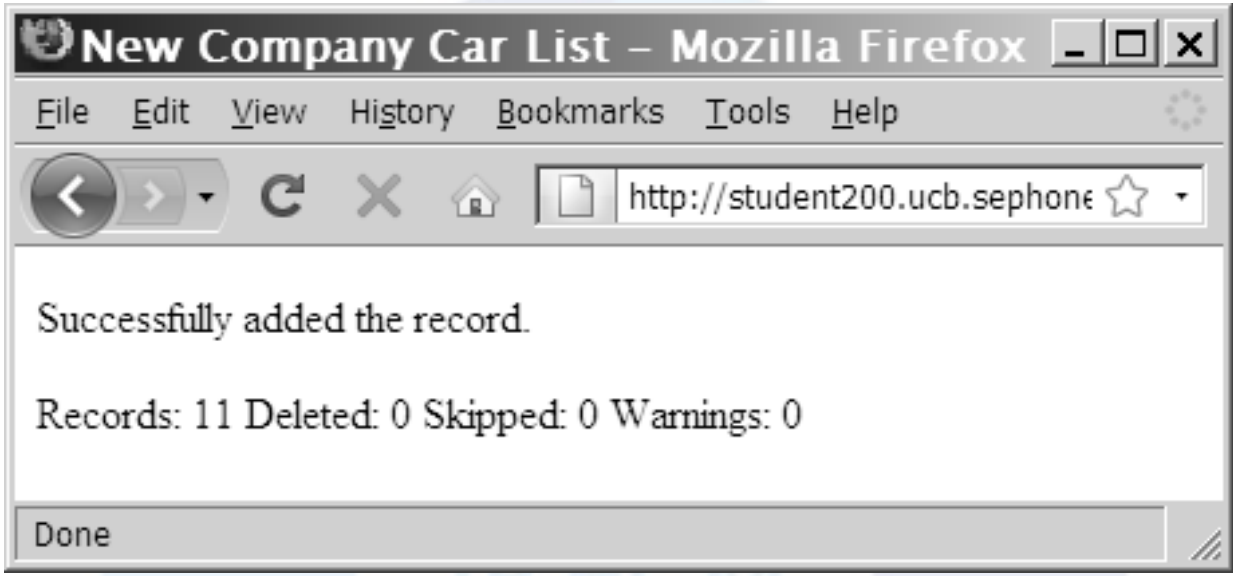
Using the `mysqli_info()` Function

- **The `mysqli_info()` function also returns information for LOAD DATA queries**

```
$SQLstring = "LOAD DATA INFILE 'company_cars.txt'
     INTO TABLE company_cars;";
$QueryResult = @mysqli_query($SQLstring, $DBConnect);
if ($QueryResult === FALSE)
      echo "<p>Unable to execute the query.</p>"
     . "<p>Error code " . mysqli_errno($DBConnect)
     . ": " . mysqli_error($DBConnect) . "</p>";
else {
     echo "<p>Successfully added the record.</p>";
     echo "<p>" . mysqli_info($DBConnect) . "</p>";
}
```

Đoạn này nó dùng câu lệnh LOAD DATA INFILE để đọc dữ liệu từ một file text (company_cars.txt) và thêm vào bảng company_cars. Sau đó, nó cũng dùng mysqli_info() để lấy thông tin về cái truy vấn LOAD DATA INFILE này.

Using the `mysqli_info()` Function



Output of `mysqli_info()` function for a `LOAD DATA` query

Working with Query Results

Function	Description
`mysql_data_seek($Result, position)`	Moves the result pointer to a specified row in the resultset
`mysql_fetch_array($Result, MYSQL_ASSOC | MYSQL_NUM | MYSQL_BOTH)`	Returns the fields in the current row of a resultset into an indexed array, associative array, or both, and moves the result pointer to the next row
`mysql_fetch_assoc($Result)`	Returns the fields in the current row of a resultset into an associative array and moves the result pointer to the next row
`mysql_fetch_lengths($Result)`	Returns the field lengths for the current row in a resultset into an indexed array
`mysql_fetch_row($Result)`	Returns the fields in the current row of a resultset into an indexed array and moves the result pointer to the next row

