

CSCI 241

Querying Web Databases

Department of CS & QM
Winthrop University
Fall 2011

Prepared by X. Wang

Topics

- Querying MySQL Database in PHP
- Handling MySQL Server Errors
- Using `require` Files
- Processing User Input
- Querying Databases with User Data
- User Behavior Diagram

Prepared by X. Wang

2

Querying Database in PHP

- In PHP, we query a database stored in a Database Server by executing SQL statements through PHP functions
- PHP also provides functions to manage result sets returned from queries, and handle errors
- Here we only discuss the PHP functions related to the MySQL database server
- For other database servers, please read the corresponding sections listed in PHP online manual

Prepared by X. Wang

3

Querying Database in PHP

- To query an MySQL database in PHP scripts, we need to have the following five steps
1. Open the database connection
 2. Select a database
 3. Construct and execute an SQL query statement
 4. Process the result set in row by row
 5. Generate output with the attribute values in a row

Prepared by X. Wang

4

Querying Database in PHP

- For demonstrating the database querying code, assume that we create the following `customers` table in the `wangx` database with the `user` and `password` as `wangx/wang123`

```
CREATE TABLE customers (
  cust_id int(8) NOT NULL auto_increment,
  lastname varchar(50) NOT NULL,
  firstname varchar(50) NOT NULL,
  email varchar(50) NOT NULL,
  password varchar(32) NOT NULL,
  address varchar(50),
  city varchar(30),
  state varchar(20),
  zipcode varchar(10),
  phone varchar(15),
  PRIMARY KEY (cust_id)
) ENGINE=MyISAM;
```

Prepared by X. Wang

5

Querying Database in PHP

- We also insert the following rows into the `customers` table

```
INSERT INTO customers VALUES
(NULL, 'Wang', 'Xusheng', 'wangx@winthrop.edu', MD5('abcd'),
'1001 University Dr', 'Rock Hill', 'SC', '29732', '(803)123-4567');
INSERT INTO customers VALUES
(NULL, 'Cook', 'Peter', 'cookp@yahoo.com', MD5('123'),
'2189 West Blvd', 'Charlotte', 'NC', '28208', '(704)987-6543);
INSERT INTO customers VALUES
(NULL, 'Perry', 'Frank', 'perryf@gmail.com', MD5('123'),
'637 Blue Moon Road', 'Houston', 'TX', '77006', '(713)222-2222');
INSERT INTO customers VALUES
(NULL, 'Freeman', 'Larry', 'lfreeman@yahoo.com', MD5('168'),
'5200 Richmand Ave', 'Atlanta', 'GA', '30303', '(404)909-8080);
```

Prepared by X. Wang

6

Querying Database in PHP

- The following code ([query01.php](#)) is a complete PHP file that shows the five steps to query data from the customers table

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<title>Customers</title>
</head>
<body>
<?php
// (1) Open the database connection with the anonymous user
$conn = mysql_connect("localhost", "wangx", "wang123");
```

Prepared by X. Wang

7

Querying Database in PHP

```
// (2) Select the test database
mysql_select_db("wangx", $conn);
// (3) Run the query on the customer table through the connection
$result = mysql_query("SELECT * FROM customers", $conn);
// (4) While there are still rows in the result set, fetch the current row into $row
while ($row = mysql_fetch_array($result, MYSQL_NUM))
{
// (5) Print out each element in $row, that is, print the values of the attributes
foreach ($row as $attribute) print "{$attribute}, ";
// Can do the same thing with
// for ($i=0; $i<count($row); $i++) print("{$row[$i]} ");
// Print a carriage return to neaten the output
print "<br />";
}
?>
</body>
</html>
```

Prepared by X. Wang

8

Querying Database in PHP

- ❑ Open the database connection
- To connect to the MySQL database server, we use the function `mysql_connect()`
- The syntax of this function is listed in the PHP manual at <http://www.php.net/manual/en/function.mysql-connect.php>

