PHP and Databases

Prof. Hyuk-Yoon Kwon
https://sites.google.com/view/seoultech-bigdata

Most parts are based on slides used in
(http://cgi.csc.liv.ac.uk/~martin/teaching/comp519/)

Variable Scope

The scope of a variable is the context within which it is defined.

```
<?php
$a = 1; /* limited variable scope */
function Test()
{
    echo $a;
  /* reference to local scope variable */
}
Test();
?>
```

view the output page

The scope is local within functions, and hence the value of $a is undefined in the “echo” statement.

```
<?php
$a = 1;
$b = 2;
function Sum()
{
    global $a, $b;
    $b = $a + $b;
}
Sum();
echo $b;
?>
```

view the output page

`global`

refers to its global version.

```
<?php
function Test()
{
    static $a = 0;
    echo $a;
    $a++;
}
Test();
Test();
Test();
?>
```

view the output page

`static`

does not lose its value.

Including Files

The include() statement includes and evaluates the specified file.

```php
//  vars.php
<?php

$color = 'green';
$fruit = 'apple';

?>

// test.php
<?php

echo "A $color $fruit"; // A

include 'vars.php';

echo "A $color $fruit"; // A green apple

?>
```

view the output page

```php
<?php

function foo()
{
   global $color;

   include ('vars.php');

   echo "A $color $fruit";
}

/* vars.php is in the scope of foo() so     *
 * $fruit is NOT available outside of this  *
 * scope.  $color is because we declared it *
 * as global.                                */

foo();                      // A green apple
echo "A $color $fruit";     // A green

?>
```

view the output page

*The scope of variables in “included” files depends on where the “include” file is added!

PHP Information

The `phpinfo()` function is used to output PHP information about the version instal led on the server, parameters selected when installed, etc.

```
<?php
// Show all PHP information
phpinfo();
?>
<?php
// Show only the general information
phpinfo(INFO_GENERAL);
?>
```

view the output page

Server Variables

The $_SERVER array variable is a reserved variable that contains all server information.

```
<html><head></head>
<body>

<?php
echo "Browser: " . $_SERVER["HTTP_USER_AGENT"] . "<br />";
echo "User's IP address: " . $_SERVER["REMOTE_ADDR"];
?>

<?php
echo "<br/><br/><br/>";
echo "<h2>All information</h2>";
foreach ($_SERVER as $key => $value)
  {
    echo $key . " = " . $value . "<br/>";
  }
?>

</body>
</html>
```

view the output page

The $_SERVER is a super global variable, i.e. it's available in all scopes of a PHP script.

File Open

The `fopen("file_name","mode")` function is used to open files in PHP.

r	Read only.	r+	Read/Write.
w	Write only.	w+	Read/Write.
a	Append.	a+	Read/Append.
x	Create and open for write only.	x+	Create and open for read/write.

```
<?php
$fh=fopen("welcome.txt","r");
?>
```

```
<?php
if
( !($fh=fopen("welcome.txt","r")) )
exit("Unable to open file!");
?>
```

For w, and a, if no file exists, it tries to create it (use with caution, i.e. check that this is the case, otherwise you'll overwrite an existing file).

For x if a file exists, this function fails (and returns 0).

If the `fopen()` function is unable to open the specified file, it returns 0 (false).

File Workings

`fclose()` closes a file.

`fgetc()` **reads a single character**

`fwrite(), fputs()` **writes a string without and with \n**

```
<?php
$myFile = "welcome.txt";
if (!($fh=fopen($myFile,'r')))
exit("Unable to open file.");
while (!feof($fh))
{
  $x=fgetc($fh);
  echo $x;
}
fclose($fh);
?>
```

view the output page

```
<?php
$lines = file('welcome.txt');
foreach ($lines as $l_num => $line)
{
 echo "Line #{$l_num}:"
.$line."<br/>";
}
?>
```

view the output page

`feof()` **determines if the end is true.**

`fgets()` **reads a line of data**

`file()` **reads entire file into an array**

```
<?php
$myFile = "welcome.txt";
$fh = fopen($myFile, 'r');
$theData = fgets($fh);
fclose($fh);
echo $theData;
?>
```