Table 8-2 Common PHP functions for accessing database results

mysqli_data_seek($Result, position): Hàm này giúp mình di chuyển cái "result pointer" tới một dòng cụ thể nào đó trong cái bảng kết quả. Ví dụ, nếu mày muốn quay lại dòng đầu tiên hoặc nhảy tới dòng thứ 5 thì dùng hàm này.
mysqli_fetch_array($Result, MYSQL_ASSOC | MYSQL_NUM | MYSQL_BOTH): Hàm này lấy dữ liệu của cái dòng hiện tại mà "result pointer" đang chỉ tới và trả về dưới dạng một mảng.
Nếu mày dùng MYSQL_ASSOC, nó sẽ trả về mảng kết hợp (key là tên cột).
Nếu mày dùng MYSQL_NUM, nó sẽ trả về mảng số (key là số thứ tự của cột, bắt đầu từ 0).
Nếu mày dùng MYSQL_BOTH, nó sẽ trả về cả hai loại mảng trên. Sau khi lấy xong, "result pointer" sẽ tự động chuyển xuống dòng kế tiếp.
mysqli_fetch_assoc($Result): Hàm này giống như mysqli_fetch_array() nhưng nó chỉ trả về mảng kết hợp thôi.
mysqli_fetch_lengths($Result): Hàm này trả về độ dài của từng trường (từng cột) trong cái dòng hiện tại dưới dạng một mảng số.
mysqli_fetch_row($Result): Hàm này cũng giống như mysqli_fetch_array() nhưng nó chỉ trả về mảng số thôi. Sau khi lấy xong, "result pointer" cũng sẽ tự động chuyển xuống dòng kế tiếp.

Retrieving Records into an Indexed Array

- The mysqli_fetch_row() function returns the fields in the current row of a result set into an indexed array and moves the result pointer to the next row

Hàm mysqli_fetch_row() sẽ trả về các trường (các cột) của dòng hiện tại trong kết quả truy vấn (resultset) thành một mảng số, và nó cũng tự động di chuyển cái "result pointer" tới dòng tiếp theo.

Mảng số ở đây có nghĩa là mình sẽ truy cập các giá trị trong mảng bằng số thứ tự của cột, bắt đầu từ 0. Ví dụ: $row[0] sẽ lấy giá trị của cột đầu tiên, $row[1] lấy giá trị cột thứ hai, v.v.

Retrieving Records into an Indexed Array

```
$q = "SELECT * FROM friends";
$result = mysqli_query($con,$q);

echo "<table border=1>";
echo "<tr><th>First</th><th>Last</th>
      <th>Address</th><th>age</th></tr>";
$Row = mysqli_fetch_row($result);
do {
   echo "<tr><td>{$Row[0]}</td>";
   echo "<td>{$Row[1]}</td>";
   echo "<td>{$Row[2]}</td>";
   echo "<td>{$Row[3]}</td></tr>";
   $Row = mysqli_fetch_row($result);
} while ($Row);
echo "</table>";
mysqli_close($con);
```

$q = "SELECT * FROM friends";: Câu truy vấn SQL này lấy tất cả các cột và tất cả các dòng từ bảng "friends".

$result = mysqli_query($con,$q);: Thực hiện câu truy vấn và lưu kết quả vào biến $result. Đoạn code sau đó tạo một bảng HTML để hiển thị dữ liệu.

$Row = mysqli_fetch_row($result);: Lấy dòng đầu tiên từ kết quả truy vấn và lưu vào biến $ Row.

Vòng lặp do...while: Lặp qua từng dòng của kết quả truy vấn.

