6.2 Технології MySQLi і PDO / MySQLi and PDO technologies

PHP 5 і наступні версії можуть працювати з СУБД MySQL за допомогою

PHP 5 and later can work with a MySQL database using

- **MySQLi**
- **PDO** (PHP Data Objects)

У більш ранніх версіях PHP використовується розширення **MySQL**, визнане застарілим з 2012 року

Earlier versions of PHP used the MySQL extension. However, this extension was deprecated in 2012

Що слід використовувати – MySQLi або PDO?

Should I use MySQLi or PDO?

6.2 Технології MySQLi і PDO / MySQLi and PDO technologies

PDO буде працювати в 12 різних системах баз даних, тоді як MySQLi буде працювати тільки з базами даних MySQL

PDO will work on 12 different database systems, whereas MySQLi will only work with MySQL databases

Таким чином, якщо вам потрібно переключити свій проект на використання іншої бази даних, PDO спростить цей процес. Вам потрібно лише змінити рядок підключення і кілька запитів. З MySQLi вам потрібно буде переписати весь код – включаючи запити.

So, if you have to switch your project to use another database, PDO makes the process easy. You only have to change the connection string and a few queries. With MySQLi, you will need to rewrite the entire code - queries included.

6.2 Технології MySQLi і PDO / MySQLi and PDO technologies

Обидва є об'єктно-орієнтованими, але MySQLi також підтримує процедурний API

Both are object-oriented, but MySQLi also offers a procedural API

Обидва підтримують підготовлені вирази (запити)

Both support prepared statements

Підготовлені оператори захищають від SQL-ін'єкцій і дуже важливі для безпеки веб-додатків.

Prepared Statements protect from SQL injection, and are very important for web application security.

6.2 Технології MySQLi і PDO / MySQLi and PDO technologies

З’єднання з MySQL / Open a connection to MySQL

MySQLi Object-oriented

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";

// Create connection
$conn = new mysqli($servername, $username, $password);

// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
echo "Connected successfully";
?>
```

6.2 Технології MySQLi і PDO / MySQLi and PDO technologies

З'єднання з MySQL / Open a connection to MySQL

MySQLi Procedural

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";

// Create connection
$conn = mysqli_connect($servername, $username, $password);

// Check connection
if (!$conn) {
    die("Connection failed: " . mysqli_connect_error());
}
echo "Connected successfully";
?>
```

6.2 Технології MySQLi і PDO / MySQLi and PDO technologies

З'єднання з MySQL / Open a connection to MySQL

PDO

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";

try {
    $conn = new PDO("mysql:host=$servername;dbname=myDB", $username, $password);
    // set the PDO error mode to exception
    $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
    echo "Connected successfully";
    }
catch(PDOException $e)
    {
    echo "Connection failed: " . $e->getMessage();
    }
?>
```

6.2 Технології MySQLi і PDO / MySQLi and PDO technologies

Закриття з'єднання / Close the connection

MySQLi Object-Oriented