view the output page

```
<?php
$myFile = "testFile.txt";
$fh = fopen($myFile, 'a') or
die("can't open file");
$stringData = "New Stuff 1\n";
fwrite($fh, $stringData);
$stringData = "New Stuff 2\n";
fwrite($fh, $stringData);
fclose($fh);
?>
```

view the output page

Form Handling

Any form element is automatically available via one of the built-in PHP variables (provided that HTML element has a "name" defined with it).

```
<html>
<body>
<form action="welcome.php" method="get">
Enter your name: <input type="text" name="name" /> <br/>
Enter your age: <input type="text" name="age" /> <br/>
<input type="submit" /> <input type="reset" />
</form>
</body>
</html>
```

view the output page

```
<html>
<body>

Welcome <?php echo $_GET["name"]."."; ?><br />
You are <?php echo $_GET["age"]; ?> years old!

</body>
</html>
```

view the output page

`$_POST`
contains all POST data.

`$_GET`
contains all GET data.

Cookie Workings

`setcookie(name,value,expire,path,domain)` creates cookies.

```
<?php
$cookie_name = "user";
$cookie_value = "John Doe";
setcookie("user", "John Doe", time() + (86400 * 30), "/"); // 86400 = 1 day
?>
```

NOTE:

setcookie() **must appear BEFORE** <html> **(or any output) as it's part of the header information sent with the page.**

```
<?php
$cookie_name = "user";
if(!isset($_COOKIE[$cookie_name])) {
    echo "Cookie named '" . $cookie_name . "' is not set!";
} else {
    echo "Cookie '" . $cookie_name . "' is set!<br>";
    echo "Value is: " . $_COOKIE[$cookie_name];
}
?>
```

view the output page

`$_COOKIE`

contains all COOKIE data.

`isset()`

finds out if a cookie is set

Getting Time and Date

`date()` and `time ()` formats a time or a date.

```
<?php
//Prints something like: Monday
echo date("l");

//Like: Monday 15th of January 2003 05:51:38 AM
echo date("l jS \of F Y h:i:s A");

//Like: Monday the 15th
echo date("l \\t\h\e jS");
?>
```

view the output page

date() returns a string formatted according to the specified format.

```
<?php
$nextWeek = time() + (7 * 24 * 60 * 60);
                    // 7 days; 24 hours; 60 mins; 60secs
echo 'Now:       '. date('Y-m-d') ."\n";
echo 'Next Week: '. date('Y-m-d', $nextWeek) ."\n";
?>
```

view the output page

time() returns current Unix timestamp

*Here is more on date/time formats: http://uk.php.net/manual/en/function.date.php

Required Fields in User-Entered Data

A multipurpose script which asks users for some basic contact information and then checks to see that the required fields have been entered.

```
<html>
<head>
<title>PHP Form example</title>
</head>
<body>
<?php
/*declare some functions*/
function print_form($f_name, $l_name, $email, $os)
{
?>

 <form action="form_checker.php" method="get">
 First Name: <input type="text" name="f_name" value="<?php echo $f_name?>" /> <br/>
 Last Name <b>*</b>:<input type="text" name="l_name" value="<?php echo $l_name?>" /> <br/>
 Email Address <b>*</b>:<input type="text" name="email" value="<?php echo $email?>" /> <br/>
 Operating System: <input type="text" name="os" value="<?php echo $os?>" /> <br/><br/>
 <input type="submit" name="submit" value="Submit" /> <input type="reset" />
 </form>

<?php
}  //**  end of "print_form" function
```

Check and Confirm Functions

```
function check_form($f_name, $l_name, $email, $os)
{
  if (!$l_name||!$email){
    echo "<h3>You are missing some required fields!</h3>";
    print_form($f_name, $l_name, $email, $os);
  }
  else{
    confirm_form($f_name, $l_name, $email, $os);
  }
}  //** end of "check_form" function
```

```
function confirm_form($f_name, $l_name, $email, $os)
{
?>

<h2>Thanks! Below is the information you have sent to us.</h2>
<h3>Contact Info</h3>

<?php
echo "Name: $f_name $l_name <br/>";
echo "Email: $email <br/>";
echo "OS: $os";
}   //** end of "confirm_form" function
```

Main Program

```
/*Main Program*/

if (!isset($_GET["submit"]))
{
?>

 <h3>Please enter your information</h3>
 <p>Fields with a "<b>*</b>" are required.</p>

<?php
 print_form("","","","");
}
else{
 check_form($_GET["f_name"],$_GET["l_name"],$_GET["email"],$_GET["os"]);
}
?>


</body>
</html>
```

view the output page

PHP sessions

- **Keep track client's information**
  - HTML and web servers don't keep track of information entered on a page when the client's browser opens another page.
  - Doing anything with the same information across several pages can sometimes be difficult.

