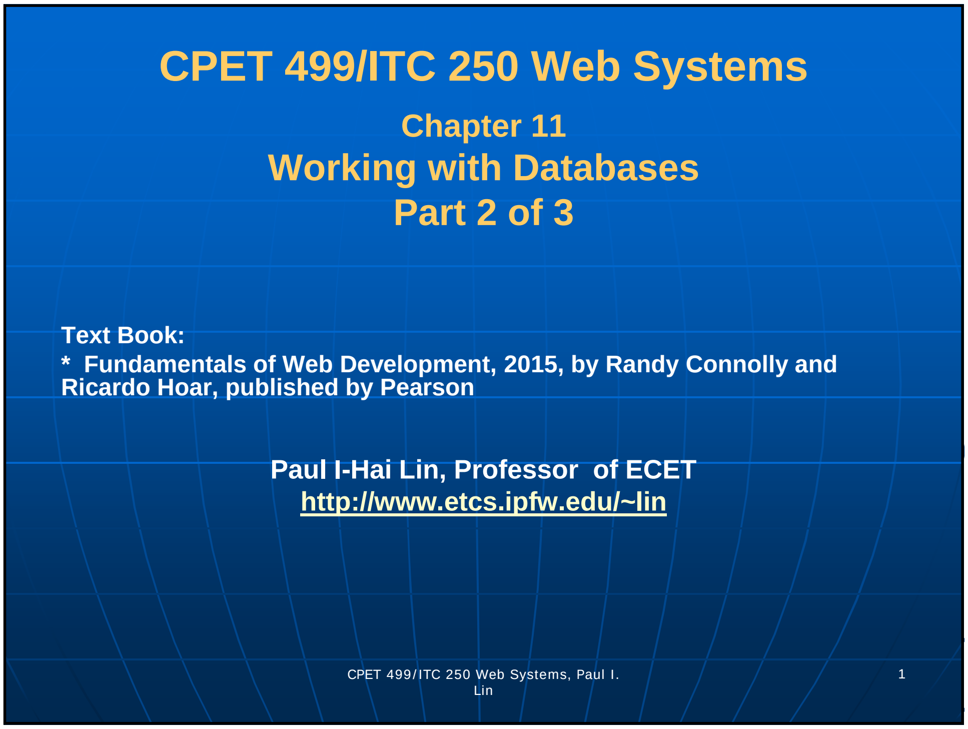
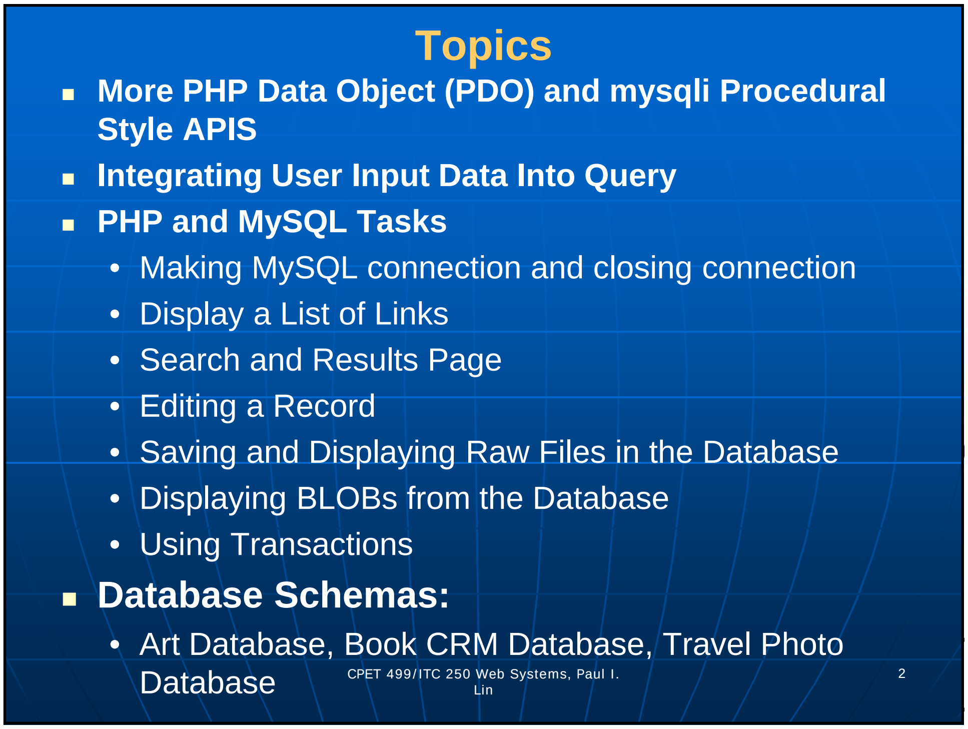


# CPET 499/ITC 250 Web Systems

## Chapter 11
## Working with Databases
## Part 2 of 3

Text Book:
* Fundamentals of Web Development, 2015, by Randy Connolly and Ricardo Hoar, published by Pearson

Paul I-Hai Lin, Professor of ECET
http://www.etcs.ipfw.edu/~lin

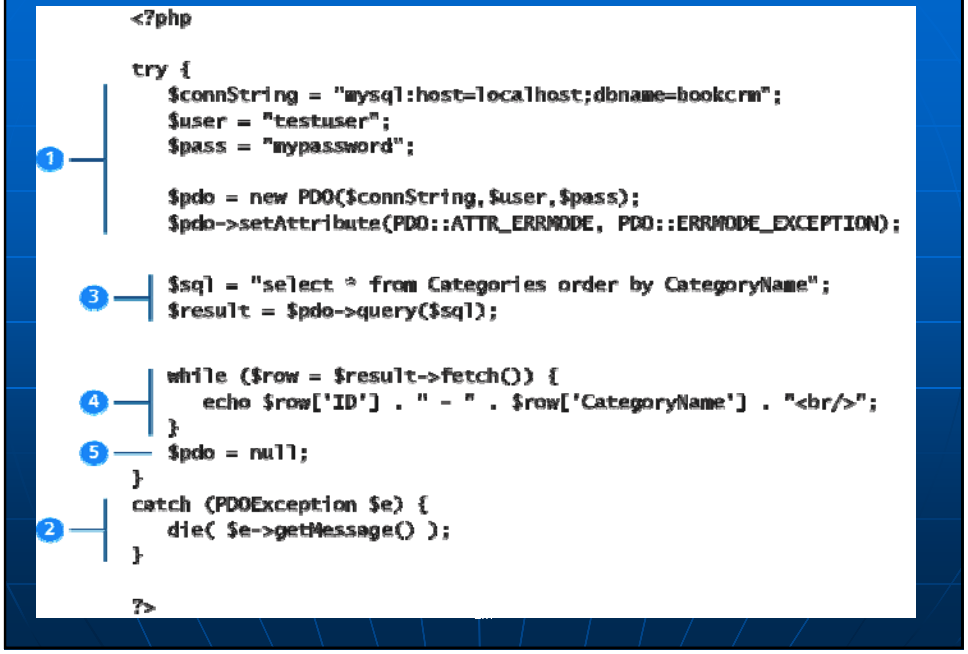


## Topics

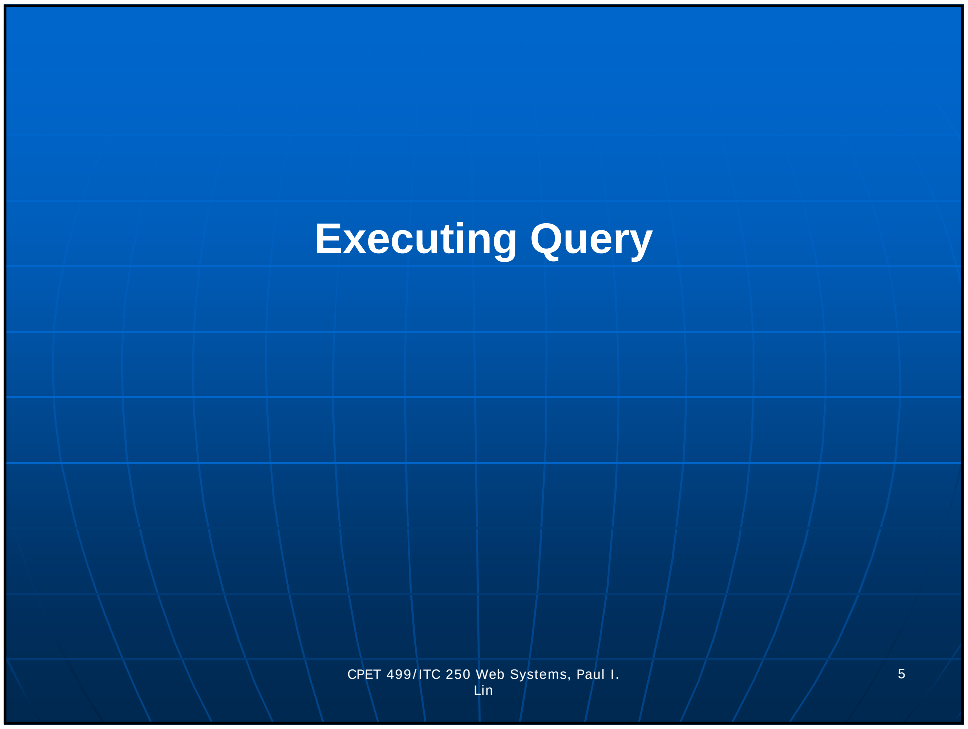
- **More PHP Data Object (PDO) and mysqli Procedural Style APIS**
- **Integrating User Input Data Into Query**
- **PHP and MySQL Tasks**
  - Making MySQL connection and closing connection
  - Display a List of Links
  - Search and Results Page
  - Editing a Record
  - Saving and Displaying Raw Files in the Database
  - Displaying BLOBs from the Database
  - Using Transactions
- **Database Schemas:**
  - Art Database, Book CRM Database, Travel Photo Database

# Connecting to MySQL Using PHP Data Object (PDO)

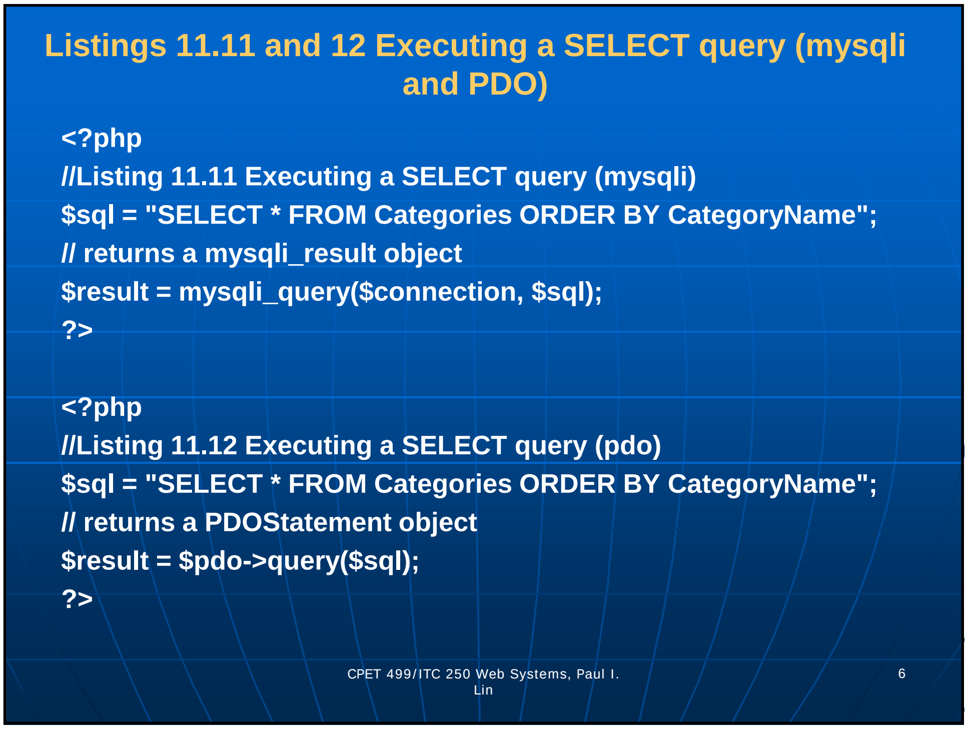
## Figure 11.20 Basic Database Connection using PHP PDO



```
<?php

try {
   $connString = "mysql:host=localhost;dbname=bookcrm";
   $user = "testuser";
   $pass = "mypassword";

   $pdo = new PDO($connString,$user,$pass);
   $pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);

   $sql = "select * from Categories order by CategoryName";
   $result = $pdo->query($sql);

   while ($row = $result->fetch()) {
      echo $row['ID'] . " - " . $row['CategoryName'] . "<br/>";
   }
   $pdo = null;
}
catch (PDOException $e) {
   die( $e->getMessage() );
}

?>
```

