PHP Data Object

PDO

Dbdavidson.php

- Create a php file, DB***<your last name>***.php and save OUTSIDE your public_html
  - Mine is DBdavidson.php
- Create 4 constants for Username, Password, Database Host, and Database Name
- ***Dbdavidson.php***

```
<?php
define("USERNAME", "<your webid>");
define("PASSWORD", "<your MySQL password>");
define ("DBHOST", "localhost");
define("DBNAME", "<your webid>");
?>
```

Replace < > entries with your account info. DO NOT include < >

PHP Data Objects (PDOs)

- Software abstraction layer that allows you to write database access code and easily change to a different underlying database without changing code.
  - **PDO is Database Neutral**
  - Only in the *connection* method
  - Works with MySQL, SQLite, MariaDB, PostgresSQL…
  - Available in PHP 5.1 and newer versions

PDO General Process

- The statement template is compiled, optimized and stored for later use.
  - NOT IMMEDIATELY EXECUTED
- Before statement executed
  - Bind actual value to parameter before executing
  - Same template can be reused without recompiling (just bind new value to execute)

PDO: Advantages

- Protects against SQL injection
- Reduces time a query takes to execute (optimize only once)
- Syntax easier to read – not a bunch of quoted parameters
- Only send parameters to server – query, itself, already stored server-side

1. Connect Using PDO

- Use Database Source Name (DSN) to connect
  - **DSN is Database Specific**
  - We will use MySQL DSN

```
<?php
    try {
            require_once('/home/kdavidso/DBdavidson.php');
            $mysqli = new PDO('mysql:host='.DBHOST.';dbname='.DBNAME,
USERNAME, PASSWORD);
    }
    catch (PDOException $e)  {
            echo "Error!: ". $e->getMessage()."<br />";
            die ("Could not connect to server ".DBNAME."<br />");
    }
.... ?>
```

Create a *Database* class

```
<?php
class Database {
   private static $mysqli = null;

   public static function dbConnect() {
      require_once("/home/kdavidso/DBdavidson.php");

      if($mysqli == null) {
         try {
            $mysqli = new PDO('mysql:host='.DBHOST.';dbname='.DBNAME,
USERNAME, PASSWORD);
            echo "Successful Connection";
         }
         catch(PDOException $e) {
            echo "Could not connect";
            die($e->getMessage());
         }
      }
      return $mysqli;
   }
}
?>
```

1. Connect Using PDO

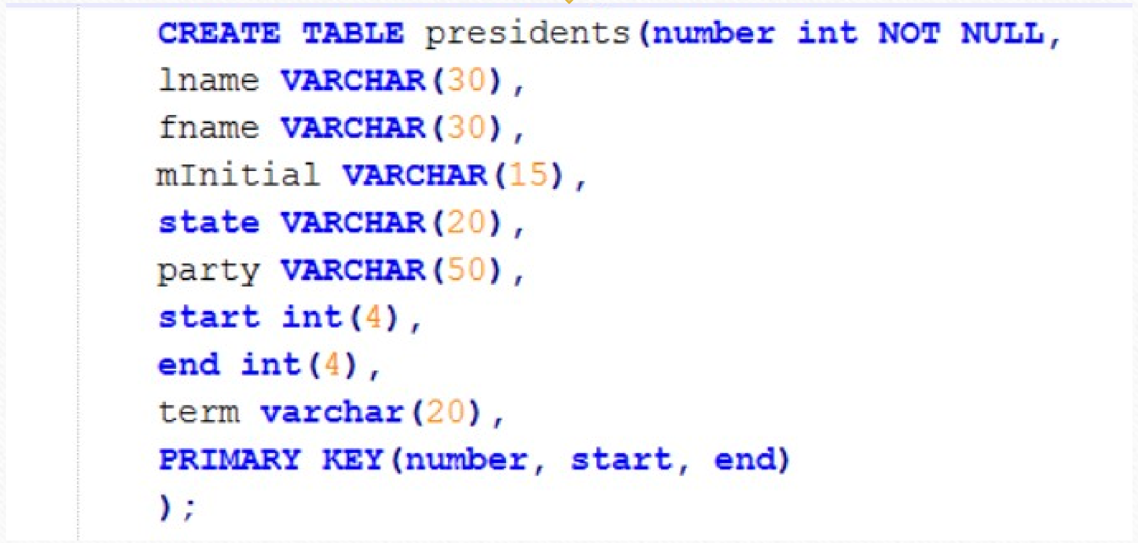
```
<?php require_once("session.php");
require_once("included_functions.php");
require_once("database.php");
      new_header("Choose Your President");
       $mysqli = Database::dbConnect();
       $mysqli ->
setAttribute(PDO::ATTR_ERRMODE,

PDO::ERRMODE_EXCEP[illegible]ON);
?>
```