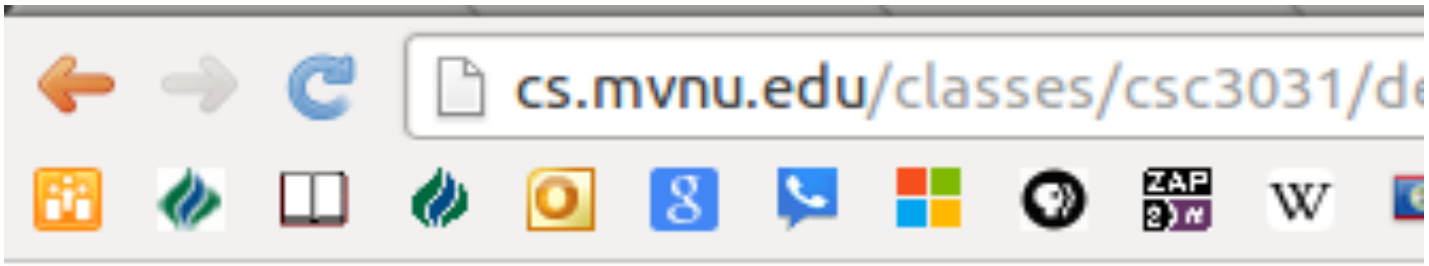
echo "<tr><td>{$Row[0]}</td>"; ... echo "<td>{$ Row[3]}</td></tr>";: In ra các giá trị của từng cột trong dòng hiện tại vào các ô <td> của bảng HTML. $Row[0] tương ứng với cột "First", $Row [1] với "Last", $Row[2] với "Address", và $Row[3] với "age".

$Row = mysqli_fetch_row($result);: Lấy dòng tiếp theo từ kết quả truy vấn. Vòng lặp tiếp tục cho đến khi không còn dòng nào để lấy.

echo "</table>";: Kết thúc việc tạo bảng HTML.

mysqli_close($con);: Đóng kết nối đến MySQL server.

Retrieving Records into an Indexed Array



First	Last	Address	age
Bill	Yeakus	123 Pine Street	67
Nancy	Bellwig	335 Snelling Ave	43
Lester	Longbottom		35
Carly	Sampson		33

Retrieving Records into an Associative Array

- The `mysqli_fetch_assoc()` function returns the fields in the current row of a resultset into an associative array and moves the result pointer to the next row
- The difference between `mysqli_fetch_assoc()` and `mysqli_fetch_row()` is that instead of returning the fields into an indexed array, the `mysqli_fetch_assoc()` function returns the fields into an associate array and uses each field name as the array key

mysqli_fetch_assoc(): Hàm này nó cũng lấy dữ liệu từ cái dòng hiện tại trong resultset, nhưng thay vì trả về mảng số, nó trả về mảng kết hợp.

Điểm khác biệt giữa mysqli_fetch_assoc() và mysqli_fetch_row() là ở chỗ đó đó. Thay vì dùng số thứ tự cột làm key của mảng, mysqli_fetch_assoc() dùng chính cái tên cột làm key luôn.
Ví dụ, để lấy giá trị của cột "firstname", mày sẽ dùng $row['firstname'] thay vì $ row[0] như khi dùng mysqli_fetch_row().

Closing Query Results

- When you are finished working with query results retrieved with the mysqli_query() function, use the mysqli_free_result() function to close the resultset
- To close the resultset, pass to the mysqli_free_result() function the variable containing the result pointer from the mysqli_query() function

Khi làm việc xong với kết quả truy vấn lấy về từ mysqli_query(), mình nên dùng hàm mysqli_free_result() để đóng cái resultset lại.

Để đóng resultset, mình chỉ cần truyền cái biến chứa result pointer (cái mà mysqli_query() trả về) vào hàm mysqli_free_result() là xong.