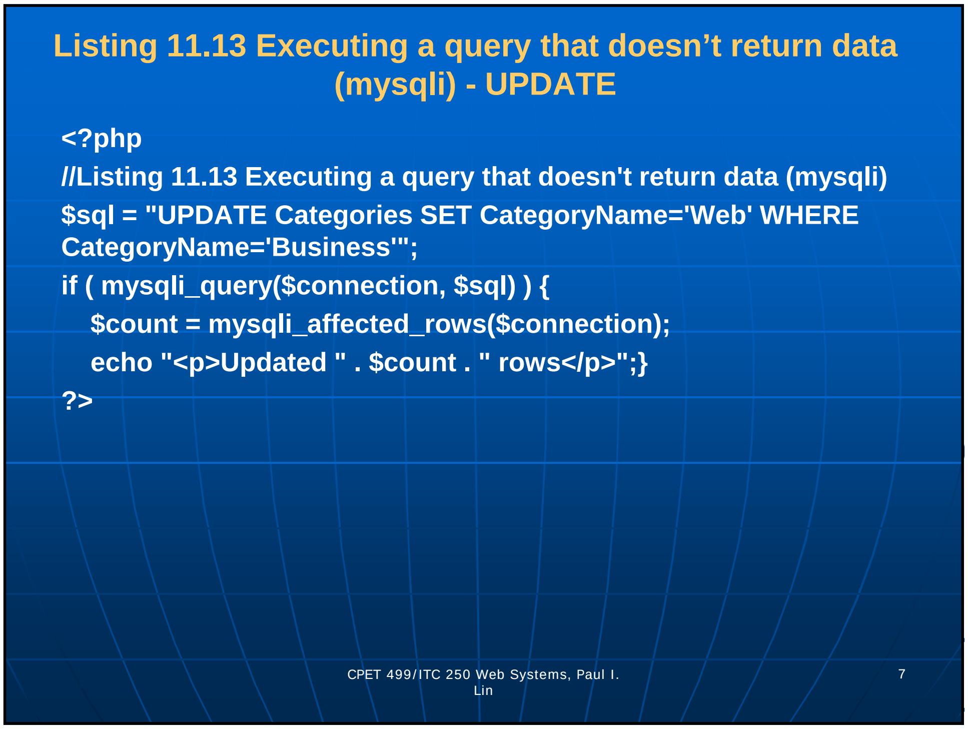


## Executing Query



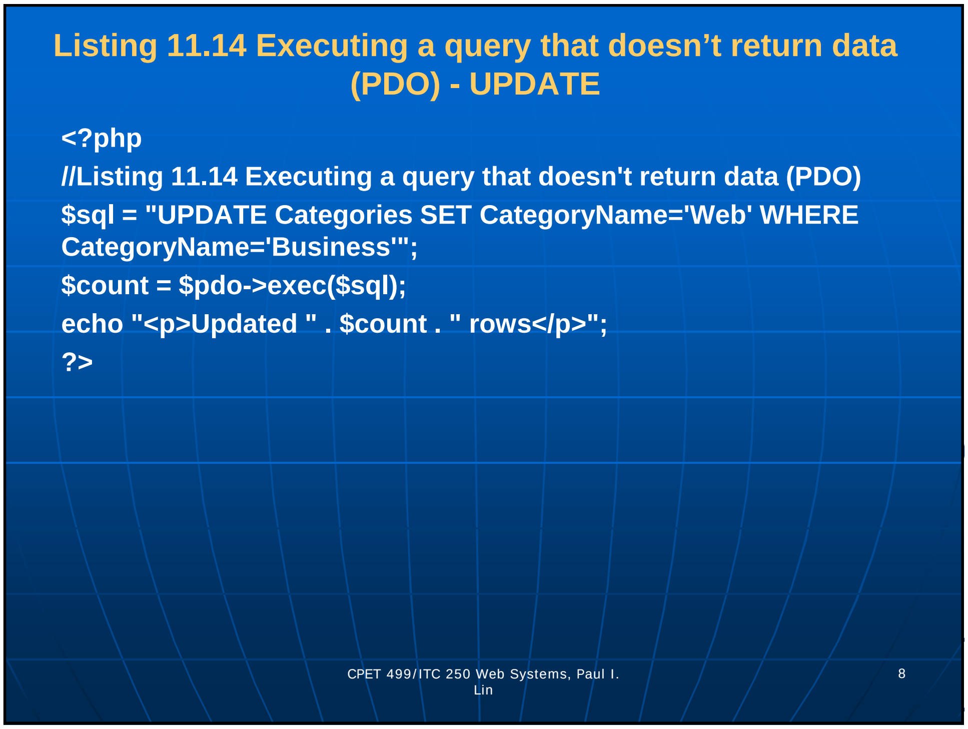
## Listings 11.11 and 12 Executing a SELECT query (mysqli and PDO)

```
<?php
//Listing 11.11 Executing a SELECT query (mysqli)
$sql = "SELECT * FROM Categories ORDER BY CategoryName";
// returns a mysqli_result object
$result = mysqli_query($connection, $sql);
?>

<?php
//Listing 11.12 Executing a SELECT query (pdo)
$sql = "SELECT * FROM Categories ORDER BY CategoryName";
// returns a PDOStatement object
$result = $pdo->query($sql);
?>
```



## Listing 11.13 Executing a query that doesn’t return data (mysqli) - UPDATE

```
<?php
//Listing 11.13 Executing a query that doesn't return data (mysqli)
$sql = "UPDATE Categories SET CategoryName='Web' WHERE
CategoryName='Business'";
if ( mysqli_query($connection, $sql) ) {
   $count = mysqli_affected_rows($connection);
   echo "<p>Updated " . $count . " rows</p>";}
?>
```

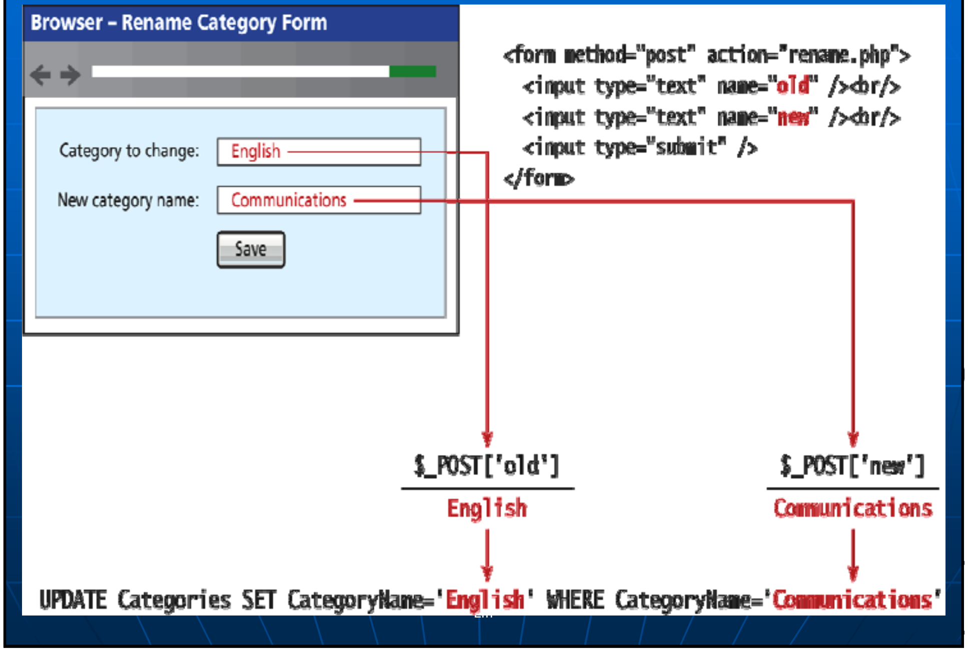


## Listing 11.14 Executing a query that doesn’t return data (PDO) - UPDATE

```
<?php
//Listing 11.14 Executing a query that doesn't return data (PDO)
$sql = "UPDATE Categories SET CategoryName='Web' WHERE
CategoryName='Business'";
$count = $pdo->exec($sql);
echo "<p>Updated " . $count . " rows</p>";
?>
```

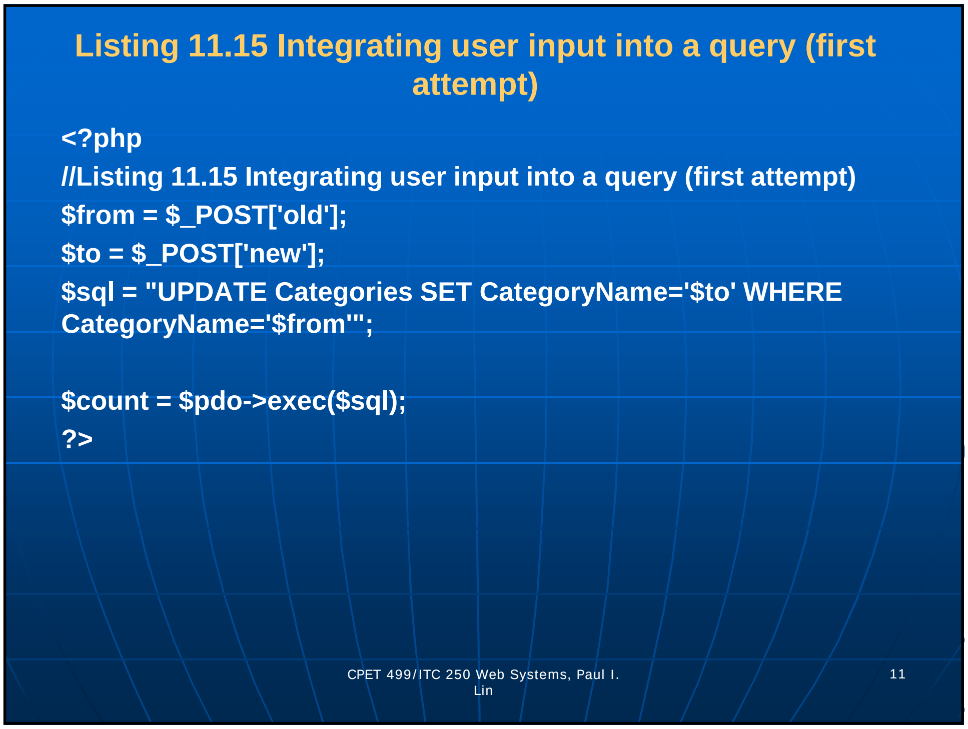
- Integrating User Data into An Query
- Sanitizing User Data
- Prepare Statements

## Figure 11.21 Integrating user input data into a query



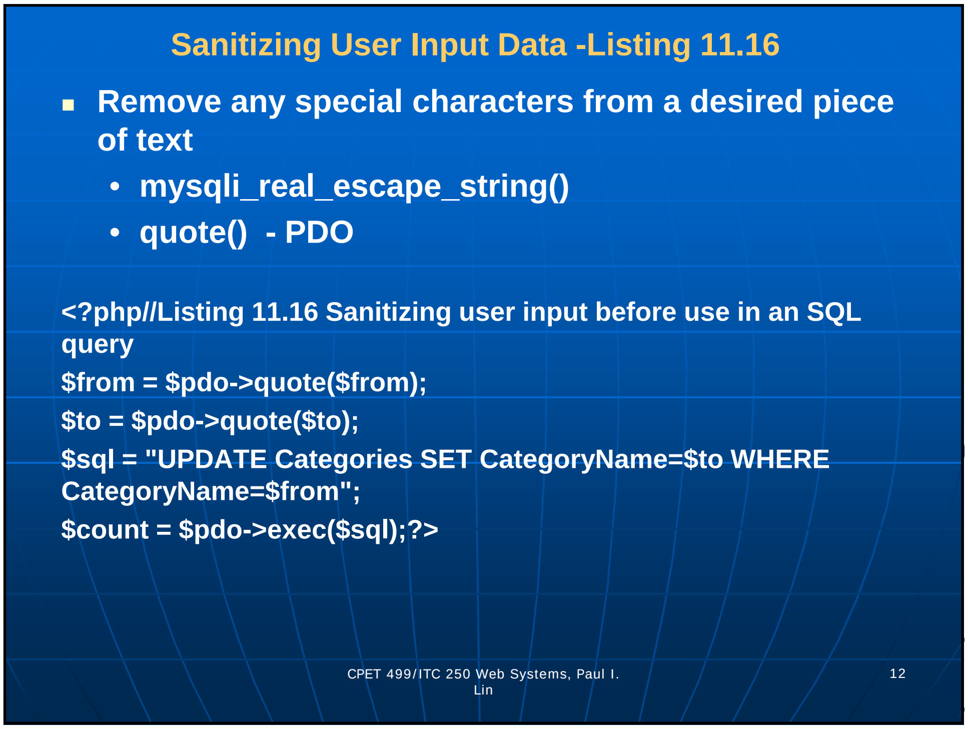
```
<form method="post" action="rename.php">
  <input type="text" name="old" /><br/>
  <input type="text" name="new" /><br/>
  <input type="submit" />
</form>
```

```
UPDATE Categories SET CategoryName='English' WHERE CategoryName='Communications'
```



