

CPET 499/ITC 250 Web Systems

Chapter 14 Working with Databases Part 3 of 3

Text Book:

* Fundamentals of Web Development, 2nd edition, by Randy Connolly and Ricardo Hoar, published by Pearson

Paul I-Hai Lin, Professor of Electrical and Computer Engineering
Technology

<http://www.etcs.pfw.edu/~lin>

Topics

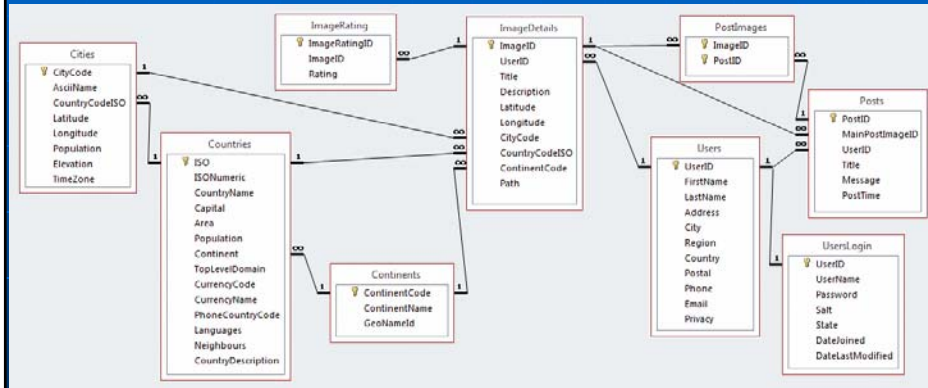
14.7 Case Study Schemas

- Travel Phot Sharing Databases
- Art Database
- Book CRM Database

14.8 Sample Databases Techniques

- Search Results Page
- Editing a Record
- Saving and Displaying Raw Files in the Database
 - Storing BLOB Data
 - Displaying BOLBs from the Databse

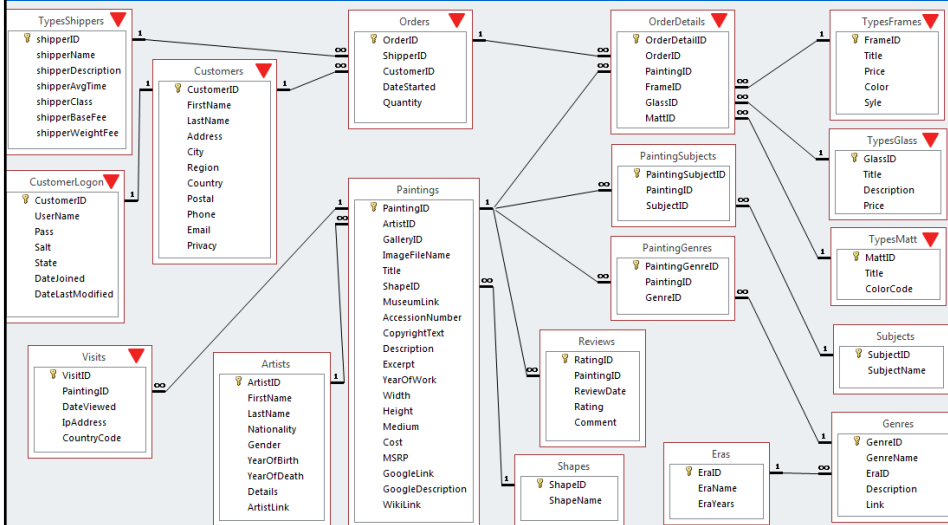
Travel Phot Sharing Database



CPET 499/ITC 250 Web Systems, Paul I. Lin

3

Art Database



CPET 499/ITC 250 Web Systems, Paul I. Lin

4

Figure 14.28 Search Results Page

```
<?php
// % --- SQL wildcard character
// Like --- SQL Operator
$searchFor = '%' . $_GET['textSearch'] . '%';
$sql = "SELECT * FROM Books WHERE Title Like ?";
$stmt = $pdo->prepare($sql);
$stmt->bindValue(1, $searchFor);
$stmt->execute();
?>
```