Accessing Query Result Information

- The mysqli_num_rows() function returns the number of rows in a query result
- The mysqli_num_fields() function returns the number of fields in a query result
- Both functions accept a database connection variable as an argument

mysqli_num_rows(): Hàm này trả về số lượng dòng (rows) trong kết quả truy vấn. Ví dụ, nếu mày chạy một câu lệnh SELECT và nó trả về 10 dòng dữ liệu, thì hàm này sẽ trả về số 10.

mysqli_num_fields(): Hàm này trả về số lượng cột (fields) trong kết quả truy vấn. Ví dụ, nếu câu lệnh SELECT của mày lấy ra 4 cột dữ liệu (ví dụ: firstname, lastname, address, age), thì hàm này sẽ trả về số 4.

Cả hai hàm này đều cần một biến kết nối database làm tham số để nó biết mình đang muốn lấy thông tin từ cái kết quả truy vấn nào.

Accessing Query Result Information

```
$SQLstring = "SELECT * FROM company_cars";
$QueryResult = @mysqli_query($DBConnect$, SQLstring);
if ($QueryResult === FALSE)
      echo "<p>Unable to execute the query.</p>"
     . "<p>Error code " . mysqli_errno($DBConnect)
     . ": " . mysqli_error($DBConnect) . "</p>";
else
     echo "<p>Successfully executed the query.</p>";
$NumRows = mysqli_num_rows($QueryResult);
$NumFields = mysqli_num_fields($QueryResult);
if ($NumRows != 0 && $NumFields != 0)
     echo "<p>Your query returned " .
   mysqli_num_rows($QueryResult) . " rows and "
     . mysqli_num_fields($QueryResult) . " fields.</p>";
else
     echo "<p>Your query returned no results.</p>";
mysqli_close($DBConnect);
```

Accessing Query Result Information

Output of the number of rows and fields returned from a query

Kết nối và đóng kết nối:
Hàm mysqli_connect() dùng để mở kết nối tới máy chủ MySQL.
Hàm mysqli_close() dùng để đóng kết nối.
Hàm mysqli_errno() trả về mã lỗi của thao tác MySQL gần nhất (nếu có).
Hàm mysqli_error() trả về thông báo lỗi chi tiết.
Dấu @ (error control operator) dùng để tắt hiển thị thông báo lỗi.
Quản lý Database:
Hàm mysqli_create_db() dùng để tạo một database mới.
Hàm mysqli_select_db() dùng để chọn một database để làm việc.
Hàm mysqli_drop_db() dùng để xóa một database.
Thực thi câu lệnh SQL:
Hàm mysqli_query() dùng để gửi các câu lệnh SQL tới MySQL.
"Result pointer" là một biến đặc biệt trỏ tới dòng dữ liệu hiện tại trong kết quả truy vấn.
Tạo và xóa bảng:
Câu lệnh CREATE TABLE (thường dùng với mysqli_query()) để tạo một bảng mới.
PRIMARY KEY clause chỉ định một hoặc nhiều trường làm khóa chính cho bảng.
AUTO_INCREMENT clause tạo một trường tự động tăng giá trị.
NOT NULL clause tạo một trường bắt buộc phải có dữ liệu.
Câu lệnh DROP TABLE (thường dùng với mysqli_query()) để xóa một bảng.
Thao tác với dữ liệu (Records):
Câu lệnh LOAD DATA (thường dùng với mysqli_query()) để thêm nhiều bản ghi từ một file text.
Câu lệnh UPDATE (thường dùng với mysqli_query()) để cập nhật các bản ghi.
Câu lệnh DELETE (thường dùng với mysqli_query()) để xóa các bản ghi.
Hàm mysqli_info() trả về thông tin về các thao tác (ví dụ: số dòng được thêm, sửa) tùy thuộc vào loại truy vấn.
Lấy dữ liệu từ kết quả truy vấn:
Hàm mysqli_fetch_row() trả về dữ liệu của dòng hiện tại dưới dạng mảng số (indexed array).
Hàm mysqli_fetch_assoc() trả về dữ liệu của dòng hiện tại dưới dạng mảng kết hợp (associative array).
Hàm mysqli_free_result() dùng để giải phóng bộ nhớ sau khi làm việc xong với kết quả truy vấn.
Thông tin về kết quả truy vấn:
Hàm mysqli_num_rows() trả về số lượng dòng trong kết quả truy vấn.
Hàm mysqli_num_fields() trả về số lượng cột trong kết quả truy vấn.
Với các truy vấn SELECT, mysqli_num_rows() cho biết số lượng bản ghi được trả về.

