Chapter 5
PHP and MySQL

Lectured by:
Nguyễn Hữu Hiếu

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 compliant
- PHP includes functionality that allows you to work directly with different types of databases, without going through 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.

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

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.

Opening and Closing a Connection

- The database connection is assigned to the $DBConnect variable

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

- Close a database connection using the mysql_close() function

```
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

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

Reporting MySQL Errors

- The mysqli_errno() function returns the error code from the last attempted MySQL function call or 0 if no error occurred
- The mysqli_error() — Returns the text of the error message from previous MySQL operation
- The mysqli_errno() and mysqli_error() functions return the results of the previous mysqli*() function

Selecting a Database

- The syntax for the mysqli_select_db() function is:

  mysqli_select_db(*connection*, *database);*
- The function returns a value of TRUE if it successfully selects a database or FALSE if it does not
- For security purposes, you may choose to use an include file to connect to the MySQL server and select a database

Sample Code

good

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

bad

```
mysqli_select_db($link, "nonexistentdb",
echo mysql_errno($link) . ": " .
mysql_error($link). "<br>";
```

good

```
mysqli_select_db( $link, "demo");
```

bad

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

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);
```

Sample Code

```
<?php

  $link = mysqli_connect('localhost', 'mysql_us
  er', 'mysql_password');
  if (!$link) {
      die('Not connected : ' . mysqli_error($li
  nk));
  }

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


15

Executing SQL Statements

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

  mysqli_query(connection, *query);*
- The mysqli_query() function returns one of three values:
  - 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

Executing SQL Statements

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

Sample Code

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

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

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


 // Perform Query
 $result = mysql_query($query);

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

 // Use result
 // Attempting to print $result won't allow access to information in the resource
 // One of the mysql result functions must be used
 // See also mysql_fetch_array(), mysql_fetch_row(), etc.
 while ($row = mysql_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
 mysql_free_result($result);
?>
```

Adding, Deleting, and Updating Records

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

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 ('Les\
ter', 'Longbottom', '35')");

mysqli_query($con,"INSERT INTO friends (FirstName,
LastName, Age) VALUES ('Carl\
y', '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

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 mysql_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);
?>
```

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

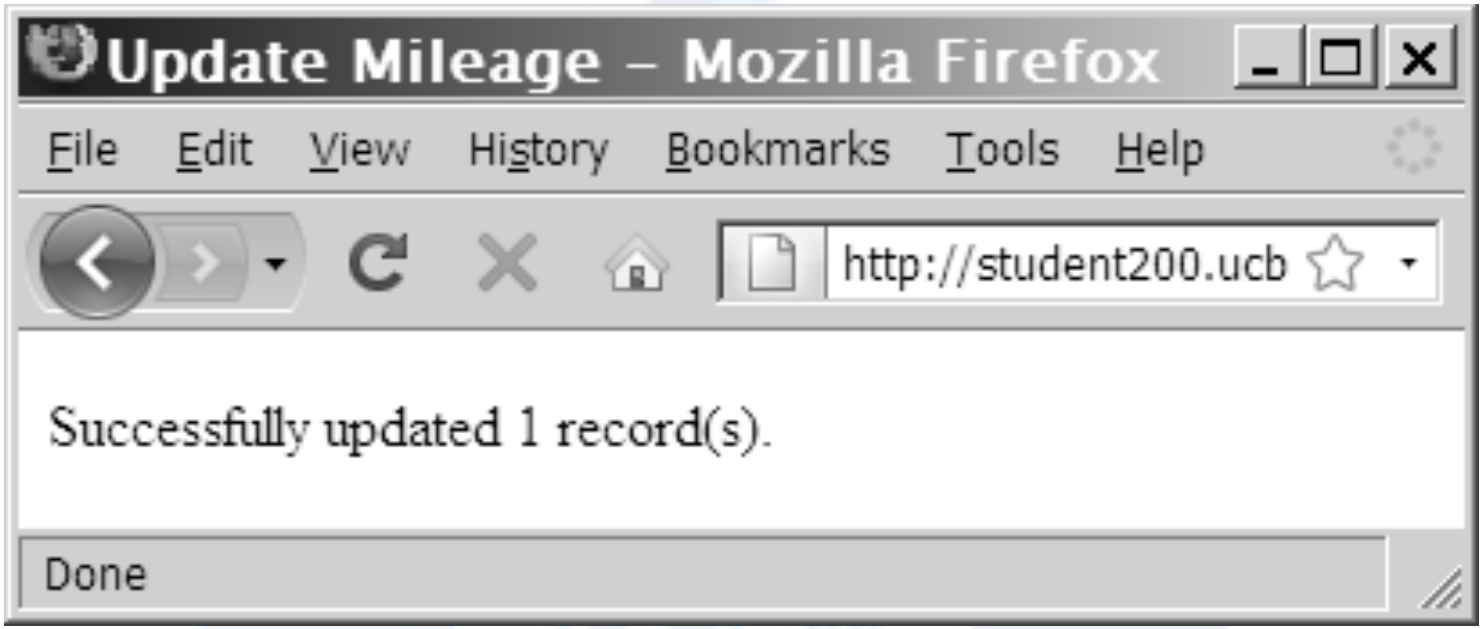
- With queries that return results (SELECT queries), use the mysqli_num_rows() function to find the number of records returned from the query
- With queries that modify tables but do not return results (INSERT, UPDATE, and DELETE queries), use the mysqli_affected_rows() function to determine the number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

Using the `mysql_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>";
mysql_close($con);
?>
```