## Listing 11.15 Integrating user input into a query (first attempt)

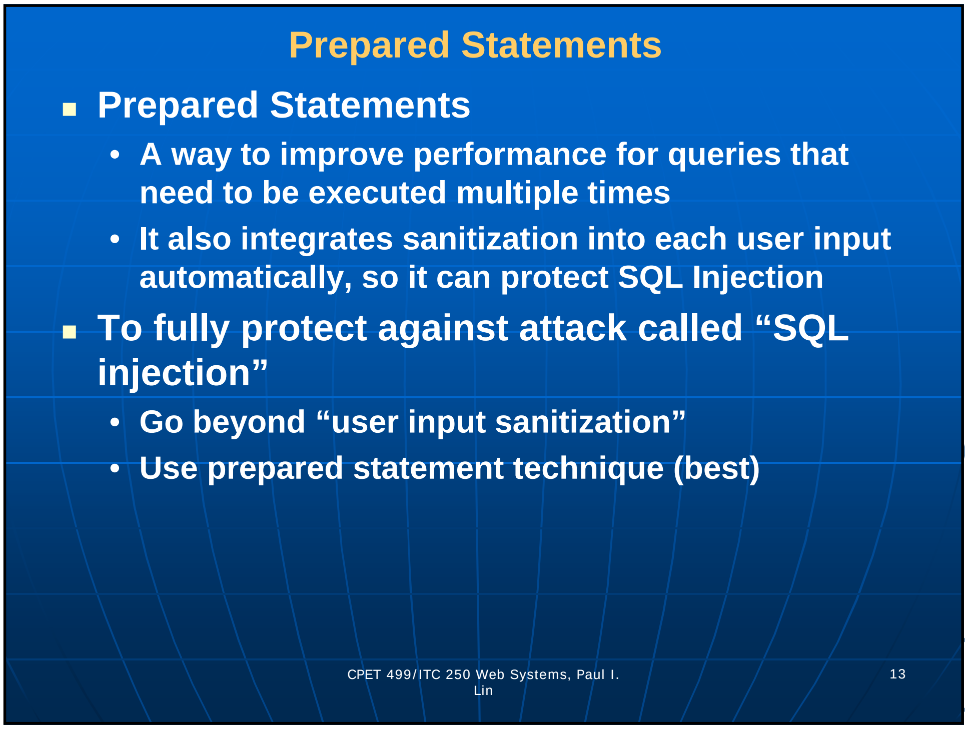
```
<?php
//Listing 11.15 Integrating user input into a query (first attempt)
$from = $_POST['old'];
$to = $_POST['new'];
$sql = "UPDATE Categories SET CategoryName='$to' WHERE
CategoryName='$from'";

$count = $pdo->exec($sql);
?>
```



## Sanitizing User Input Data -Listing 11.16

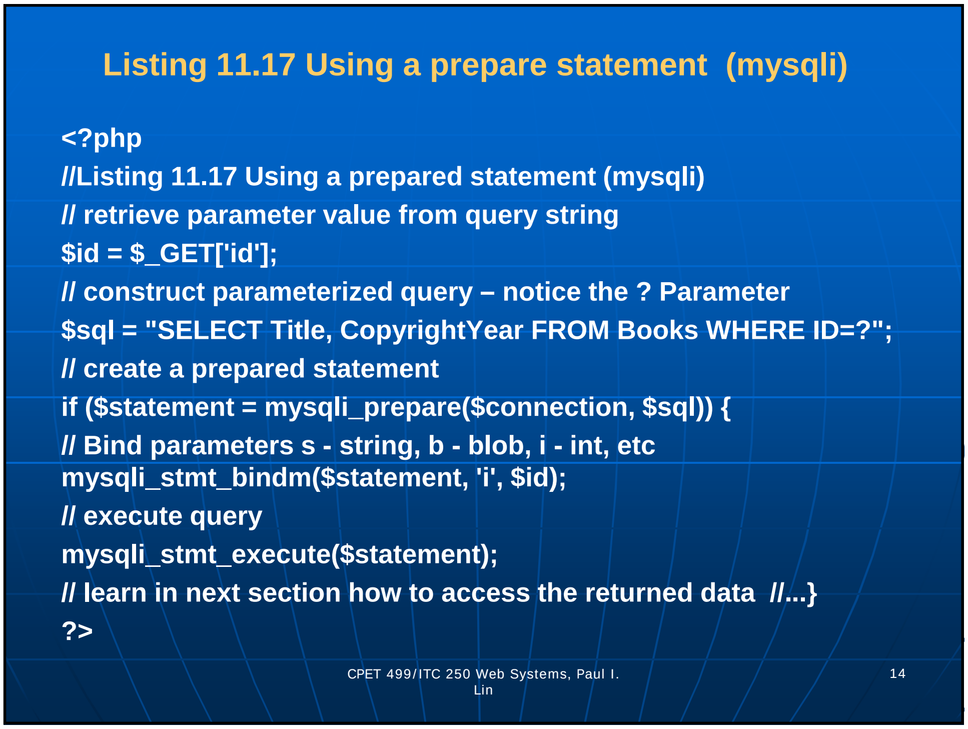
- Remove any special characters from a desired piece of text
  - mysqli_real_escape_string()
  - quote() - PDO

```
<?php//Listing 11.16 Sanitizing user input before use in an SQL
query
$from = $pdo->quote($from);
$to = $pdo->quote($to);
$sql = "UPDATE Categories SET CategoryName=$to WHERE
CategoryName=$from";
$count = $pdo->exec($sql);?>
```



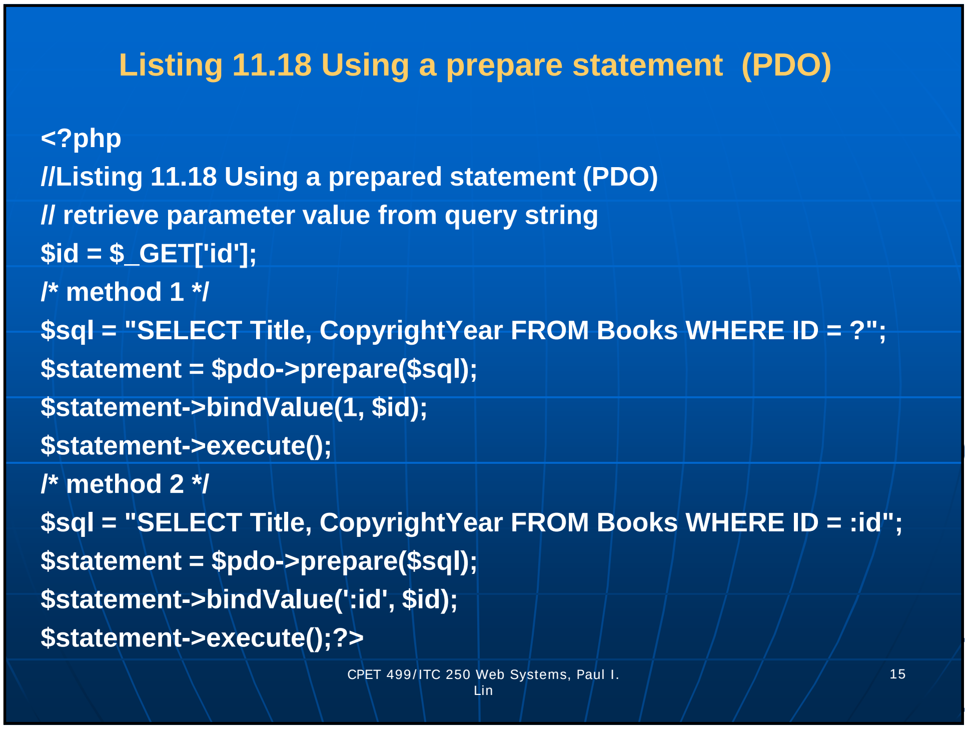
## Prepared Statements

- Prepared Statements
  - A way to improve performance for queries that need to be executed multiple times
  - It also integrates sanitization into each user input automatically, so it can protect SQL Injection
- To fully protect against attack called “SQL injection”
  - Go beyond “user input sanitization”
  - Use prepared statement technique (best)



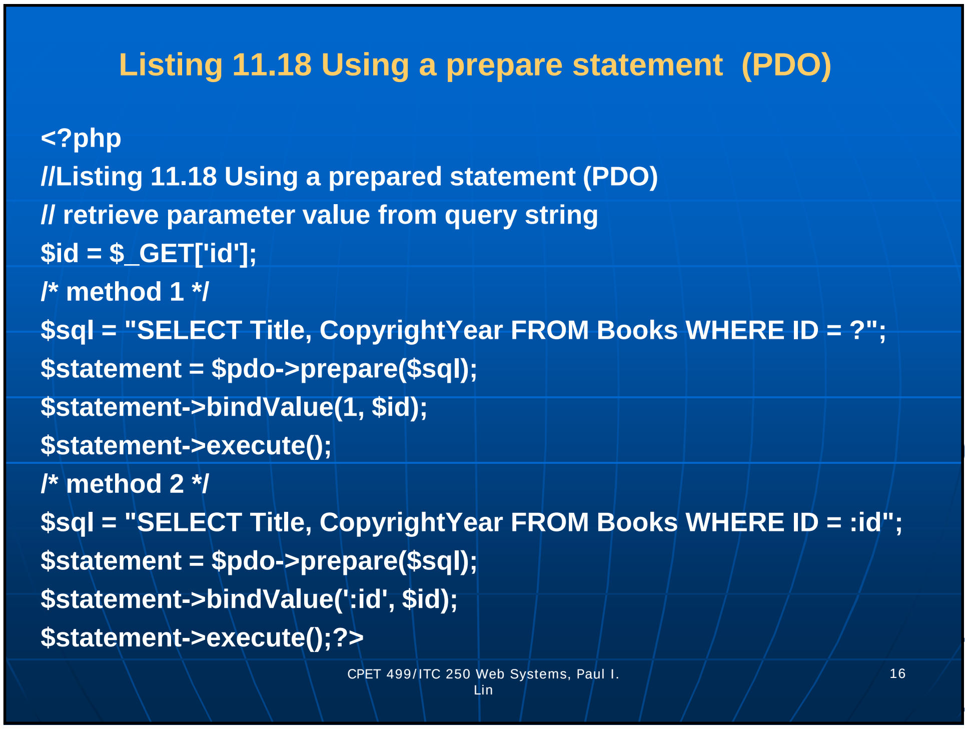
## Listing 11.17 Using a prepare statement (mysqli)

```
<?php
//Listing 11.17 Using a prepared statement (mysqli)
// retrieve parameter value from query string
$id = $_GET['id'];
// construct parameterized query – notice the ? Parameter
$sql = "SELECT Title, CopyrightYear FROM Books WHERE ID=?";
// create a prepared statement
if ($statement = mysqli_prepare($connection, $sql)) {
// Bind parameters s - string, b - blob, i - int, etc
mysqli_stmt_bindm($statement, 'i', $id);
// execute query
mysqli_stmt_execute($statement);
// learn in next section how to access the returned data //...}
?>
```