Summary

- The `mysqli_connect()` function opens a connection to a MySQL database server
- The `mysqli_close()` function closes a database connection
- The `mysqli_errno()` function returns the error code from the last attempted MySQL function call or zero if no error occurred

Summary (continued)

- The `mysqli_error()` function returns the error message from the last attempted MySQL function call or an empty string if no error occurred
- The error control operator (@) suppresses error messages
- You use the `mysqli_create_db()` function to create a new database
- The `mysqli_select_db()` function selects a database

Summary (continued)

- You use the `mysqli_drop_db()` function to delete a database
- The `mysqli_query()` function sends SQL statements to MySQL
- A result pointer is a special type of variable that refers to the currently selected row in a resultset
- You use the CREATE TABLE statement with the `mysqli_query()` function to create a table

Summary (continued)

- The PRIMARY KEY clause indicates a field or fields that will be used as a referential index for the table
- The AUTO_INCREMENT clause creates a field that is automatically updated with the next sequential value for that column
- The NOT NULL clause creates a field that must contain data
- You use the DROP TABLE statement with the `mysqli_query()` function to delete a table

Summary (continued)

- You use the LOAD DATA statement and the `mysqli_query()` function with a local text file to add multiple records to a database – MAY NOT WORK ON PARADOX
- You use the UPDATE statement with the `mysqli_query()` function to update records in a table
- You use the DELETE statement with the `mysqli_query()` function to delete records from a table

Summary (continued)

- The `mysqli_info()` function returns the number of operations for various types of actions, depending on the type of query.
- The `mysqli_fetch_row()` function returns the fields in the current row of a resultset into an indexed array and moves the result pointer to the next row.

Summary (continued)

- The `mysqli_fetch_assoc()` function returns the fields in the current row of a resultset into an associative array and moves the result pointer to the next row
- The `mysqli_free_result()` function closes a resultset

Summary (continued)

- The `mysqli_num_rows()` function returns the number of rows in a query result, and the `mysqli_num_fields()` function returns the number of fields in a query result
- With queries that return results, such as SELECT queries, you can use the `mysqli_num_rows()` function to find the number of records returned from the query

Creating and Deleting Tables

- Use the CREATE TABLE statement with the mysqli_query() function to create a new table
- Use the mysqli_select_db() function before executing the CREATE TABLE statement to verify that you are in the right database

Tạo bảng:

Để tạo một bảng mới, mình dùng câu lệnh CREATE TABLE kết hợp với hàm mysqli_query() của PHP.

Trước khi chạy câu lệnh CREATE TABLE, nên dùng hàm mysqli_select_db() để đảm bảo là mình đang ở đúng cái database cần tạo bảng.

Để tránh việc cố gắng tạo lại một bảng đã tồn tại, mình có thể dùng lệnh SHOW TABLES LIKE để kiểm tra xem bảng đó đã có chưa.