Figure 14.28 Search Results Page

```
<input type="search"
      name="textSearch"
      placeholder="Enter Search String"
      value=<?php echo $_GET['textSearch']; ?>
/>
```

Listing 14.25 search-results.php (Incomplete code)

```
<?php
// defines a constant for query string parameter name
define('SEARCHBOX', 'txtSearch');
// define a function to return the value of the search parameter
function getSearchFor(){
    // this function is missing something . . . do you know what it is?
    return $_GET[SEARCHBOX];
}
function getDB(){
    //...
    $pdo = new PDO($connString,$user,$pass);
    $pdo->setAttribute(PDO::ATTR_ERRMODE,
        PDO::ERRMODE_EXCEPTION);
    return $pdo;
}
```

CPET 499/ITC 250 Web Systems, Paul I. Lin

9

Listing 14.25 search-results.php (Incomplete code)

```
function getResults()
{
    try {
        $db = getDB();
        // add SQL wildcard characters to search term
        $searchFor = '%' . getSearchFor() . '%';
        $sql = "SELECT * FROM Books WHERE Title Like ?";
        $statement = $db->prepare($sql);
        $statement->bindValue(1, $searchFor);
        $statement->execute();
        return $statement;
    }
    catch (PDOException $e) { die($e->getMessage()); }?>
```

CPET 499/ITC 250 Web Systems, Paul I. Lin

10

Listing 14.25 search-results.php (Incomplete code)

```
<html>
<body>
<form method="get" action="search-results.php" >
  <fieldset>
    <legend>Search Title</legend>
    <input type="search"
      name="<?php echo SEARCHBOX; ?>"
      placeholder="Enter search string"
      value="<?php echo getSearchFor(); ?>" />
    <input type="submit" />
  </fieldset>
</form>
```

Listing 14.25 search-results.php (Incomplete code)

```
<table border="1">
<?php
if (! empty($_GET[SEARCHBOX]) && $result = getResults() ) {
  while ($row = $result->fetch()) {
    ?>
    <tr>
      <td><?php echo $row['ISBN10']; ?></td>
      <td><?php echo $row['Title']; ?></td>
      <td><?php echo $row['CopyrightYear']; ?>
      </td> </tr><?php }} ?>
    </table>
  </body>
</html>
```

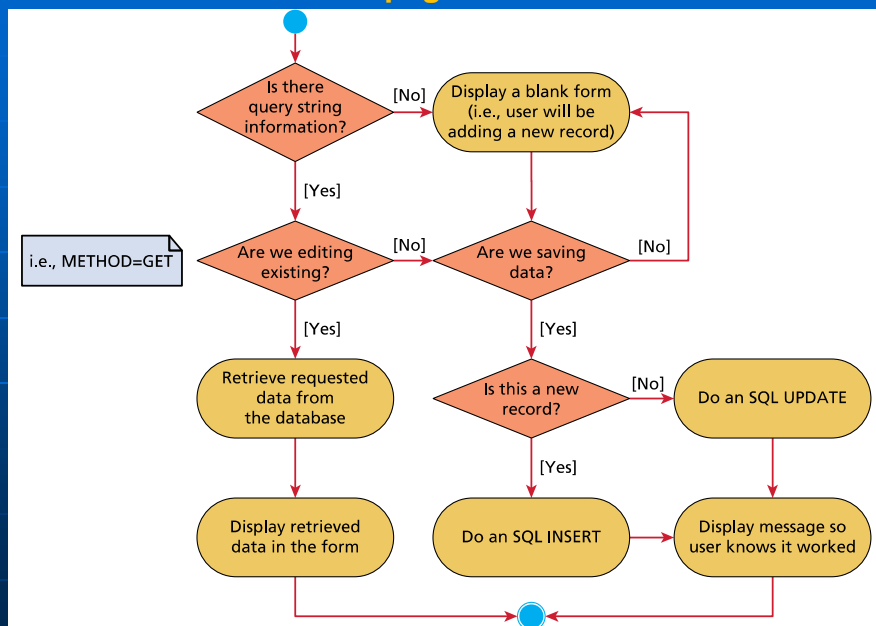
Listing 14.28 Solution to search results page problem

```
function getSearchFor()
{
    $value = "";
    if (isset($_GET[SEARCHBOX]))
    {
        $value = $_GET[SEARCHBOX];
    }
    return $value;
}
```