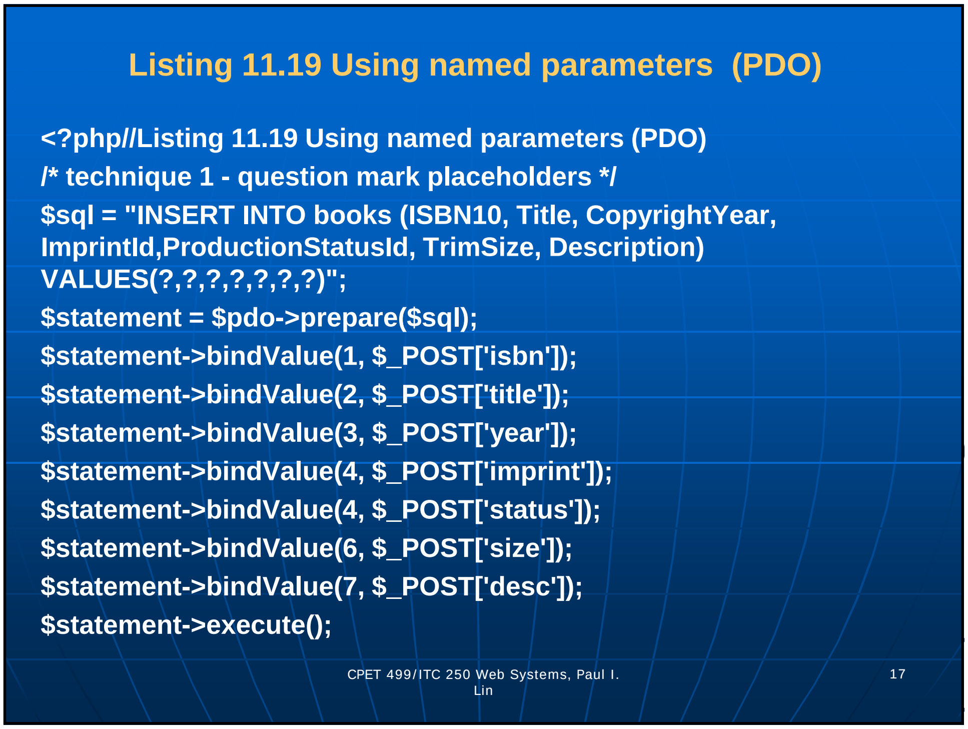
## Listing 11.18 Using a prepare statement  (PDO)

```
<?php
//Listing 11.18 Using a prepared statement (PDO)
// retrieve parameter value from query string
$id = $_GET['id'];
/* method 1 */
$sql = "SELECT Title, CopyrightYear FROM Books WHERE ID = ?";
$statement = $pdo->prepare($sql);
$statement->bindValue(1, $id);
$statement->execute();
/* method 2 */
$sql = "SELECT Title, CopyrightYear FROM Books WHERE ID = :id";
$statement = $pdo->prepare($sql);
$statement->bindValue(':id', $id);
$statement->execute();?>
```



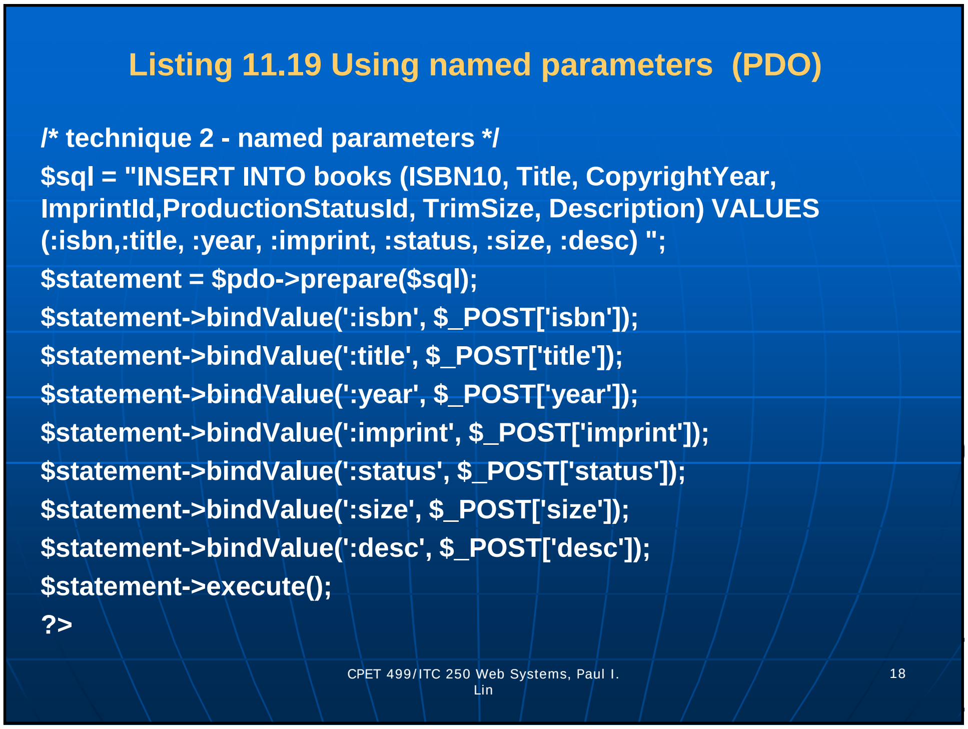
## Listing 11.18 Using a prepare statement  (PDO)

```
<?php
//Listing 11.18 Using a prepared statement (PDO)
// retrieve parameter value from query string
$id = $_GET['id'];
/* method 1 */
$sql = "SELECT Title, CopyrightYear FROM Books WHERE ID = ?";
$statement = $pdo->prepare($sql);
$statement->bindValue(1, $id);
$statement->execute();
/* method 2 */
$sql = "SELECT Title, CopyrightYear FROM Books WHERE ID = :id";
$statement = $pdo->prepare($sql);
$statement->bindValue(':id', $id);
$statement->execute();?>
```



## Listing 11.19 Using named parameters (PDO)

```
<?php//Listing 11.19 Using named parameters (PDO)
/* technique 1 - question mark placeholders */
$sql = "INSERT INTO books (ISBN10, Title, CopyrightYear,
ImprintId,ProductionStatusId, TrimSize, Description)
VALUES(?,?,?,?,?,?,?)";
$statement = $pdo->prepare($sql);
$statement->bindValue(1, $_POST['isbn']);
$statement->bindValue(2, $_POST['title']);
$statement->bindValue(3, $_POST['year']);
$statement->bindValue(4, $_POST['imprint']);
$statement->bindValue(4, $_POST['status']);
$statement->bindValue(6, $_POST['size']);
$statement->bindValue(7, $_POST['desc']);
$statement->execute();
```



## Listing 11.19 Using named parameters (PDO)

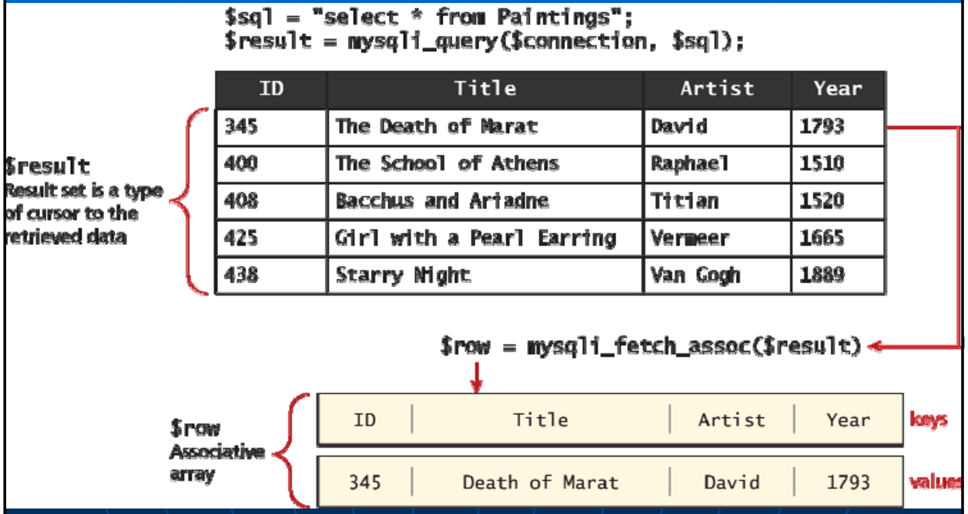
```
/* technique 2 - named parameters */
$sql = "INSERT INTO books (ISBN10, Title, CopyrightYear,
ImprintId,ProductionStatusId, TrimSize, Description) VALUES
(:isbn,:title, :year, :imprint, :status, :size, :desc) ";
$statement = $pdo->prepare($sql);
$statement->bindValue(':isbn', $_POST['isbn']);
$statement->bindValue(':title', $_POST['title']);
$statement->bindValue(':year', $_POST['year']);
$statement->bindValue(':imprint', $_POST['imprint']);
$statement->bindValue(':status', $_POST['status']);
$statement->bindValue(':size', $_POST['size']);
$statement->bindValue(':desc', $_POST['desc']);
$statement->execute();
?>
```

## PHP mysql APIs

Process the Query Results

- Displaying Content from a Result Set
- Perform Calculation
- Search for Something in it
- Other Operation

## Figure 11.22 Fetching From a Result Set



```
$sql = "select * from Paintings";
$result = mysqli_query($connection, $sql);
```

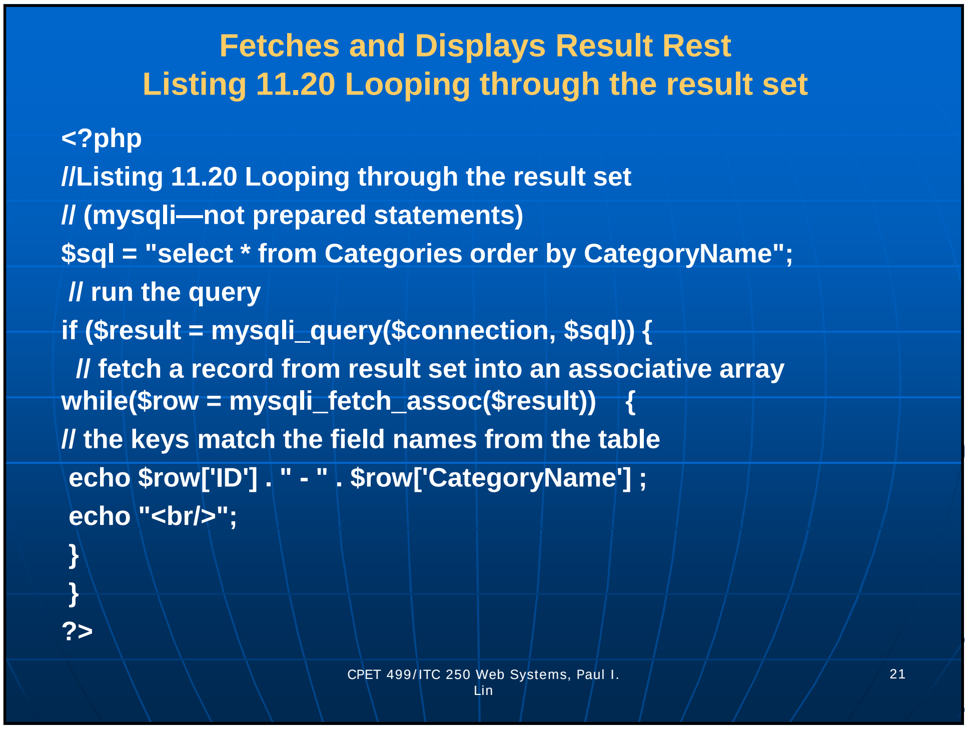
| ID | Title | Artist | Year |
|---|---|---|---|
| 345 | The Death of Marat | David | 1793 |
| 400 | The School of Athens | Raphael | 1510 |
| 408 | Bacchus and Ariadne | Titian | 1520 |
| 425 | Girl with a Pearl Earring | Vermeer | 1665 |
| 438 | Starry Night | Van Gogh | 1889 |

$result
Result set is a type of cursor to the retrieved data

```
$row = mysqli_fetch_assoc($result)
```

