HIT6323/3323 – Web Programming

Module 08 – MySQL Databases with PHP

Delivered by Dr. Jinjun Chen
(jchen@ict.swin.edu.au)

Faculty of ICT

Outline

Manipulating MySQL Databases with PHP

- Connect to MySQL from PHP
- Learn how to handle MySQL errors
- Execute SQL statements with PHP
- Use PHP to work with MySQL databases and tables
- Use PHP to manipulate database records

Reading: Textbook Chapter 9

PHP Overview

- 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
- PHP supports SQLite, database abstraction layer functions, and PEAR DB
  - PEAR stands for PHP Extension and Application Repository, a library of open source PHP code. One of the most popular code modules is PEAR DB. PEAR DB performs similar functions as ODBC, but designed specifically to work with PHP.

Opening and Closing a MySQL 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 MySQL Connection (continued)

- The syntax for the `mysqli_connect()` function is:

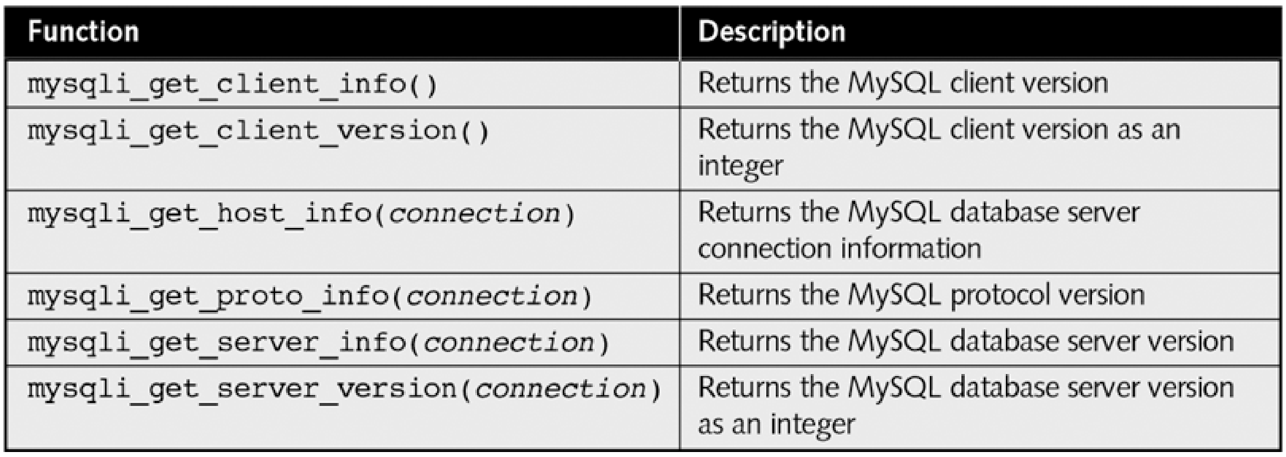
```
$connection = mysqli_connect("host"[, "user ", "password", "database"])
```

- 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
- The *database* argument selects a database with which to work
- Close a connection to a MySQL database server with the `mysqli_close()` function

```
mysqli_close($connection);
```

