CONNECT TO A MYSQL SERVER

To enable a PHP page to access a database stored on a MySQL server, you must create a connection to the server. The `mysql_connect` function allows you to create a connection to a MySQL server.

The `mysql_connect` function takes three arguments—the host name or IP address of the computer running the MySQL server, a user name and a password. If the MySQL server is running on the same computer as the Web server, you can specify the host name localhost or the IP address 127.0.0.1 to connect to the MySQL server. If the MySQL server does not require a user name and password, you can enter empty strings ("") for these arguments.

When a connection to the MySQL server is made successfully, the `mysql_connect` function returns a link identifier. A link identifier is a positive integer that labels the connection to the server. If a connection is not made successfully, the function will return a value of false.

The link identifier can be used by other functions in the PHP script, such as `mysql_select_db` and `mysql_query`, to send commands to the MySQL server. Specifying a link identifier in a function allows you to perform an operation using a specific MySQL server connection. If no link identifier is specified, the last successful connection made in the script will be used.

If you include more than one `mysql_connect` function with the same arguments in a script, new connections will not be created when the subsequent functions are called. Instead, the functions will use the link identifier for the first connection.

When a PHP script that opens a MySQL server connection terminates, the connection to the server is automatically closed. You can use the `mysql_close` function to explicitly close a connection.

Extra

Once a connection to a MySQL server has been established, you can use the `mysql_select_db` function to select the database you want to work with. To use the `mysql_select_db` function, you must specify the name of the database you want to access and the link identifier for the connection. This function returns a value of true if the specified database is successfully selected or false if the selection is not successful.

Example:

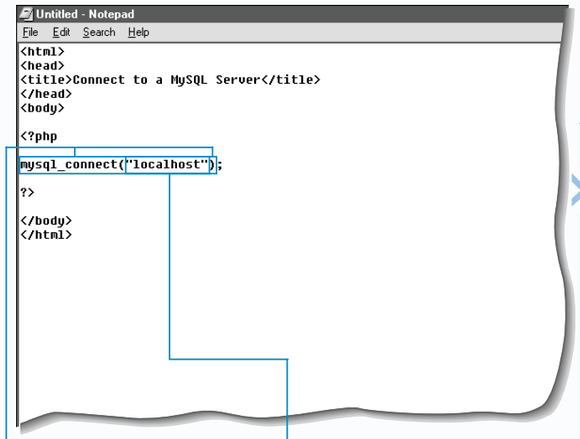
```
$linkID = mysql_connect("localhost", "", "");

if (mysql_select_db("mysql", $linkID) != FALSE)
{
    print "The database was successfully selected.";
}
else
{
    print "The specified database cannot be selected.";
}
```

The error control operator (@) is often used with the `mysql_connect` function. This operator prevents a PHP error message from appearing in a user's Web browser when a connection to the MySQL server cannot be established. Also, since a script is unlikely to continue working properly if a connection to the MySQL server fails, you may want to use the `die` function to terminate the script and display a customized error message.

Example:

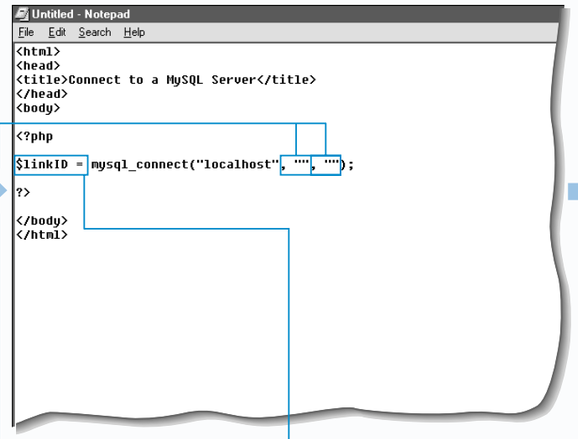
```
$linkID = @mysql_connect("localhost", "", "") or
die("Connection to the database cannot be made.");
```

CONNECT TO A MYSQL SERVER



1 To connect to a MySQL server, type **mysql_connect()**.

2 Between the parentheses, type the host name or IP address of the computer running the MySQL server enclosed in quotation marks.

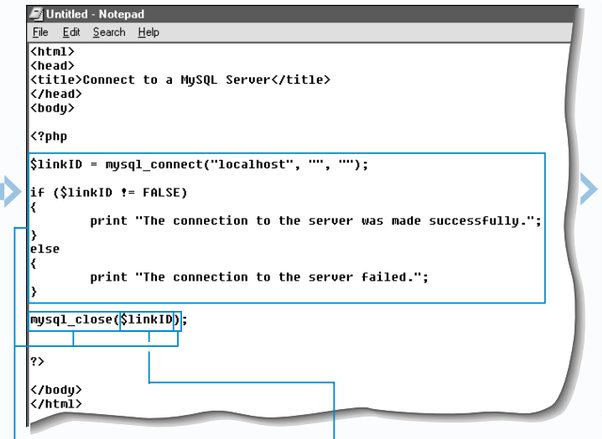


3 Type a comma followed by a user name enclosed in quotation marks.

4 Type a comma followed by a password enclosed in quotation marks.

Note: If a user name or password is not required, enter an empty string.

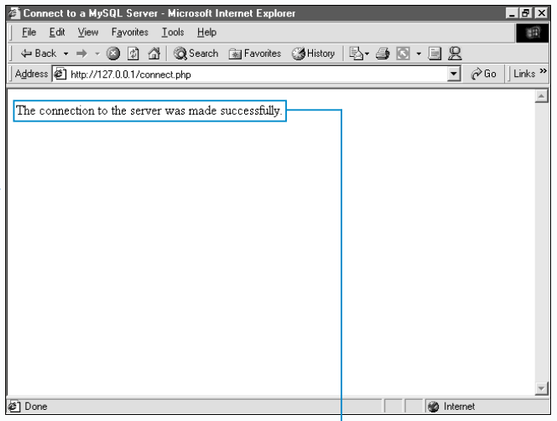
5 Type the code that assigns the link identifier returned by the `mysql_connect` function to a variable.



6 To test if a connection to the MySQL server was made successfully, type the code that checks the value of the link identifier.

7 To close the connection to the MySQL server, type **mysql_close()**.

8 Between the parentheses, type the name of the variable that stores the link identifier.



9 Display the PHP page in a Web browser.

■ The Web browser displays the results of connecting to a MySQL server.

CREATE A PERSISTENT CONNECTION TO A MYSQL SERVER

You can use the `mysql_pconnect` function to create a persistent connection to a MySQL server. When a persistent connection is established, the connection will remain open even after the PHP script is finished processing. When the `mysql_pconnect` function is called again in another PHP script using the same arguments as the original `mysql_pconnect` function call, the original connection will be used instead of creating a new connection.

The `mysql_pconnect` function will work only if PHP is set up as a module. For information about setting up PHP as an Apache module, see the top of page 11. To set up PHP as a module on a different type of Web server, refer to PHP's documentation.

The `mysql_pconnect` function takes three arguments—the host name or IP address of the computer running the MySQL server, a user name and a password. If a user name or password is not required, you can enter empty strings ("") for these arguments.

When a connection to the server is made successfully, the `mysql_pconnect` function returns a link identifier, which is a positive number that labels the connection to the server. If a connection is not made successfully, the function will return a value of false.

A connection created using the `mysql_pconnect` function cannot be closed using the `mysql_close` function. A connection using the `mysql_pconnect` function will eventually close after it has been idle for a specific amount of time.

The `mysql_pconnect` function is useful in instances where many connections to the database have to be made in a short period of time using the same user name and password. Opening and closing database connections can use a substantial amount of system resources and thus slow down a server. Using a persistent connection to a database is more efficient and improves the performance of the application.

Extra

When specifying the host argument for the `mysql_pconnect` function, you may specify a port number by typing the host name or IP address followed by a colon (:) and the port number you want to use. By default, PHP uses port number 3306. You typically would not have to specify a port number when connecting to a MySQL server, unless the MySQL server is not set to the default port number. If multiple MySQL servers are running on the same machine, a port number needs to be specified in order to connect to the appropriate MySQL server. You may also specify a port number when calling the `mysql_connect` function.

Example:

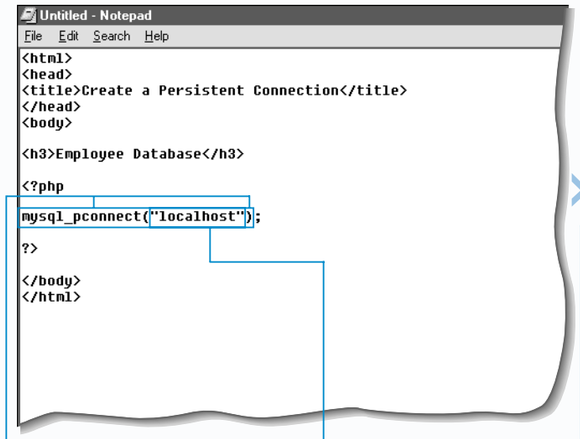
```
$linkID = mysql_pconnect("localhost:3307", "martine", "secret");
```

When working in the Unix environment, you can specify the path of a socket being used by a MySQL server. A socket provides a means for the client to communicate with the server and is similar to a network port number. The path of a socket can be specified in the host argument for the `mysql_pconnect` function by typing the host name or IP address followed by a colon (:) and the path to the socket file. The default socket is /tmp/mysql.sock. You may also specify a path to a socket when calling the `mysql_connect` function.

