## Week 7 Practical: PHP & Multimedia

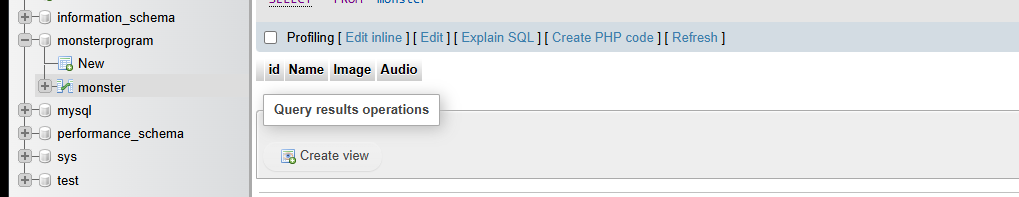
During this week’s practical you will learn how to save image and audio files within a database and then retrieve them, in order to play / display them within a web page. The application you will build this week saves and displays details about various mythological and modern monsters.

**Exercise 1A**

Within this exercise you will create a new table to hold image and audio files. Login to MySQL via phpMyAdmin and select your database.

Now run the SQL below in the “SQL” tab of phpMyAdmin (<https://intweb.bucks.ac.uk/sqladmin/>) on your database to create the table.





The table contains four fields, the id is automatically generated as new records are inserted. The name field contains the monster’s name and the image field a jpeg image of the monster. The audio field contains a related sound in the form of a wav file. Both the image and audio fields are of type blob, which stands for Binary Large Object and is used to store binary data. Both the wav and jpeg file types can be store within a field of this type.

**Exercise 1B**

Write the equivalent table creation code in PHP using a Phinx migration class. You do not have to run the migration but include your code below. See <https://book.cakephp.org/phinx/0/en/migrations.html> for guidance.

Phinx CODE

<?php

use Phinx\Migration\AbstractMigration;

class CreateMonsterTable extends AbstractMigration

{

public function change()

{

// Create the 'monster' table

$table = $this->table('monster');

// Add columns

$table->addColumn('Name', 'string', ['limit' => 20, 'null' => false])

->addColumn('Image', 'binary', ['null' => false])

->addColumn('Audio', 'binary', ['null' => false])

->addPrimaryKey('id') // Add primary key to 'id'

->create();

}

}

**Exercise 2**

The next step is to create a form through which we can select the image and audio file, to store within the new monster table. Create a new HTML document through NotePad named monsterform.html and include the markup below within it.

Note, we are including Bootstrap into this page as per the instructions found on <http://getbootstrap.com/getting-started/>



The form contains a text box within which the monster’s name can be entered and two file inputs that allow the user to select a file via a file dialog box. The two file controls will allow the user to select the jpeg image and wav file of the monster they are saving within the table.

After the user has entered the relevant information into the form and clicks the submit button the form data is sent to the savemonster.php script. The source code for this script is listed on the next page. When creating this file do not forget to substitute your own MySQL login, password and database name.



Code using my database, password, etc.

<?php

$db = mysqli\_connect("localhost", "root", "root", "monsterprogram");

// Obtain the file sent to the server within the response.

$image = $\_FILES['monsterimage']['tmp\_name'];

$audio = $\_FILES['monsteraudio']['tmp\_name'];

// Get the file binary data

$imagedata = addslashes(fread(fopen($image, "r"), filesize($image)));

$audiodata = addslashes(fread(fopen($audio, "r"), filesize($audio)));

$sql = "INSERT INTO monster";

$sql .= "(name, image, audio) ";

$sql .= "VALUES ('$\_POST[txtname]', '$imagedata','$audiodata');";

$result = mysqli\_query($sql, $db);

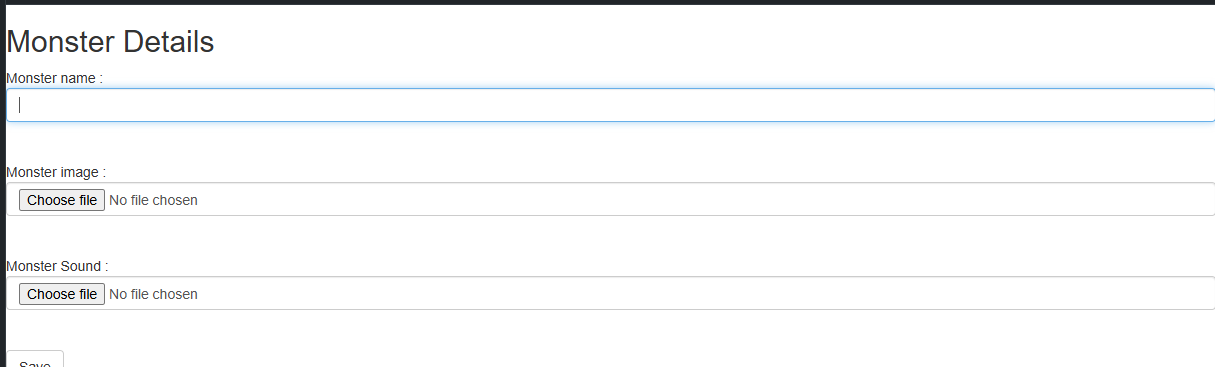
mysqli\_close();

?>

(You can download the bundle of sample image and audio files for this exercise here: <https://github.com/iamjonjackson/bnu-php-tutorial-multimedia-files>)

After copying the two files to the server, display monsterform.html within your browser. Enter “Alien” into the monster description and select the **Alien.jpg** image followed by the **Xfiles.wav** audio file (these files are included in the bundle on github linked to above).

Click the submit button to save the form data within the monster table. If there are no errors within your script you should see a blank page displayed within the browser.



**Exercise 3**

In order to display the contents of a blob field it is necessary to write a new script that extracts the raw binary data from within the field and sends it back to the browser embedded in the response. Recreate the **getjpg.php** file below.



The browser must be told what type of data is being sent to it in order to be able to render it correctly, hence the header function is used to embed the **Content-type: image/jpeg** within the response.

Create a second script named getwav.php in line with the code below, to extract the audio data.



Finally we will create a script named displaymonster.php that will display the first monster saved within the monster table.



Load the script displaymonster.php within your browser.

Does it display the image of the first monster you saved within exercise 1?

No, it did not display the image of the first monster I saved within exercise 1.

**Exercise 4**

Now modify displaymonster.php in line with the script below, so that it will display all the records within the monster table.



Before you run this script add one or two more monsters to the table using the image and audio files within the MultimediaFilesForPractical bundle available on GitHub.

**Exercise 5**

Combine the two files created in exercise 2 into a **single file** that uses a recursive call.

**Exercise 6**

Draw the UML design for the script created within exercise 5.