CPET 499/ITC 250 Web Systems, Paul I. Lin

13

Fig. 14.30 Program flow in record editor (save & retrieve), page 676



Listing 14.27 Author class

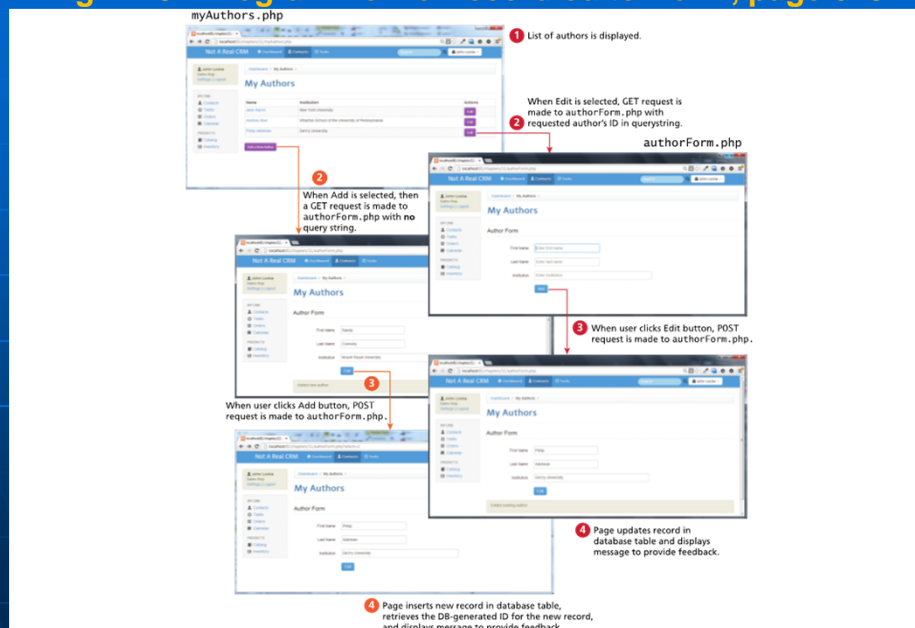
```
<?php
class Author {
    public $id = "";
    public $firstName = "";
    public $lastName = "";
    public $institution = "";
    function __construct($id,$first,$last,$institute) {
        $this->id = $id;
        $this->firstName = $first;
        $this->lastName = $last;
        $this->institution = $institute; }
    // Returns true if this is a new author, false otherwise
    function isNew() {
        if (empty($this->id) )    return true;
        else    return false; }}

?>
```

CPET 499/ITC 250 Web Systems, Paul I. Lin

15

Fig. 14.31 Program flow of record editor form, page 678



Listing 14.28 processAuthorFormInfo() function

```
<?php
function processAuthorFormInfo($pdo) {
    // first let us see if there is any query string information
    // . . . if not return empty author object
    if (!isThereQueryStringInfo()) { return new Author("", "", "", "");
}
// are we editing an existing author . . .
if ( areEditingExisting() ) {
    // since request method is GET, then this is either request for
    // inserting new or a request to edit if id attribute
    // NOTE: we are assuming ID in query string is ok
    // (should actually test it in real site)
    $which = $_GET['which'];
    // retrieve data from database
    return retrieveAuthor($pdo, $which); }
}
```

CPET 499/ITC 250 Web Systems, Paul I. Lin

17

Listing 14.28 processAuthorFormInfo() function

```
// . . . or are we saving an author
if ( areSaving() ) {
    // if here then saving a record
    // we are going to use the existence of an ID querystring to
    // determine whether we should be inserting or updating
    $id = "";
    if ( isset($_POST['id']) ) {
        $id = $_POST['id'];
    }
    $author = saveAuthor( $pdo, $id, $_POST['firstname'],
        $_POST['lastname'], $_POST['institution'] );
    return $author;
}
?>
```