Example:

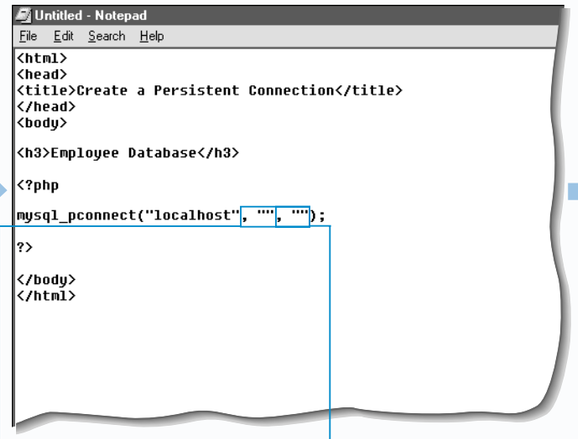
```
$linkID = mysql_pconnect("localhost:/tmp/mysql2.sock", "martine", "secret");
```

CREATE A PERSISTENT CONNECTION TO A MYSQL SERVER



1 To create a persistent connection to a MySQL server, type **mysql_pconnect()**.

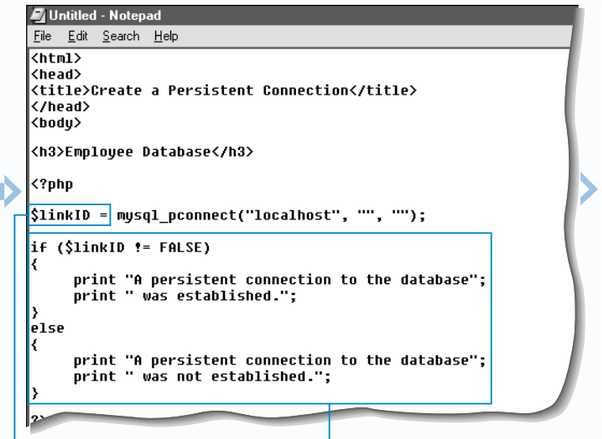
2 Between the parentheses, type the host name or IP address of the computer running the MySQL server enclosed in quotation marks.



3 Type a comma followed by a user name enclosed in quotation marks.

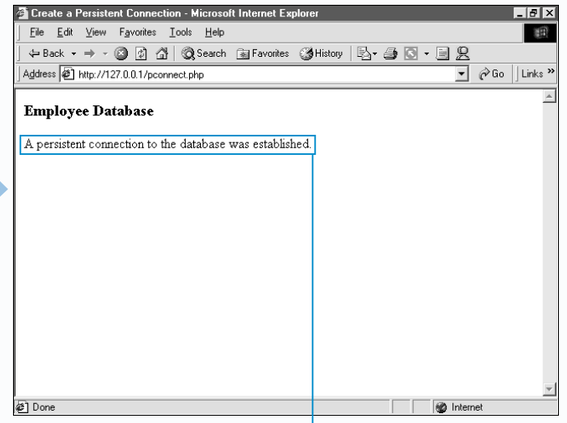
4 Type a comma followed by a password enclosed in quotation marks.

Note: If a user name or password is not required, enter an empty string.



5 Type the code that assigns the link identifier returned by the `mysql_pconnect` function to a variable.

6 To test if the persistent connection to the MySQL server was made successfully, type the code that checks the value of the link identifier.



7 Display the PHP page in a Web browser.

■ The Web browser displays the results of creating a persistent connection to a MySQL server.

ISSUE AN SQL STATEMENT TO A MYSQL SERVER

After creating a connection to a MySQL server in a PHP page and selecting the database you want to work with, you can use the `mysql_query` function to issue an SQL statement to the server. Issuing an SQL statement to a MySQL server allows you to manipulate a database and perform tasks such as creating a table, retrieving data or adding records.

To use the `mysql_query` function, you need to specify a valid SQL statement. You can also specify the link identifier for the MySQL server connection you want to use.

If the SQL statement is successfully executed, the `mysql_query` function will return a value of true or a result identifier. If the statement is not successfully executed, the function will return a value of false. SQL statements that perform an operation but do not retrieve data from a database, such as `CREATE TABLE` and `UPDATE`, return a value of true if successfully executed.

SQL statements used to retrieve data from a database, such as `SELECT` and `SHOW TABLES`, return a result identifier if successfully executed. The result identifier is a positive integer that points to the data retrieved from the database, which is temporarily stored in a *result set*. To access the data from the result set, you specify the result identifier as an argument for the function you want to use, such as the `mysql_fetch_row` function.

You should keep in mind that the database program you are using determines the SQL statements you can issue. Certain SQL features are not available in MySQL in order to improve the database program's performance and speed. There are also special SQL statements that can be used only on a MySQL database. You can consult the www.mysql.com Web site to find more information about the specific features of MySQL.

Extra

The `mysql_create_db` function can be used in a PHP script to create a database. To use this function, you need to specify the name of the database you want to create followed by the link identifier for the MySQL server connection. The `mysql_create_db` function returns a value of true if the database was successfully created or false if the operation fails. In order to create a database, the user name used to make the MySQL server connection must have the appropriate permissions.

Example:

```
if (mysql_create_db("accounting", $linkID) == TRUE)
{
    print "The database was created successfully.";
}
else
{
    print "The database could not be created.";
}
```

The `mysql_drop_db` function allows you to delete a database from a MySQL server. If the database is deleted successfully, a value of true is returned. Otherwise, a value of false is returned.

Example:

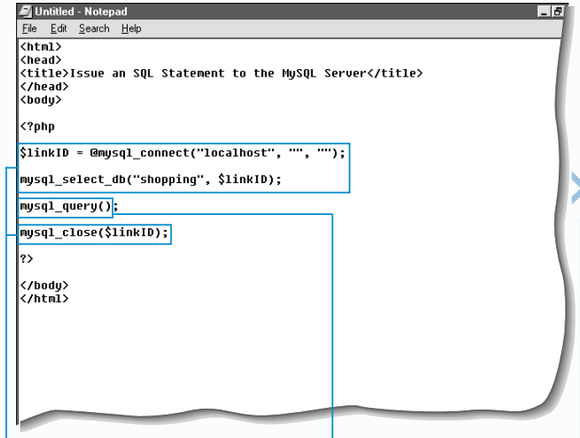
```
$resultID = mysql_drop_db("customers", $linkID);
```

You can use the `mysql_db_query` function to send an SQL statement directly to a database after creating a connection to the server. The `mysql_db_query` function takes three arguments—the name of the database you want to query, an SQL statement and the link identifier for the MySQL server connection.

Example:

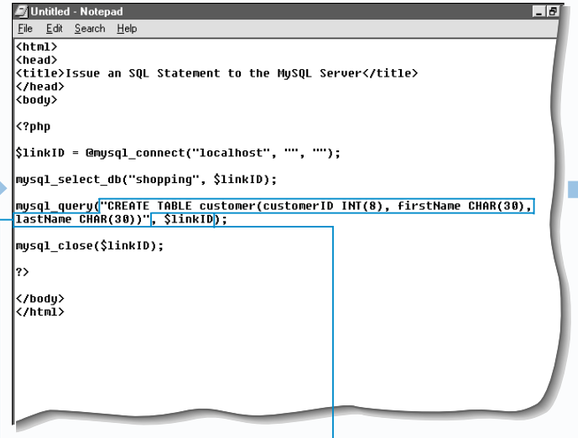
```
$resultID = mysql_db_query("accounting",
"CREATE TABLE employee(id INT(4), name CHAR(15),
extension INT(4))", $linkID);
```

ISSUE AN SQL STATEMENT TO A MYSQL SERVER



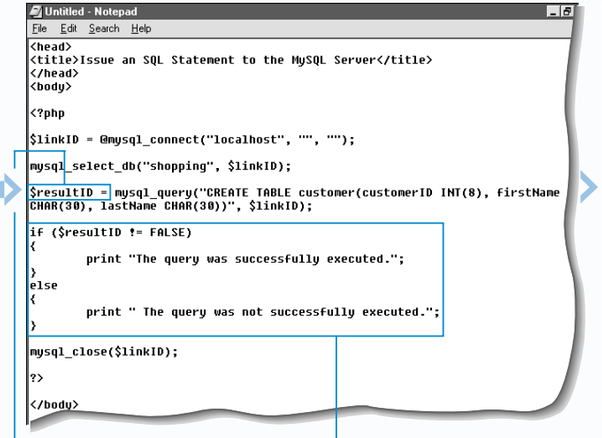
1 Type the code that connects the PHP page to the MySQL server and selects the database you want to work with.

2 To issue an SQL statement to the MySQL server, type **mysql_query**().



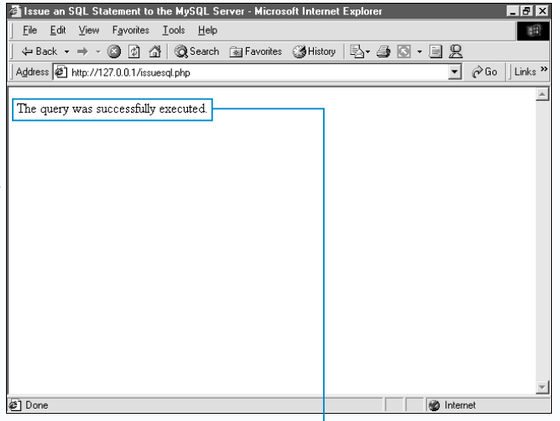
3 Between the parentheses, type the SQL statement you want to issue, enclosed in quotation marks.