```
$conn->close();
```

MySQLi Procedural

```
mysqli_close($conn);
```

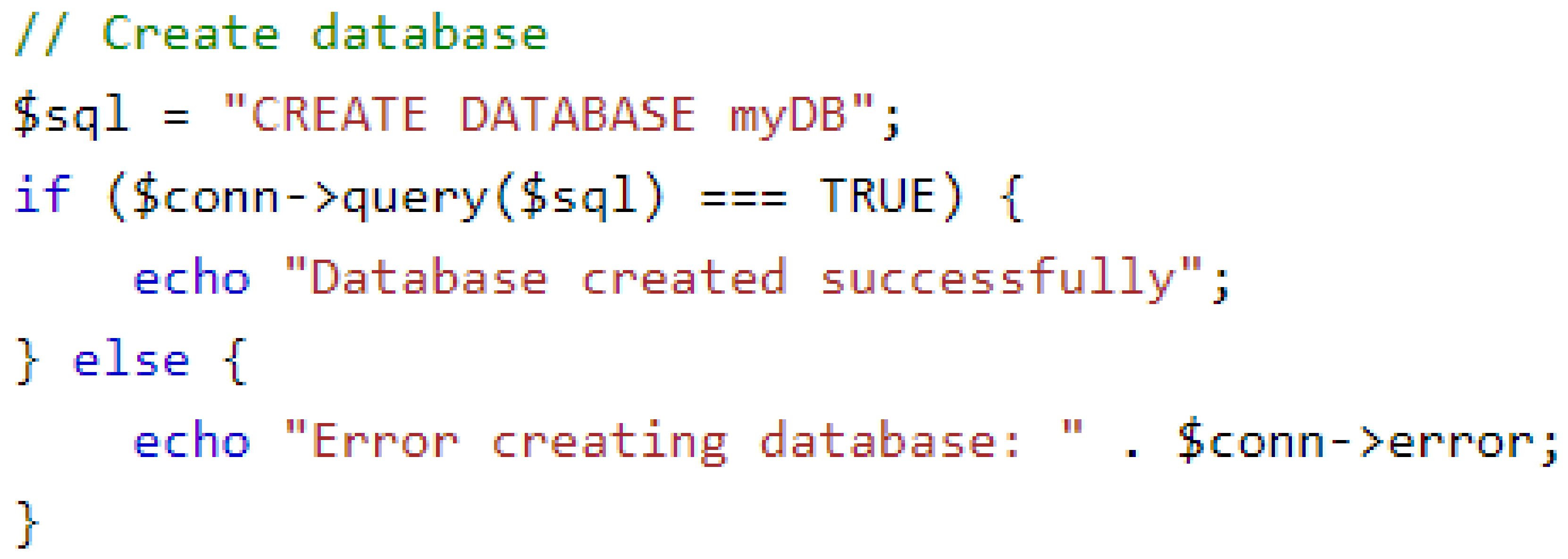
PDO

```
$conn = null;
```

6.2 Технології MySQLi і PDO / MySQLi and PDO technologies

Створення бази даних / Create a database

MySQLi Object-Oriented

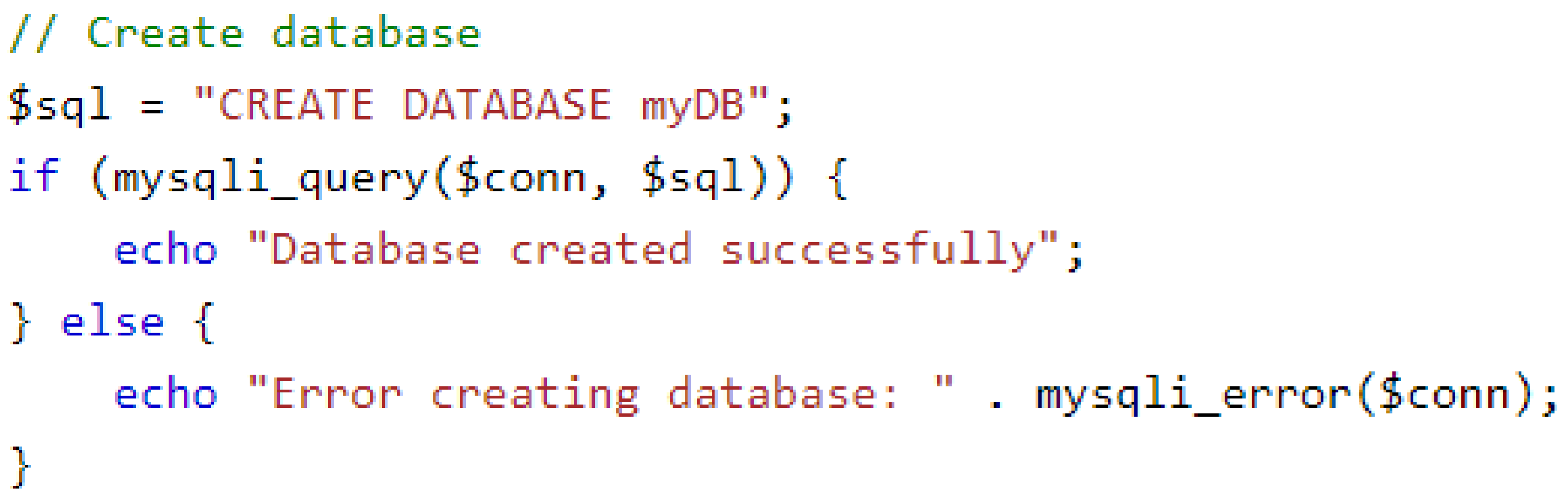