- **PHP sessions**
  - *Sessions* maintains data during a user's visit
  - You can use session variables for storing information
  - E.g., "shopping cart" function can work for an online shop

- **Session identifier**
  - Servers keep track of users' sessions by using a session identifier, which is generated by the server when a session starts an d is then used by the browser when it requests a page from the server.
  - This session ID can be sent through a cookie or by passing the session ID in the URL string.

- **Sessions only store information temporarily, so if you need to preserve information, say, between visits to the same site, you should likely consider a method such as using a cookie or a database to store such informatio n.**

PHP sessions (cont.)

- To start a session, use the function session_start() at the beginning of your PHP script before you store or ac cess any data.

```php
<?php
// Start the session
session_start();
?>
<!DOCTYPE html>
<html>
<body>

<?php
// Set session variables
$_SESSION["favcolor"] = "green";
$_SESSION["favanimal"] = "cat";
echo "Session variables are set.";
?>

</body>
</html>
```

view the output page

Using session variables

- Once a session variable has been defined, you can access it from other pages.

```
<?php
session_start();
?>
<!DOCTYPE html>
<html>
<body>

<?php
// Echo session variables that were set on previous
page
echo "Favorite color is
" . $_SESSION["favcolor"] . ".<br>";
echo "Favorite animal is
" . $_SESSION["favanimal"] . ".";
?>

</body>
</html>
```

view the output page

- Another way to show all the session variable values for a user session

```php
<?php
session_start();
?>
<!DOCTYPE html>
<html>
<body>

<?php
print_r($_SESSION);
?>

</body>
</html>
```

view the output page

Modifying session variables

- To change a session variable, just overwrite it

```php
<?php
session_start();
?>
<!DOCTYPE html>
<html>
<body>

<?php
// to change a session variable, just overwrite it
$_SESSION["favcolor"] = "yellow";
print_r($_SESSION);
?>

</body>
</html>
```

view the output page

More on session variables

- **You need to include a call to the session_start() function for each page on which you want to access the sessi on variables.**
- **A session will end once you quit the browser**
  - You can call the session_destroy() function.
  - Note, however, even after calling the session_destroy() function, session variables are still available to the rest of the currently executing PHP page.
- **The function session_unset() removes all session variables. If you want to remove one variable, use the unse t($var) function call.**
- **The default timeout for session files is 24 minutes. It's possible to change this timeout.**

```php
<?php
session_start();
?>
<!DOCTYPE html>
<html>
<body>

<?php
print_r($_SESSION);
// remove all session variables
session_unset();
print_r($_SESSION);

// destroy the session
session_destroy();
?>
```

view the output page

```
</body>
</html>
```

Practice - Session

- **Use the session variables in PHP page according to the following guidelines**
  - Determine a specific time period to maintain the session (e.g., 100 seconds)
  - For the first access, make a session variable to maintain the current time (Hint: use time())
  - If the session variable is set, check the passed time is satisfied with a given session period
    - If we have more time, then just print current passed time from the first access (e.g., 50 seconds)
    - Otherwise, delete session variable and print the session variable is deleted

Objects in PHP

Defining a class

- Use the “class” keyword which includes the class name (case-insensitive, but otherwise following the rules for PHP identifiers)

.

```
<?php
class Car {
   function Car($model, $price) {
      $this->model = $model;
      $this->price = $price;
   }
   function discount($rate) {
      $this->discounted_price = $this->price * $rate;
   }
   function print_car() {
            echo "model: $this->model" . ", ";
            echo "price: $this->price" . ", ";
            echo "discounted_price: $this->discounted_price" . "<br>";
   }
}
…
```

Creating an Object

- **Use the “new” keyword to create an instance of the object class**
  - Note that the object must be defined before you instantiate it.

```
// create an object
$bm = new Car("BM", 100000);
$ben = new Car("Ben", 120000);
```