CPET 499/ITC 250 Web Systems, Paul I. Lin

18

Listing 14.29 Helper functions for Listing 14.28

```
/* Checks if there is any query string information passed in GET
or POST*/
function isThereQueryStringInfo() {
    if ( areEditingExisting() ) { return true; }
    if ( areSaving() ) { return true; } return false;
}

/* Checks if query string info tells us whether we are editing
existing author*/
function areEditingExisting() {
    if ($ _SERVER['REQUEST_METHOD'] == 'GET' &&
        isset($_GET['which'])) { return true; }
}
```

CPET 499/ITC 250 Web Systems, Paul I. Lin

19

Listing 14.29 Helper functions for Listing 14.28

```
/* Checks if query string info tells us whether we are saving
author info*/
function areSaving() {
    if ($ _SERVER['REQUEST_METHOD'] ==
        'POST' && isset($_POST['firstname']) &&
        isset($_POST['lastname'])) { return true; }
}

/* Retrieve a populated author from the database*/
function retrieveAuthor($pdo, $id){
    $sql = "SELECT * FROM Authors WHERE ID=:id";
    $statement = $pdo->prepare($sql);
    $statement->bindValue(':id', $id);
    $statement->execute();
    $row = $statement->fetch(PDO::FETCH_ASSOC);
    return new Author($row['ID'], $row['FirstName'], row['LastName'],
    $row['Institution']);}
```

CPET 499/ITC 250 Web Systems, Paul I. Lin

20

Listing 14.29 Helper functions for Listing 14.28

```
/* Actually perform the database insert or update*/
function saveAuthor($pdo, $id, $first, $last, $institute){
    $GLOBALS['updateStatus'] = "";
    $author = new Author($id, $first, $last, $institute);
    // set up sql statement and page's message
    if ( $author->isNew() ) {
        $sql = "INSERT INTO authors (FirstName,LastName,Institution)
VALUES (:first,:last,:institute)";
        $GLOBALS['saveMessage'] = 'Added new ';
    }
    else {
        $sql = "UPDATE authors SET FirstName=:first,LastName=:last,
Institution=:institute WHERE ID=:id";
        $GLOBALS['saveMessage'] = 'Edited existing ';
    }
}
```

CPET 499/ITC 250 Web Systems, Paul I. Lin

21

Listing 14.29 Helper functions for Listing 14.28

```
// setup the parameters for the query
$stmt = $pdo->prepare($sql);
$stmt->bindValue(':first', $first);
$stmt->bindValue(':last', $last);
$stmt->bindValue(':institute', $institute);
if ( ! $author->isNew() ) $stmt->bindValue(':id', $id);
// execute the query
$stmt->execute();
// retrieve auto generated id if this was an insert and update
// author object
if ( $author->isNew() ) {
    $author->id = $pdo->lastInsertId();
}
return $author;
}
```

CPET 499/ITC 250 Web Systems, Paul I. Lin

22

Listing 14.30 authorForm.php page

```
<?php
// initialize page globals
require_once('includes/config-books.inc.php');
require_once('includes/Author.class.php');
// class name for hiding a <div>
$GLOBALS['updateStatus'] = 'hide';
// the message to be displayed after saving
$GLOBALS['saveMessage'] = ' ';
```

Listing 14.30 authorForm.php page

```
try {
// set up the PDO connection to database
$pdo = new PDO(DBCONNECTION,DBUSER,DBPASS);
$pdo->setAttribute(PDO::ATTR_ERRMODE,
PDO::ERRMODE_EXCEPTION);
// perform the algorithm and return populated Author object
$author = processAuthorFormInfo($pdo);
// change form Submit button text based on author object
if ( $author->isNew() ) { $buttonText = 'Add'; }
else { $buttonText = 'Edit'; }}
catch (PDOException $e) { die( $e->getMessage() );
}
//...
?>
```

