Chapter 4

How to use PHP with a MySQL database

Objectives

Applied

1. Given the specifications for a database application that requires only the skills that are presented in this chapter, develop the application. That includes:
   - Connecting to a MySQL database
   - Handling PDO exceptions
   - Using prepared statements to execute SQL statements
   - Getting the data from the result sets that are returned by SQL statements

Objectives (continued)

Knowledge

1. Describe the PHP code for creating a PDO object that connects to a MySQL database.
2. Describe the PHP code for handling the PDO exceptions that may occur when you try to create a PDO object.
3. List two reasons you should use prepared statements for production applications.
4. Describe how to use a prepared statement to execute a SELECT, INSERT, UPDATE, or DELETE statement.
5. Describe a PHP array and the way that numeric and string indexes are used to access the data in a PHP array.
6. Distinguish between the fetch() and fetchAll() methods of the PDOStatement class.
7. Describe how to use a foreach statement to get data from all rows of a result set.

The syntax for creating an object from any class

```
new ClassName(arguments);
```

The syntax for creating a database object from the PDO class

```
new PDO($dsn, $username, $password);
```

The syntax for a DSN (Data Source Name) for a MySQL database

```
mysql:host=host_address;dbname=database_name
```


How to connect to a MySQL database

```
$dsn = 'mysql:host=localhost;dbname=my_guitar_shop1';
$username = 'mgs_user';
$password = 'pa55word';

// creates PDO object
$db = new PDO($dsn, $username, $password);
```


Key terms

- class
- object
- new keyword
- argument
- PDO object
- DSN (Data Source Name)
- PDO (PHP Data objects) extension

The syntax for a try/catch statement

```
try {
    // statements that might throw an exception
} catch (ExceptionClass $exception_name) {
    // statements that handle the exception
}
```

The syntax for executing a method of any object

```
$objectName->methodName(argumentList)
```


How to handle a PDO exception

```
try {
    $db = new PDO($dsn, $username, $password);
    echo '<p>You are connected to the database!</p>';
} catch (PDOException $e) {
    $error_message = $e->getMessage();
    echo "<p>An error occurred while connecting to
            the database: $error_message </p>";
}
```

How to handle any type of exception

```
try {
    // statements that might throw an exception
} catch (Exception $e) {
    $error_message = $e->getMessage();
    echo "<p>Error message: $error_message </p>";
}
```


Key terms

- exception
- exception handling
- throw an exception
- try/catch statement
- try block
- catch block
- method
- Exception class
- PDOException class

A method of the PDO class for preparing a SQL statement

```
prepare($select_statement)
```

Two methods of the PDOStatement class for executing a statement

```
bindValue($param, $value)
execute()
```


How to execute a SQL statement that doesn't have parameters

```
$query = 'SELECT * FROM products';
$statement = $db->prepare($query);
$statement->execute();
```

How to execute a SQL statement that has a parameter

```
$query = 'SELECT * FROM products
          WHERE categoryID = :category_id';
$statement = $db->prepare($query);
$statement->bindValue(':category_id', $category_id);
$statement->execute();
```


Two more methods of the PDOStatement class

```
fetch()
closeCursor()
```

Code that returns a result set that has one row

```
$query = 'SELECT productCode, productName, listPrice
          FROM products
          WHERE productID = :product_id';
$statement = $db->prepare($query);
$statement->bindValue(':product_id', $product_id);
$statement->execute();
$product = $statement->fetch();
$statement->closeCursor();
```

Code that uses a string index to access each column

```
$product_code = $product['productCode'];
$product_name = $product['productName'];
$product_list_price = $product['listPrice'];
```

Code that uses a numeric index to access each column

```
$product_code = $product[0];
$product_name = $product[1];
$product_list_price = $product[2];
```


Key terms

- array
- elements
- index

Another method of the PDOStatement class

```
fetchAll()
```

Code that returns a result set of two or more rows

```
$query = 'SELECT productCode, productName, listPrice
          FROM products
          WHERE categoryID = :category_id;'
$statement = $db->prepare($query);
$statement->bindValue(':category_id', $category_id);
$statement->execute();
$products = $statement->fetchAll();
$statement->closeCursor();
```


How to use a foreach statement to display the result set in an HTML table

```
<?php foreach ($products as $product) { ?>
<tr>
    <td><?php echo $product['productCode']; ?></td>
    <td><?php echo $product['productName']; ?></td>
    <td><?php echo $product['listPrice']; ?></td>
</tr>
<?php } ?>
```