Accessing properties and methods

- Once you have an object, you access methods and properties (variables) of the object using the –> notation.

```
// call methods in the class
$bm->discount(0.95);
$ben->discount(0.98);

// show object properties
$bm->print_car();
$ben->print_car();
```

Practice - Object

- Write a PHP Calculator class which will accept two values as arguments, then add them, subtract them, multiply them together, or divide them on request
  - *For example :*

    $mycalc = new MyCalculator(12, 6);

    echo $mycalc- > add(); // Displays 18

    echo $mycalc- > multiply(); // Displays 72

PHP and MySQL Database

- MySQL is the most popular database system used with PHP
  - Ref.) https://www.w3schools.com
  - Entire list supported by PHP: http://php.net/manual/en/refs.database.vendors.php
- PHP + MySQL Database System
  - PHP combined with MySQL are cross-platform
- Facts About MySQL Database
  - MySQL supports web sites with HUGE volumes of both data and end-users (like Facebook, Twitter, and Wikipedia).
  - Another great thing about MySQL is that it can be scaled down to support embedded database applications.

Logging into MySQL

```
C:\wamp\bin\mysql\mysql5.7.2\bin> mysql.exe –u root
```

```
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 209201 to server version: 5.0.22

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql>
```

```
mysql> create database practiceDB;
```

- **After you created the database once, you can use the database**

```
mysql> use practiceDB;
```

Creating a Table

```
mysql> CREATE TABLE students(
    -> num INT NOT NULL AUTO_INCREMENT,
    -> f_name VARCHAR(48),
    -> l_name VARCHAR(48),
    -> student_id INT,
    -> email VARCHAR(48),
    -> PRIMARY KEY(num));
```

Hit **Enter** after each line (if you want). MySQL doesn't try to interpret the command itself until it sees a semicolon (;)

(The "->" characters you see are not typed by you.)

```
Query OK, 0 rows affected (0.02 sec)
```

Viewing The Table Structure

```
mysql> DESCRIBE students;
```

```
+------------+-------------+------+-----+---------+----------------+
| Field      | Type        | Null | Key | Default | Extra          |
+------------+-------------+------+-----+---------+----------------+
| num        | int(11)     | NO   | PRI | NULL    | auto_increment |
| f_name     | varchar(48) | YES  |     | NULL    |                |
| l_name     | varchar(48) | YES  |     | NULL    |                |
| student_id | int(11)     | YES  |     | NULL    |                |
| email      | varchar(48) | YES  |     | NULL    |                |
+------------+-------------+------+-----+---------+----------------+
```

Inserting Data

```
mysql> INSERT INTO students
    ->  VALUES(NULL,'Russell','Martin',396640,'martin@csc.liv.ac.uk');
```

```
Query OK, 1 row affected (0.00 sec)
```

Using `SELECT FROM` you select some data from a table.

```
mysql> SELECT * FROM students;
```

```
+-----+---------+--------+------------+----------------------+
| num | f_name  | l_name | student_id | email                |
+-----+---------+--------+------------+----------------------+
|   1 | Russell | Martin |     396640 | martin@csc.liv.ac.uk |
+-----+---------+--------+------------+----------------------+
1 row in set (0.00 sec)
```

Inserting Some More Data

```
mysql> INSERT INTO students
    -> VALUES(NULL,'James','Bond',007,'bond@csc.liv.ac.uk');
Query OK, 1 row affected (0.01 sec)


mysql> SELECT * FROM students;
+-----+---------+--------+------------+----------------------+
| num | f_name  | l_name | student_id | email                |
+-----+---------+--------+------------+----------------------+
|   1 | Russell | Martin |     396640 | martin@csc.liv.ac.uk |
|   2 | James   | Bond   |          7 | bond@csc.liv.ac.uk   |
+-----+---------+--------+------------+----------------------+
2 rows in set (0.00 sec)
```

Note: The value “NULL” in the “num” field is automatically replaced by the SQL interpreter as the “auto_increment” option was selected when the table was defined.