```
// Create database
$sql = "CREATE DATABASE myDB";
if ($conn->query($sql) === TRUE) {
    echo "Database created successfully";
} else {
    echo "Error creating database: " . $conn->error;
}
```

6.2 Технології MySQLi і PDO / MySQLi and PDO technologies

Створення бази даних / Create a database

MySQLi Procedural

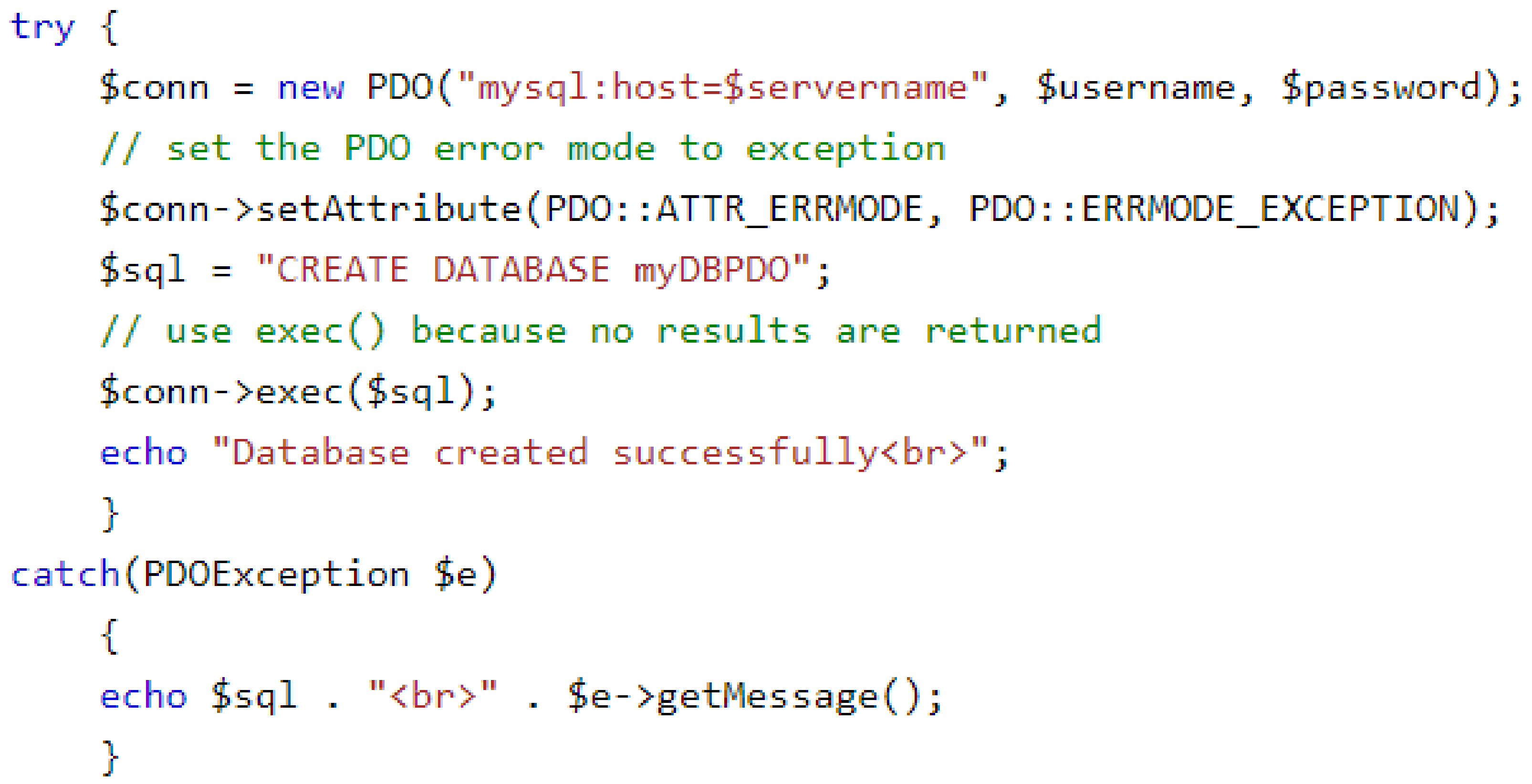