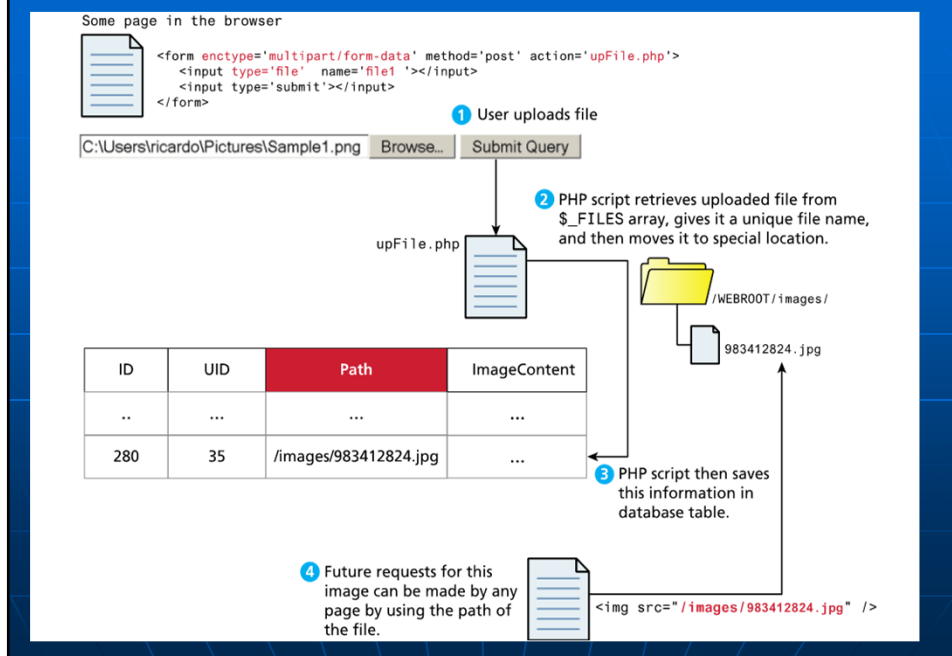
Listing 14.30 authorForm.php page

```
<!DOCTYPE html>
<html>
<head lang="en">
  ..
<form class="form-horizontal" method="post"
action="authorForm.php">
<fieldset>
<legend>Author Form</legend>
<input type="hidden" name="id" value="<?php echo $author->id
?>" />
<label>First Name</label>
<input type="text" name="firstname" placeholder="Enter first
name" value=" <?php echo $author->firstName; ?>">
```

Listing 14.30 authorForm.php page

```
<label>Last Name</label>
<input type="text" name="lastname" placeholder="Enter last
name" value="<?php echo $author->lastName; ?>">
<label>Institution</label>
<input type="text" name="institution" placeholder="Enter
Institution" value="<?php echo $author->institution; ?>">
<button type="submit" >
  <?php echo $buttonText; ?>
</button>
</fieldset>
</form>
<div class="alert alert-info <?php echo
$GLOBALS['updateStatus']; ?>">
<p> <?php echo $GLOBALS['saveMessage']; ?> author</p>
</div>
```

Fig. 14.32 Storing file location in the database, page 684



■ Saving and Displaying Raw Files in the Database

- Upload an image and save it within BLOB field
- \$_FILES --- superglobal array

```
<form enctype='multipart/form-data' method='post'
action='upFile.php'>
```