Altering the Table

```
mysql> ALTER TABLE students ADD date DATE;
```

```
Query OK, 2 rows affected (0.00 sec)
Records: 2  Duplicates: 0  Warnings: 0
```

```
mysql> SELECT * FROM students;
+-----+---------+--------+------------+----------------------+------+
| num | f_name  | l_name | student_id | email                | date |
+-----+---------+--------+------------+----------------------+------+
|   1 | Russell | Martin |     396640 | martin@csc.liv.ac.uk | NULL |
|   2 | James   | Bond   |          7 | bond@csc.liv.ac.uk   | NULL |
+-----+---------+--------+------------+----------------------+------+
2 rows in set (0.00 sec)
```

Updating the Table

```
mysql> UPDATE students SET date='2007-11-15' WHERE num=1;
```

```
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

```
mysql> SELECT * FROM students;
+-----+---------+--------+------------+----------------------+------------+
| num | f_name  | l_name | student_id | email                | date       |
+-----+---------+--------+------------+----------------------+------------+
|   1 | Russell | Martin |     396310 | martin@csc.liv.ac.uk | 2012-11-15 |
|   2 | James   | Bond   |          7 | bond@csc.liv.ac.uk   | NULL       |
+-----+---------+--------+------------+----------------------+------------+
2 rows in set (0.00 sec)
```

Note that the default date format is “YYYY-MM-DD” and I don’t believe this default setting can be changed.

Deleting Some Data

```
mysql> DELETE FROM students WHERE l_name='Bond';
```

```
Query OK, 1 row affected (0.00 sec)
```

```
mysql> SELECT * FROM students;
+-----+---------+--------+------------+----------------------+------------+
| num | f_name  | l_name | student_id | email                | date       |
+-----+---------+--------+------------+----------------------+------------+
|   1 | Russell | Martin |     396310 | martin@csc.liv.ac.uk | 2012-11-15 |
+-----+---------+--------+------------+----------------------+------------+
1 row in set (0.00 sec)
```

Backup/restoring a MySQL Database

- **Backup an entire database with a command**
  - cmd> mysqldump –u root practiceDB > backup.sql

- **Restore the database**
  - Create a new database practiceDB2
  - cmd> mysql –u root practiceDB2 < backup.sql

PHP Connect to MySQL

- Open a Connection to MySQL
  - Before we can access data in the MySQL database, we need to be able to connect to the server

- MySQLi Object-Oriented

```php
<?php
$servername = "localhost";
$username = “root";
$password = "";

// Create connection
$conn = new mysqli($servername, $username, $password);

// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
echo "Connected successfully";
?>
```

MySQLi Procedural

```php
<?php
$servername = "localhost";
$username = “root ";
$password = "";

// Create connection
$conn = mysqli_connect($servername, $username, $password);

// Check connection
if (!$conn) {
    die("Connection failed: " . mysqli_connect_error());
}
echo "Connected successfully";
?>
```

Close the Connection

- MySQLi Object-Oriented

```
$conn->close();
```

- MySQLi Procedural

```
mysqli_close($conn);
```

PHP Create a MySQL Database

- The CREATE DATABASE statement is used to create a database in MySQL.

```php
<?php
$servername = "localhost";
$username = "root";
$password = "";

// Create connection
$conn = new mysqli($servername, $username, $password);
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

// Create database
$sql = "CREATE DATABASE myDB";
if ($conn->query($sql) === TRUE) {
    echo "Database created successfully";
} else {
    echo "Error creating database: " . $conn->error;
}

$conn->close();
?>
```

PHP Create MySQL Tables

- The CREATE TABLE statement is used to create a table in MySQL.
- We will create a table named "MyGuests", with five columns: "id", "firstname", "lastname", "email" and "reg_date":

```
CREATE TABLE MyGuests (
id INT(6) UNSIGNED AUTO_INCREMENT PRIMARY KEY,
firstname VARCHAR(30) NOT NULL,
lastname VARCHAR(30) NOT NULL,
email VARCHAR(50),
reg_date TIMESTAMP
)
```

```php
<?php
…

// sql to create table
$sql = "CREATE TABLE MyGuests (
id INT(6) UNSIGNED AUTO_INCREMENT PRIMARY KEY,
firstname VARCHAR(30) NOT NULL,
lastname VARCHAR(30) NOT NULL,
email VARCHAR(50),
reg_date TIMESTAMP
)";
if ($conn->query($sql) === TRUE) {
    echo "Table MyGuests created successfully";
} else {
    echo "Error creating table: " . $conn->error;
}
$conn->close();
?>
```

PHP Insert Data Into MySQL