Another syntax for the foreach statement that works better within PHP tags

```
<?php foreach ($products as $product) : ?>
<tr>
    <td><?php echo $product['productCode']; ?></td>
    <td><?php echo $product['productName']; ?></td>
    <td><?php echo $product['listPrice']; ?></td>
</tr>
<?php endforeach; ?>
```


Key terms

- foreach statement
- foreach loop

How to execute an INSERT statement

```
$category_id = 1;
$code = 'strat';
$name = 'Fender Stratocaster';
$price = 699.99;

$query = "INSERT INTO products
          (categoryID, productCode, productName,
listPrice)
       VALUES
          (:category_id, :code, :name, :price)";

$statement = $db->prepare($query);
$statement->bindValue(':category_id', $category_id);
$statement->bindValue(':code', $code);
$statement->bindValue(':name', $name);
$statement->bindValue(':price', $price);
$statement->execute();
$statement->closeCursor();
```

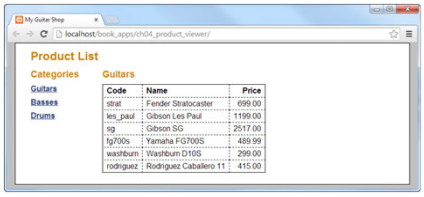
How to execute an UPDATE statement

```
$product_id = 4;
$price = 599.99;

$query = "UPDATE products
          SET listPrice = :price
          WHERE productID = :product_id";

$statement = $db->prepare($query);
$statement->bindValue(':price', $price);
$statement->bindValue(':product_id', $product_id);
$statement->execute();
$statement->closeCursor();
```

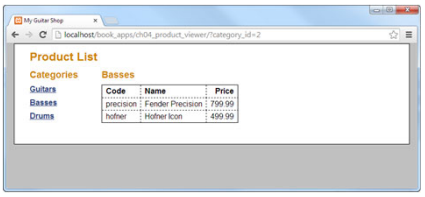

How to execute a DELETE statement

```
$product_id = 4;

$query = "DELETE FROM products
          WHERE productID = :product_id";

$statement = $db->prepare($query);
$statement->bindValue(':product_id', $product_id);
$statement->execute();
$statement->closeCursor();
```


The user interface



The user interface after the user selects a new category



The database.php file

```
<?php
    $dsn = 'mysql:host=localhost;dbname=my_guitar_shop1';
    $username = 'mgs_user';
    $password = 'pa55word';

    try {
        $db = new PDO($dsn, $username, $password);
    } catch (PDOException $e) {
        $error_message = $e->getMessage();
        include('database_error.php');
        exit();
    }
?>
```


The database_error.php file

```
<!DOCTYPE html>
<html>
    <!-- the head section -->
    <head>
        <title>My Guitar Shop</title>
        <link rel="stylesheet" type="text/css" href="main.css">
    </head>

    <!-- the body section -->
    <body>
    <main>
        <h1>Database Error</h1>
        <p>There was an error connecting to the database.</p>
        <p>The database must be installed as described
           in appendix A.</p>
        <p>The database must be running as described
           in chapter 1.</p>
        <p>Error message: <?php echo $error_message; ?></p>
    </main>
    </body>
</html>
```

The index.php file

```
<?php
require('database.php');

// Get category ID
$category_id = filter_input(INPUT_GET, 'category_id',
                            FILTER_VALIDATE_INT);
if ($category_id == NULL || $category_id == FALSE) {
    $category_id = 1;
}

// Get name for selected category
$queryCategory = 'SELECT * FROM categories
                  WHERE categoryID = :category_id';
$statement1 = $db->prepare($queryCategory);
$statement1->bindValue(':category_id', $category_id);
$statement1->execute();
$category = $statement1->fetch();
$category_name = $category['categoryName'];
$statement1->closeCursor();
```


The index.php file (continued)

```
// Get all categories
$queryAllCategories = 'SELECT * FROM categories
                       ORDER BY categoryID';
$statement2 = $db->prepare($queryAllCategories);
$statement2->execute();
$categories = $statement2->fetchAll();
$statement2->closeCursor();

// Get products for selected category
$queryProducts = 'SELECT * FROM products
                  WHERE categoryID = :category_id
                  ORDER BY productID';
$statement3 = $db->prepare($queryProducts);
$statement3->bindValue(':category_id', $category_id);
$statement3->execute();
$products = $statement3->fetchAll();
$statement3->closeCursor();
?>
```