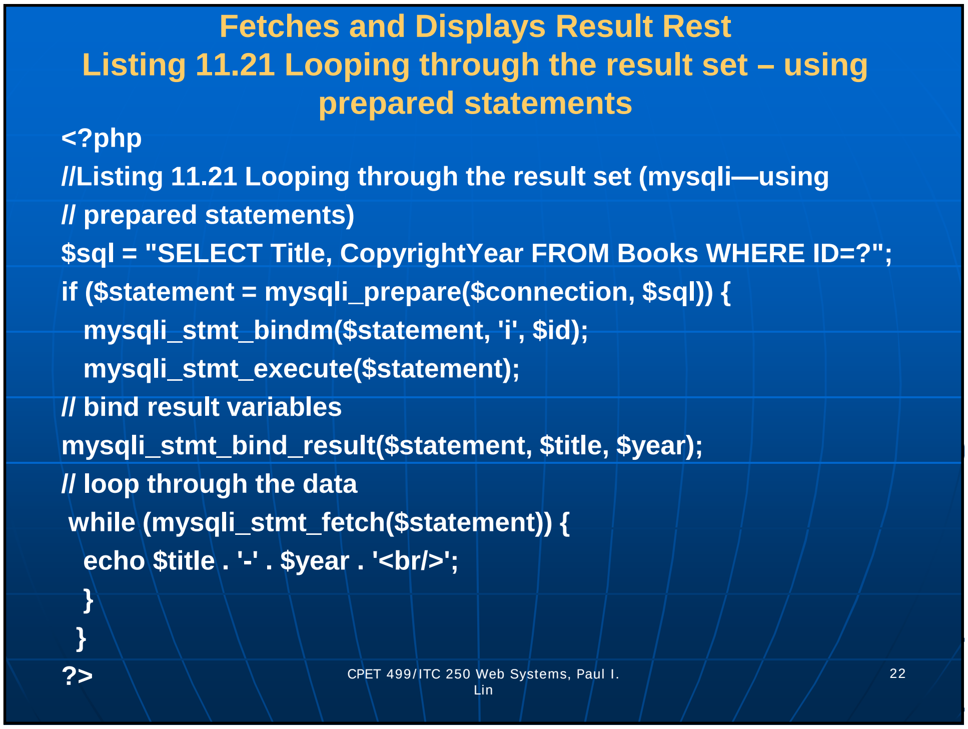
$row
Associative array

| ID | Title | Artist | Year |
|---|---|---|---|
| 345 | Death of Marat | David | 1793 |



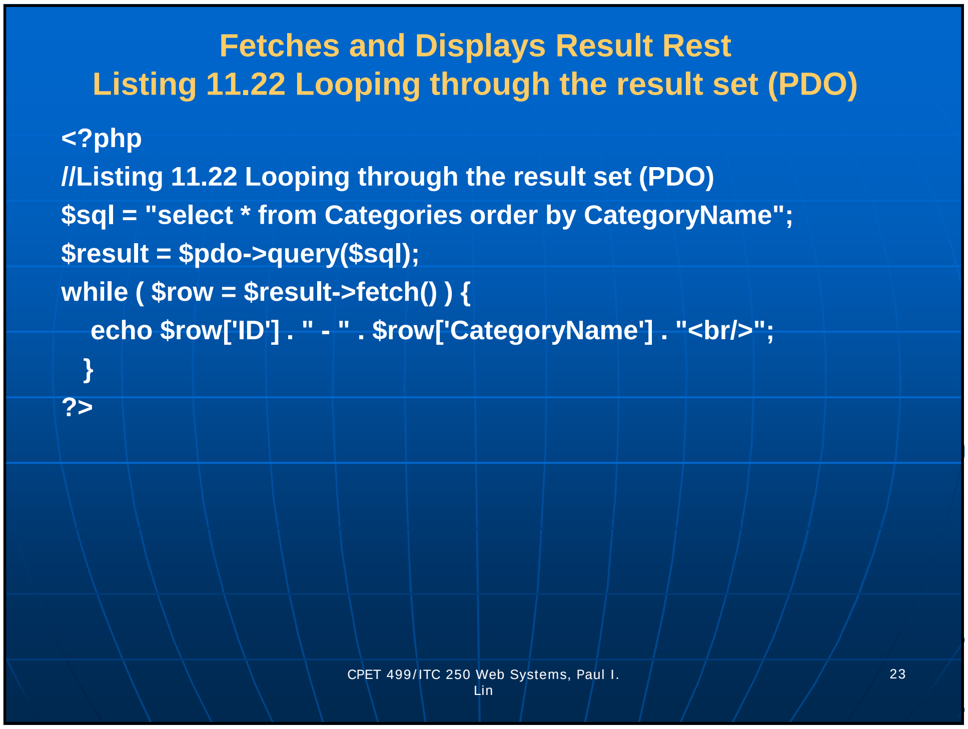
## Fetches and Displays Result Rest
## Listing 11.20 Looping through the result set

```
<?php
//Listing 11.20 Looping through the result set
// (mysqli—not prepared statements)
$sql = "select * from Categories order by CategoryName";
 // run the query
if ($result = mysqli_query($connection, $sql)) {
  // fetch a record from result set into an associative array
while($row = mysqli_fetch_assoc($result))    {
// the keys match the field names from the table
 echo $row['ID'] . " - " . $row['CategoryName'] ;
 echo "<br/>";
 }
 }
?>
```



## Fetches and Displays Result Rest
## Listing 11.21 Looping through the result set – using prepared statements

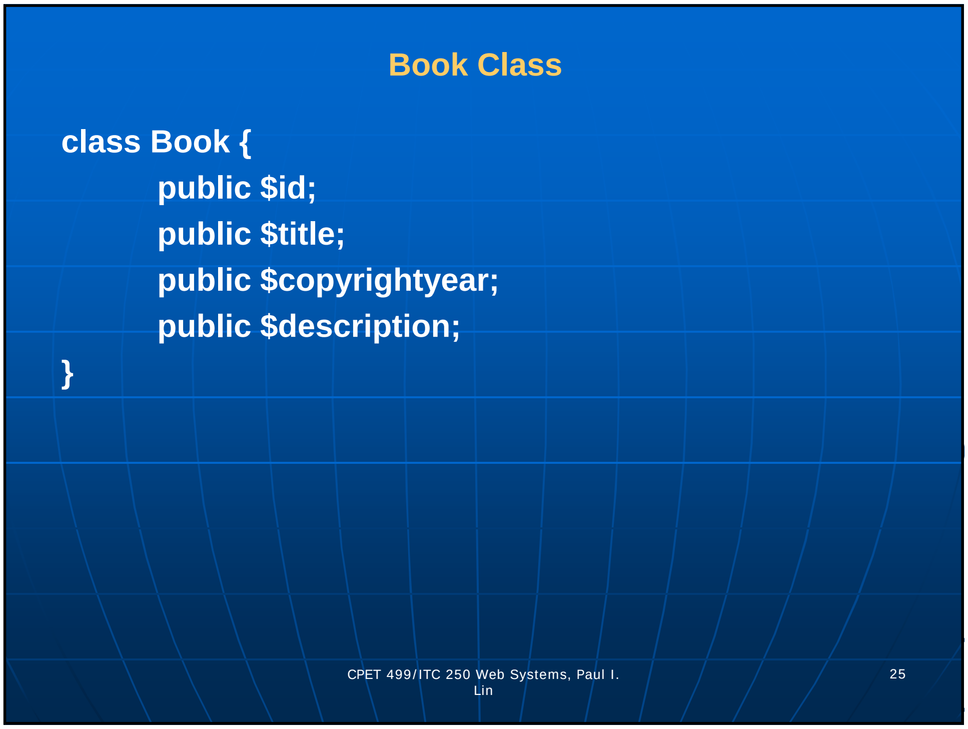
```
<?php
//Listing 11.21 Looping through the result set (mysqli—using
// prepared statements)
$sql = "SELECT Title, CopyrightYear FROM Books WHERE ID=?";
if ($statement = mysqli_prepare($connection, $sql)) {
  mysqli_stmt_bindm($statement, 'i', $id);
  mysqli_stmt_execute($statement);
// bind result variables
mysqli_stmt_bind_result($statement, $title, $year);
// loop through the data
 while (mysqli_stmt_fetch($statement)) {
  echo $title . '-' . $year . '<br/>';
  }
 }
?>
```



## Fetches and Displays Result Rest
## Listing 11.22 Looping through the result set (PDO)

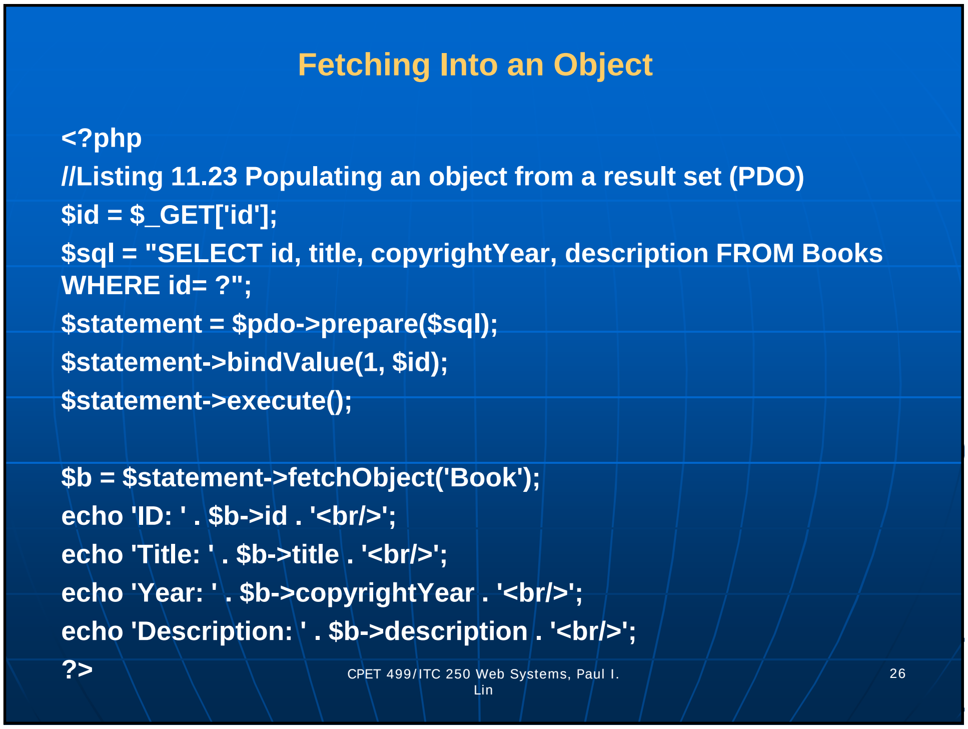
```
<?php
//Listing 11.22 Looping through the result set (PDO)
$sql = "select * from Categories order by CategoryName";
$result = $pdo->query($sql);
while ( $row = $result->fetch() ) {
   echo $row['ID'] . " - " . $row['CategoryName'] . "<br/>";
  }
?>
```

# Fetching Into An Object



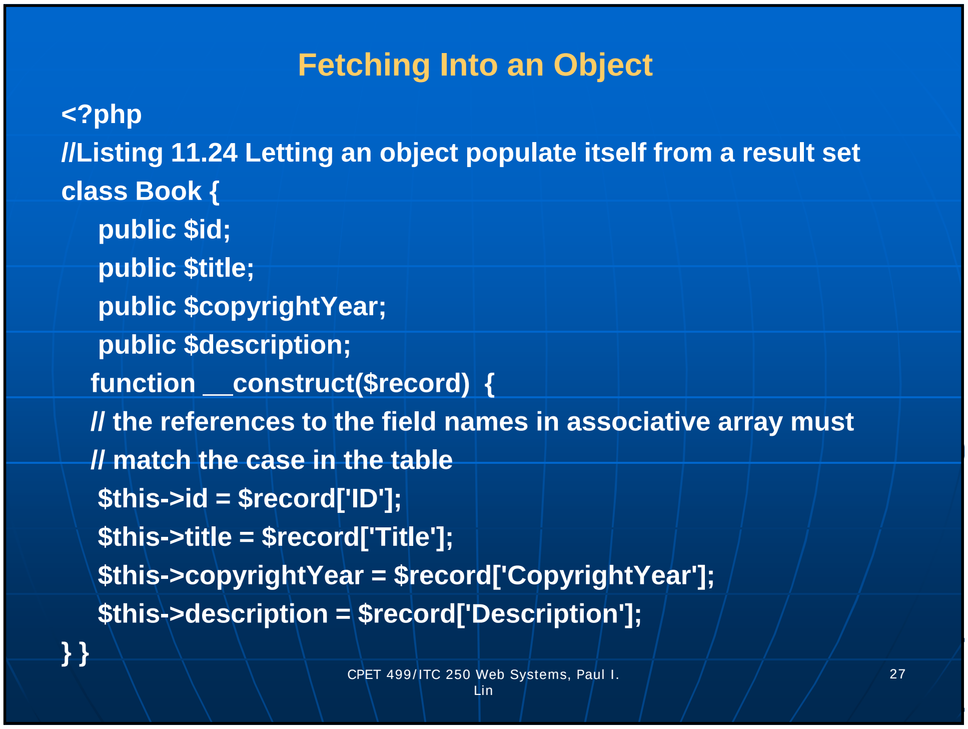
## Book Class

```
class Book {
        public $id;
        public $title;
        public $copyrightyear;
        public $description;
}
```