4 Type a comma followed by the name of the variable that stores the link identifier.



5 Type the code that assigns the value returned by the `mysql_query` function to a variable.

6 To test if the SQL statement was issued successfully, type the code that checks the value of the variable storing the `mysql_query` function.



7 Display the PHP page in a Web browser.

The Web browser displays the result of issuing an SQL statement to a MySQL server.

RETRIEVE INFORMATION FROM A DATABASE

The SQL SELECT statement allows you to retrieve information from a database. This is useful when you want to display the contents of a table. The mysql_query function is used to issue a SELECT statement to the database.

The SELECT statement allows you to specify the data you want to retrieve from a table. You can specify the data you want to retrieve by name or use an asterisk (*) to retrieve all the data in a table. The SELECT statement uses the FROM clause to specify the name of the table that stores the information you want to retrieve.

When the SELECT statement has been executed successfully, the data retrieved from the database is temporarily stored in a result set, which has the same format as a table. The fields specified in the SELECT statement make up the columns of the result set and the values from each field form the rows.

The mysql_query function returns a result identifier, which points to the result set. If the SELECT statement is not executed successfully, the function returns a value of false.

Once the SELECT statement has been successfully issued to the database, you can use the mysql_fetch_row function to access the data in the result set. You must specify the variable that stores the result identifier as the argument for the mysql_fetch_row function. The mysql_fetch_row function returns an array containing the values from one row of the result set. To retrieve subsequent rows in the result set, you can use the mysql_fetch_row function in a loop.

Information retrieved from a database is commonly placed in an HTML table. This allows you to neatly display the information on a Web page.

Extra

You can use the list statement with the mysql_fetch_row function to assign an array value to a variable with a meaningful name. When you want to use the array value, you can use the variable name to access the value instead of referencing the value by its numeric key. This can help simplify your code and make your arrays easier to work with.

Example:

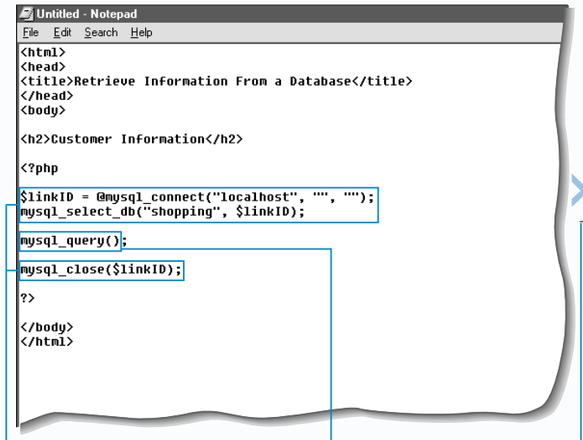
```
$resultID = mysql_query("SELECT firstName, extension FROM employees", $linkID);
print "<table>";
print "<tr><th>First Name</th><th>Extension</th></tr>";
while (list($firstName, $extension) = mysql_fetch_row($resultID))
{
     print "<tr><td>$firstName</td><td>$extension</td></tr>";
}
print "</table>";
```

You can use the WHERE clause with a SELECT statement to specify a condition for retrieving data. To specify multiple conditions, you can also use the OR and AND operators. For more information about using the WHERE clause, consult the MySQL documentation at the www.mysql.com Web site.

Example:

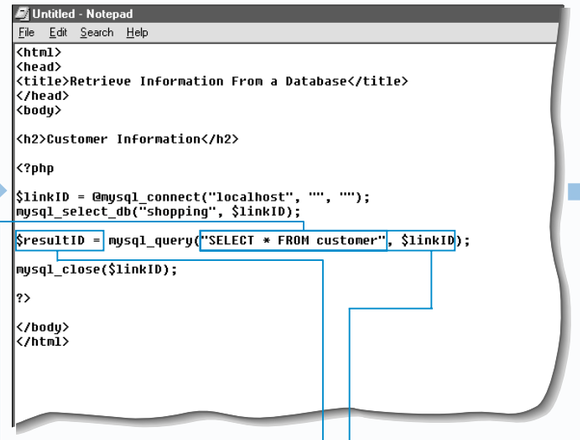
```
SELECT invoiceNumber, totalCost FROM orders WHERE totalCost = 50
  OR (totalCost >= 100 AND totalCost <= 300)
```

RETRIEVE INFORMATION FROM A DATABASE



1 Type the code that connects the PHP page to the MySQL server and selects the database you want to work with.

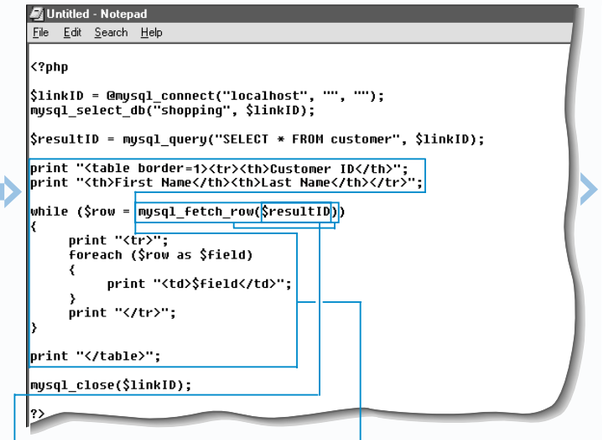
2 To issue an SQL statement to the MySQL server, type **mysql_query()**.



3 Between the parentheses, type **SELECT * FROM** followed by the name of the table that stores the information you want to retrieve. Enclose the SELECT statement in quotation marks.

4 Type a comma followed by the name of the variable that stores the link identifier.

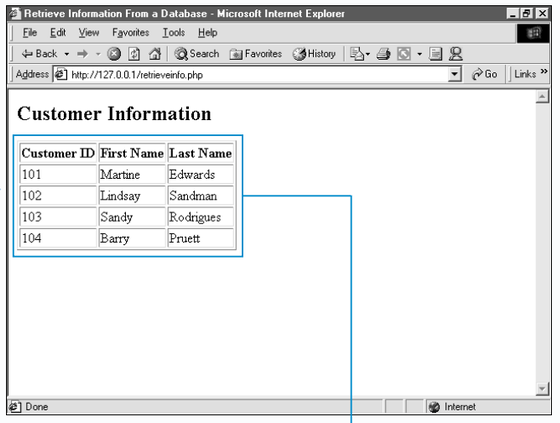
5 Type the code that assigns the value returned by the mysql_query function to a variable.



6 To access the data retrieved from the database, type **mysql_fetch_row()**.

7 Between the parentheses, type the name of the variable that stores the result identifier.

8 Type the code that processes the result of the mysql_fetch_row function and formats the information for display.



9 Display the PHP page in a Web browser.

■ The Web browser displays the result of retrieving information from a database.

ADD A RECORD

Once you establish a connection with a database, you can add records to the database using the INSERT statement.

When adding a record to a database, you use the mysql_query function to issue an INSERT statement to the MySQL server. The mysql_query function returns a value of true if the new records are added successfully. Otherwise, the function returns a value of false.

The INSERT statement uses the INTO clause to specify the name of the table into which you want to insert records. After the table name, you indicate the fields into which you want to insert data.

The VALUES clause is used to specify the values for each record you want to add to the table. You must specify a value that corresponds to each field name indicated in the INTO clause. You must place the values in the same order the fields were specified in the INTO clause. String values should be enclosed in single quotation marks (').

If you do not specify the names of the fields you want to affect in the INTO clause, MySQL assumes that you want to insert data into all the fields in the table. The values must be specified in the same order the fields appear in the table.

You may insert multiple records using the same INSERT statement. After the values for the first record are specified, values for subsequent records are included, enclosed in parentheses. Each record you add must be separated by a comma.

The MySQL server requires that appropriate permissions be granted before a script can add information to a database. For information about setting permissions in a MySQL server, consult the MySQL documentation at the www.mysql.com Web site.

Extra

When adding a string value that contains quotation marks, you must escape the quotation marks by placing the backslash character (\) before each quotation mark. If the quotation marks in a string are not escaped, an error may occur when the INSERT statement is executed.

Example:

```
mysql_query("INSERT INTO customer (customerID, firstName,
      lastName) VALUES (111, 'Tim', 'O\'Brien')", $linkID);
```

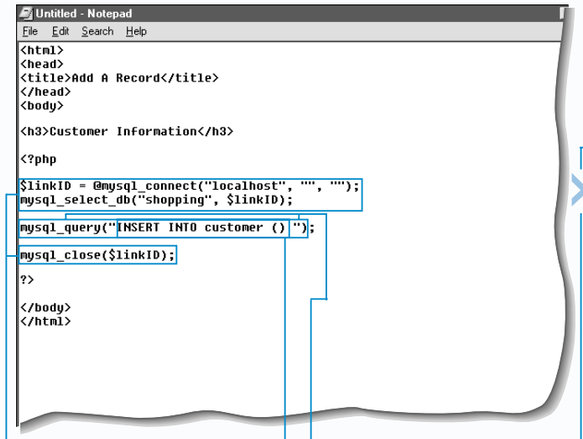
You can insert a string value stored in a variable into a record in your database. You should use the addslashes function to escape the quotation marks in a string stored in a variable before it is used with an INSERT statement. The addslashes function takes the variable that contains a string value as an argument. When a variable is used in an SQL statement, the variable name is enclosed in single quotation marks.