```
resource mysql_connect ([string server [, string username [, string password [, bool new_link [, int client_flags]]]])
```
- In this function, we can specify the DB server name, username and password
- If MySQL server is installed on the same computer as the PHP engine, the server name is "localhost"

Prepared by X. Wang

9

Querying Database in PHP

- ❑ Open the database connection
- The function `mysql_connect()` returns a *connection resource* (one special data type in PHP)
- This resource can be used later for other functions to work with the database server
- The variable `$connection` keeps the returned resource
- Note: all functions related to MySQL are beginning with "mysql"

Prepared by X. Wang

10

Querying Database in PHP

- ❑ Select a database
- In MySQL, we need to select the database before we can do a query
- The function `mysql_select_db()` is used to select a database. Its syntax is

```
bool mysql_select_db(string database_name [, resource link_identifier])
```
- This function just matches the `use` command in the MySQL Command Interpreter
- The first parameter of the function is the *database name*
- The second parameter is optional. It is the *connection resource* returned in the `mysql_connect()`

Prepared by X. Wang

11

Querying Database in PHP

- ❑ Construct and execute an SQL Query
- The function `mysql_query()` sends a query to the MySQL server. Its syntax is:

```
resource mysql_query(string query [, resource link_identifier])
```
- The first parameter is the *SQL query statement*
- The second parameter is optional. It is the *connection resource* returned in the `mysql_connect()`
- This function returns a *result set resource* if the query is successful, otherwise a *false*
- This resource will be used in the next step to retrieve data rows

Prepared by X. Wang

12

Querying Database in PHP

- Process the result set in row by row
- This step usually includes a loop
- The function `mysql_fetch_array()` fetches a row from the result set as a numeric array, an associate array, or both


```
array mysql_fetch_array ( resource result [, int result_type] )
```
- If the fetch is successful, it returns an array that corresponds to the fetched row
- If there are no more rows, it returns `false`
- The first parameter is the `result set resource` returned by the function `mysql_query()`

Prepared by X. Wang

13

Querying Database in PHP

- Process the result set in row by row
- The second parameter is optional. It specifies the returned array type with different constants: `MYSQL_BOTH` (the default type), `MYSQL_NUM` and `MYSQL_ASSOC`
- If the second parameter is `MYSQL_NUM`, it is the same as the function `mysql_fetch_row()`. The returned array is indexed with the numbers
- If the second parameter is `MYSQL_ASSOC`, it is the same as the function `mysql_fetch_assoc()`. The returned array is indexed with the associative keys

Prepared by X. Wang

14

Querying Database in PHP

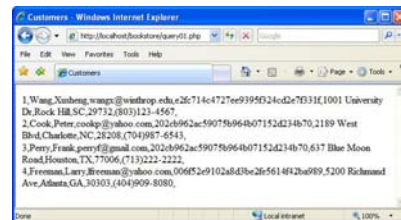
- Generate output with the attributes in a row
- In this step, we process the array for the current row to generate output for the HTML document
- We may need to use different functions of arrays and strings to process the attribute values
- Based on the array type, we may access array elements through `number index` or `associative keys`
- It may also need to use a loop to traverse all attribute values

Prepared by X. Wang

15

Querying Database in PHP

- Query result: This example will display the customer information as follows



Prepared by X. Wang

16

Querying Database in PHP

- Another example (we use the default parameters for the functions) (`query02.php`)
- ```
<?php
$connection = mysql_connect("localhost", "wangx", "wang123");
mysql_select_db("wangx");

$result = mysql_query("SELECT cust_id, lastname, firstname FROM customers");

while ($row = mysql_fetch_array($result))
{
 printf("Customer-ID: %s Name: %s %s\n\n", $row[0], $row[2],
 $row["lastname"]);
}
?>
```

Prepared by X. Wang

17

## Handling MySQL Errors

- Database functions can fail. PHP code needs to handle the errors during function operations
- PHP provides two error-handling functions: `mysql_error()` and `mysql_errno()` for detecting and reporting errors
- `mysql_error()` shows a string message about an error
- `mysql_errno()` shows a number about an error
- Another function `die()` outputs the message and then gracefully ends the script code

Prepared by X. Wang

18

## Handling MySQL Errors

- After a connection is established, any function that returns `false` also sets an error code
- The error code can be retrieved by `mysql_errno()`
- Its associated error message string can be obtained by `mysql_error()`
- The `mysql_connect()` and `mysql_pconnect()` functions **don't** set either the error code or error string on failure
- This failure must be handled manually. We can directly use the `die()` function to output an error message

Prepared by X. Wang

19

## Handling MySQL Errors

- Let's rewrite the first example with error handling (`query03.php`)