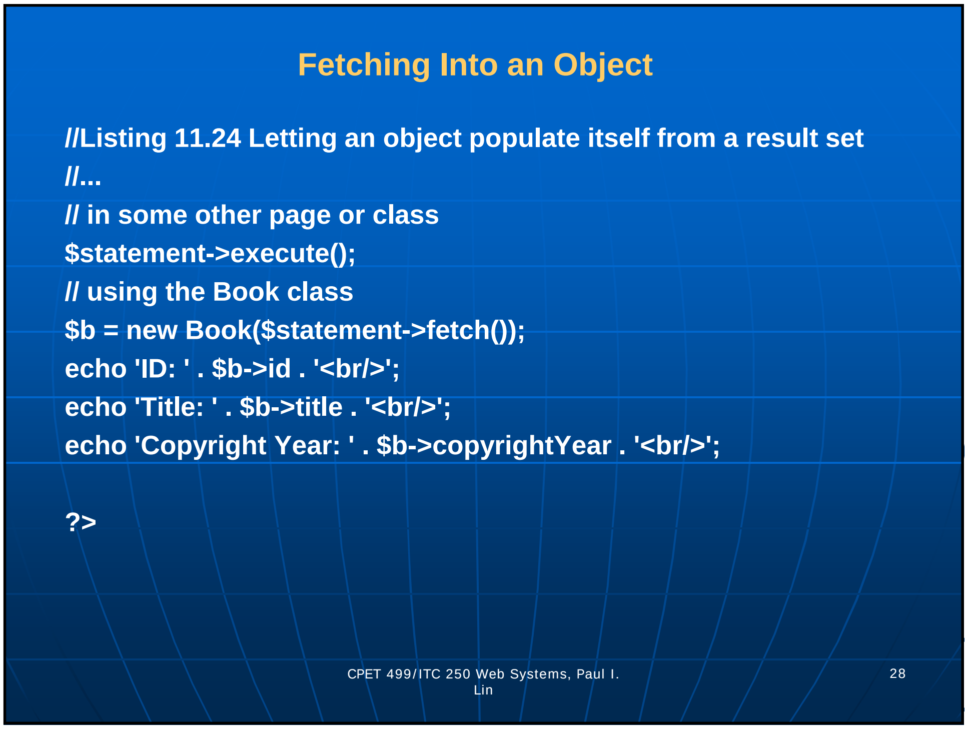
## Fetching Into an Object

```
<?php
//Listing 11.23 Populating an object from a result set (PDO)
$id = $_GET['id'];
$sql = "SELECT id, title, copyrightYear, description FROM Books
WHERE id= ?";
$statement = $pdo->prepare($sql);
$statement->bindValue(1, $id);
$statement->execute();

$b = $statement->fetchObject('Book');
echo 'ID: ' . $b->id . '<br/>';
echo 'Title: ' . $b->title . '<br/>';
echo 'Year: ' . $b->copyrightYear . '<br/>';
echo 'Description: ' . $b->description . '<br/>';
?>
```



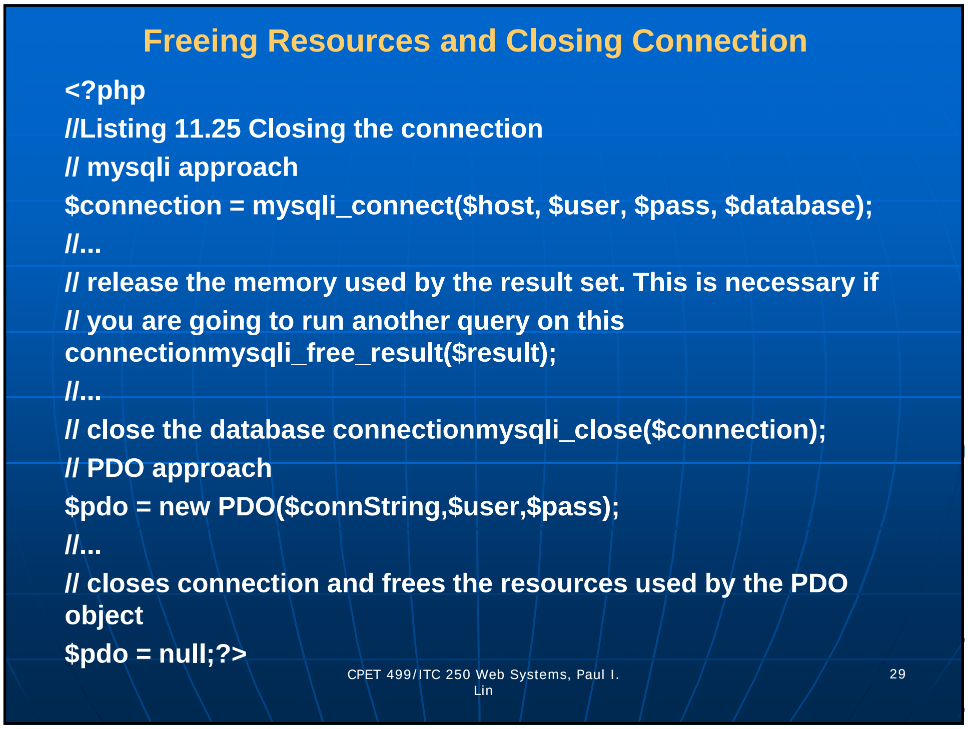
## Fetching Into an Object

```
<?php
//Listing 11.24 Letting an object populate itself from a result set
class Book {
    public $id;
    public $title;
    public $copyrightYear;
    public $description;
   function __construct($record)  {
   // the references to the field names in associative array must
   // match the case in the table
    $this->id = $record['ID'];
    $this->title = $record['Title'];
    $this->copyrightYear = $record['CopyrightYear'];
    $this->description = $record['Description'];
} }
```



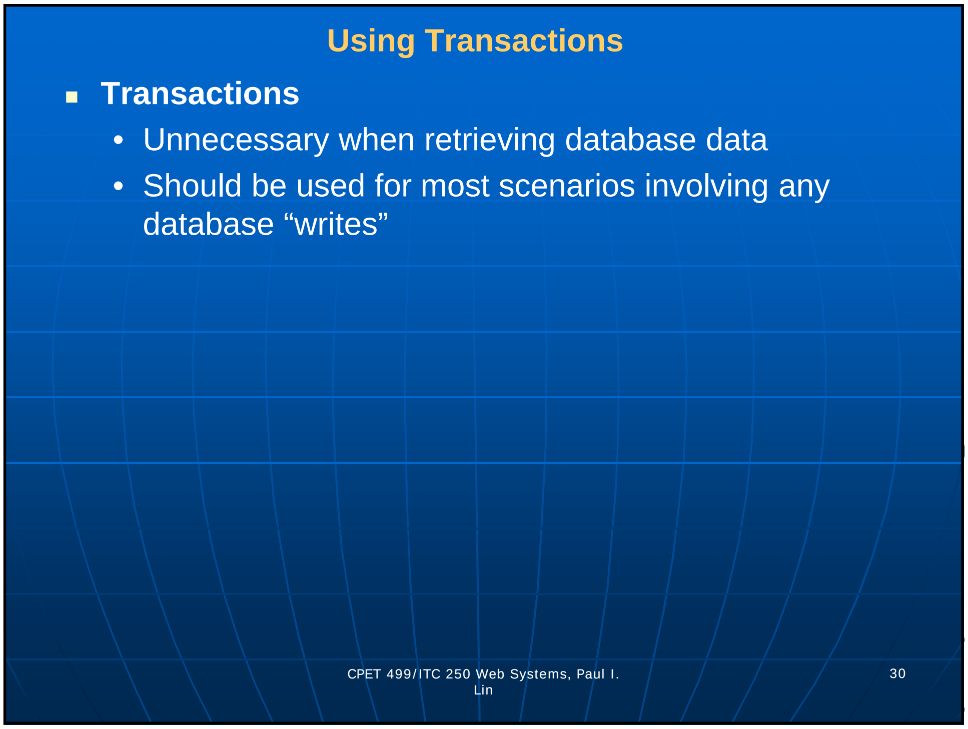
## Fetching Into an Object

```
//Listing 11.24 Letting an object populate itself from a result set
//...
// in some other page or class
$statement->execute();
// using the Book class
$b = new Book($statement->fetch());
echo 'ID: ' . $b->id . '<br/>';
echo 'Title: ' . $b->title . '<br/>';
echo 'Copyright Year: ' . $b->copyrightYear . '<br/>';

?>
```



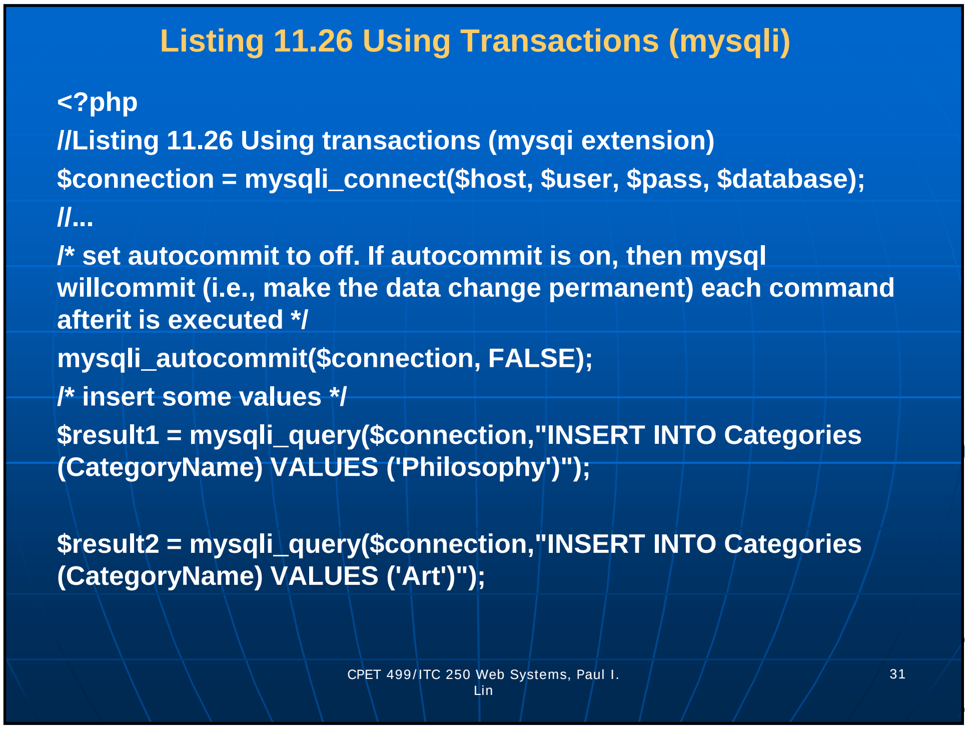
## Freeing Resources and Closing Connection

```
<?php
//Listing 11.25 Closing the connection
// mysqli approach
$connection = mysqli_connect($host, $user, $pass, $database);
//...
// release the memory used by the result set. This is necessary if
// you are going to run another query on this
connectionmysqli_free_result($result);
//...
// close the database connectionmysqli_close($connection);
// PDO approach
$pdo = new PDO($connString,$user,$pass);
//...
// closes connection and frees the resources used by the PDO
object
$pdo = null;?>
```