Example:

```
$text = "O'Brien";
$escaped = addslashes($text);
mysql_query("INSERT INTO customer (customerID, firstName,
      lastName) VALUES (111, 'Tim', '$escaped')", $linkID);
```

You can make changes to the php.ini file so that quotation marks, backslash characters and NULL characters in strings are automatically escaped when the strings are used in a script. When the magic_quotes_gpc directive in the php.ini file is enabled, string values that are retrieved from a form or a cookie are escaped automatically. When the magic_quotes_runtime directive is enabled, functions that retrieve information from external sources, such as databases and text files, return strings that are escaped automatically.

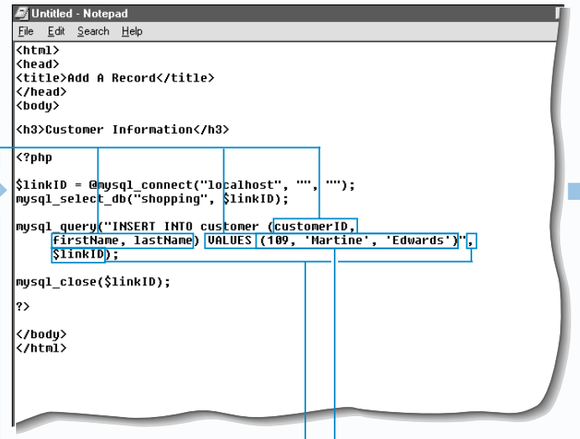
ADD A RECORD



1 Type the code that connects the PHP page to the MySQL server and selects the database you want to work with.

2 To add a record in the database, type **mysql_query("")**.

3 Between the quotation marks, type **INSERT INTO** followed by the name of the table to which you want to add records. Then type ().

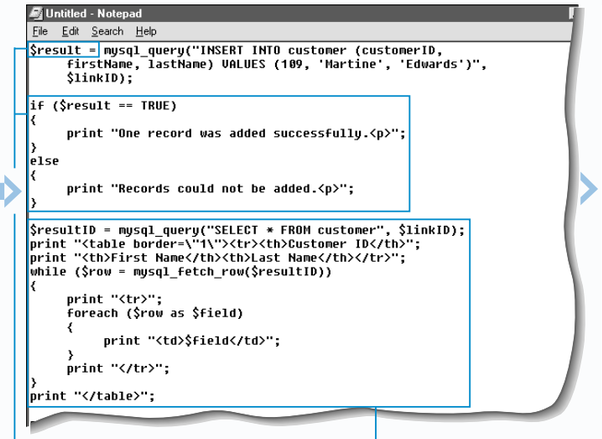


4 Between the parentheses, type the names of the fields in the table, separating each field name with a comma.

5 To specify the values for a record you want to add, type **VALUES**.

6 Type the values you want to assign to each field in the record, enclosed in parentheses. A comma separates each value.

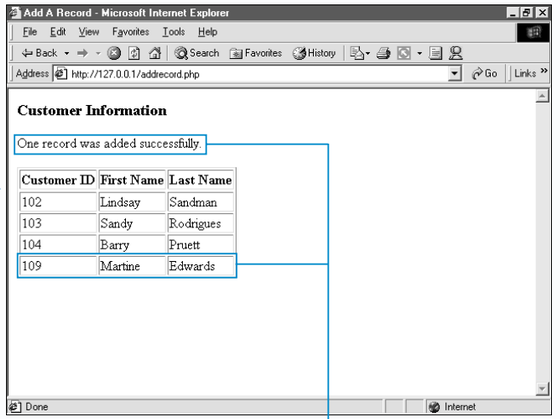
7 Outside the quotation marks, type a comma followed by the link identifier.



8 Type the code that assigns the value returned by the mysql_query function to a variable.

9 To test if the records were added successfully, type the code that checks the value of the variable storing the mysql_query function.

10 Type the code that displays information from the database in the PHP page.



11 Display the PHP page in a Web browser.

■ The Web browser displays the result of adding a record to a database.

UPDATE A RECORD

Once you establish a connection with a database, you can edit the information contained in the database. Editing the information in a database allows you to keep the information up-to-date.

To update records in a database, you use the `mysql_query` function to issue an `UPDATE` statement to the MySQL server. The `mysql_query` function returns a value of true if the update is made successfully.

After the `UPDATE` statement, you indicate the name of the table that contains the records you want to modify. The `SET` clause specifies which field in a record needs to be changed and the new value. You can specify more than one field to be changed. The `UPDATE` statement uses the `WHERE` clause to specify the data that identifies the record to be modified.

Depending on the data specified in the `WHERE` clause, you may update the value of one or more records in a table. To update a single record, you would typically indicate the unique ID for the record in the `WHERE` clause. If a `WHERE` clause is not specified when using an `UPDATE` statement, MySQL will update all the records in the table.

When specifying the value of a field, a string value must be enclosed in single quotation marks (`''`). If the value contains quotation marks, you must escape the quotation marks by placing the backslash character (`\`) before each quotation mark, such as `\'` and `\"`.

The MySQL server requires that appropriate permissions be granted before a script can update information in a database. For information about setting update permissions in a MySQL server, consult the MySQL documentation at the www.mysql.com Web site.

Extra

The `mysql_affected_rows` function may be used to determine the number of records that were affected by an `UPDATE` statement. The link identifier to the MySQL connection is passed as an argument to the `mysql_affected_rows` function. The code that calls the `mysql_affected_rows` function is placed immediately after the line that updates the database. The `mysql_affected_rows` function may also be used to determine the number of records affected by an `INSERT` or `DELETE` statement.

Example:

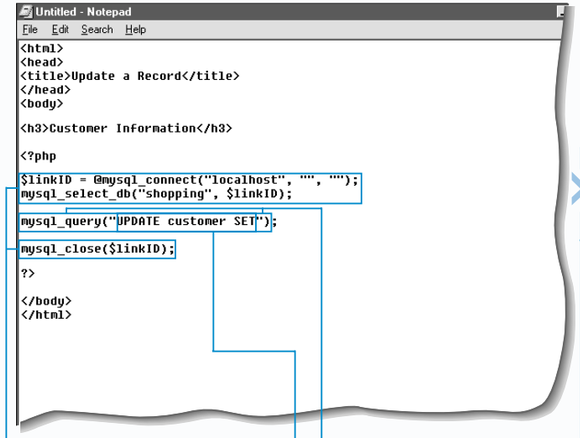
```
$result = mysql_query("UPDATE customer SET lastName = 'Edwards'
      WHERE firstName = 'Martine'", $linkID);
print mysql_affected_rows($linkID) . " record(s) were updated.";
```

You may want to store SQL statements in variables and then pass a variable as an argument to the `mysql_query` function. Using variables to store SQL statements is a common programming practice and can make your code easier to read and update. Storing SQL statements in variables also provides a convenient way to switch between statements when troubleshooting a script. You simply place two slashes (`//`) in front of each statement you temporarily do not want to use.

Example:

```
// $query = "UPDATE customer SET lastName = 'Smith' WHERE customerID = 109";
// $query = "UPDATE customer SET firstName = 'Joe' WHERE customerID = 107";
$query = "UPDATE customer SET customerID = 103 WHERE firstName > 'Sandy'";
$result = mysql_query($query, $linkID);
```