Double-Colon or Scope Resolution Operator is used to reference constants or static methods of a class.
ClassName::static/constant

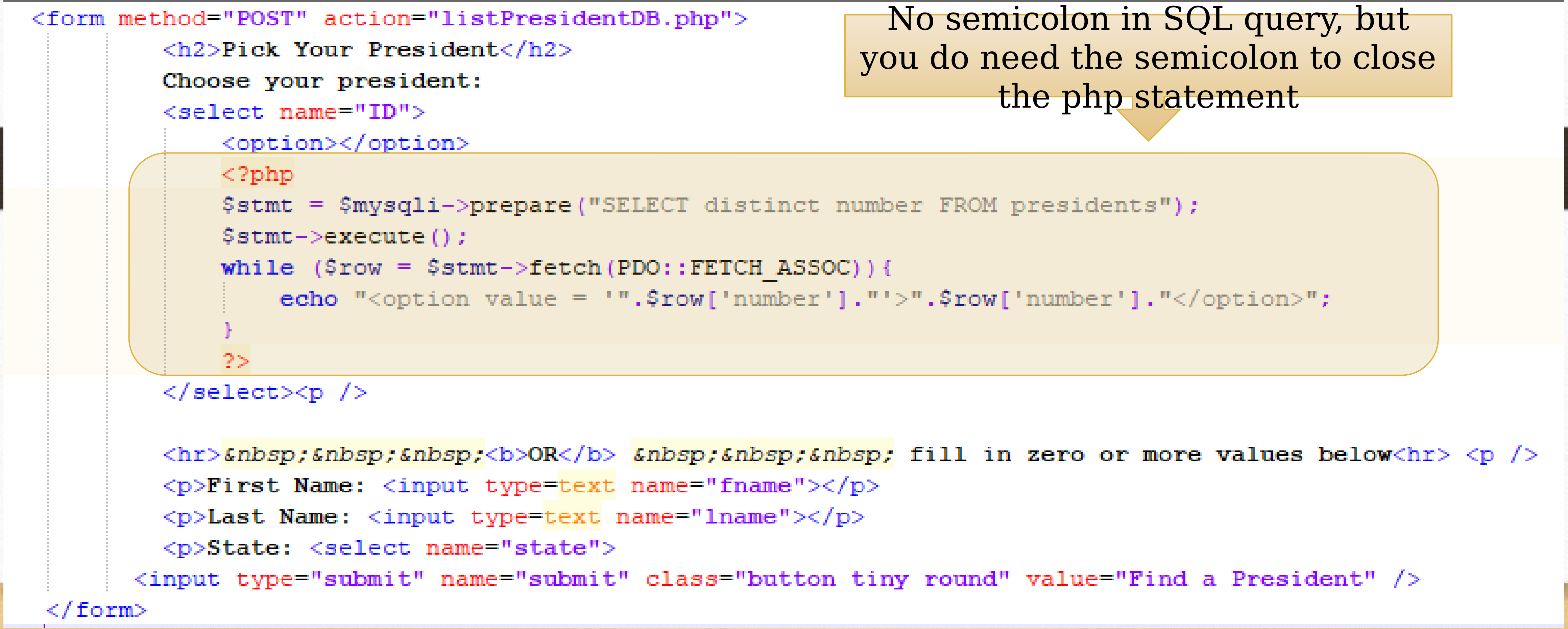
Presidents DB

presidents SQL



```
CREATE TABLE presidents(number int NOT NULL,
lname VARCHAR(30),
fname VARCHAR(30),
mInitial VARCHAR(15),
state VARCHAR(20),
party VARCHAR(50),
start int(4),
end int(4),
term varchar(20),
PRIMARY KEY(number, start, end)
);
```