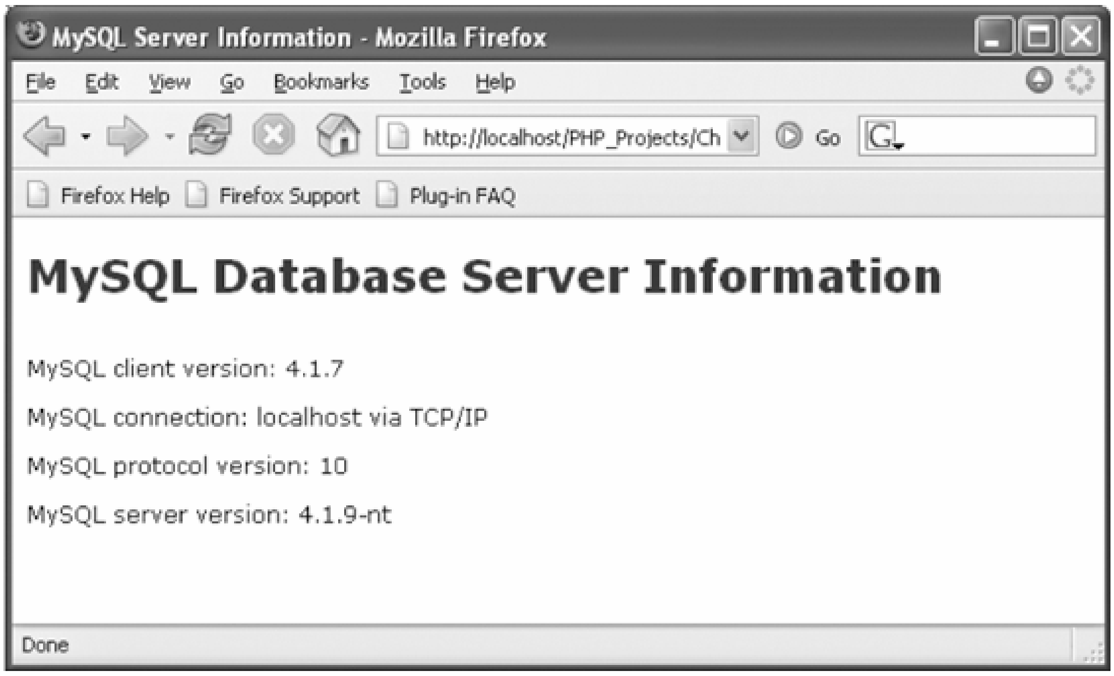
Opening and Closing a MySQL Connection (continued)

Table 9-1 MySQL server information functions



Function	Description
`mysqli_get_client_info()`	Returns the MySQL client version
`mysqli_get_client_version()`	Returns the MySQL client version as an integer
`mysqli_get_host_info(`*`connection`*`)`	Returns the MySQL database server connection information
`mysqli_get_proto_info(`*`connection`*`)`	Returns the MySQL protocol version
`mysqli_get_server_info(`*`connection`*`)`	Returns the MySQL database server version
`mysqli_get_server_version(`*`connection`*`)`	Returns the MySQL database server version as an integer

Opening and Closing a MySQL Connection (continued)


MySQL Server Information - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://localhost/PHP_Projects/Ch
Go
Firefox Help
Firefox Support
Plug-in FAQ
Done

MySQL Database Server Information

MySQL client version: 4.1.7

MySQL connection: localhost via TCP/IP

MySQL protocol version: 10

MySQL server version: 4.1.9-nt

Figure 9-1 MySQLInfo.php in a Web browser

Selecting a Database

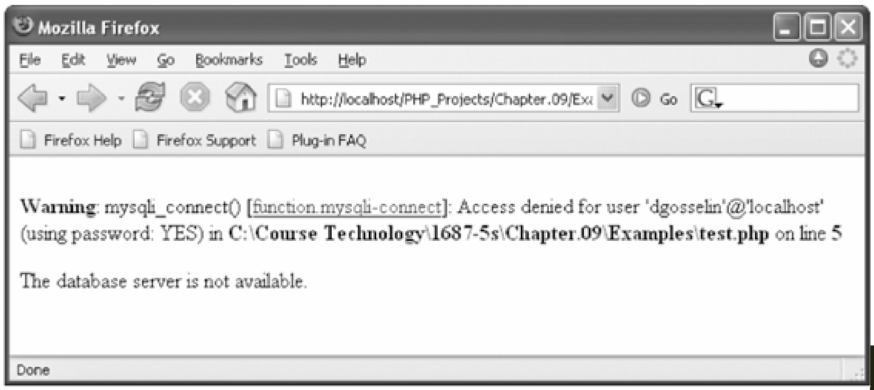
- Select a database with the `use` *`database`* statement when you log on to the MySQL Monitor
- The syntax for the `mysqli_select_db()` function is:

  mysqli_select_db(*connection*, *database*)
- The function returns a value of true if it successfully selects a database or false if it does not

Handling 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

  e.g. `if (!$DBConnect) ...`


Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://localhost/PHP_Projects/Chapter.09/Exa
Go
Firefox Help
Firefox Support
Plug-in FAQ
Done

```
Warning: mysqli_connect() [function.mysqli-connect]: Access denied for user 'dgosselin'@'localhost'
(using password: YES) in C:\Course Technology\1687-5s\Chapter.09\Examples\test.php on line 5

The database server is not available.
```