The index.php file (continued)

```
<!DOCTYPE html>
<html>
<!-- the head section -->
<head>
    <title>My Guitar Shop</title>
    <link rel="stylesheet" type="text/css" href="main.css" />
</head>
<!-- the body section -->
<body>
<main>
    <h1>Product List</h1>
    <aside>
        <!-- display a list of categories -->
        <h2>Categories</h2>
        <nav>
        <ul>
            <?php foreach ($categories as $category) : ?>
            <li>
                <a href="?category_id=<?php echo
                    $category['categoryID']; ?>">
                    <?php echo $category['categoryName']; ?>
                </a>
            </li>
```

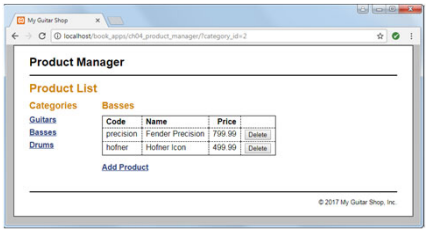

The index.php file (continued)

```
            <?php endforeach; ?>
        </ul>
        </nav>
    </aside>

    <section>
        <!-- display a table of products -->
        <h2><?php echo $category_name; ?></h2>
        <table>
            <tr>
                <th>Code</th>
                <th>Name</th>
                <th class="right">Price</th>
            </tr>
```

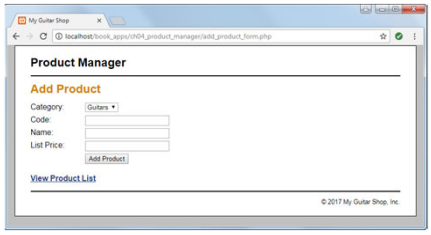

The index.php file (continued)

```
            <?php foreach ($products as $product) : ?>
            <tr>
                <td><?php echo $product['productCode']; ?></td>
                <td><?php echo $product['productName']; ?></td>
                <td class="right"><?php echo
                                  $product['listPrice']; ?></td>
            </tr>
            <?php endforeach; ?>
        </table>
    </section>
</main>
<footer></footer>
</body>
</html>
```


The Product List page



The Add Product page



The index.php file

```
<?php
require_once('database.php');

// Get category ID
if (!isset($category_id)) {
    $category_id = filter_input(INPUT_GET, 'category_id',
            FILTER_VALIDATE_INT);
    if ($category_id == NULL || $category_id == FALSE) {
        $category_id = 1;
    }
}

// Get name for selected category
$queryCategory = 'SELECT * FROM categories
                  WHERE categoryID = :category_id';
$statement1 = $db->prepare($queryCategory);
$statement1->bindValue(':category_id', $category_id);
$statement1->execute();
$category = $statement1->fetch();
$category_name = $category['categoryName'];
$statement1->closeCursor();
```


The index.php file (continued)

```
// Get all categories
$queryAllCategories = 'SELECT * FROM categories
                       ORDER BY categoryID';
$statement2 = $db->prepare($queryAllCategories);
$statement2->execute();
$categories = $statement2->fetchAll();
$statement2->closeCursor();

// Get products for selected category
$queryProducts = 'SELECT * FROM products
                  WHERE categoryID = :category_id
                  ORDER BY productID';
$statement3 = $db->prepare($queryProducts);
$statement3->bindValue(':category_id', $category_id);
$statement3->execute();
$products = $statement3->fetchAll();
$statement3->closeCursor();
?>
```


The index.php file (continued)

```
<!DOCTYPE html>
<html>

<!-- the head section -->
<head>
    <title>My Guitar Shop</title>
    <link rel="stylesheet" type="text/css" href="main.css" />
</head>

<!-- the body section -->
<body>
<header><h1>Product Manager</h1></header>
<main>
    <h1>Product List</h1>
```


The index.php file (continued)

```
    <aside>
        <!-- display a list of categories -->
        <h2>Categories</h2>
        <nav>
        <ul>
           <?php foreach ($categories as $category) : ?>
           <li><a href="?category_id=<?php echo
            $category['categoryID']; ?>">
                <?php echo $category['categoryName']; ?>
            </a>
           </li>
           <?php endforeach; ?>
        </ul>
        </nav>
    </aside>
```