2. Executing a Query (choosePresident.php)



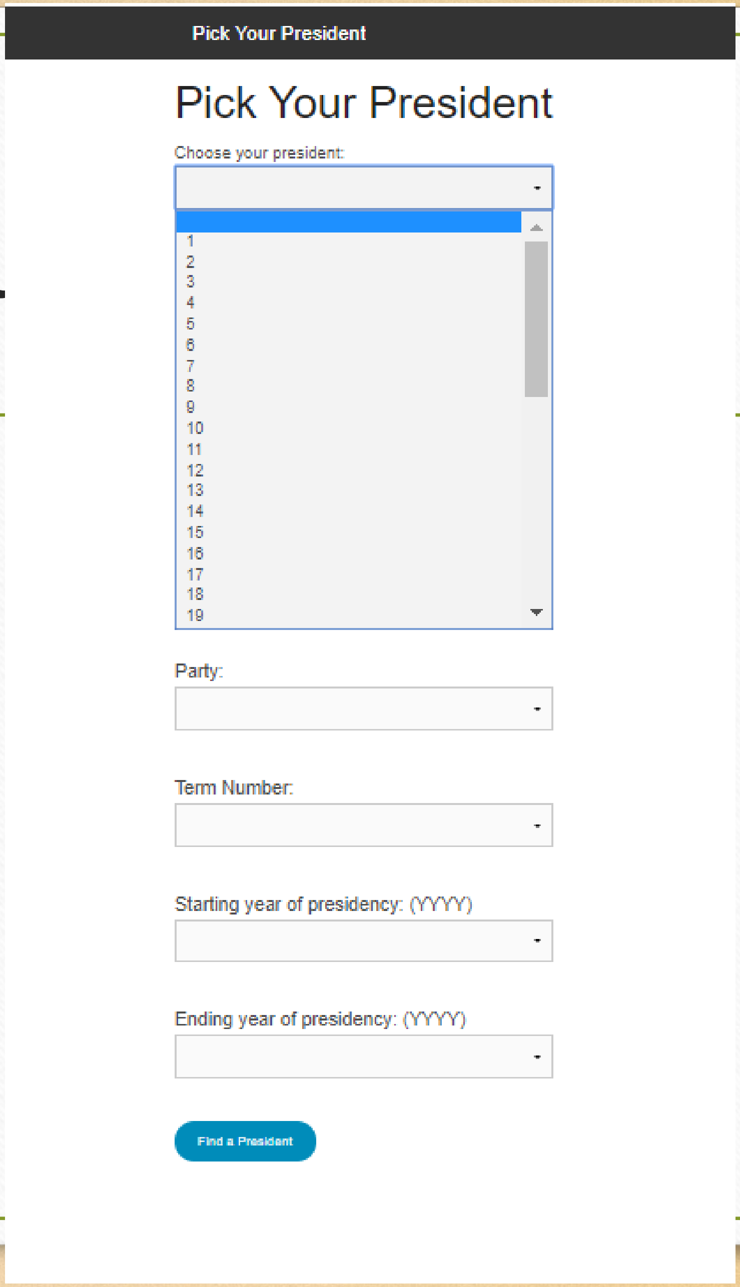
```
<form method="POST" action="listPresidentDB.php">
        <h2>Pick Your President</h2>
        Choose your president:
        <select name="ID">
            <option></option>
            <?php
            $stmt = $mysqli->prepare("SELECT distinct number FROM presidents");
            $stmt->execute();
            while ($row = $stmt->fetch(PDO::FETCH_ASSOC)){
                echo "<option value = '".$row['number']."'>".$row['number']."</option>";
            }
            ?>
        </select><p />

        <hr>&nbsp;&nbsp;&nbsp;<b>OR</b> &nbsp;&nbsp;&nbsp; fill in zero or more values below<hr> <p />
        <p>First Name: <input type=text name="fname"></p>
        <p>Last Name: <input type=text name="lname"></p>
        <p>State: <select name="state">
      <input type="submit" name="submit" class="button tiny round" value="Find a President" />
  </form>
```

No semicolon in SQL query, but you do need the semicolon to close the php statement

2. Executing a Qu

choosePresdient.php



Pick Your President

Pick Your President

Choose your president:

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19

Party:

Term Number:

Starting year of presidency: (YYYY)

Ending year of presidency: (YYYY)

2. Executing a Query (Numeric)

```
        $query = "SELECT * ";
        $query .= "FROM presidents ";
        $query .= "WHERE number = ?";
        $stmt = $mysqli -> prepare($query);
    $ID = $_POST['ID'];
        $stmt -> execute([$ID]);
```

This MUST be an array

NOTE:

? Designates an ANONYMOUS or UNNAMED parameter

Only strings and numerics are bound.