Using the `mysql_affected_rows()` Function



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

Using the `mysql_info()` Function

- 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

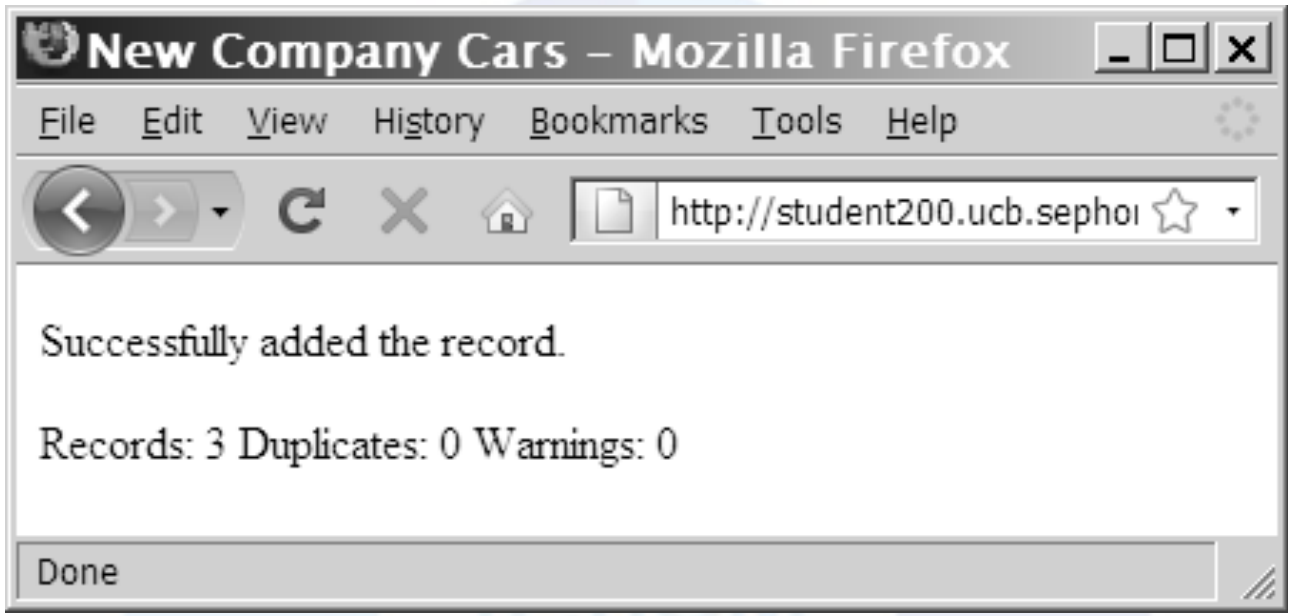
Using the `mysqli_info()` Function

- **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 mysql_info() function returns an empty string**