Figure 9-2 Database connection error message

Suppressing Errors with the Error Control Operator

- Writing code that anticipates and handles potential problems is often called **bulletproofing**
- Bulletproofing techniques include:
  - Validating submitted form data

    e.g. `if (isset($_GET['height']) ...`
  - Using the **error control operator (@)** to suppress error messages

    e.g.
    ```
    $DBConnect = @mysqli_connect(...);
    if (!$DBConnect) ...
    ```

Terminating Script Execution

- The `die()` and `exit()` functions terminate script execution
- The `die()` version is usually used when attempting to access a data source
- Both functions accept a single string argument
- Call the `die()` and `exit()` functions as separate statements or by appending either function to an expression with the `Or` operator

Terminating Script Execution (continued)

```
$DBConnect = @mysqli_connect("localhost", "root", "paris");
if (!$DBConnect)
      die("<p>The database server is not available.</p>");
echo "<p>Successfully connected to the database server.</p>";
$DBSelect = @mysqli_select_db($DBConnect, "flightlog");
if (!$DBSelect)
      die("<p>The database is not available.</p>");
echo "<p>Successfully opened the database.</p>";
// additional statements that access the database
mysqli_close($DBConnect);
```

No `else` required here

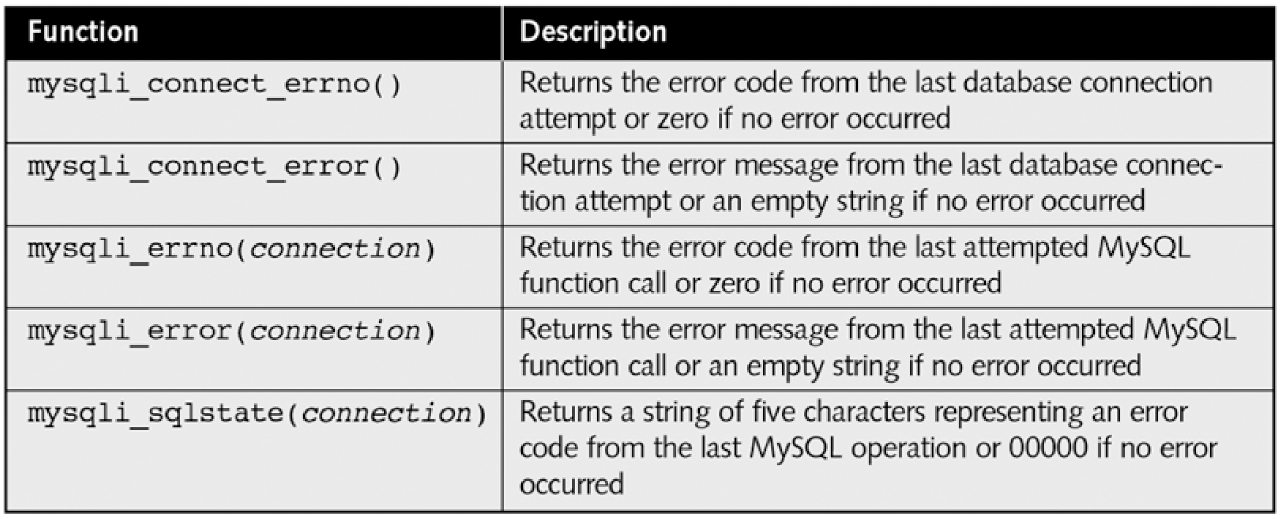
Terminating Script Execution (continued)

```
$DBConnect = @mysqli_connect("localhost", "dongosselin",
"rosebud")
     Or die("<p>The database server is not available.</p>");
// the above is one statement: connected OK or die
echo "<p>Successfully connected to the database server.</p>";
@mysqli_select_db($DBConnect, "flightlog")
     Or die("<p>The database is not available.</p>");
echo "<p>Successfully opened the database.</p>";
// additional statements that access the database server
mysqli_close($DBConnect);
```