```php
// Create database
$sql = "CREATE DATABASE myDB";
if (mysqli_query($conn, $sql)) {
    echo "Database created successfully";
} else {
    echo "Error creating database: " . mysqli_error($conn);
}
```

6.2 Технології MySQLi і PDO / MySQLi and PDO technologies

Створення бази даних / Create a database

PDO



```
try {
    $conn = new PDO("mysql:host=$servername", $username, $password);
    // set the PDO error mode to exception
    $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
    $sql = "CREATE DATABASE myDBPDO";
    // use exec() because no results are returned
    $conn->exec($sql);
    echo "Database created successfully<br>";
    }
catch(PDOException $e)
    {
    echo $sql . "<br>" . $e->getMessage();
    }
```

6.2 Технології MySQLi і PDO / MySQLi and PDO technologies

Створення таблиці / Create a table

MySQLi Object-Oriented

```
// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

// sql to create table
$sql = "CREATE TABLE MyGuests (
id INT(6) UNSIGNED AUTO_INCREMENT PRIMARY KEY,
firstname VARCHAR(30) NOT NULL,
lastname VARCHAR(30) NOT NULL,
email VARCHAR(50),
reg_date TIMESTAMP
)";
```

```
if ($conn->query($sql) === TRUE) {
    echo "Table MyGuests created successfully";
} else {
    echo "Error creating table: " . $conn->error;
}
```

6.2 Технології MySQLi і PDO / MySQLi and PDO technologies

Створення таблиці / Create a table

MySQLi Procedural

```
// Create connection
$conn = mysqli_connect($servername, $username, $password, $dbname);
// Check connection
if (!$conn) {
    die("Connection failed: " . mysqli_connect_error());
}

// sql to create table
$sql = "CREATE TABLE MyGuests (
id INT(6) UNSIGNED AUTO_INCREMENT PRIMARY KEY,
firstname VARCHAR(30) NOT NULL,
lastname VARCHAR(30) NOT NULL,
email VARCHAR(50),
reg_date TIMESTAMP
)";
```

```
if (mysqli_query($conn, $sql)) {
    echo "Table MyGuests created successfully";
} else {
    echo "Error creating table: " . mysqli_error($conn);
}
```

6.2 Технології MySQLi і PDO / MySQLi and PDO technologies

Створення таблиці / Create a table

PDO

```
$conn = new PDO("mysql:host=$servername;dbname=$dbname", $username, $password);
// set the PDO error mode to exception
$conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);

// sql to create table
$sql = "CREATE TABLE MyGuests (
id INT(6) UNSIGNED AUTO_INCREMENT PRIMARY KEY,
firstname VARCHAR(30) NOT NULL,
lastname VARCHAR(30) NOT NULL,
email VARCHAR(50),
reg_date TIMESTAMP
)";

// use exec() because no results are returned
$conn->exec($sql);
echo "Table MyGuests created successfully";
```

6.2 Технології MySQLi і PDO / MySQLi and PDO technologies

INSERT

MySQLi Object-Oriented

```
$sql = "INSERT INTO MyGuests (firstname, lastname, email)
VALUES ('John', 'Doe', 'john@example.com')";

if ($conn->query($sql) === TRUE) {
    echo "New record created successfully";
} else {
    echo "Error: " . $sql . "<br>" . $conn->error;
}
```

MySQLi Procedural

```
$sql = "INSERT INTO MyGuests (firstname, lastname, email)
VALUES ('John', 'Doe', 'john@example.com')";

if (mysqli_query($conn, $sql)) {
    echo "New record created successfully";
} else {
    echo "Error: " . $sql . "<br>" . mysqli_error($conn);
}
```

PDO

```
$sql = "INSERT INTO MyGuests (firstname, lastname, email)
VALUES ('John', 'Doe', 'john@example.com')";
// use exec() because no results are returned
$conn->exec($sql);
echo "New record created successfully";
```

6.2 Технології MySQLi і PDO / MySQLi and PDO technologies

DELETE

MySQLi Object-Oriented

```
// sql to delete a record
$sql = "DELETE FROM MyGuests WHERE id=3";

if ($conn->query($sql) === TRUE) {
    echo "Record deleted successfully";
} else {
    echo "Error deleting record: " . $conn->error;
}
```

MySQLi Procedural