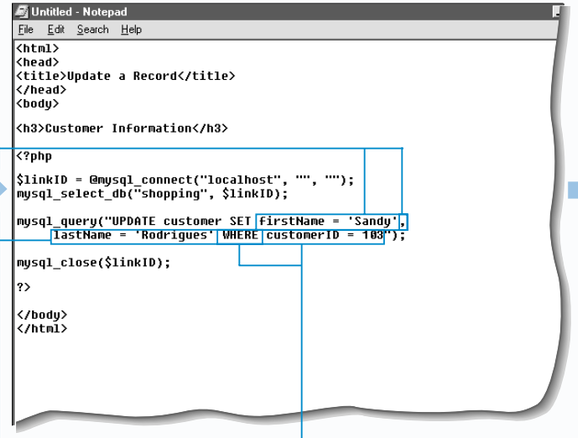
UPDATE A RECORD



1 Type the code that connects the PHP page to the MySQL server and selects the database you want to work with.

2 To update a record in the database, type **mysql_query("")**.

3 Between the quotation marks, type **UPDATE** followed by the name of the table you want to update. Then type **SET**.

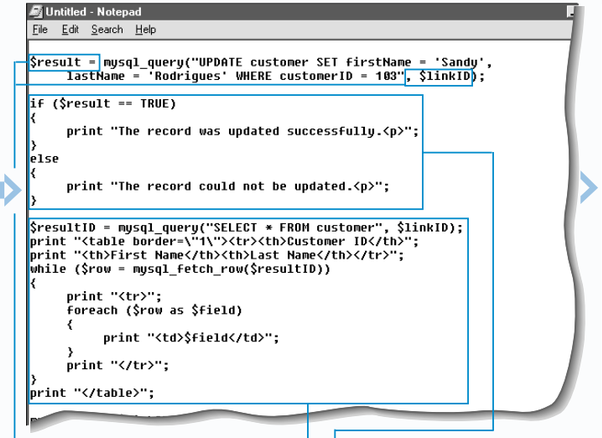


4 Type the name of the field you want to change followed by **=**. Then type the new value you want to assign.

5 Repeat step 4 for each field you want to change, separating each field with a comma.

6 To specify which records to update, type **WHERE**.

7 Type the name of the field followed by **=**. Then type the value that identifies the records to be modified.

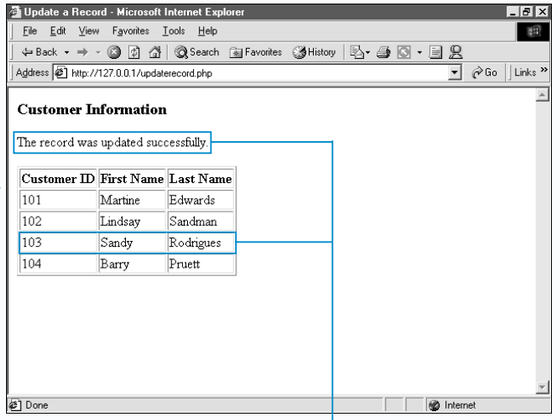


8 Outside the quotation marks, type a comma followed by the link identifier.

9 Type the code that assigns the result of the `mysql_query` function to a variable.

10 To test if the update was made successfully, type the code that checks the value of the variable storing the `mysql_query` function.

11 Type the code that displays the information in the database.



12 Display the PHP page in a Web browser.

The Web browser displays the result of updating a record in a database.

DELETE A RECORD

After creating a connection to a MySQL server in a PHP page and selecting the database you want to work with, you can use the DELETE statement to remove a record from the database.

When deleting a record from a database, you use the mysql_query function to issue a DELETE statement to MySQL server. The mysql_query function returns a value of true if the record is deleted successfully. Otherwise, the function returns a value of false.

The DELETE statement uses the FROM clause to specify the name of the table that stores the record you want to delete. The WHERE clause indicates information that uniquely identifies the record you want to delete. You should be careful when identifying the record you want to delete, as information accidentally deleted from a database often cannot be recovered.

When identifying the information in the record you want to delete, a string value must be enclosed in single quotation marks (''). If the value contains quotation marks, you must escape the quotation marks by placing the backslash character (\) before each quotation mark, such as \' and \".

After the DELETE statement, you can specify the link identifier for the MySQL server connection you want to use. If you simply want to use the last connection that was made to the server, you can omit the link identifier from the mysql_query function.

If you want to remove all the records from a table at one time, you can use the DELETE statement without including the WHERE clause. This will delete all the data in the table but will not delete the table from the database.

The MySQL server requires that appropriate permissions be granted before a script can delete information from a database. For information about setting permissions in a MySQL server, consult the MySQL documentation at the www.mysql.com Web site.

Extra

Multiple records can be removed from a table at one time by specifying a range of values in the WHERE clause of the DELETE statement. For example, if you have a field that stores the last date on which a customer made a purchase, you can use an expression with the WHERE clause to delete all the records of customers who made their last purchase before a specific date.

Example:

```
$result = mysql_query("DELETE FROM customer
      WHERE lastPurchase < '2000-04-23'", $linkID);
```

To delete an entire table and all the data stored in the table from a database, you can use the DROP TABLE statement with the mysql_query function. After the DROP TABLE statement, you must specify the name of the table you want to delete. You should be careful not to delete a table that you might need at a later time.

Example:

```
$result = mysql_query("DROP TABLE oldCustomers", $linkID);
```

You can remove a field from a table by using the ALTER TABLE statement with the mysql_query function. After specifying the table you want to affect, the DROP clause is used to specify the field you want to remove. When a field is removed, all the data in the field is also deleted.

Example:

```
$result = mysql_query("ALTER TABLE customer
      DROP middleName", $linkID);
```

DELETE A RECORD

Untitled - Notepad

```
<html>
<head>
<title>Delete A Record</title>
</head>
<body>

<h3>Customer Information</h3>

<?php

$linkID = mysql_connect("localhost", "", "");
mysql_select_db("shopping", $linkID);

mysql_query("DELETE FROM customer");

mysql_close($linkID);

?>

</body>
</html>
```

1 Type the code that connects the PHP page to the MySQL server and selects the database you want to work with.

2 To delete a record from the database, type **mysql_query("")**.

3 Between the quotation marks, type **DELETE FROM** followed by the name of the table from which you want to delete a record.

Untitled - Notepad

```
<html>
<head>
<title>Delete A Record</title>
</head>
<body>

<h3>Customer Information</h3>

<?php

$linkID = mysql_connect("localhost", "", "");
mysql_select_db("shopping", $linkID);

mysql_query("DELETE FROM customer
      WHERE customerID = 103", $linkID);

mysql_close($linkID);

?>

</body>
</html>
```

4 To specify the record to delete, type **WHERE**.

5 Type the name of the field followed by **=**. Then type the value in the field that identifies the record to be deleted.

6 Outside the quotation marks, type a comma followed by the link identifier.

Untitled - Notepad

```
mysql_select_db("shopping", $linkID);

$result = mysql_query("DELETE FROM customer
      WHERE customerID = 103", $linkID);

if ($result == TRUE)
{
      print "The record was successfully deleted.<p>";
}
else
{
      print "The record could not be deleted.<p>";
}

$resultID = mysql_query("SELECT * FROM customer", $linkID);
print "<table border=\"1\"><tr><th>Customer ID</th>";
print "<th>First Name</th><th>Last Name</th></tr>";
while ($row = mysql_fetch_row($resultID))
{
      print "<tr>";
      foreach ($row as $field)
      {
            print "<td>$field</td>";
      }
      print "</tr>";
}
print "</table>";
```

7 Type the code that assigns the value returned by the mysql_query function to a variable.

8 To test if the record was deleted successfully, type the code that checks the value of the variable storing the mysql_query function.

9 Type the code that displays information from the database in the PHP page.

Delete A Record - Microsoft Internet Explorer

Address http://127.0.0.1/deleterecord.php

Customer Information

The record was successfully deleted.

Customer ID	First Name	Last Name
102	Lindsay	Sandman
104	Barry	Pruett
109	Martine	Edwards

10 Display the PHP page in a Web browser.

■ The Web browser displays the result of deleting a record from a database.

RETRIEVE INFORMATION AS AN ASSOCIATIVE ARRAY

After using the SELECT statement with the mysql_query function to retrieve information from a database and place it in a result set, you can use the mysql_fetch_assoc function to retrieve information from the result set as an associative array.

The mysql_fetch_assoc function returns an associative array in which the keys are the same as the names of the fields in the result set and the array values correspond to the values in the fields. Many programmers assign the result of the mysql_fetch_assoc function to a variable to make the associative array easier to work with.

Accessing data as an associative array can make your PHP code more readable because it identifies exactly which fields are being used instead of identifying the fields only by number. Using associative arrays also means that you do not need to consider the order in which the fields appear in the result set. This may help prevent problems that may arise if the structure of the table were to change in the future.

To call the mysql_fetch_assoc function, you must pass the result identifier that points to the data stored in the result set as an argument. If the function is successful, the associative array is returned. If there is no information in the result set, the mysql_fetch_assoc function returns a value of false.

To access a value from the associative array, you specify the name of the variable that stores the result of the mysql_fetch_assoc function followed by brackets []. Between the brackets, specify the name of the field that stores the value you want.

The mysql_fetch_assoc function is often used with a loop to allow access to all the information from the result set. You can use the mysql_num_rows function in the loop code to indicate when the loop should end. The mysql_num_rows function determines the number of rows returned by the SELECT statement issued by the mysql_query function.

Extra

The mysql_fetch_object function allows you to retrieve information from a result set as an object. This function returns an object with property names that are the same as the field names in the table. To access the value of a field, specify the name of the variable that stores the object followed by the member access operator (->) and the field name.

Example:

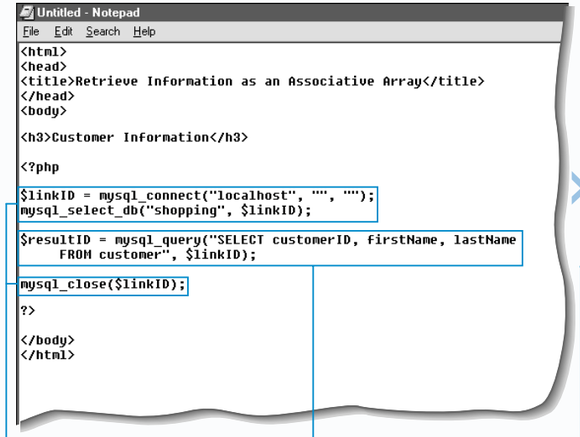
```
while ($row = mysql_fetch_object($resultID))
{
      print "<tr>";
      print "<td>" . $row->customerID . "</td>";
      print "<td>" . $row->firstName . "</td>";
      print "<td>" . $row->lastName . "</td>";
      print "</tr>";
}
```

The mysql_fetch_array function is used to retrieve information from a database as an array that contains both numeric and associative keys. The mysql_fetch_array function returns an array in which each field value appears twice—once with a numeric key and once with an associative key. This duplication of values makes the mysql_fetch_array function inefficient for retrieving large amounts of information from a result set. You should use this function only when both the numeric order and the field names to which the values belong are needed.

