**Lab 05 Server-side Programming using PHP (5)**

**Module Intended Learning Outcome (#2):**

**On completion of the module, students are expected to be able to:**

**• Use appropriate development tools to examine and develop dynamic web applications**

**with international standards.**

**Lesson Intended Learning Outcome:**

**On completion of this lab, students are expected to be able to:**

**• Create database-driven website with PHP and MySQL.**

redirect

redirect

redirect

post

post

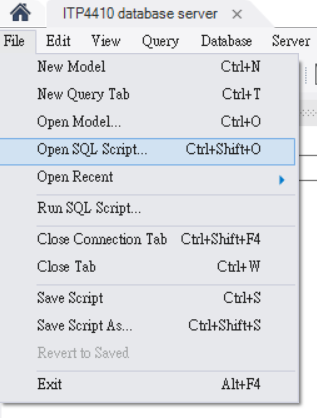
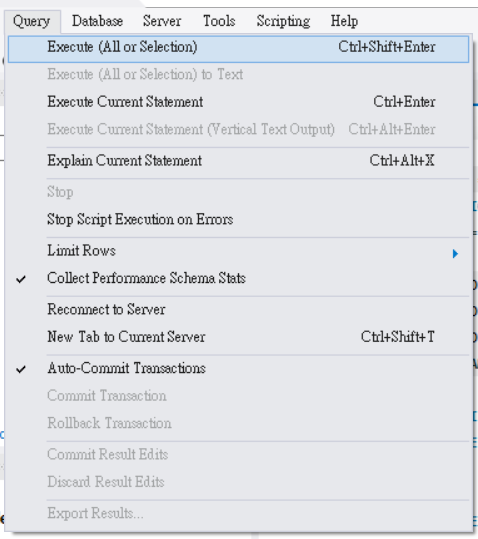
get

Structure of the website:

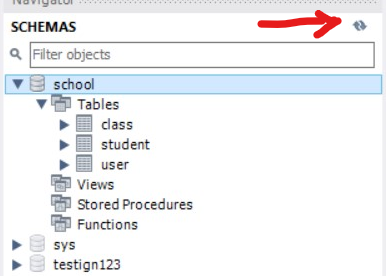
* Main files:
  + frm*xxx*.php : HTML Form
  + db*xxx*.php : Retrieve data from DB
  + msg*xxx*.php : Reply message to the user
* Include files:
  + db.inc.php : connect database; used by all db*xxx*.php and frmupdate.php

**Import database from file:**

Start MySQL workbench, download datafile from Moodle and unzip it. Click File 🡪 open SQL Script 🡪 select the school.sql file 🡪 SQL file opened 🡪 Click Query 🡪 Execute(All or Selection)

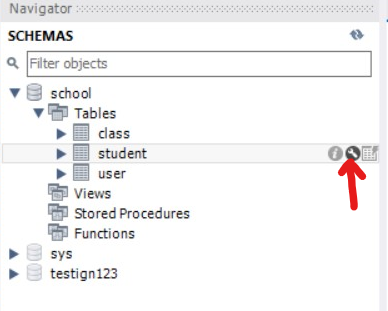
 

After executing the script, refresh the schemas, you will see a 'school' database is created.