```
<?php
function showerror()
{
 die("Error " . mysql_errno() . " : " . mysql_error());
}

// (1) Open the database connection
if (!($connection = @mysql_connect("localhost", "wangx", "wang123")))
 die("Could not connect");

// (2) Select the test database
if (!(@mysql_select_db("wangx", $connection)))
 showerror();
```

Prepared by X. Wang

20

## Handling MySQL Errors

```
// (3) Run the query on the customer table through the connection
// NOTE : 'SELECT' is deliberately misspelt to cause an error
if (!($result = @mysql_query("SELEC * FROM customer", $connection)))
 showerror();

// (4) While there are still rows in the result set,
// fetch the current row into the array $row
while ($row = @mysql_fetch_array($result, MYSQL_NUM))
{
 // (5) Print out each element in $row, that is, print the values of
 // the attributes
 foreach ($row as $attribute)
 print "{$attribute}, ";

 // Print a new line to neaten the output
 print "
";
}
?>
```

Prepared by X. Wang

21

## Handling MySQL Errors

- For a query like the `mysql_query()` function, it may return no results. This is a normal result. It is not an error
- If we want to show a message about this case, we can use the `mysql_num_rows()` function to detect whether the number of returned rows is 0
- In the previous example, we add an `@` character before the function. This will make the system default error message be discarded. Instead, we only want to show an error message with our own string

Prepared by X. Wang

22

## Using require Files

- For developing a Web application, we can save common functions, variables and constants in an external file
- In the following example, we save the `hostname`, `database name`, `user name`, `password` and the `showerror()` function in an external include file `db.inc`

```
<?php
$hostname = "localhost";
$dbusername = "wangx";
$dbpassword = "wang123";
$databasename = "wangx";
function showerror() {
 die("Error " . mysql_errno() . " : " . mysql_error());
}
?>
```

Prepared by X. Wang

23

## Using require Files

- With this external file, we can rewrite the first code as follows (`query04.php`)

```
<?php
require "db.inc";
if (!($connection = @mysql_connect($hostname, $dbusername, $dbpassword)))
 die("Could not connect");
if (!(@mysql_select_db($databasename, $connection)))
 showerror();
if (!($result = @mysql_query("SELECT * FROM customers", $connection)))
 showerror();
while ($row = @mysql_fetch_array($result, MYSQL_NUM))
{
 foreach ($row as $attribute) print "{$attribute}, ";
 print "
";
}
?>
```

Prepared by X. Wang

24

## Using require Files

- In the above example, we store `db.inc` file in the same folder as the PHP source code file
- This means that another web user can directly access this file and find the password for the user
- This presents a minor security problem
- To solve this problem, we have three ways
  1. Save the `db.inc` file to a folder that is not under the root folder for the web server
  2. Configure the web server so that the files with the extension `.inc` are forbidden to be retrieved
  3. Change the extension to `.php` (the simplest way)

Prepared by X. Wang

25

## Formatting Results

- In the previous examples, we simply display results using the `<pre></pre>` element or natural HTML flow
- To create better display structure, we need to use other HTML elements to format the query results
- The HTML `table` is a good way for formatting query results
- The following is an example using table to format the query result (`query05.php`)
- We also change the include file from `db.inc` to `db.php`

Prepared by X. Wang

26

## Formatting Results

```
<?php
require "db.php";
if (!($connection = @ mysql_connect($hostname , $dbusername, $dbpassword)))
 die("Could not connect");

if (!(@ mysql_select_db($databasename, $connection)))
 showerror();

if (!($result = @ mysql_query ("SELECT * FROM customers", $connection)))
 showerror();