No `if` required here

Reporting MySQL Errors

Table 9-2 MySQL error reporting functions

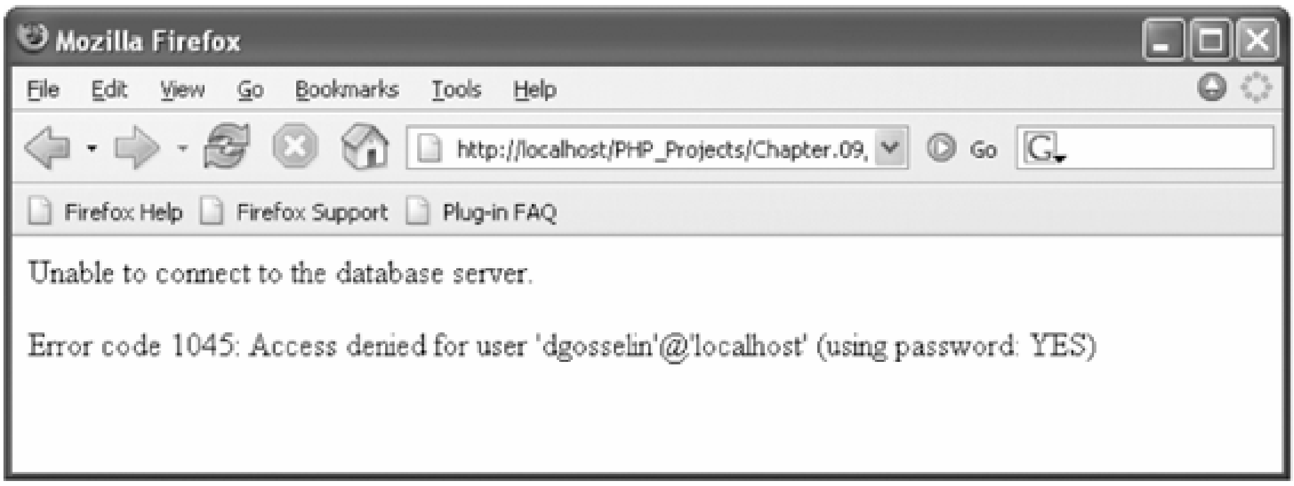


Function	Description
`mysqli_connect_errno()`	Returns the error code from the last database connection attempt or zero if no error occurred
`mysqli_connect_error()`	Returns the error message from the last database connection attempt or an empty string if no error occurred
`mysqli_errno(`*`connection`*`)`	Returns the error code from the last attempted MySQL function call or zero if no error occurred
`mysqli_error(`*`connection`*`)`	Returns the error message from the last attempted MySQL function call or an empty string if no error occurred
`mysqli_sqlstate(`*`connection`*`)`	Returns a string of five characters representing an error code from the last MySQL operation or 00000 if no error occurred

Reporting MySQL Errors (continued)

```
$User = $_GET['username'];