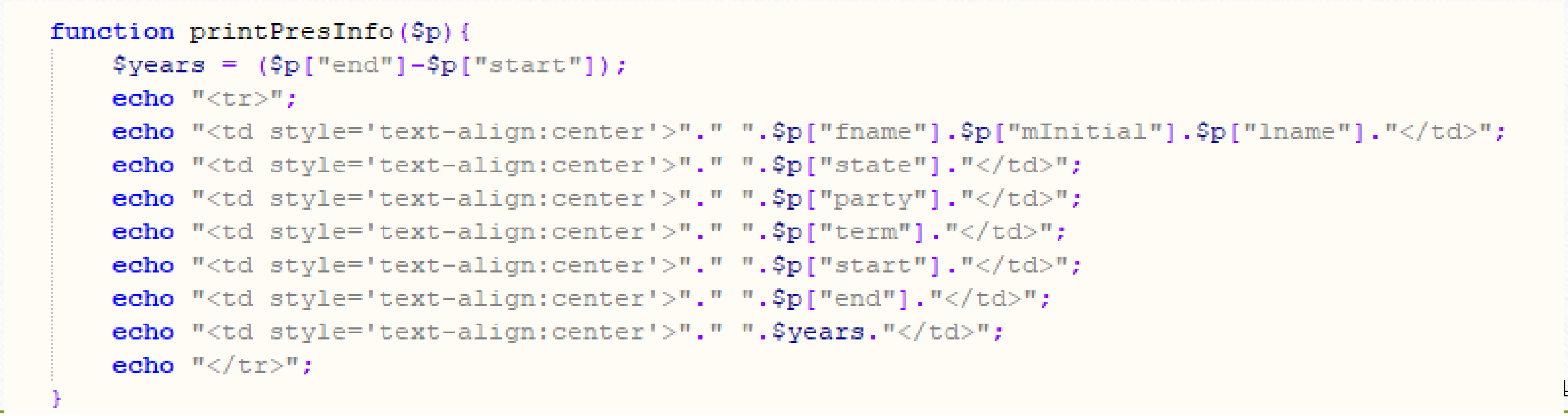
2. Executing a Query (String)

```
$name = $_POST['name']'

$query = "SELECT * ";
$query .= "FROM Country ";
$query .= "WHERE Name Like ? ";
$stmt = $mysqli -> 
prepare($query);
$stmt -> execute([“%UN%”]);
```