$num_result = mysql_num_rows($result);

print("<p>The Customer List</P>");
print("<p>The number of customers: $num_result</p>");
print ' <table border="1"> <tr> <th>Customer ID</th> <th>Name</th> <th>
 Email</th> <th>Address</th> <th>Phone</th>';
```

Prepared by X. Wang

27

## Formatting Results

```
while ($row = @ mysql_fetch_array($result))
{
 print "<tr>";
 print "<td>{$row['cust_id']}</td>";
 print "<td>{$row['firstname']} {$row['lastname']}</td>";
 print "<td>{$row['email']}</td>";
 print "<td>{$row['address']} {$row['city']}, {$row['state']} {$row['zipcode']}</td>";
 print "<td>{$row['phone']}</td>";
 print "</tr>";
}
print "</table>";
?>
```

Prepared by X. Wang

28

## Formatting Results

- The following is a new query result presented with the table element

The screenshot shows a web browser window titled 'Customers - Windows Internet Explorer'. The address bar shows 'http://localhost/bookstore/query05.php'. The page content displays 'The Customer List' and 'The number of customers: 4'. Below this is a table with 5 columns: Customer ID, Name, Email, Address, and Phone. The table contains 4 rows of customer data.

| Customer ID | Name          | Email              | Address                                | Phone         |
|-------------|---------------|--------------------|----------------------------------------|---------------|
| 1           | Xinsheng Wang | wangx@winthrop.edu | 1001 University Dr Rock Hill, SC 29732 | (803)123-4567 |
| 2           | Peter Cook    | ccookp@yahoo.com   | 2189 West Blvd Charlotte, NC 28208     | (704)987-6543 |
| 3           | Frank Perry   | peryf@gmail.com    | 637 Blue Moon Road Houston, TX 77006   | (713)223-2222 |
| 4           | Larry Freeman | lfreema@yahoo.com  | 5200 Richmond Ave Atlanta, GA 30305    | (404)909-8080 |

Prepared by X. Wang

29

## Processing User Input

- Usually, a query result needs to be based on user's input information. This creates user-driven querying
- To create user-driven querying, we need to know the following skills
  1. Pass data from a web browser to a web server
  2. Access user data in PHP scripts
  3. Secure interactive query systems (to be discussed in the future)
  4. Query data using user input data

Prepared by X. Wang

30

## Processing User Input

- Pass data from a web browser to a web server
- Three techniques can be used to pass data from a browser to a server
  1. Use HTML `form` element to enter data
  2. Enter a URL with parameter data (query string) in the browser's address input field
  3. Embed [hypertext links](#) that can be clicked to retrieve a PHP script resource and provide parameters to the script
- Using an HTML form and clicking on hypertext links are two most common techniques for providing user input to a web database application

Prepared by X. Wang

31

## Processing User Input

- Pass data from a web browser to a web server
- With the HTML form, we can pass data in one of two methods: GET or POST
- In the GET method, data is passed as part of the requested URL
- In the POST method, data is encoded separately from the URL and forms part of the body of the HTTP request
- An embedded hypertext link or a manually entered URL with parameters **always uses** the GET method

Prepared by X. Wang

32

## Processing User Input

- Pass user data using the HTML form
- Users enter data into an HTML form that is then encoded by the browser as part of an HTTP request
- The following is an HTML code (`zipcodeform.html`)

```
<body>
<form name="form1" id="form1" method="GET" action="querybyzipcode.php">
Search customers with the zipcode:

Enter a zipcode: <input type="text" name="zipcode" value="All"/>
(Type All to see all zipcodes)

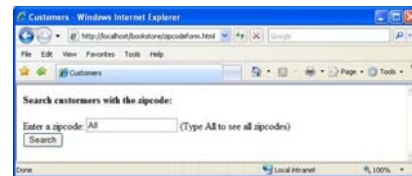
<input type="submit" value="Search" />
</form>
</body>
```

Prepared by X. Wang

33

## Processing User Input

- Pass user data using the HTML form
- The code creates a form as follows



Prepared by X. Wang

34

## Processing User Input

- Pass user data using the HTML form or URL directly
- When clicking the Search button, the browser generates an URL that include the server side program name and parameter strings as follows
 