Example:

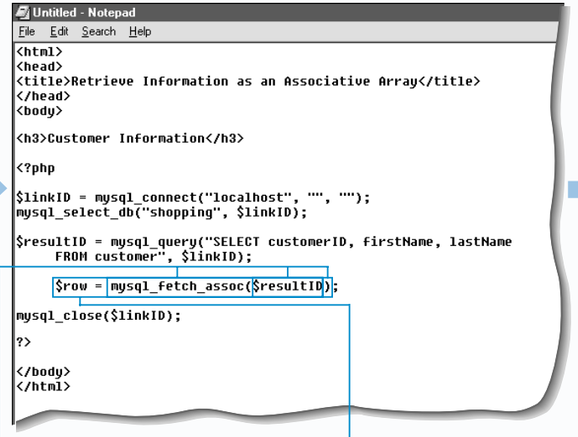
```
while ($row = mysql_fetch_array($resultID))
{
      print "<tr>";
      print "<td>" . $row[0] . "</td>";
      print "<td>" . $row[firstName] . "</td>";
      print "<td>" . $row[2] . "</td>";
      print "</tr>";
}
```

RETRIEVE INFORMATION AS AN ASSOCIATIVE ARRAY



1 Type the code that connects the PHP page to the MySQL server and selects the database you want to work with.

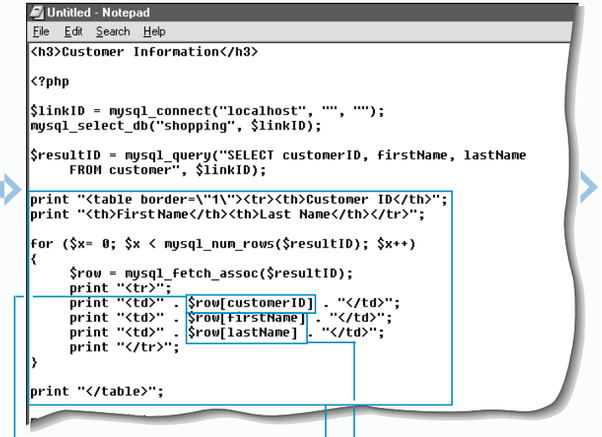
2 Type the code that uses a mysql_query function to retrieve information from the database and assigns the result identifier returned by the mysql_query function to a variable.



3 To retrieve information from the result set as an associative array, type **mysql_fetch_assoc()**.

4 Between the parentheses, type the name of the variable that stores the result identifier.

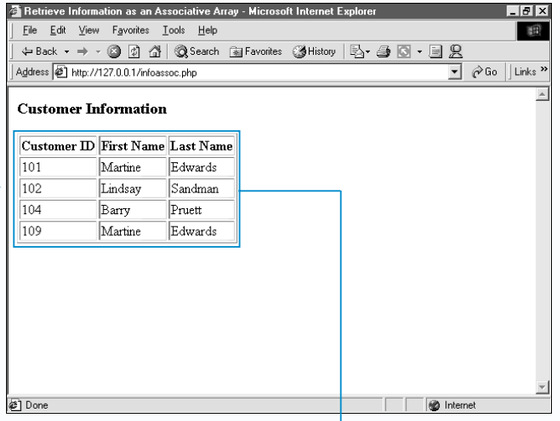
5 Type the code that assigns the result of the mysql_fetch_assoc function to a variable.



6 To access a value from the result set, type the name of the variable that stores the associative array. Then type the name of the field that stores the value you want to access, enclosed in brackets [].

7 Repeat step 6 for each value you want to retrieve.

8 Type the code that uses the information retrieved as an associative array and formats the information for display.



9 Display the PHP page in a Web browser.

■ The Web browser displays the result of retrieving information from a database as an associative array.

RETRIEVE INFORMATION FROM MULTIPLE TABLES

One of the most useful features of a relational database is the ability to retrieve data from multiple tables and then combine the data into a single result set.

You use the SELECT statement to combine related data from multiple tables. This operation is referred to as a join. To select a field for a join, you specify the name of the table containing the field and the name of the field, separated by a dot(.). You can select as many fields as you need for a join. You use the FROM clause to specify the names of the two tables from which you want to retrieve information.

The WHERE clause allows you to specify the relationship between the tables being joined. Each table has a field that stores a unique identifier for each record, referred to as a primary key. To specify the relationship between the two tables, you specify the name of the first table and the name of the primary key for that table. You then enter the name of the second table and the name of the field that stores the same values as those stored in the primary key of the first table. If the unique identifier for a record in the first table matches information in the second table, then the record will be retrieved from both tables. For example, you may want to join data from two tables called customer and orders. The customer table has a primary key field that stores a unique customer id for each customer, as well as fields containing information such as first and last names. The orders table contains the orders made by each customer and includes each customer's id. You can join data from the two tables to create a report that includes specific information for each customer who has placed an order.

Extra

You can use the MySQL AS construct to specify a different name for a field you are retrieving. Specifying a different name for a field can be particularly useful when using the mysql_fetch_assoc function. The associative array returned by this function is indexed using only field names. This can cause problems if you retrieve fields that have the same name from different tables.

Example:

```
SELECT orders.totalCost AS orderTotal,
shipping.totalCost AS shippingTotal FROM orders,
shipping WHERE orders.customerID = shipping.customerID
```

You can use the MySQL CONCAT function to combine data from different fields into a single field. To use the CONCAT function, you must specify the names of the fields you want to retrieve, separated by commas. You can also include a string, such as a space (' '), to separate the fields. You can then use the AS construct to specify a name for the new field.

Example:

```
SELECT CONCAT(firstName, ' ', lastName) AS fullName,
customerID FROM customer
```

When you want to display all the records from one of the tables in a join, you can use the LEFT JOIN clause. This clause allows you to retrieve all the records from a table, regardless of whether the records have corresponding entries in the second table. To use the LEFT JOIN clause, specify the name of the table from which you want to retrieve all the records directly following the FROM clause. Type LEFT JOIN followed by the name of the second table. Then type ON followed by the relationship between the two tables.

Example:

```
SELECT customer.firstName, customer.lastName,
orders.itemDescription, orders.quantity,
orders.purchaseDate FROM orders LEFT JOIN
customer ON customer.customerID =
orders.customerID
```

RETRIEVE INFORMATION FROM MULTIPLE TABLES

Untitled - Notepad

File Edit Search Help

```
<html>
<head>
<title>Join Two Tables</title>
</head>
<body>

<?php

$linkID = @mysql_connect("localhost", "", "");

mysql_select_db("shopping");

mysql_query("SELECT customer.firstName,
customer.lastName, orders.itemDescription,
orders.quantity, orders.purchaseDate");

mysql_close($linkID);

?>

</body>
</html>
```

1 To issue an SQL statement to the MySQL server, type **mysql_query**("").

2 Between the quotation marks, type **SELECT**.

3 To specify a field for the join, type the name of the table followed by a dot and the name of the field.

4 Repeat step 3 until you have specified all the fields for the join, separating each table and field name pair with a comma.

Untitled - Notepad

File Edit Search Help

```
<html>
<head>
<title>Join Two Tables</title>
</head>
<body>

<?php

$linkID = @mysql_connect("localhost", "", "");

mysql_select_db("shopping");

mysql_query("SELECT customer.firstName,
customer.lastName, orders.itemDescription,
orders.quantity, orders.purchaseDate FROM customer,
orders WHERE customer.customerID");

mysql_close($linkID);

?>

</body>
</html>
```

5 Type **FROM** followed by the names of the tables containing information you want to retrieve, separated by commas.

6 To specify the relationship between the tables, type **WHERE**.

7 Type the name of the first table followed by a dot and the name of the primary key field for the table.

Untitled - Notepad

File Edit Search Help

```
$resultID = mysql_query("SELECT customer.firstName,
customer.lastName, orders.itemDescription,
orders.quantity, orders.purchaseDate FROM customer,
orders WHERE customer.customerID = orders.customerID");

print "<table>\n";
print "<tr align=\"left\"><th>Name</th>
<th>Description</th><th>
<th>Quantity</th><th>Date</th></tr>\n";

while ($row = mysql_fetch_row($resultID))
{
        print "<tr>\n";
        foreach($row as $field)
        {
                print "<td>$field</td>\n";
        }
        print "</tr>\n";
}

print "</table>\n";

mysql_close($linkID);
```

8 Type = followed by the name of the second table and a dot. Then type the name of the field that stores the same values as those stored in the primary key of the first table.

9 Type the code that assigns the result of the mysql_query function to a variable.

10 Type the code that uses the mysql_fetch_row function to access and retrieve data and formats the data for display.

Join Two Tables - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Back Search Favorites History

Address http://127.0.0.1/retrieve.php Go Links

Name	Surname	Description	Quantity	Date
Barry	Pruett	Golf Balls	2	2001-04-23
Martine	Edwards	Tennis Balls	4	2001-04-24
Martine	Edwards	Tennis Rackets	3	2001-04-24
Barry	Pruett	Hockey Pucks	5	2001-04-24

Done Internet

11 Display the PHP page in a Web browser.

■ The Web browser displays the result of retrieving information from multiple tables.

ACCESS A PORTION OF A RESULT SET

After information has been retrieved from a database and placed in a result set, you can use the `mysql_data_seek` function to access a portion of the result set. The `mysql_data_seek` function advances the row pointer of the result set to a row you specify. This allows you to work with a small portion of a large result set without having to iterate through the entire result set.