$Password = $_GET['password'];
$DBConnect = @mysqli_connect("localhost", $User, $Password)
     Or die("<p>Unable to connect to the database server.</p>"
     . "<p>Error code " . mysqli_connect_errno()
     . ": " . mysqli_connect_error()) . "</p>";
echo "<p>Successfully connected to the database server.</p>";
@mysqli_select_db($DBConnect, "flightlog")
     Or die("<p>The database is not available.</p>");
echo "<p>Successfully opened the database.</p>";
// additional statements that access the database
mysqli_close($DBConnect);
```

Reporting MySQL Errors (continued)


Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://localhost/PHP_Projects/Chapter.09,
Go
Firefox Help Firefox Support Plug-in FAQ

Unable to connect to the database server.

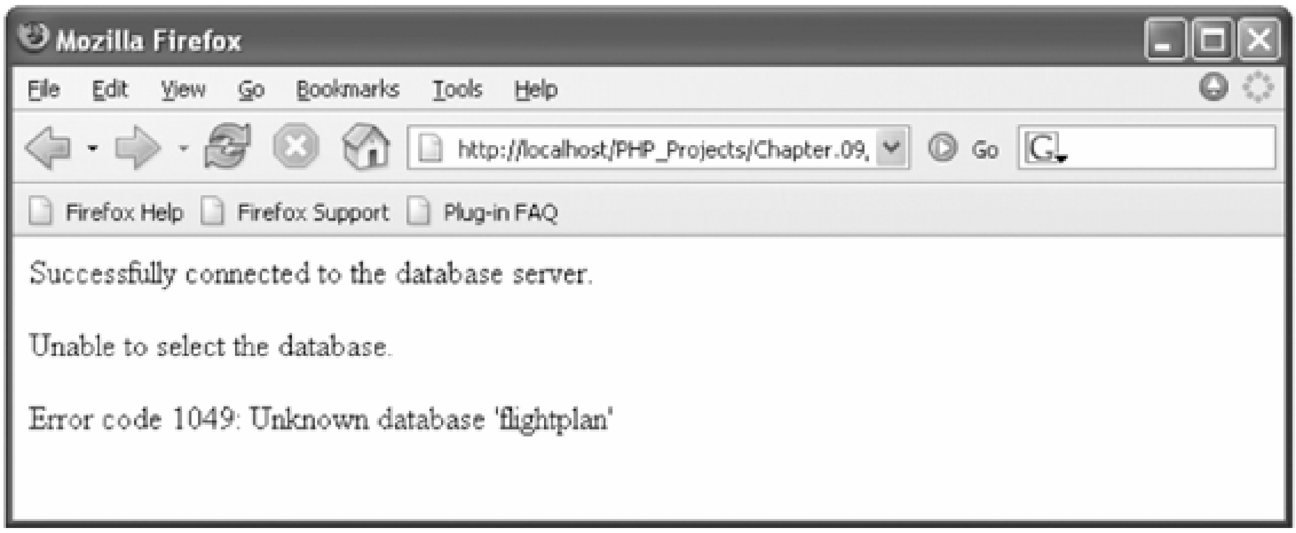
Error code 1045: Access denied for user 'dgosselin'@'localhost' (using password: YES)