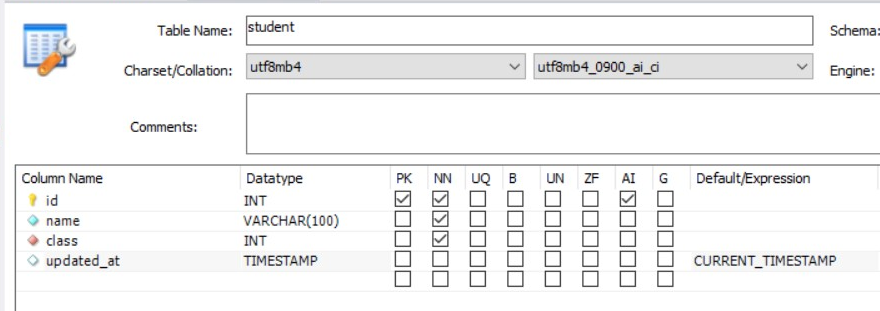


**Show table structure:**

Click student 🡪 Click spanner icon



You will see the following table structure

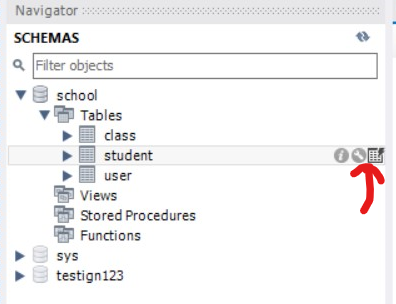


When new record has been created:

* AUTO\_INCREMENT (AI): generate ID automatically
* CURRENT\_TIMESTAMP: generate current date and time automatically

**Browse table record:**

Click student 🡪 Click table icon

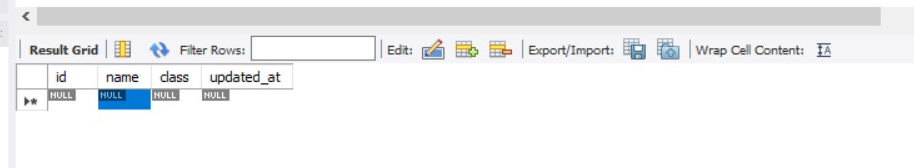


**Insert record into table:**

Click student 🡪 Insert, create the following students:

|  |  |  |
| --- | --- | --- |
| id |  |  |
| name | Peter | Sam |
| class | 1 | 2 |
| update\_at | CURRENT\_TIMESTAMP | CURRENT\_TIMESTAMP |
|  |  |  |

You can insert the above record by SQL statement or insert it by GUI interface on MySQL workbench



Or

use school;

insert into student(name,class) values("Peter",1);

insert into student(name,class) values("Sam",2);

**db.inc.php**

* Create a new folder '**inc**' under lab5 folder,
* Create db.inc.php in **/www/lab5/inc**
* Copy the source codes from Lecture 5 P.8 and paste them into db.inc.php
* Update the database name and password (no password if you are using wampserver):

<?php

$dsn = 'mysql:dbname=**school**;host=localhost;charset=UTF8';

$dbuser = 'root';

$dbpwd = '';

**dbstu.php**

Create dbstu.php in /www/lab5

**1. Connect to the database:**

Import the file db.inc.php, then call the dbconnect() function to get a PDO object, the function has already defined on the file:

<?php

require\_once "./inc/db.inc.php";

$pdo = dbconnect();

?>

* './' means current folder

**2. Prepare and execute SQL**

2.1. Create a PHP try-catch block:

$pdo = dbconnect();

try {

} catch (PDOException $e) {

  die($e->getMessage());

}

?>

2.2. Refer to the Lecture 5 P.16. Copy and paste the source codes and paste it into the try block:

try {

  // paste here

} catch (PDOException $e) {

Modify the source code to retrieve all students from the database:

$sql = "SELECT id, name, class, updated\_at FROM student ";

~~$sql = "SELECT \* FROM books";~~

$stmt = $pdo->prepare($sql);

$stmt->execute();

2.3. Create basic HTML document under the PHP close tag:

?>

<html>

<head></head>

<body>

<h1>Select All Students</h1>

</body></html>

2.4. Under the <h1> heading, create a new PHP block:

<h1>Select All Students</h1>

<?php

?>

</body></html>

**3. Check result**

echo the number of records found:

<?php

$numFound = $stmt->rowCount();

echo "<p>".$numFound." Student(s) found</p>";

?>

Browse <http://localhost/lab5/dbstu.php>





**4. Show result**

Create a table if there is at least one student:

echo "<p>".$numFound." Student(s) found</p>";

if( $numFound > 0 ) {

echo "<table border='1'>\n";

echo "<tr>\n";

echo "<td>ID</td>\n";

echo "<td>Name</td>\n";

echo "<td>Class</td>\n";

echo "<td>Updated At</td>\n";

echo "</tr>\n";

echo "</table>\n";