The `mysql_data_seek` function is typically used to separate a large number of records to be displayed on a Web page. For example, if a user's query retrieves a large number of records from a database, you may want to display the records on the Web page in small segments at a time instead of displaying all the records at once. Handling a large result set in this manner is very efficient, since you retrieve only the data you need.

To use the `mysql_data_seek` function, you must specify the result identifier that points to the data stored in the result set. You also specify the offset position of the row you want to access in the result set. The offset positions of the rows in a result set are numbered starting from zero (0). If you specify an offset position that is equal to or greater than the number of rows in the result set, an error will be generated.

The `mysql_data_seek` function will return a value of true or false depending on whether the function was executed successfully.

Once the pointer is positioned at the first row you want to access in the result set, you can call the `mysql_fetch_row` function to retrieve the information in the row. The pointer is then advanced to the next row. To retrieve subsequent rows, you can call the `mysql_fetch_row` function repeatedly using code that creates a loop, such as a `for` statement.

Extra

You can use the `mysql_result` function to retrieve a single value from a result set. To use the `mysql_result` function, you specify the result identifier for the result set and the offset position of the row that contains the value you want to retrieve. The `mysql_result` function will return the value of the first field in the specified row.

TYPE THIS:

```
$resultID = mysql_query("SELECT customer.firstName, customer.lastName,
  orders.itemDescription, orders.quantity, orders.purchaseDate
  FROM customer, orders WHERE customer.customerID = orders.customerID",
  $linkID);
print mysql_result($resultID, 3);
```

RESULT:

Martine

When using the `mysql_result` function, you can specify the value you want to retrieve from a row. To retrieve a value from a specific field in the row, you can include a third argument to specify the offset position of the field. You can also specify the name of the field instead of the offset position of the field. Another option allows you to prefix the field name with the name of the table from which you want to retrieve the data. This is useful when working with a result set created from multiple tables.

Example:

```
mysql_result($resultID, 3, 4);
```

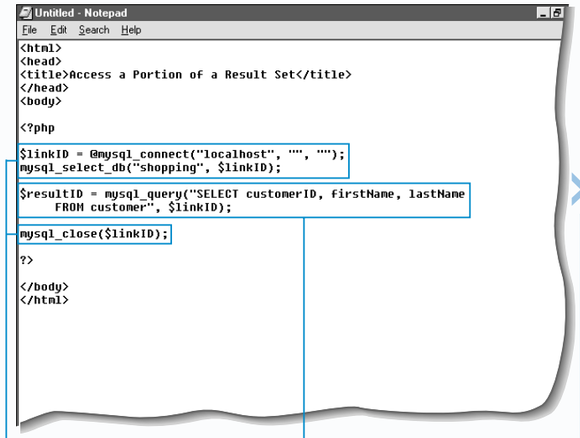
Can be typed as:

```
mysql_result($resultID, 3, "purchaseDate");
```

Or

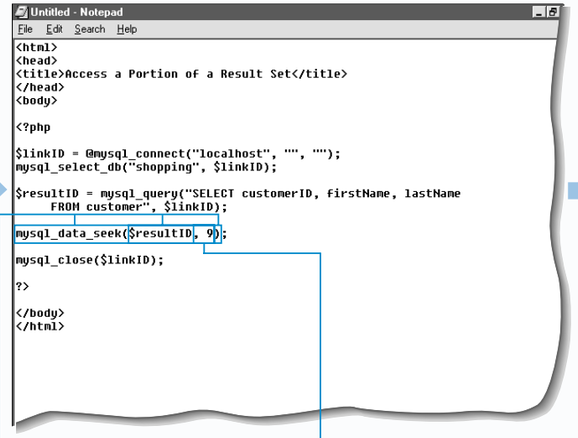
```
mysql_result($resultID, 3, "orders.purchaseDate");
```

ACCESS A PORTION OF A RESULT SET



1 Type the code that connects the PHP page to the MySQL server and selects the database you want to work with.

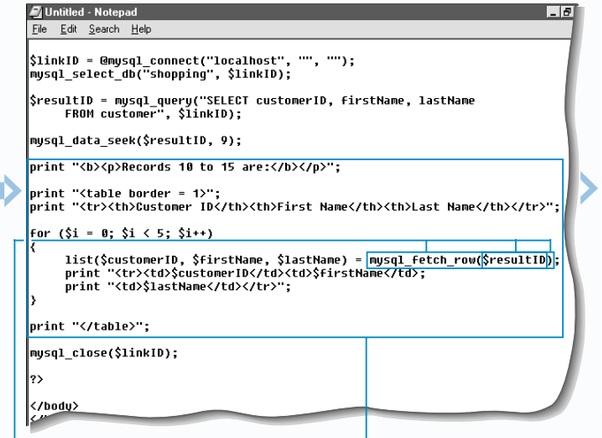
2 Type the code that uses a `mysql_query` function to retrieve information from the database and assigns the result identifier returned by the `mysql_query` function to a variable.



3 To move the row pointer to a specific row in the result set, type **mysql_data_seek**().

4 Between the parentheses, type the name of the variable that stores the result identifier.

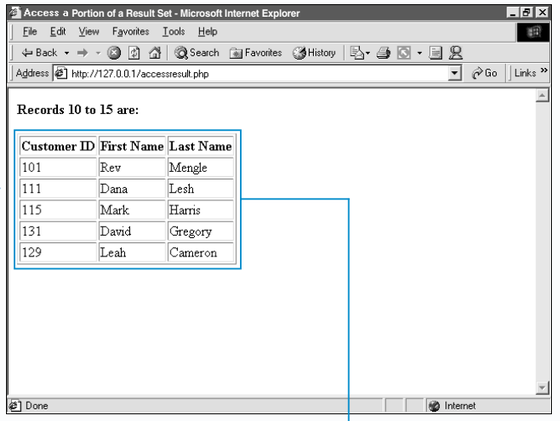
5 Type a comma, followed by the number that specifies the offset position of the row where you want to place the pointer.



6 To retrieve the row at which the pointer is positioned, type **mysql_fetch_row**().

7 Between the parentheses, type the name of the variable that stores the result identifier.

8 Type the code that processes the result of the `mysql_fetch_row` function and formats the information for display.



9 Display the PHP page in a Web browser.

■ The Web browser displays the result of accessing a portion of a result set.

GET INFORMATION ABOUT FIELDS

The `mysql_list_fields` function returns a result identifier that can be used to obtain information about the fields in a table. You may find it useful to retrieve information such as the names or data types of fields before making an SQL query. The `mysql_list_fields` function takes three arguments—the name of a database, the name of the table containing the fields you want to work with and the link identifier of the MySQL server connection. If you do not specify a link identifier, the last successful connection to the MySQL server will be used. The `mysql_list_fields` function will return a result identifier if successful, otherwise it will return a value of false.

You can obtain the number of fields in a table by using the `mysql_num_fields` function. When calling the `mysql_num_fields` function, you must pass the result identifier obtained from the `mysql_list_fields` function as an argument. The `mysql_num_fields` function is often used as a counter in a loop that cycles through each field in a table. This type of loop allows you to access information about each field.

There are a number of functions that use the result identifier obtained from the `mysql_list_fields` function to retrieve information about a field. The `mysql_field_name` function is used to retrieve the name of a field. You can use the `mysql_field_type` function to retrieve the data type of a field. The `mysql_field_len` function is used to retrieve the maximum length of a field. You can use the `mysql_field_flags` function to retrieve other information about a field. These functions all take two arguments—the result identifier obtained from the `mysql_list_fields` function and the offset position of the field in the table. The offset positions of the fields are numbered starting from zero (0) and are determined by where the fields appear in the table. An error will occur if an offset position that is equal to or greater than the number of fields in a table is specified.

Extra

You can use functions to obtain information about the tables in a database. To obtain a list of tables in a database, you can use the `mysql_list_tables` function. The function takes two arguments—the name of the database and the link identifier of the MySQL server connection. To determine the number of tables in a database, you can use the `mysql_num_rows` function, which takes the result identifier obtained from the `mysql_list_tables` function as an argument. You can also determine the name of a table by using the `mysql_tablename` function. You must specify the result identifier obtained from the `mysql_list_tables` function and the offset position of the table in the database as arguments for the `mysql_tablename` function.

Example:

```
$resultID = mysql_list_tables("shopping", $linkID);
for ($tablePos = 0; $tablePos < mysql_num_rows($resultID); $tablePos++)
{
      print mysql_tablename($resultID, $tablePos) . "<br>";
}
```

You can obtain a list of databases on a MySQL server by using the `mysql_list_dbs` function. The `mysql_list_dbs` function takes the link identifier of the MySQL server connection as an argument. You can then use the `mysql_num_rows` function to determine the number of databases in the list. You can also determine the name of a database by using the `mysql_tablename` function.