3. Processing Results (Presidents)

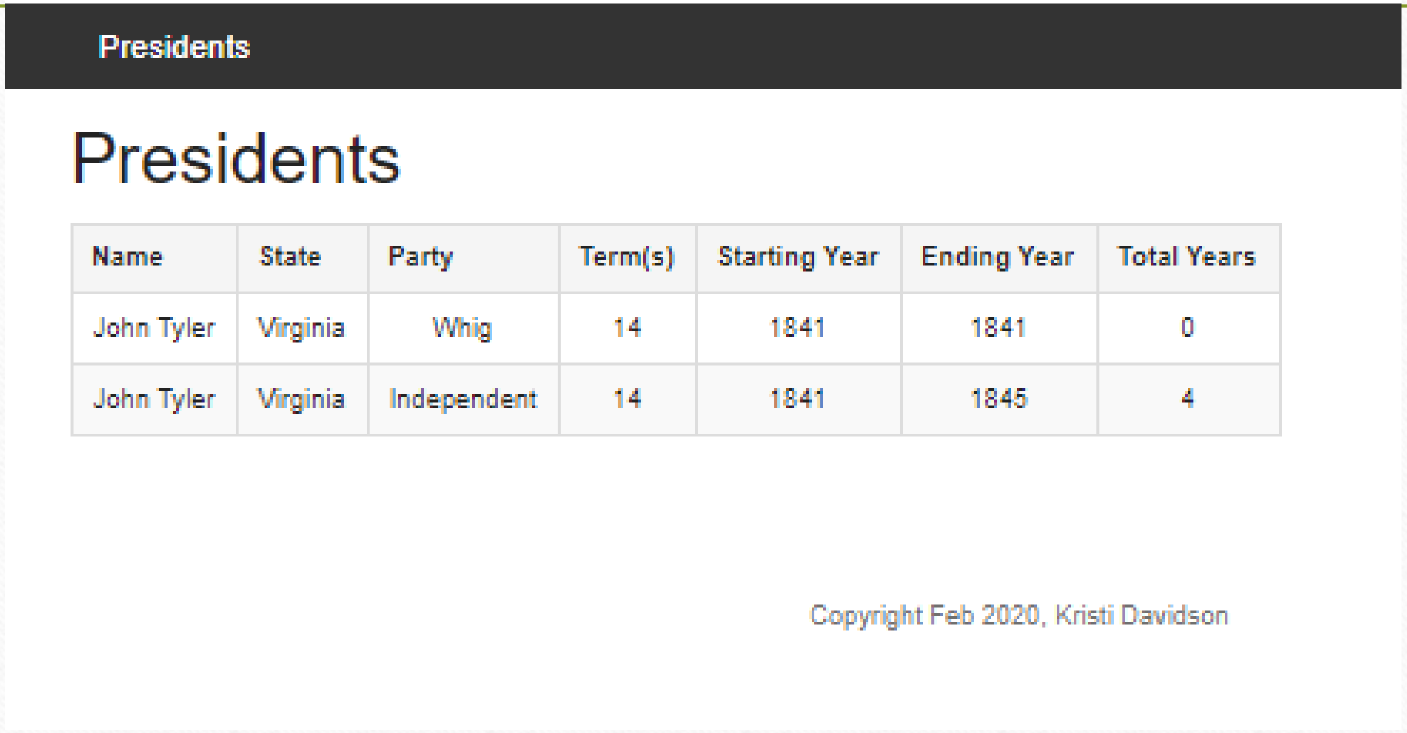
```
while($row = $stmt->fetch(PDO::FETCH_ASSOC)) {
    //Call a function instead
        printPresident($row);
```



```
function printPresInfo($p){
    $years = ($p["end"]-$p["start"]);
    echo "<tr>";
    echo "<td style='text-align:center'>"." ".$p["fname"].$p["mInitial"].$p["lname"]."</td>";
    echo "<td style='text-align:center'>"." ".$p["state"]."</td>";
    echo "<td style='text-align:center'>"." ".$p["party"]."</td>";
    echo "<td style='text-align:center'>"." ".$p["term"]."</td>";
    echo "<td style='text-align:center'>"." ".$p["start"]."</td>";
    echo "<td style='text-align:center'>"." ".$p["end"]."</td>";
    echo "<td style='text-align:center'>"." ".$years."</td>";
    echo "</tr>";
}
```

3. Processing Results (listPresident.php)

$ID = 14


Presidents

Presidents

Name	State	Party	Term(s)	Starting Year	Ending Year	Total Years
John Tyler	Virginia	Whig	14	1841	1841	0
John Tyler	Virginia	Independent	14	1841	1845	4

Copyright Feb 2020, Kristi Davidson

3. Processing Results (Countries)

```
                while($row = $stmt->fetch(PDO::FETCH_ASSOC))
{
                        echo "<tr>";
                        echo "<td style='text-align:center'>".
$row['Name']."</td>";
                        echo "</tr>";
            }
```

4. Releasing Results

$stmt -> close();

AUTOMATIC so we will NOT include
(throws an error on CLI if you do)

5. Closing Connection

Database::dbDisconnect();

```
<?php
class Database {
	...
	public static function dbDisconnect() {
		$mysqli = null;
	}
}
?>
```

PDO: Named Parameters

- Use ***:name*** rather than ***?*** to designate parameters
- Order doesn't matter since *key* is used to reference *value*

```
$query = "SELECT * ";
$query .= "FROM presidents ";
$query .= "WHERE number = :num";
$stmt = $mysqli -> prepare($query);
$params = [“:num” => $_POST[‘ID’]];
$stmt -> execute($params);
```

The colon (:) is REQUIRED here. Tells PHP this is a placeholder

The colon (:) is OPTIONAL here. In executing and binding the : is inferred as a placeholder

PDO: Named Parameters

```
$query = "SELECT Name, Capital,
Continent ";
$query .= "FROM Country ";
$query .= "WHERE Name LIKE :name";
$stmt = $mysqli->prepare($query);
$params = array( “:name” => “%UN
%”);
$stmt -> execute($params);
```