}

Use a While loop to create table row for showing record data

(Hints:  ):

echo "</tr>\n";

while( $result = $stmt->fetch() ) {

echo "<tr>\n";

echo "<td>".$result[\_\_??\_\_]."</td>\n";

echo "<td>".$result[\_\_??\_\_]."</td>\n";

echo "<td>".$result[\_\_??\_\_]."</td>\n";

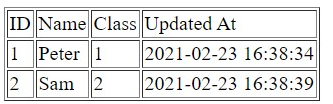
echo "<td>".$result[\_\_??\_\_]."</td>\n";

echo "</tr>\n";

}

echo "</table>\n";

The following is the expected output:



**5. Join tables in SQL:**

5.1. If you want to show the class names instead of index numbers, you should retrieve data from table student and table class. Modify the SQL statement to join the tables:

$sql = 'SELECT student.id,

student.name,

class.name AS **classname,**

student.updated\_at FROM student ';

$sql .= ' INNER JOIN class on student.class = class.id ';

. . .

while( $result = $stmt->fetch() ) {

echo "<tr>\n";

echo "<td>".$result[\_\_??\_\_]."</td>\n";

echo "<td>".$result[\_\_??\_\_]."</td>\n";

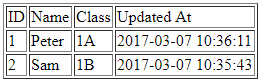
echo "<td>".$result[**'classname'**]."</td>\n";

echo "<td>".$result[\_\_??\_\_]."</td>\n";

echo "</tr>\n";

}

The following is the expected result:



5.2. Create links for editing pages, add links for those editing pages:

echo "<td>Updated At</td>\n";

echo "<td>Action</td>\n";

echo "</tr>\n";

. . .

echo "<td>".$result['updated\_at']."</td>\n";

echo "<td>";

echo "<a href='frmupdate.php?id=".$result[\_\_??\_\_]."'>Update</a> ";

echo "<a href='dbdelete.php?id=".$result[\_\_??\_\_]."'>Delete</a>";

echo "</td>\n";

echo "</tr>\n";

}

echo "</table>\n";

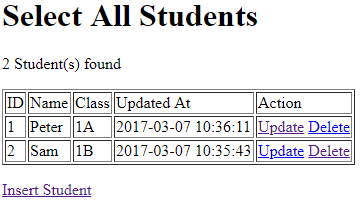
}

?>

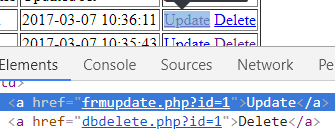
<p><a href="frminsert.php">Insert Student</a></p>

</body>

The following is the expected output:



Check the links if correct IDs are used.

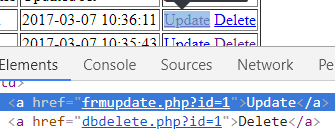


**dbdelete.php**

1. Create dbdelete.php in /www/lab5

Refer to the notes Lecture 5 P.6. Copy and paste the template to dbdelete.php.

**2. Validate and get data**

Get student id from the URL:

/\* 2. Validate and get data \*/

if( !array\_key\_exists('id',\_\_??\_\_) ) {

header('location: dbstu.php');

exit();

}

$id = \_\_??\_\_['id'];

**3. Prepare and execute SQL**

Refer to the notes Lecure5 P.17. Copy and paste the source codes under the comment:

/\* 3. Prepare and execute SQL . . . \*/

~~$sql = "SELECT \* FROM user";~~

~~$sql .= " WHERE email = :email";~~

~~$sql .= " AND password = :password";~~

$sql = "DELETE FROM student ";

$sql .= " WHERE id = :id ";

$stmt = $pdo->prepare($sql);

~~$stmt->bindParam(":email", $\_POST['email']);~~

~~$stmt->bindParam(":password", $\_POST['password']);~~

$stmt->bindParam(":id", $id);