Using the `mysql_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 " . mysql_errno($DBConnect)
     . ": " . mysqli_error($DBConnect) . "</p>";
else {
     echo "<p>Successfully added the record.</p>";
     echo "<p>" . mysqli_info($DBConnect) . "</p>";
}
```

Using the `mysql_info()` Function



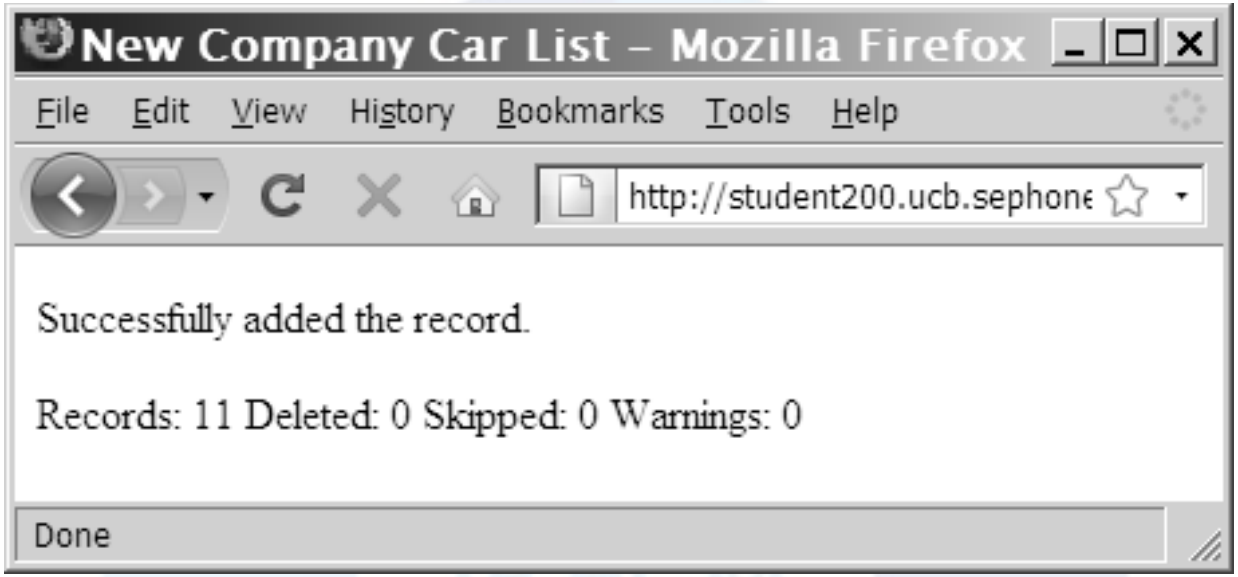
Output of `mysqli_info()` function for an `INSERT` query that adds multiple records

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>";
}
```

Using the `mysql_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

Retrieving Records into an Indexed Array

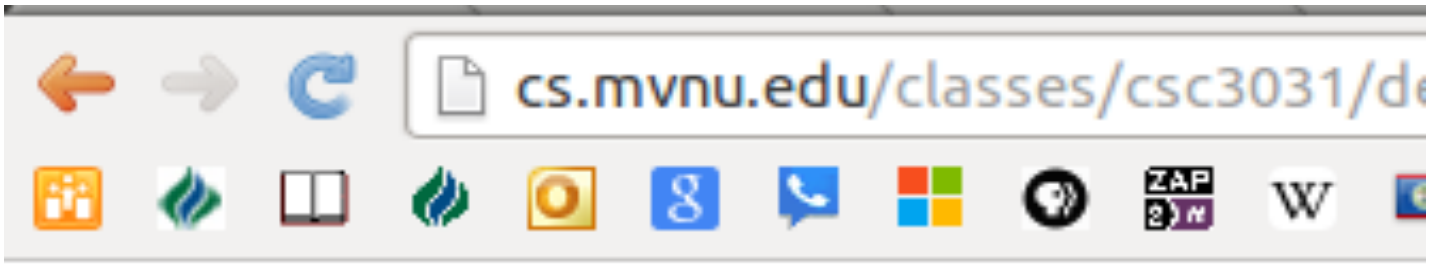
- The mysqli_fetch_row() function returns the fields in the current row of a result set into an indexed array and moves the result pointer to the next row

