CSCI 2910
Client/Server-Side Programming

Topic: MySQL in PHP

Reading: Williams & Lane pp. 171–188

Today's Goals

- JavaScript gives us the ability to add dynamic content to our XHTML pages along with the capability to verify the data that a user input into a form
- MySQL gives us the ability to create tables, insert, delete, and view (select) data from a database
- PHP gives us the ability to execute code on the server
- The link between the user and the PHP scripts is provided through the arrays $_GET and $_POST (Note that a "get" can be simulated with a simple URL since a form with a method equal to "get" simply sends the data as a URL)
- The last link to be addressed is between the PHP script at the MySQL. That's what we're doing today!

MySQL Process

Remember the process for accessing data from a database using MySQL:

- Log onto MySQL:
  "mysql -u username -p password"
- Select a database to work with:
  "use database"
- Send a query to one or more tables:
  "select ..."
- MySQL displays results in text on the display
- When your finished, exit MySQL using "exit"

PHP Access to MySQL

The PHP libraries contain functions that allow us to do each of the MySQL operations

- Logging onto MySQL:
  $connection = mysql_connect ("host_URL", "username", "password");
- Selecting a database:
  mysql_select_db("dbname", $connection);
- Querying a table:
  $result = mysql_query("SELECT * FROM tablename", $connection);
- Receiving results: use $result to access data
- Exiting MySQL:
  mysql_close ($connection);

Logging onto MySQL Using PHP

- Syntax:
  $connection = mysql_connect ("host_URL", "username", "password");
- Connecting to the server using the function mysql_connect() takes three parameters:
  - $connection is a variable that is used as a reference to the connection once it has been made.
  - host_URL is the domain name of the MySQL host. "localhost" can be used if MySQL is installed on the same server as the PHP engine
  - "username" represents the username that has privileges to access the database
  - "password" is the password for the username

Selecting a MySQL Database Using PHP

- Syntax:
  mysql_select_db("dbname", $connection);
- Selecting a database using the function mysql_select_db() takes two parameters:
  - "dbname" identifies the name of the database.. For your accounts, your database name is the same as your z-name
  - $connection identifies the connection resource you declared when you established a connection to the MySQL server

Querying a Table Using PHP

- Syntax:
  $result = mysql_query("SELECT * FROM tablename", $connection);
- Querying a table is as simple as creating a string representing the select statement and passing it to the table.
- The first parameter of the function is the MySQL statement in the form of a string.
- The second parameter of the function identifies the connection resource.

Retrieving the Query Data

- Unfortunately, the output $result from the previous function doesn't provide you with anything beyond a reference to the resource where you can find the results. In other words, this isn't just an array of returned records.
- We need to use the function mysql_fetch_array() to access the records returned from the query. This is done one record at a time.
- Syntax:
  $record = mysql_fetch_array($result [, int result_type])

Retrieving the Query Data (continued)

- Pulling a record from the result of a query requires at least one parameter:
  - $result is the reference to the query performed by calling the function mysql_query()
  - result_type is an optional field that defines how the array will be returned.
    - Using **MYSQL_NUM** here will return an array with integer indices/keys.
    - Using **MYSQL_ASSOC** here will return an array using the field names as indices/keys.
    - Using **MYSQL_BOTH** here will return an array with two elements for every field, one with integer indices/keys and one using the field names.
    - Default is **MYSQL_BOTH.**

Retrieving the Query Data (continued)

- The function mysql_fetch_array() returns either the next record in the returned query or a "false" if there are no more records.
- By returning a "false", a while loop can be used to process until there are no more records.

Example of mysql_fetch_array()

```
$i=0;
while($record = mysql_fetch_array($result,
  MYSQL_ASSOC))
{
  print "---------- Record $i ----------<br />";
  foreach ($record as $index => $field)
     print "Field ".$index." = ".$field."<br />";
  $i++;
}
```


Closing the Connection

- A connection should close automatically when the PHP script completes, but as all good programmers know, you always close anything you open.
- To close a connection, use the mysql_close() function.
- Syntax:
  boolean = mysql_close($connection);
- $connection is the connection resource assigned with mysql_connect()
- The return value is true on success, false on failure.

MySQL Errors

- If you made any syntax errors when doing our exercises in MySQL, you know that MySQL outputs a cryptic message identifying the error.
- Errors will occur for a number of reasons both during development and after deploying the software
- PHP has a number of functions to assist the programmer in handling MySQL errors.

mysql_errno()

- `int mysql_errno($connection)` returns the numerical value of the error message from the last MySQL operation.
- A zero returned means there was no error.
- A list of the integer error codes can be found at: http://dev.mysql.com/doc/refman/5.0/en/error-handling.html

mysql_error()

- If the error number is too cryptic, the programmer can always use mysql_error()
- `string mysql_error($connection)` returns the text of the error message from last MySQL operation.
- This message is similar to the message you received after a syntax error at the command line MySQL.

die() or exit()

- The functions die() and exit() allow a script to exit gracefully.
- The two functions are equivalent, i.e., "die" and "exit" are interchangeable.
- Syntax:
  `void exit ( [string or int status] )`
- If status is a string, exit prints the string before stopping the script
- If status is an integer, it will be returned to calling application.
  - Status must be between 0 and 254.
  - 255 is reserved for PHP.
  - 0 indicates successful operation.

Other MySQL PHP Functions

- int **mysql_num_fields** ($result) retrieves the number of fields from a query.
- int mysql_num_rows ($result) retrieves the number of rows from a result set. Only works with a SELECT statement.
- int mysql_affected_rows ($result) retrieves the number of rows affected by the last INSERT, UPDATE or DELETE query.