$stmt->execute();

**4. Check result**

If record has been deleted, send the HTTP redirection 'msgdelete.php?**t**=xxx'

Otherwise send HTTP redirection 'msgdelete.php?**f**=xxx'

/\* 4. Check result . . . \*/

if( $stmt->rowCount() == 1 ) {

$queryString = '**t**='.$id; // ok

}

else {

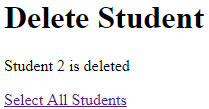
$queryString = '**f**='.$id; // fail

}

header('location: msgdelete.php?'.$queryString);

exit();

**5. Go back to the dbstu.php. Try to delete student 2.**



**dbinsert.php**

1. Create dbinsert.php in /www/lab5

Refer to Lecture 5. Copy and paste the template to dbinsert.php.

**2. Validate and get data**

Open frminsert.php and read the form elements. In dbinsert.php, collect data from the form:

/\* 2. Validate and get data \*/

if( !array\_key\_exists(\_\_??\_\_, \_\_??\_\_) ||

!array\_key\_exists(\_\_??\_\_, \_\_??\_\_) ) {

header('location: dbstu.php');

 exit();

}

**$name** = \_\_??\_\_;

**$class** = \_\_??\_\_;

**3. Prepare and execute SQL:**

Refer to Lecture 5 P.17. Copy and paste the source codes under the comment:

/\* 3. Prepare and execute SQL . . . \*/

~~$sql = "SELECT \* FROM students ";~~

~~$sql .= " WHERE class = :class ";~~

$sql = 'INSERT INTO student (name, class) ';

$sql .= ' VALUES (:name, :class) ';

$stmt = $pdo->prepare($sql);

~~$stmt->bindParam(":class", $\_POST['class']);~~

$stmt->bindParam(\_\_??\_\_, \_\_??\_\_);

$stmt->bindParam(\_\_??\_\_, \_\_??\_\_);

$stmt->execute();

**4. Check result**

If record has been deleted, send the HTTP redirect ‘msginsert.php?**t**=xxx’

Otherwise send HTTP redirect ‘msginsert.php?**f**=xxx’

/\* 4. Check result . . . \*/

if( $stmt->rowCount() == 1 ) {

$queryString = '**t**='.$pdo->lastInsertId(); // ok

}

else {

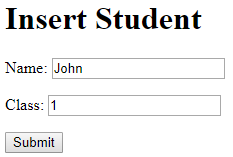
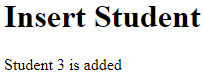
$queryString = '**f**='.$name; // fail

}

header('location: msginsert.php?'.$queryString);

exit();

**5.** Browse frminsert.php. Try to create a new student**.**

  

**frmUpdate.php**

1. Open frmupdate.php. The program should search the student record before showing the form.

Refer to **Lecture 5 P.7**. Copy and paste the template at the start of frmupdate.php.

<?php

/\* 1. Connect to the database \*/

. . .

die($e->getMessage());

}

?>

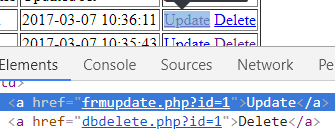
~~<html><head>...~~

<!-- 5. Show result . . . -->

<html>

<head>

**2. Validate and get data**

Get student id from the URL:

/\* 2. Validate and get data \*/

if( !array\_key\_exists(\_\_??\_\_, \_\_??\_\_) ) {

header('location: dbstu.php');

exit();

}

**$id** = \_\_??\_\_;

**3. Prepare and execute SQL:**

Refer to Lecture 5 P.17. Copy and paste the source codes under the comment:

/\* 3. Prepare and execute SQL . . . \*/

~~$sql = "SELECT \* FROM user";~~

$sql = "SELECT id, name, class FROM student

~~$sql .= " WHERE email = :email";~~

~~$sql .= " AND password = :password";~~

$sql .= " WHERE id = :id ";