```php
<?php

…

$sql = "INSERT INTO MyGuests (firstname, lastname, email)
VALUES ('John', 'Doe', 'john@example.com')";

if ($conn->query($sql) === TRUE) {
    echo "New record created successfully";
} else {
    echo "Error: " . $sql . "<br>" . $conn->error;
}

$conn->close();
?>
```

PHP Get ID of Last Inserted Record

- If we perform an INSERT or UPDATE on a table with an AUTO_INCREMENT field, we can get the ID of the last inserted/updated record immediately.
  - In the table "MyGuests", the "id" column is an AUTO_INCREMENT field

```
<?php
…

$sql = "INSERT INTO MyGuests (firstname, lastname, email)
VALUES ('John', 'Doe', 'john@example.com')";

if ($conn->query($sql) === TRUE) {
    $last_id = $conn->insert_id;
    echo "New record created successfully. Last inserted ID is: " .
$last_id;
} else {
    echo "Error: " . $sql . "<br>" . $conn->error;
}

$conn->close();
?>
```

PHP Insert Multiple Records Into MySQL

- **Multiple SQL statements must be executed with the mysqli_multi_query() function.**

```
<?php
…

$sql = "INSERT INTO MyGuests (firstname, lastname, email)
VALUES ('John', 'Doe', 'john@example.com');";
$sql .= "INSERT INTO MyGuests (firstname, lastname, email)
VALUES ('Mary', 'Moe', 'mary@example.com');";
$sql .= "INSERT INTO MyGuests (firstname, lastname, email)
VALUES ('Julie', 'Dooley', 'julie@example.com')";

if ($conn->multi_query($sql) === TRUE) {
    echo "New records created successfully";
} else {
    echo "Error: " . $sql . "<br>" . $conn->error;
}

$conn->close();
?>
```

PHP Prepared Statements

- A prepared statement is a feature used to execute the same (or similar) SQL statements repeatedly with high efficiency.

```php
<?php
…
// prepare and bind
$stmt = $conn->prepare("INSERT INTO MyGuests (firstname, lastname, email) VALUES (?, ?, ?)");
$stmt->bind_param("sss", $firstname, $lastname, $email);
// set parameters and execute
$firstname = "John";
$lastname = "Doe";
$email = "john@example.com";
$stmt->execute();

$firstname = "Mary";
$lastname = "Moe";
$email = "mary@example.com";
$stmt->execute();

echo "New records created successfully";
$stmt->close();
$conn->close();
?>
```

Character	Description
i	corresponding variable has type integer
d	corresponding variable has type double
s	corresponding variable has type string
b	corresponding variable is a blob and will be sent in packets

PHP Select Data From MySQL

```php
<?php
…
$sql = "SELECT id, firstname, lastname FROM MyGuests";
$result = $conn->query($sql);

if ($result->num_rows > 0) {
    // output data of each row
    while($row = $result->fetch_assoc()) {
        echo "id: " . $row["id"]. " - Name: " . $row["firstname"]. " " . $row["lastname"]. "<br>";
    }
} else {
    echo "0 results";
}
$conn->close();
?>
```

PHP Delete Data From MySQL

```php
<?php
…

// sql to delete a record
$sql = "DELETE FROM MyGuests WHERE id=3";

if ($conn->query($sql) === TRUE) {
    echo "Record deleted successfully";
} else {
    echo "Error deleting record: " . $conn->error;
}

$conn->close();
?>
```

PHP Update Data in MySQL

```php
<?php
…

$sql = "UPDATE MyGuests SET lastname='Doe' WHERE id=2";

if ($conn->query($sql) === TRUE) {
    echo "Record updated successfully";
} else {
    echo "Error updating record: " . $conn->error;
}

$conn->close();
?>
```

HW Assignment #3 – Login using the database

Description

- To correctly run the given php files, we need proper database having the required schema and the corresponding some values
- Analyze the php files and then make the required databases correctly
- The detailed steps are as follows:

Make the database according the following directions:

1. Download login.rar from eclass
2. Place uncompressed php files
3. Access login.php
4. Analyze the php codes and make a proper database for login using command line
   - Take care the sessions
5. Test login/logout

- **Submission: a word file consisting of 1) the used SQL, 2) the captured result for the final result, and 3) the explanation**