Example:

```
$resultID = mysql_list_dbs($linkID);
for ($dbPos = 0; $dbPos < mysql_num_rows($resultID); $dbPos++)
{
      print mysql_tablename($resultID, $dbPos) . "<br>";
}
```

GET INFORMATION ABOUT FIELDS

Untitled - Notepad

File Edit Search Help

```
<html>
<head>
<title>Get Information About Fields</title>
</head>
<body>

<?php

$linkID = mysql_connect("localhost", "", "");

mysql_list_fields("shopping", "customer", $linkID);

print "<table border=\"1\">";
print "<tr><th>Name</th><th>Type</th>";
print "<th>Length</th><th>Flags</th></tr>";

print "</table>";

mysql_close($linkID);

?>

</body>
</html>
```

1 Type the code that connects the PHP page to the MySQL server.

2 To retrieve information about a table, type **mysql_list_fields**().

3 Between the parentheses, type the name of the database, enclosed in quotation marks.

4 Type a comma followed by the name of the table, enclosed in quotation marks. Then type a comma followed by the link identifier.

Untitled - Notepad

File Edit Search Help

```
<html>
<head>
<title>Get Information About Fields</title>
</head>
<body>

<?php

$linkID = mysql_connect("localhost", "", "");

$resultID = mysql_list_fields("shopping", "customer", $linkID);

print "<table border=\"1\">";
print "<tr><th>Name</th><th>Type</th>";
print "<th>Length</th><th>Flags</th></tr>";

for ($fieldPos = 0; $fieldPos < mysql_num_fields($resultID);
     $fieldPos++)
{

}

print "</table>";
```

5 Type the code that assigns the result identifier returned by the `mysql_list_fields` function to a variable.

6 To obtain the number of fields in a table, type **mysql_num_fields**().

7 Between the parentheses, type the result identifier.

8 Type the code that uses the `mysql_num_fields` function.

Untitled - Notepad

File Edit Search Help

```
print "<tr><th>Name</th><th>Type</th>";
print "<th>Length</th><th>Flags</th></tr>";

for ($fieldPos = 0; $fieldPos < mysql_num_fields($resultID);
     $fieldPos++)
{
     print "<tr><td>";
     print mysql_field_name($resultID, $fieldPos) . "</td><td>";
     print mysql_field_type($resultID, $fieldPos) . "</td><td>";
     print mysql_field_len($resultID, $fieldPos) . "</td><td>";
     $flags = mysql_field_flags($resultID, $fieldPos);
     if ($flags != FALSE)
     {
          print $flags;
     }
     else
     {
          print "none";
     }
     print "</td></tr>";
}

print "</table>";
```

9 To obtain information about a specific field, type the name of the function you want to call followed by **()**.

10 Between the parentheses, type the result identifier. Then type a comma followed the offset position of the field.

11 Repeat steps 9 and 10 for each function you want to call.

12 Type the code the uses the results of the functions you called.

Get Information About Fields - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Back Search Favorites History

Address http://127.0.0.1/fieldsinfo.php Go Links

Name	Type	Length	Flags
customerID	int	4	none
firstName	string	15	none
lastName	string	25	none

Done Internet

13 Display the PHP page in a Web browser.

■ The Web browser displays the result of obtaining information about the fields in a table.

USE A FORM TO WORK WITH A DATABASE

Forms provide an easy-to-use interface for manipulating data in a database and can be used to issue SQL statements to a MySQL database. The form passes an SQL statement to a PHP page that connects to the database, processes the statement and displays the result.

When creating the code for a form that will be used to issue SQL statements to a database, you must specify the name of the PHP page that connects to the database in the `action` attribute of the `<form>` tag. You can use the `<textarea>` tag to create a text area element on the form that users can enter SQL statements into. The PHP page will use the value of the `name` attribute for the `<textarea>` tag to access the SQL statement passed by the form.

The PHP page used to process the form must have a connection to the MySQL server and select the database you want to work with. The PHP page must also contain the `mysql_query` function, which will process the SQL statement passed by the form.

If the result of the `mysql_query` function is of the boolean data type and has a value of false, the SQL statement passed by the form contains an error. You should create an error message that will be displayed in this situation.

If the returned result is of the boolean data type and has a value of true, the SQL statement was successfully executed. You may want to use the `mysql_affected_rows` function to display the number of records that were affected by the query.

When the returned result is not of the boolean data type, the SQL statement returned a result set. You can use the `mysql_num_rows` function to display the number of rows in the result set. You can also retrieve the records in the result set using the `mysql_fetch_row` function.

Apply It

When a user submits an SQL statement that generates an error, you can have the PHP page display information about the error that the user may find helpful. You can use the `mysql_errno` function to retrieve the error number from the MySQL server and the `mysql_error` function to retrieve the error message.

Example:

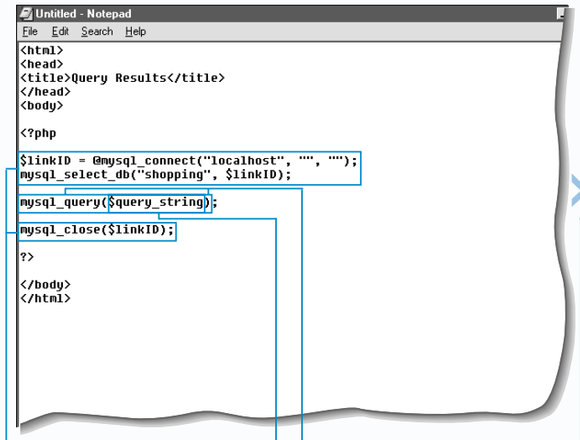
```
if ((gettype($resultID) == "boolean") && ($resultID == FALSE))
{
print "There was an error in the SQL statement.<br>";
print "Error #" . mysql_errno($linkID) . ": " . mysql_error($linkID);
}
```

When creating a table that displays retrieved records, you can use the `mysql_num_fields` function to retrieve the number of fields in the result set. You can use the result of this function to create a `for` loop that uses the `mysql_field_name` function to retrieve all the field names.

Example:

```
print "<table border=\"1\">";

print "<tr>";
for ($i = 0; $i < mysql_num_fields($resultID); $i++)
{
    print "<th>" . mysql_field_name($resultID, $i) . "</th>";
}
print "</tr>";
```

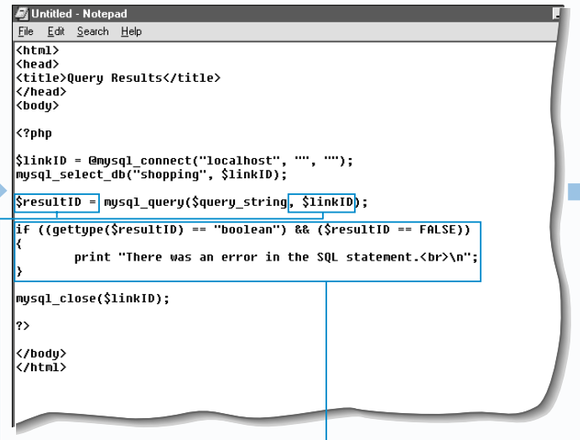
USE A FORM TO WORK WITH A DATABASE



1 Type the code that connects the PHP page to the MySQL server and selects the database you want to work with.

2 To have the PHP page accept and process an SQL statement passed by a form, type **mysql_query()**.

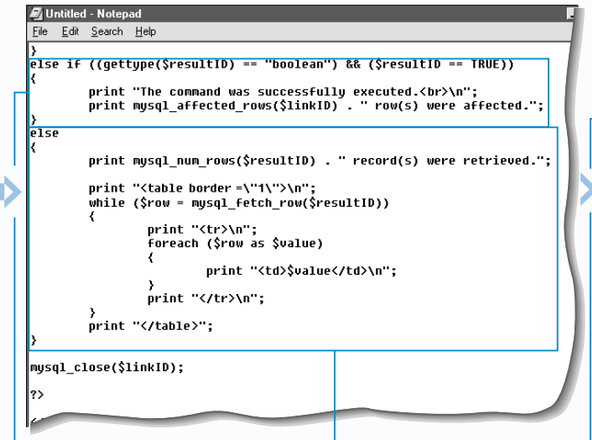
3 Between the parentheses, type **$** followed by the value assigned to the name attribute of the `<textarea>` tag in the form.



4 Type a comma followed by the name of the variable that stores the link identifier.

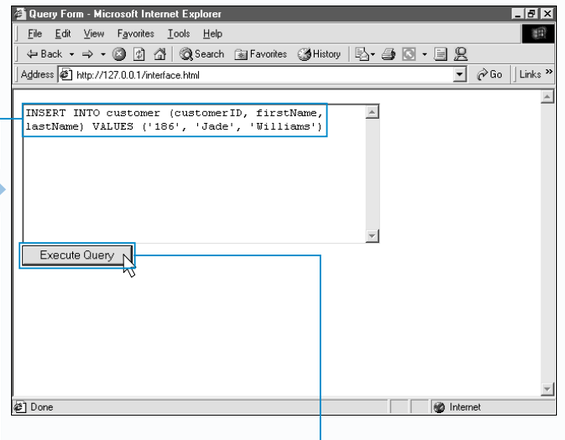
5 Type the code that assigns the result of the `mysql_query` function to a variable.

6 Type the code that determines if the result of the `mysql_query` function is of the boolean data type and is false. Then type the code you want to process in this situation.



7 Type the code that determines if the result of the `mysql_query` function is of the boolean data type and is true. Then type the code you want to process in this situation.

8 Type the code you want to process when the result of the `mysql_query` function is of a data type other than boolean.



9 In a Web browser, display the form you created to issue SQL statements to the database.

10 Enter an SQL statement into the form.

11 Click the submit button to pass the SQL statement to the PHP page.

■ The PHP page that processes the SQL statement will appear and the database will be modified.