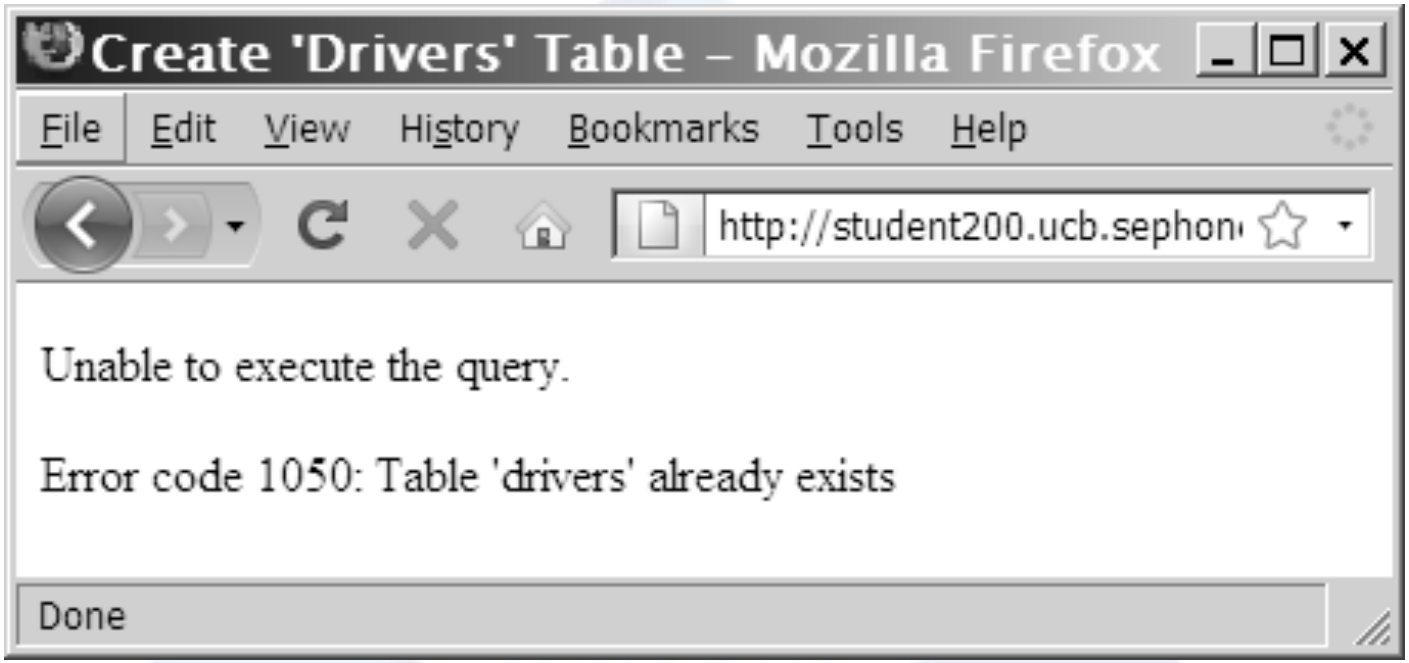
Nếu bảng chưa tồn tại, hàm mysqli_num_rows() sẽ trả về giá trị 0.

Khi tạo bảng, mình có thể chỉ định cột nào là khóa chính (primary key) bằng cách dùng từ khóa PRIMARY KEY.

Mình cũng có thể dùng từ khóa AUTO_INCREMENT để tạo một cột mà giá trị của nó sẽ tự động tăng lên cho mỗi dòng mới.

Từ khóa NOT NULL dùng để chỉ định một cột là bắt buộc phải có giá trị.

Creating and Deleting Tables



Error code and message that displays when you attempt to create a table that already exists

Creating and Deleting Tables

- Use the SHOW TABLES LIKE command to prevent code from trying to create a table that already exists.
- If the table does not exist, the mysqli_num_rows()function will return a value of 0 rows

```
$TableName = "subscribers";
$SQLstring = "SHOW TABLES LIKE
'$TableName'";
$QueryResult = @mysqli_query($DBConnect,
$SQLstring);
```

Creating and Deleting Tables

- To identify a field as a primary key in MySQL, include the PRIMARY KEY keywords when you define a field with the CREATE TABLE statement
- The AUTO_INCREMENT keyword is often used with a primary key to generate a unique ID for each new row in a table
- The NOT NULL keywords are often used with primary keys to require that a field include a value

Creating and Deleting Tables

- To delete a table, use the DROP TABLE statement with the mysqli_query() function

Xóa bảng:

Để xóa một bảng, mình dùng câu lệnh DROP TABLE kết hợp với hàm mysqli_query().