```
querybyzipcode.php?zipcode=All
```
- This will make the web server to access and execute `querybyzipcode.php`
- We can also type the URL using the above string to run the same PHP code with the same parameter

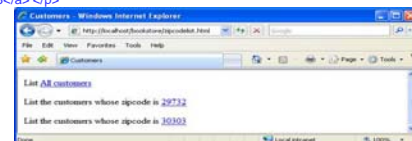
Prepared by X. Wang

35

## Processing User Input

- Pass user data using embedded links
- The following HTML code (`zipodelist.html`) creates links in the web page to allow querying different results

```
<body>
<p>List All customers</p>
<p>List the customers whose zipcode is
29732</p>
<p>List the customers whose zipcode is
30303</p>
</body>
```



Prepared by X. Wang

36

## Processing User Input

- ❑ Access user input data in PHP scripts
- PHP creates an array `$_GET` to save all the parameters passed with the GET method
- For the POST method, the parameters are saved in the array `$_POST`
- The elements in both arrays are key and value pairs
- The key is the `name` property of a form component
- To obtain the zipcode entered by users, we can use `$_GET["zipcode"]`;
- Here `zipcode` is the name of the input text-field

Prepared by X. Wang

37

## Processing User Input

- ❑ Query database using user input data
- After getting user data in scripts, we need to construct SQL statements. The following code shows how to construct a query statement based on the entered zipcode (in `querybyzipcode.php`)

```
<body>
<?php
require "db.php";
if (!($connection = @ mysql_connect($hostname , $dbname, $dbpassword)))
die("Could not connect");

if (!(@ mysql_select_db($databasename, $connection)))
showerror();
```

Prepared by X. Wang

38

## Processing User Input

- ❑ Query database using user input data
- ```
print("<p>The Customer Search Result</p>");
if (!isset($_GET["zipcode"]) || empty($_GET["zipcode"]))
{ print("<p>Please enter a zipcode or All</p></body></html>"); exit; }
$zipcode = $_GET["zipcode"];

$query = "SELECT * FROM customers";
if (isset($zipcode) && $zipcode != "All")
$query .= " WHERE zipcode=({$zipcode})";

if (!($result = @ mysql_query($query, $connection)))
showerror();

$num_result = mysql_num_rows($result);

if ($num_result == 0)
{ print("<p>No customer is found</p></body></html>"); exit; }
```

Prepared by X. Wang

39

Processing User Input

- ❑ Query database using user input data
- ```
print("<p>The number of customers: $num_result</p>");
print "<table border='1'><tr><th>Customer ID</th><th>Name</th>
<th>Email</th><th>Address</th><th>Phone</th>";

while ($row = @ mysql_fetch_array($result))
{
print "<tr>";
print "<td>{$row['cust_id']}</td>";
print "<td>{$row['firstname']} {$row['lastname']}</td>";
print "<td>{$row['email']}</td>";
print "<td>{$row['address']} {$row['city']}, {$row['state']} {$row['zipcode']}</td>";
print "<td>{$row['phone']}</td>";
print "</tr>";
}
print "</table>";
?>
</body>
```

Prepared by X. Wang

40

## Processing User Input

- ❑ Query database using user input data
- The following is the output when zipcode is All

Customer ID	Name	Email	Address	Phone
1	Xiaohua Wang	wangx@windrop.edu	1001 University Dr Rock Hill, SC 29732	(803)121-4567
2	Pratt Cook	cookp@yahoo.com	2189 West Blvd Charlotte, NC 28208	(704)987-6543
3	Frank Perry	perryf@gmail.com	637 Blue Moon Road Houston, TX 77006	(713)222-2222
4	Larry Freeman	lfreema@yahoo.com	5200 Richmond Ave Atlanta, GA 30301	(404)909-8080

Prepared by X. Wang

41

## User Behavior Diagram

- For a database-driven web site, a user may visit different web pages to complete different operations
- We can use a diagram to describe the user's possible behaviors for accessing a web site
- This diagram is called a User Behavior Diagram (UBD)
- This diagram also shows the transition relationships among different web pages in a database-driven web site

Prepared by X. Wang

42