Figure 9-4 Error number and message generated by an invalid username and password

Reporting MySQL Errors (continued)

```
$User = $_GET['username'];
$Password = $_GET['password'];
$DBConnect = @mysqli_connect("localhost", $User, $Password)
     Or die("<p>Unable to connect to the database server.</p>"
     . "<p>Error code " . mysqli_connect_errno()
     . ": " . mysqli_connect_error()) . "</p>";
echo "<p>Successfully connected to the database server.</p>";
@mysqli_select_db($DBConnect, "flightplan")
     Or die("<p>Unable to select the database.</p>"
     . "<p>Error code " . mysqli_errno($DBConnect)
     . ": " . mysqli_error($DBConnect)) . "</p>";
echo "<p>Successfully opened the database.</p>";
// additional statements that access the database
mysqli_close($DBConnect);
```

Reporting MySQL Errors (continued)


Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://localhost/PHP_Projects/Chapter.09.
Go
Firefox Help
Firefox Support
Plug-in FAQ

```
Successfully connected to the database server.

Unable to select the database.

Error code 1049: Unknown database 'flightplan'
```

Figure 9-5 Error code and message generated when attempting to select a database that does not exist

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)
```

Note:

```
$Make = “Ovation”;
$SQLString = “SELECT model, quantity FROM
    $DBTable WHERE model = $Make”; (Wrong)
```

vs

```
$Make = “Ovation”;
$SQLString = “SELECT model, quantity FROM
    $DBTable WHERE model = ‘$Make’ ”;
```

Executing SQL Statements (continued)

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
- 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

Working with Query Results

Table 9-3 Common PHP functions for accessing database results

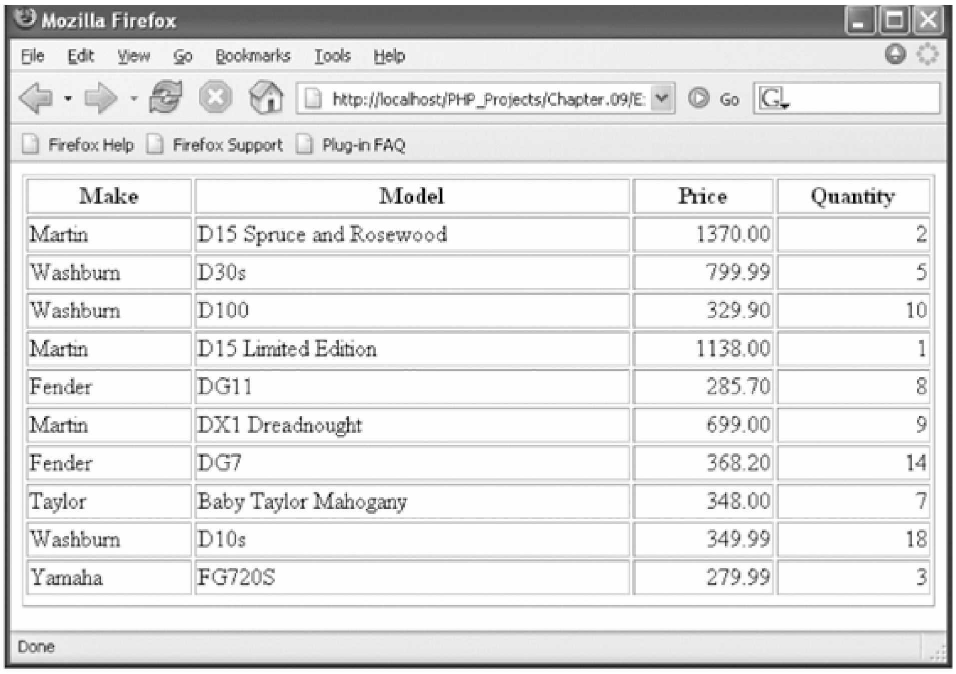
Function	Description
`mysqli_data_seek($Result, position)`	Moves the result pointer to a specified row in the resultset
`mysqli_fetch_array($Result, MYSQLI_ASSOC | MYSQLI_NUM | MYSQLI_BOTH)`	Returns the fields in the current row of a resultset into an indexed array, associative array, or both and moves the result pointer to the next row
`mysqli_fetch_assoc($Result)`	Returns the fields in the current row of a resultset into an associative array and moves the result pointer to the next row
`mysqli_fetch_lengths($Result)`	Returns the field lengths for the current row in a resultset into an indexed array
`mysqli_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

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 width='100%‘ border='1'>";
echo "<tr><th>Make</th><th>Model</th>
     <th>Price</th><th>Quantity</th></tr>";
$Row = mysqli_fetch_row($QueryResult);
do {
     echo "<tr><td>{$Row[0]}</td>";
     echo "<td>{$Row[1]}</td>";
     echo "<td align='right'>{$Row[2]}</td>";
     echo "<td align='right'>{$Row[3]}</td></tr>";
     $Row = mysqli_fetch_row($QueryResult);
} while ($Row);
```