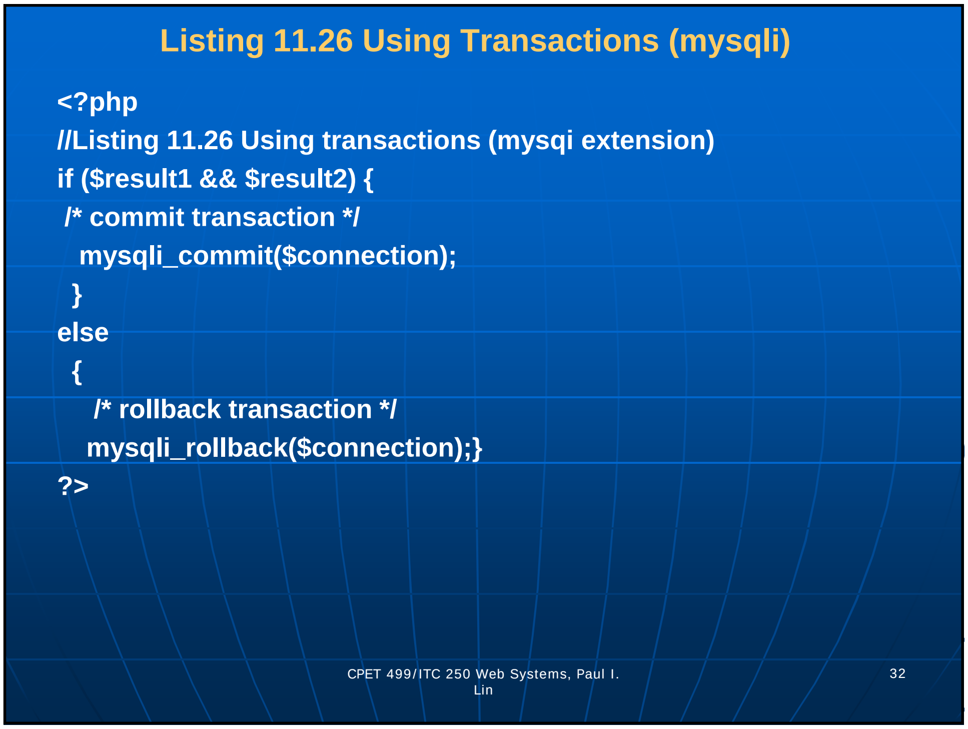
## Using Transactions

- **Transactions**
  - Unnecessary when retrieving database data
  - Should be used for most scenarios involving any database “writes”



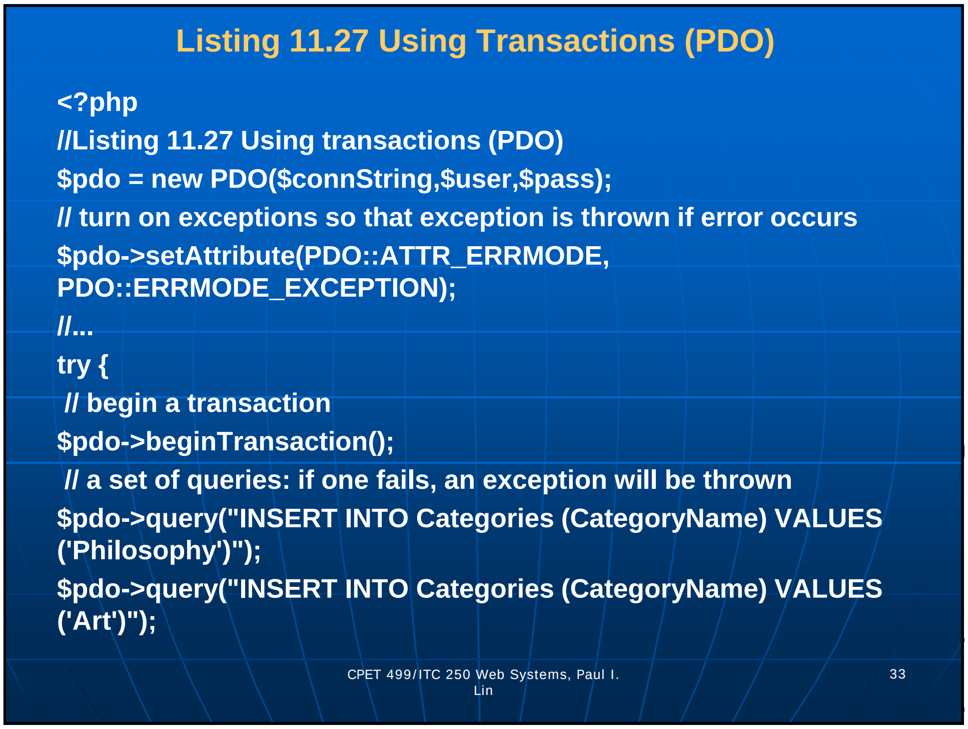
## Listing 11.26 Using Transactions (mysqli)

```
<?php
//Listing 11.26 Using transactions (mysqi extension)
$connection = mysqli_connect($host, $user, $pass, $database);
//...
/* set autocommit to off. If autocommit is on, then mysql
willcommit (i.e., make the data change permanent) each command
afterit is executed */
mysqli_autocommit($connection, FALSE);
/* insert some values */
$result1 = mysqli_query($connection,"INSERT INTO Categories
(CategoryName) VALUES ('Philosophy')");

$result2 = mysqli_query($connection,"INSERT INTO Categories
(CategoryName) VALUES ('Art')");
```



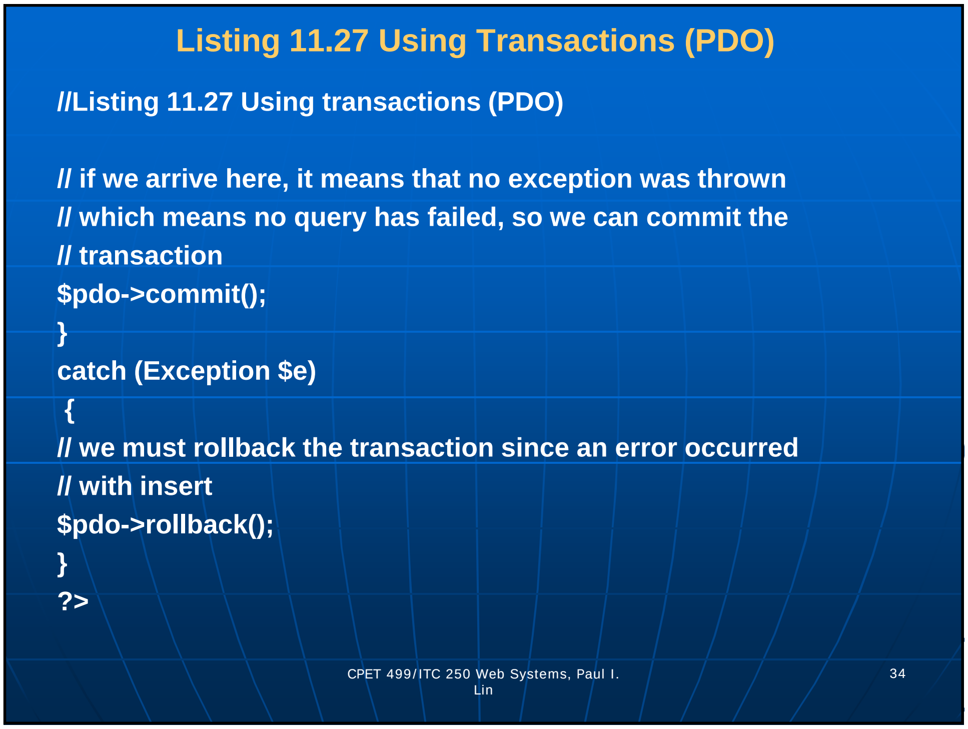
## Listing 11.26 Using Transactions (mysqli)

```
<?php
//Listing 11.26 Using transactions (mysqi extension)
if ($result1 && $result2) {
 /* commit transaction */
   mysqli_commit($connection);
  }
else
  {
     /* rollback transaction */
    mysqli_rollback($connection);}
?>
```



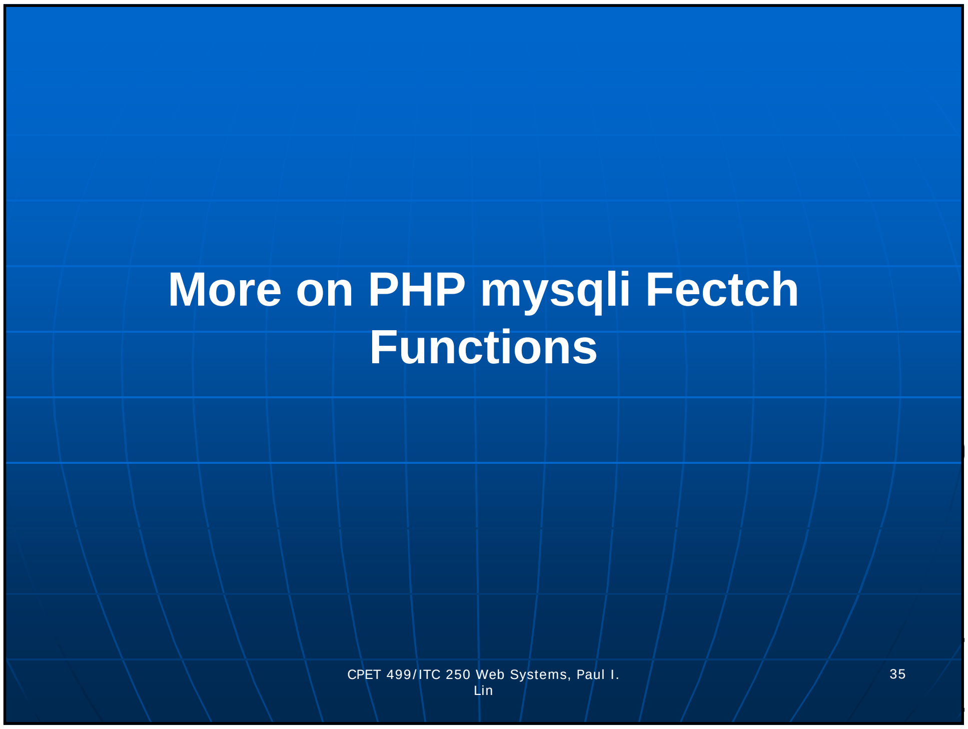
## Listing 11.27 Using Transactions (PDO)

```
<?php
//Listing 11.27 Using transactions (PDO)
$pdo = new PDO($connString,$user,$pass);
// turn on exceptions so that exception is thrown if error occurs
$pdo->setAttribute(PDO::ATTR_ERRMODE,
PDO::ERRMODE_EXCEPTION);
//...
try {
 // begin a transaction
$pdo->beginTransaction();
 // a set of queries: if one fails, an exception will be thrown
$pdo->query("INSERT INTO Categories (CategoryName) VALUES
('Philosophy')");
$pdo->query("INSERT INTO Categories (CategoryName) VALUES
('Art')");
```

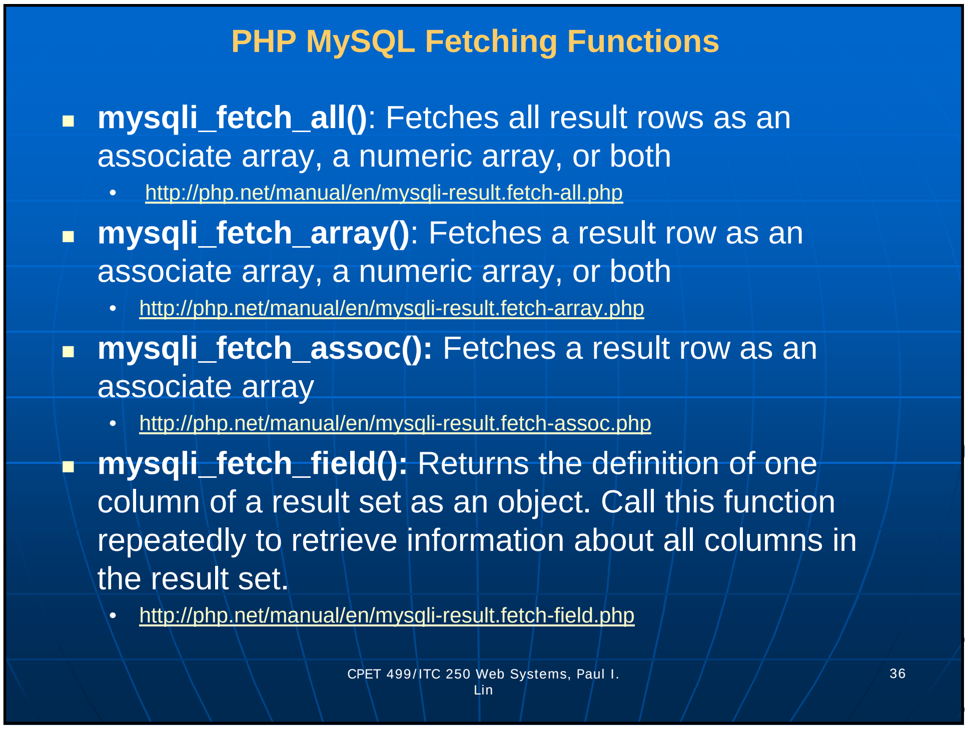


## Listing 11.27 Using Transactions (PDO)

```
//Listing 11.27 Using transactions (PDO)

// if we arrive here, it means that no exception was thrown
// which means no query has failed, so we can commit the
// transaction
$pdo->commit();
}
catch (Exception $e)
 {
// we must rollback the transaction since an error occurred
// with insert
$pdo->rollback();
}
?>
```