Creating a Database

- Use the mysqli_create_db() function to create a new database
- The basic syntax for the mysqli_create_db() is:

  $result = mysqli_create_db(*connection,* "*dbname*");
- The mysqli_create_db() returns a Boolean TRUE if successful or FALSE if there was an error
- In most cases we will use mysql monitor, PhpMyAdmin or Workbench to create databases.

Tạo database:

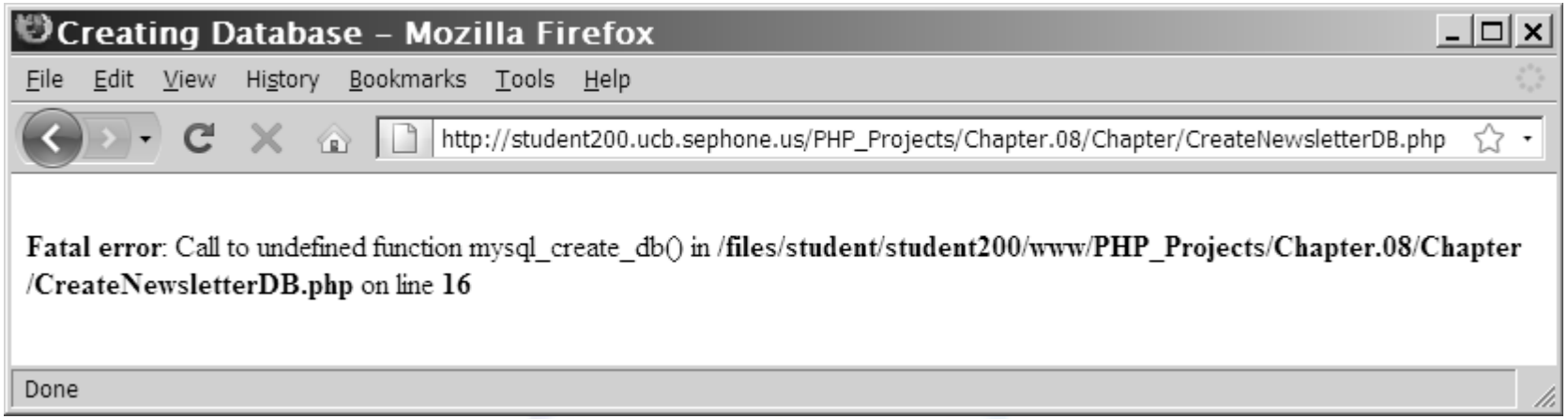
Để tạo một database mới, mình dùng hàm mysqli_create_db().

Cú pháp cơ bản của hàm mysqli_create_db() là: $result = mysqli_create_db(connection, " dbname");.

Hàm mysqli_create_db() trả về TRUE nếu tạo thành công, hoặc FALSE nếu có lỗi.

Thông thường, người ta sẽ dùng các công cụ như mysql monitor, PhpMyAdmin hoặc Workbench để tạo database thay vì dùng hàm này.

Creating a Database (continued)



Error message when the `mysqli_create_db()` function is unavailable because of insufficient privileges

Deleting a Database

- To delete a database, use the mysqli_drop_db() function.
- The format for the mysqli_drop_db() function is:

  $Result = mysqli_drop_db($connection, "dbname");
- The function returns a value of TRUE if it successfully drops a database or FALSE if it does not

Xóa database:

Để xóa một database, mình dùng hàm mysqli_drop_db().

Cú pháp của hàm mysqli_drop_db() là: $Result = mysqli_drop_db($connection, "dbname ");.

Hàm này trả về TRUE nếu xóa thành công, hoặc FALSE nếu không thành công.

Nguồn và nội dung liên quan

Tài Liệu Tham Khảo

- [1] Stepp,Miller,Kirst. Web Programming Step by Step.(1st Edition, 2009) Companion Website: http://www.webstepbook.com/
- [2] W3Schools, http://www.w3schools.com/html/default.asp