Retrieving Records into an Indexed Array (continued)


Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://localhost/PHP_Projects/Chapter.09/E
Go
Firefox Help
Firefox Support
Plug-in FAQ
Done

Make	Model	Price	Quantity
Martin	D15 Spruce and Rosewood	1370.00	2
Washburn	D30s	799.99	5
Washburn	D100	329.90	10
Martin	D15 Limited Edition	1138.00	1
Fender	DG11	285.70	8
Martin	DX1 Dreadnought	699.00	9
Fender	DG7	368.20	14
Taylor	Baby Taylor Mahogany	348.00	7
Washburn	D10s	349.99	18
Yamaha	FG720S	279.99	3

Figure 9-6 Output of the inventory table in a Web browser

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

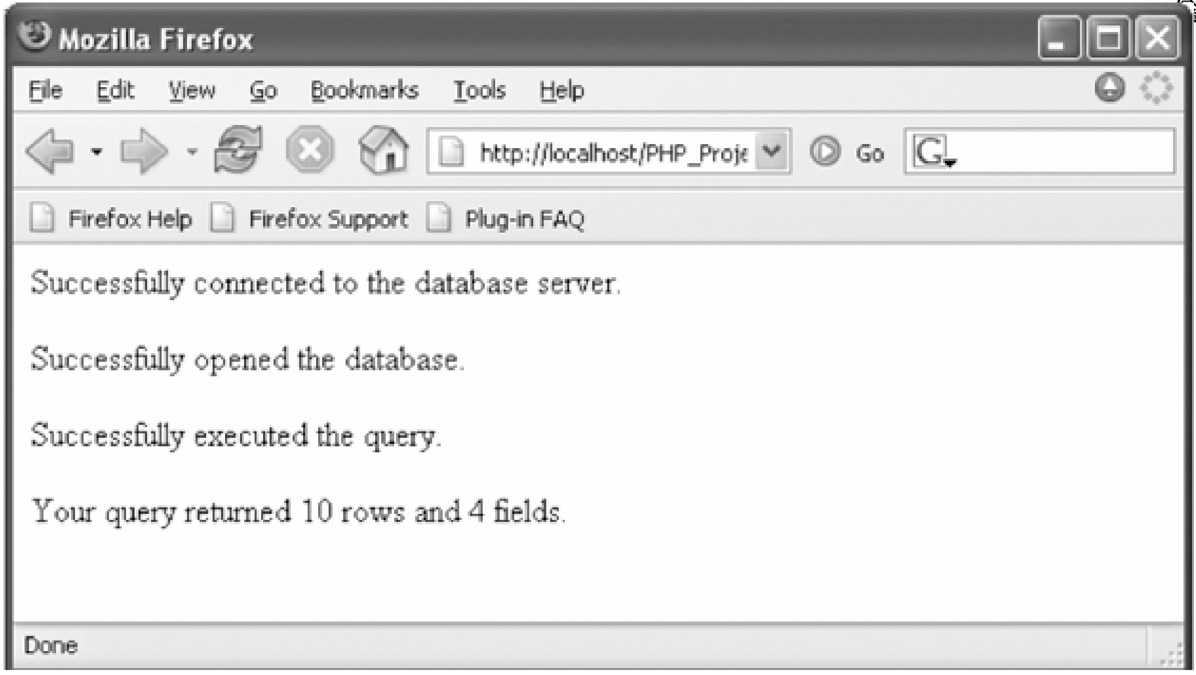
Accessing Query Result Information

- 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 (continued)

```
$SQLstring = "SELECT * FROM inventory";
$QueryResult = @mysqli_query($DBConnect, $SQLstring)
     Or die("<p>Unable to execute the query.</p>"
     . "<p>Error code " . mysqli_errno($DBConnect)
     . ": " . mysqli_error($DBConnect)) . "</p>";
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 (continued)


Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://localhost/PHP_Proje
Go
Firefox Help
Firefox Support
Plug-in FAQ
Done

```
Successfully connected to the database server.

Successfully opened the database.

Successfully executed the query.

Your query returned 10 rows and 4 fields.
```

Figure 9-8 Output of the number of rows and fields returned from a query

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

Creating and Deleting Databases

- Use the CREATE DATABASE statement with the `mysqli_query()` function to create a new database

```
$SQLstring = "CREATE DATABASE real_estate";
$QueryResult = @mysqli_query($DBConnect, $SQLstring)
     Or die("<p>Unable to execute the query.</p>"
     . "<p>Error code " . mysqli_errno($DBConnect)
     . ": " . mysqli_error($DBConnect)) . "</p>";
echo "<p>Successfully executed the query.</p>";
mysqli_close($DBConnect);
```

Creating and Deleting Databases (continued)

- Use the `mysqli_db_select()` function to check whether a database exists before you create or delete it
- To use a new database, you must select it by executing the `mysqli_select_db()` function
- Deleting a database is almost identical to creating one, except use the DROP DATABASE statement instead of the CREATE DATABASE statement with the `mysqli_query()` function

Creating and Deleting Databases (continued)

```
$DBName = "real_estate";
...
if (@!mysqli_select_db($DBConnect, $DBName))
     echo "<p>The $DBName database does not exist!</p>";
else {
     $SQLstring = "DROP DATABASE $DBName";
     $QueryResult = @mysqli_query($DBConnect, $SQLstring)
           Or die("<p>Unable to execute the query.</p>"
           . "<p>Error code “ . mysqli_errno($DBConnect)
           . ": “ . mysqli_error($DBConnect)) . "</p>";
     echo "<p>Successfully deleted the database.</p>";
}
mysqli_close($DBConnect);
```

Creating and Deleting Tables

- To create a table, use the `CREATE TABLE` statement with the `mysqli_query()` function
- Execute the `mysqli_select_db()` function before executing the `CREATE TABLE` statement or the new table might be created in the wrong database
- To prevent code from attempting to create a table that already exists, use a `mysqli_query()` function that attempts to select records from the table

Creating and Deleting Tables (continued)