The index.php file (continued)

```
    <section>
        <!-- display a table of products -->
        <h2><?php echo $category_name; ?></h2>
        <table>
            <tr>
                <th>Code</th>
                <th>Name</th>
                <th class="right">Price</th>
                <th> </th>
            </tr>

            <?php foreach ($products as $product) : ?>
            <tr>
                <td><?php echo $product['productCode']; ?></td>
                <td><?php echo $product['productName']; ?></td>
                <td class="right"><?php echo $product['listPrice'];
             ?></td>
```

The index.php file (continued)

```
            <td><form action="delete_product.php" method="post">
                <input type="hidden" name="product_id"
                       value="<?php echo $product['productID']; ?>">
                <input type="hidden" name="category_id"
                       value="<?php echo $product['categoryID']; ?>">
                <input type="submit" value="Delete">
               </form></td>
            </tr>
            <?php endforeach; ?>
        </table>
        <p><a href="add_product_form.php">Add Product</a></p>
    </section>
</main>
<footer>
    <p>&copy; <?php echo date("Y"); ?> My Guitar Shop, Inc.</p>
</footer>
</body>
</html>
```


The delete_product.php file

```
<?php
require_once('database.php');

$product_id = filter_input(INPUT_POST, 'product_id',
                           FILTER_VALIDATE_INT);
$category_id = filter_input(INPUT_POST, 'category_id',
                            FILTER_VALIDATE_INT);

// Delete the product from the database
if ($product_id != false && $category_id != false) {
    $query = 'DELETE FROM products
              WHERE productID = :product_id';
    $statement = $db->prepare($query);
    $statement->bindValue(':product_id', $product_id);
    $success = $statement->execute();
    $statement->closeCursor();
}

// Display the Product List page
include('index.php');
```


The add_product_form.php file

```
<?php
require('database.php');
$query = 'SELECT *
          FROM categories
          ORDER BY categoryID';
$statement = $db->prepare($query);
$statement->execute();
$categories = $statement->fetchAll();
$statement->closeCursor();
?>
```


The add_product_form.php file (continued)

```
<!DOCTYPE html>
<html>
<head>
<title>My Guitar Shop</title>
    <link rel="stylesheet" type="text/css" href="main.css">
</head>

<body>
    <header><h1>Product Manager</h1></header>
    <main>
        <h1>Add Product</h1>
        <form action="add_product.php" method="post"
              id="add_product_form">

            <label>Category:</label>
            <select name="category_id">
            <?php foreach ($categories as $category) : ?>
                <option value="<?php echo $category['categoryID']; ?>">
                    <?php echo $category['categoryName']; ?>
                </option>
            <?php endforeach; ?>
            </select><br>
```


The add_product_form.php file (continued)

```
            <label>Code:</label>
            <input type="text" name="code"><br>

            <label>Name:</label>
            <input type="text" name="name"><br>

            <label>List Price:</label>
            <input type="text" name="price"><br>

            <label> </label>
            <input type="submit" value="Add Product"><br>
        </form>
        <p><a href="index.php">View Product List</a></p>
    </main>

    <footer>
        <p>&copy; <?php echo date("Y"); ?> My Guitar Shop, Inc.</p>
    </footer>
</body>
</html>
```


The add_product.php file

```
<?php
// Get the product data
$category_id = filter_input(INPUT_POST, 'category_id',
                            FILTER_VALIDATE_INT);
$code = filter_input(INPUT_POST, 'code');
$name = filter_input(INPUT_POST, 'name');
$price = filter_input(INPUT_POST, 'price', FILTER_VALIDATE_FLOAT);

// Validate inputs
if ($category_id == null || $category_id == false || $code == null
    || $name == null || $price == null || $price == false ) {
    $error = "Invalid product data. Check all fields and try
    again.";
    include('error.php');
} else {
    require_once('database.php');
```

The add_product.php file (continued)

```
    // Add the product to the database
    $query = 'INSERT INTO products
                 (categoryID, productCode, productName, listPrice)
              VALUES
                 (:category_id, :code, :name, :price)';
    $statement = $db->prepare($query);
    $statement->bindValue(':category_id', $category_id);
    $statement->bindValue(':code', $code);
    $statement->bindValue(':name', $name);
    $statement->bindValue(':price', $price);
    $statement->execute();
    $statement->closeCursor();

    // Display the Product List page
    include('index.php');
}
?>
```