$stmt = $pdo->prepare($sql);

~~$stmt->bindParam(":email", $\_POST['email']);~~

~~$stmt->bindParam(":password", $\_POST['password']);~~

$stmt->bindParam(":id", $id);

$stmt->execute();

**4. Check result**

If record cannot be found, back dbstu.php

Otherwise, prepare record data into variables

$stmt->execute();

/\* 4. Check result . . . \*/

if( $stmt->rowCount() == 0 ) {

// student not found

header('location: dbstu.php');

exit();

}

$result = $stmt->\_\_??\_\_;

**$id** = $result[\_\_??\_\_];

**$name** = $result[\_\_??\_\_];

**$class** = $result[\_\_??\_\_];

} catch (PDOException $e) {

**5. Show result**

Show and add record data in the corresponding form elements:

<form method="post" action="dbupdate.php**?id=<?=$id?>**">//use get ?key value

<p>ID: <input type="text" name="id"

**value="<?=$id?>"** disabled /></p>

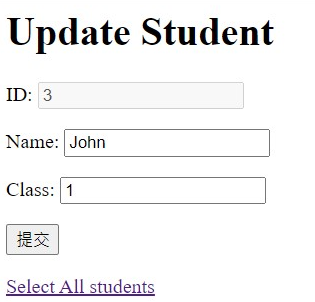
<p>Name: <input type="text" name="name"

**value="\_\_??\_\_"** /></p>

<p>Class: <input type="text" name="class"

**value="\_\_??\_\_"** /></p>

Browse <http://localhost/lab5/frmupdate.php?id=3>



**\*\*\*disable cannot submit to server**

**dbupdate.php**

1. Create dbupdate.php in **/www/lab5**

Refer to **Lecture 5 P.6**. Copy and paste the template to dbupdate.php.

**2. Validate and get data**

In dbupdate.php, collect data from the **form and URL**:

/\* 2. Validate and get data \*/

if( !array\_key\_exists(\_\_??\_\_, \_\_??\_\_) ||

!array\_key\_exists(\_\_??\_\_, \_\_??\_\_) ||

!array\_key\_exists(\_\_??\_\_, \_\_??\_\_) ) {

header('location: dbstu.php');

exit();

}

**$id** = \_\_??\_\_;

**$name** = \_\_??\_\_;

**$class** = \_\_??\_\_;

**3. Prepare and execute SQL:**

Refer to Lecture 5 P.17. Copy and paste the source codes under the comment:

/\* 3. Prepare and execute SQL . . . \*/

~~$sql = "SELECT \* FROM user";~~

~~$sql .= " WHERE email = :email";~~

~~$sql .= " AND password = :password";~~

$sql = 'UPDATE student ';

$sql .= ' SET name=:name, class=:class, updated\_at=NOW()';

$sql .= ' WHERE id=:id ';

$stmt = $pdo->prepare($sql);

~~$stmt->bindParam(":email", $\_POST['email']);~~

~~$stmt->bindParam(":password", $\_POST['password']);~~

$stmt->bindParam(\_\_??\_\_, \_\_??\_\_);

$stmt->bindParam(\_\_??\_\_, \_\_??\_\_);

$stmt->bindParam(\_\_??\_\_, \_\_??\_\_);

$stmt->execute();

**4. Check result**

If record has been updated, send HTTP redirect ‘msgupdate.php?**t**=xxx’

Otherwise send HTTP redirect ‘msgupdate.php?**f**=xxx’

/\* 4. Check result . . . \*/

if( $stmt->\_\_??\_\_ == 1 ) {

$queryString = '**t**='.$id; // ok

}

else {

$queryString = '**f**='.$id; // fail

}

\_\_??\_\_('location: msgupdate.php?'.\_\_??\_\_);

exit();

**5. Browse** [**http://localhost/lab5/frmupdate.php?id=3**](http://localhost/lab5/frmupdate.php?id=3) **. Try to update the student: from John to *YOUR\_NAME***

  