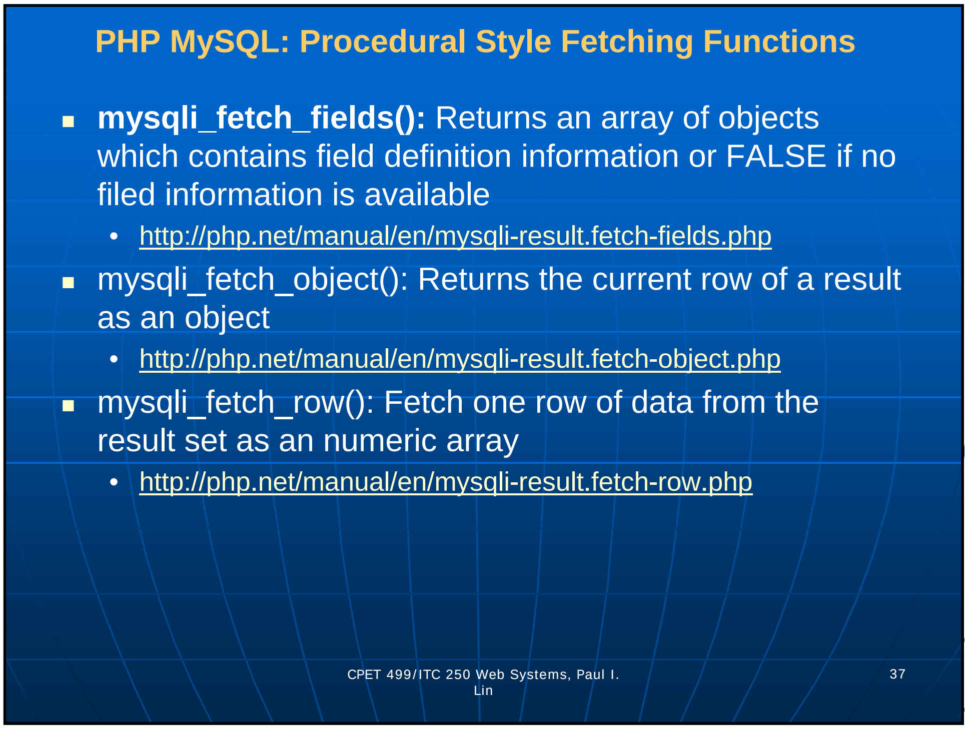


# More on PHP mysqli Fectch Functions



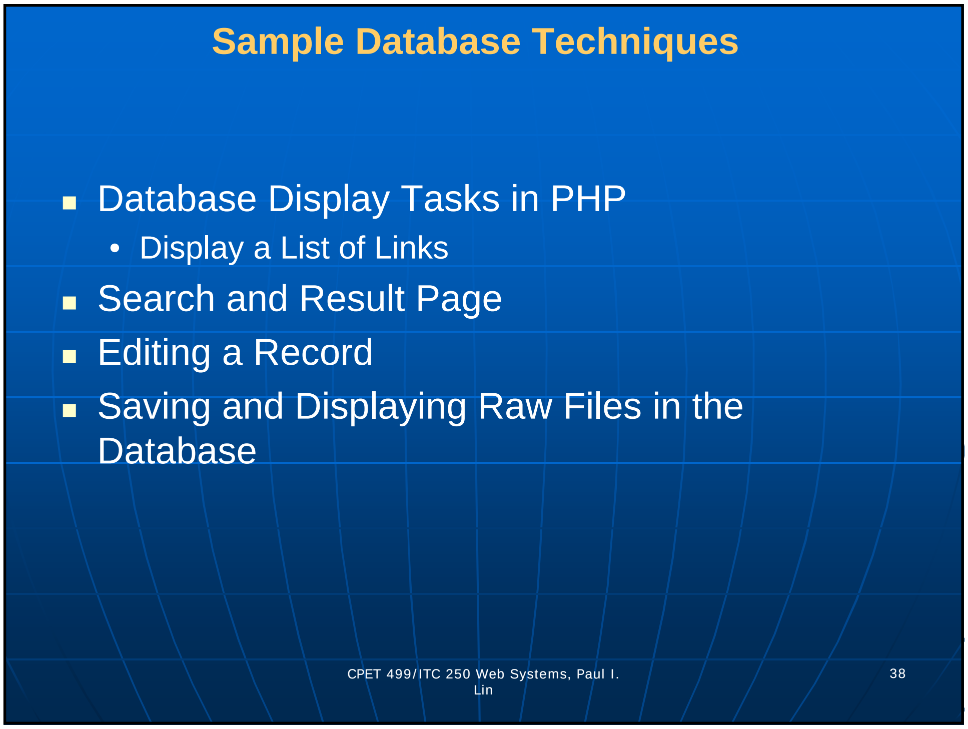
## PHP MySQL Fetching Functions

- **mysqli_fetch_all()**: Fetches all result rows as an associate array, a numeric array, or both
  - http://php.net/manual/en/mysqli-result.fetch-all.php
- **mysqli_fetch_array()**: Fetches a result row as an associate array, a numeric array, or both
  - http://php.net/manual/en/mysqli-result.fetch-array.php
- **mysqli_fetch_assoc():** Fetches a result row as an associate array
  - http://php.net/manual/en/mysqli-result.fetch-assoc.php
- **mysqli_fetch_field():** Returns the definition of one column of a result set as an object. Call this function repeatedly to retrieve information about all columns in the result set.
  - http://php.net/manual/en/mysqli-result.fetch-field.php



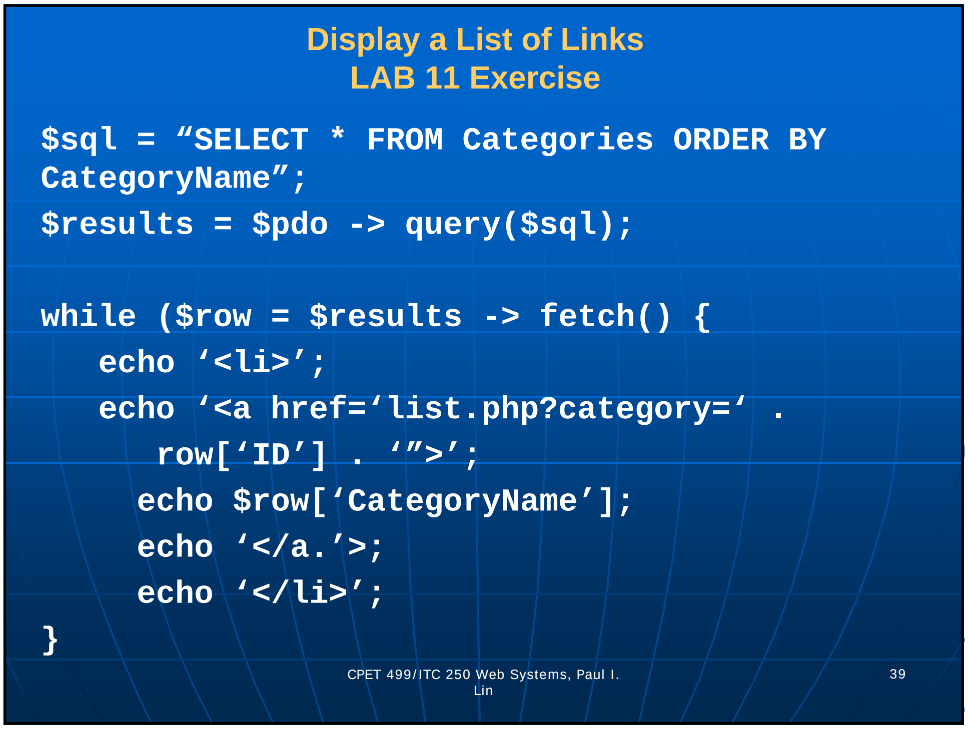
## PHP MySQL: Procedural Style Fetching Functions

- **mysqli_fetch_fields():** Returns an array of objects which contains field definition information or FALSE if no filed information is available
  - http://php.net/manual/en/mysqli-result.fetch-fields.php
- mysqli_fetch_object(): Returns the current row of a result as an object
  - http://php.net/manual/en/mysqli-result.fetch-object.php
- mysqli_fetch_row(): Fetch one row of data from the result set as an numeric array
  - http://php.net/manual/en/mysqli-result.fetch-row.php



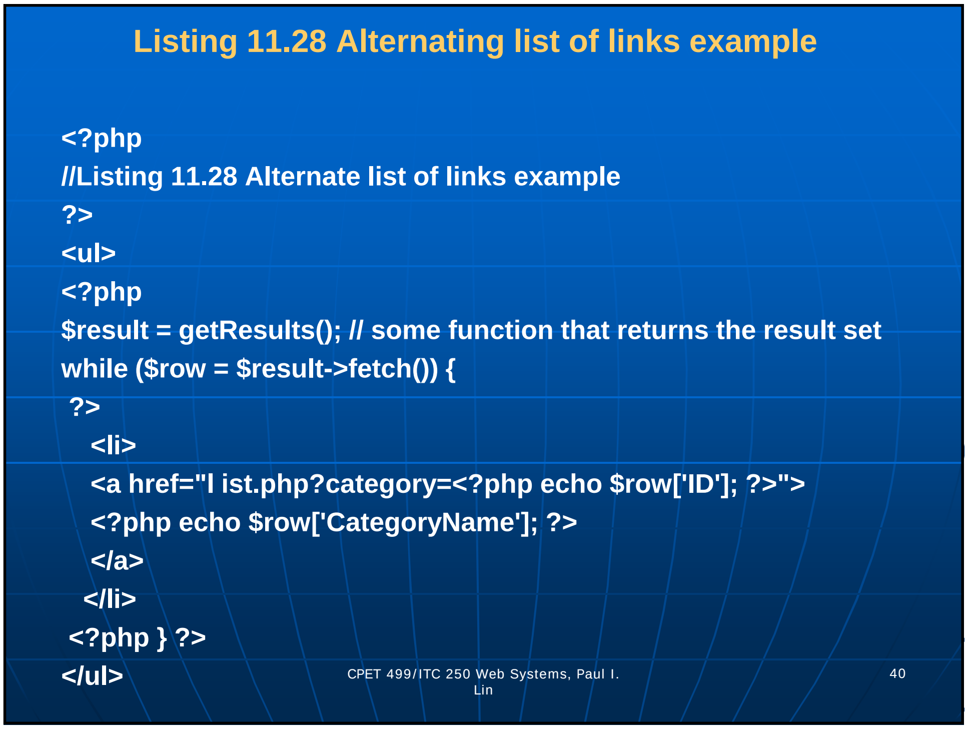
## Sample Database Techniques

- Database Display Tasks in PHP
  - Display a List of Links
- Search and Result Page
- Editing a Record
- Saving and Displaying Raw Files in the Database



## Display a List of Links
## LAB 11 Exercise

```
$sql = “SELECT * FROM Categories ORDER BY
CategoryName”;
$results = $pdo -> query($sql);

while ($row = $results -> fetch() {
   echo ‘<li>’;
   echo ‘<a href=‘list.php?category=‘ .
      row[‘ID’] . ‘”>’;
     echo $row[‘CategoryName’];
     echo ‘</a.’>;
     echo ‘</li>’;
}
```



## Listing 11.28 Alternating list of links example

```
<?php
//Listing 11.28 Alternate list of links example
?>
<ul>
<?php
$result = getResults(); // some function that returns the result set
while ($row = $result->fetch()) {
 ?>
   <li>
   <a href="l ist.php?category=<?php echo $row['ID']; ?>">
   <?php echo $row['CategoryName']; ?>
   </a>
  </li>
 <?php } ?>
</ul>
```

## Markup List Generation

```
<ul>
 <li><a href=“list.php?category=7”>Business</a></li>
 <li><a href=“list.php?category=2”>Computer Science</a></li>
 <li><a href=“list.php?category=3”>Economics</a></li>
 <li><a href=“list.php?category=9”>Engineering</a></li>
 <li><a href=“list.php?category=4”>English</a></li>
 <li><a href=“list.php?category=6”>Mathematics</a></li>
 <li><a href=“list.php?category=8”>Statistics</a></li>
 <li><a href=“list.php?category=5”>Student Success</a></li>
</ul>
```

## Q & A