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>";
   $Row = mysql_fetch_row($result);
} while ($Row);
echo "</table>";
mysqli_close($con);
```

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

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

39

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

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

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

46

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

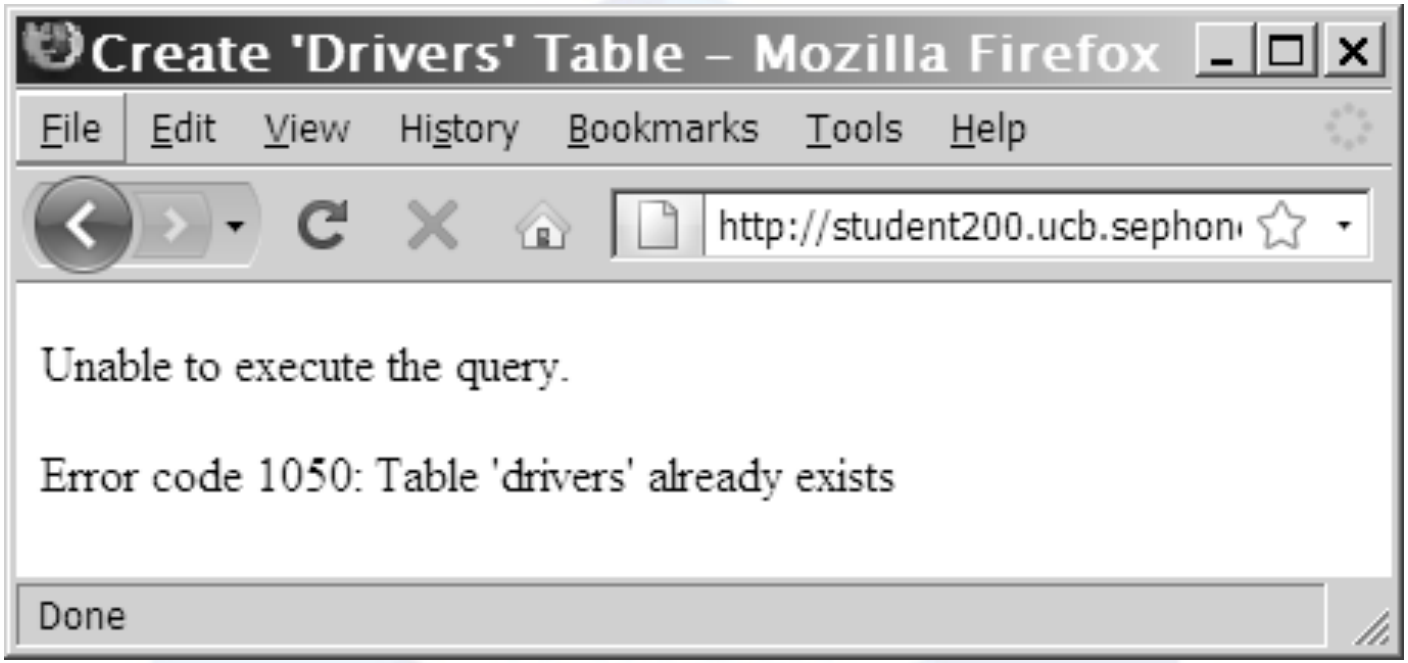
Summary (continued)

- 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

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

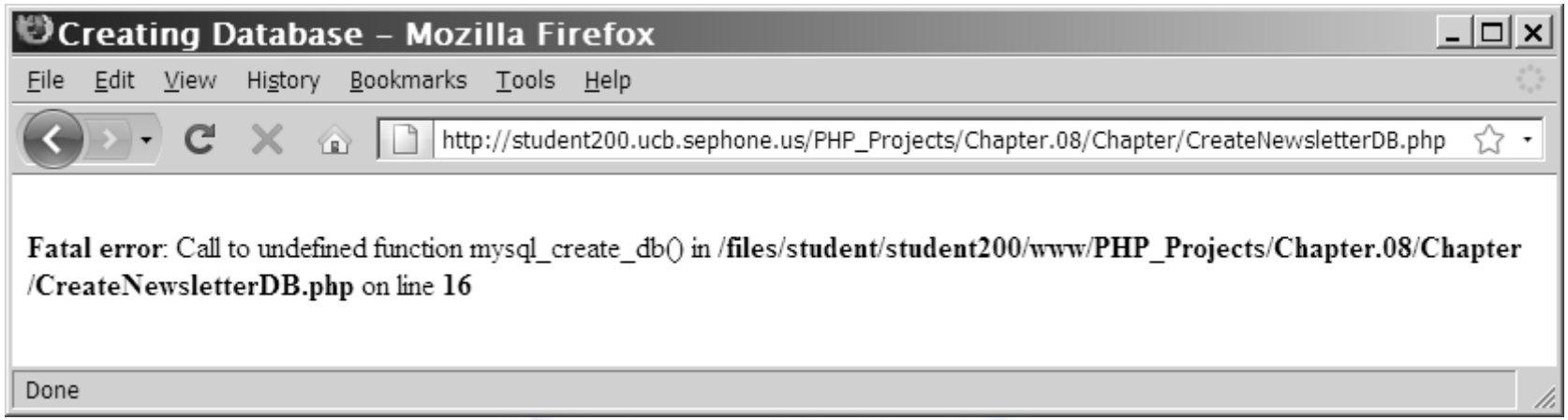
Creating a Database

- 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.

56

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 mysql_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

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