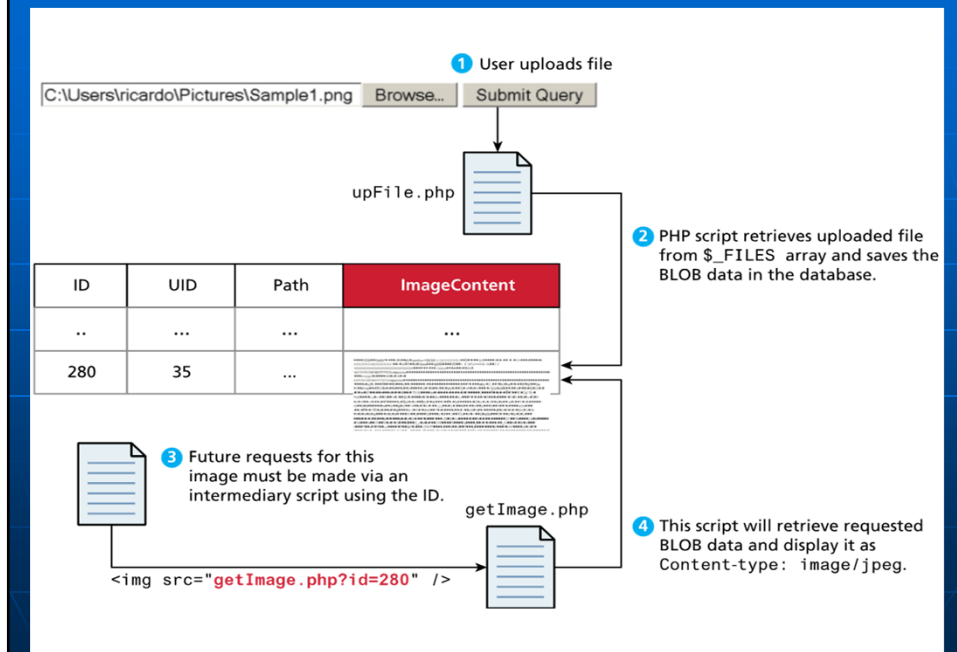
```
<input type = "file" name='file1'> </input>
```

```
</form>
```

Image File Storage: ID, UID, Path, ImageContent

- ImageContent: a BLOB field for binary data of the image
- TravellImage Path: /images/983412824.jpg

Fig. 14.33 Using BLOBs to store image data, page 685



**Listings 14-31 Code to save file contents in a BLOB field
: read a file into memory and store it into the database**

```
<?php
$fileContent = file_get_contents("someImage.jpg");
$sql = "INSERT INTO TrabellImage (ImageContent)
VALUES (:data)";

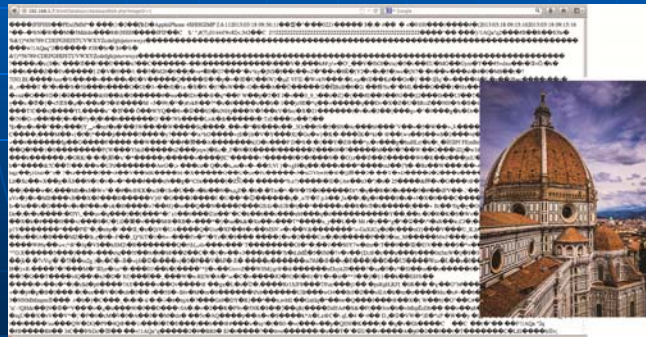
$stmt = $pdo -> prepare($sql);
$stmt -> bindParam(':data', $fileContent,
PDO::PARAM_LOB);
$stmt -> execute();
?>
```

Displaying BLOBs from the Database

File Location

PHP Script

Fig. 14.34 Output of raw data without correct header being sent, rather than the image (inset)



CPET 499/ITC 250 Web Systems, Paul I. Lin

31

Listings 14-32 Code to fetch and echo BLOB image

```
// retrieve blob content from database
$sql = "SELECT * FROM TravellImage WHERE ImageID=:id";
$stmt = $pdo->prepare($sql);
$stmt->bindParam(':id', $_GET['id']);
$stmt->execute();
$result = $stmt->fetch(PDO::FETCH_ASSOC);
if ($result) {
    // Output the MIME header
    header("Content-type: image/jpeg");
    // Output the image
    echo ($result["ImageContent"]);
}
```

CPET 499/ITC 250 Web Systems, Paul I. Lin

32

Q & A

CPET 499/ITC 250 Web Systems, Paul I.
Lin

33