```
$DBName = "real_estate";
...
$SQLstring = "CREATE TABLE commercial (city
  VARCHAR(25), state
VARCHAR(25), sale_or_lease VARCHAR(25), type_of_use
  VARCHAR(40),
Price INT, size INT)";
$QueryResult = @mysqli_query($DBConnect, $SQLstring)
    Or die("<p>Unable to execute the query.</p>"
    . "<p>Error code " . mysqli_errno($DBConnect)
    . ": " . mysqli_error($DBConnect)) . "</p>";
echo "<p>Successfully created the table.</p>";
mysqli_close($DBConnect);
```

Adding, Deleting, and Updating Records

Note: refer to Chapter 8

Add

- To add records to a table, use the `INSERT` and `VALUES` keywords with the `mysqli_query()` function
- The values entered in the `VALUES` list must be in the same order in which you defined the table fields
- You must specify `NULL` in any fields for which you do not have a value
  e.g. for `AUTO_INCREMENT` field
- To add multiple records to a database, use the `LOAD DATA` statement and the `mysqli_query()` function with a local text file containing the records you want to add

Adding, Deleting, and Updating Records (continued)

Delete

- To delete records in a table, use the `DELETE` and `WHERE` keywords with the `mysqli_query()` function
- The `WHERE` keyword determines which records to delete in the table

Adding, Deleting, and Updating Records (continued)

Update

- To update records in a table, use the `UPDATE`, `SET`, and `WHERE` keywords with the `mysqli_query()` function
- 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` keyword

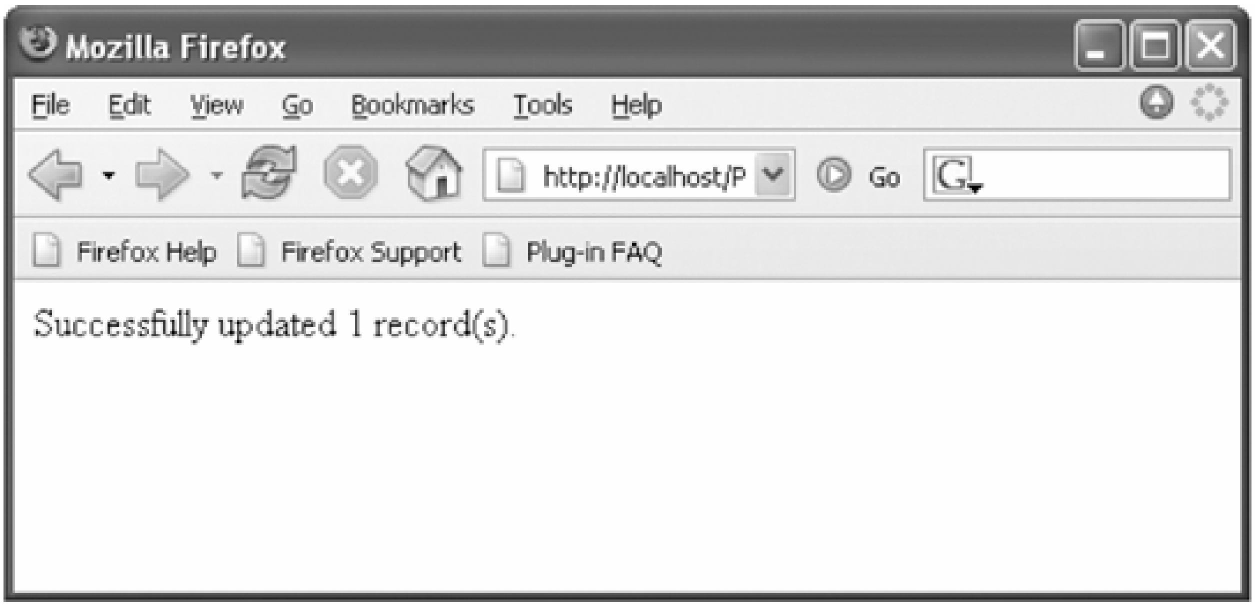
Using the **`mysqli_affected_rows()`** Function

- 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 **`mysqli_affected_rows()`** Function (continued)

```
$SQLstring = "UPDATE inventory SET price=368.20
      WHERE make='Fender' AND model='DG7'";
$QueryResult = @mysqli_query($DBConnect, $SQLstring)
      Or die("<p>Unable to execute the query.</p>"
      . "<p>Error code " . mysqli_errno($DBConnect)
      . ": " . mysqli_error($DBConnect)) . "</p>";
echo "<p>Successfully updated "
      . mysqli_affected_rows($DBConnect) . " record(s).</p>";
```

Using the `mysqli_affected_rows()` Function (continued)



Successfully updated 1 record(s).

Figure 9-16 Output of `mysqli_affected_rows()` function for an `UPDATE` query

Summary

- PHP includes functionality that allows you to work directly with different types of databases, without going through ODBC
- Writing code that anticipates and handles potential problems is often called bulletproofing
- The error control operator (@) suppresses error messages
- A result pointer is a special type of variable that refers to the currently selected row in a resultset

Summary (continued)

- Use the `mysqli_query()` function to send SQL statements to MySQL
- To identify a field as a primary key in MySQL, include the `PRIMARY KEY` keywords when you first 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
- You use the `LOAD DATA` statement and the `mysqli_query()` function with a local text file to add multiple records to a database
- 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