```
// sql to delete a record
$sql = "DELETE FROM MyGuests WHERE id=3";

if (mysqli_query($conn, $sql)) {
    echo "Record deleted successfully";
} else {
    echo "Error deleting record: " . mysqli_error($conn);
}
```

PDO

```
// sql to delete a record
$sql = "DELETE FROM MyGuests WHERE id=3";

// use exec() because no results are returned
$conn->exec($sql);
echo "Record deleted successfully";
```

6.2 Технології MySQLi і PDO / MySQLi and PDO technologies

UPDATE

MySQLi Object-Oriented

```
$sql = "UPDATE MyGuests SET lastname='Doe' WHERE id=2";

if ($conn->query($sql) === TRUE) {
    echo "Record updated successfully";
} else {
    echo "Error updating record: " . $conn->error;
}
```

MySQLi Procedural

```
$sql = "UPDATE MyGuests SET lastname='Doe' WHERE id=2";

if (mysqli_query($conn, $sql)) {
    echo "Record updated successfully";
} else {
    echo "Error updating record: " . mysqli_error($conn);
}
```

PDO

```
$sql = "UPDATE MyGuests SET lastname='Doe' WHERE id=2";

// Prepare statement
$stmt = $conn->prepare($sql);

// execute the query
$stmt->execute();

// echo a message to say the UPDATE succeeded
echo $stmt->rowCount() . " records UPDATED successfully";
```

6.2 Технології MySQLi і PDO / MySQLi and PDO technologies

SELECT

MySQLi Object-Oriented

```
$sql = "SELECT id, firstname, lastname FROM MyGuests";
$result = $conn->query($sql);

if ($result->num_rows > 0) {
    echo "<table><tr><th>ID</th><th>Name</th></tr>";
    // output data of each row
    while($row = $result->fetch_assoc()) {
        echo "<tr><td>".$row["id"]."</td><td>".$row["firstname"]." ".$row["lastname"]."</td>
</tr>";
    }
    echo "</table>";
} else {
    echo "0 results";
}
```

6.2 Технології MySQLi і PDO / MySQLi and PDO technologies

SELECT

MySQLi Procedural

```
$sql = "SELECT id, firstname, lastname FROM MyGuests";
$result = mysqli_query($conn, $sql);

if (mysqli_num_rows($result) > 0) {
    // output data of each row
    while($row = mysqli_fetch_assoc($result)) {
        echo "id: " . $row["id"]. " - Name: " . $row["firstname"]. " " . $row["lastname"]. "
<br>";
    }
} else {
    echo "0 results";
}
```

6.2 Технології MySQLi і PDO / MySQLi and PDO technologies

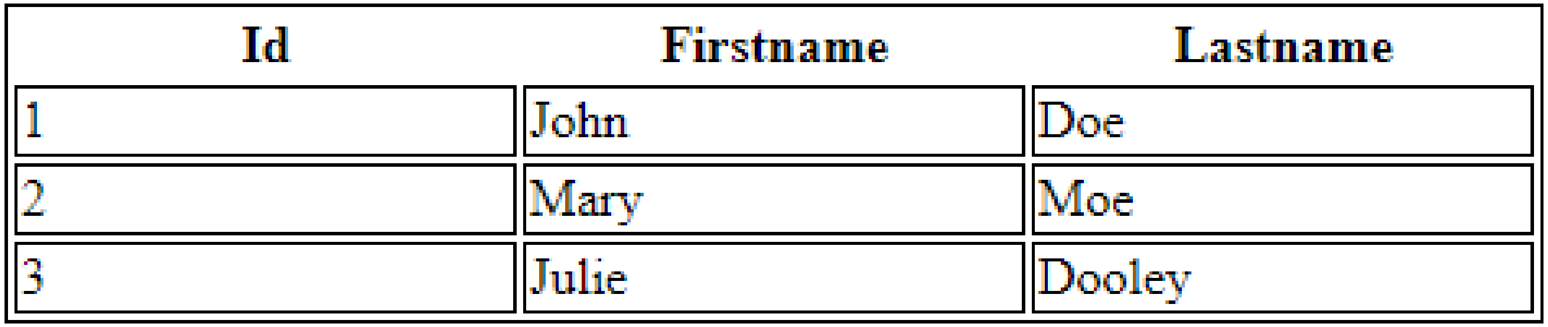
SELECT

PDO

```
echo "<table style='border: solid 1px black;'>";
echo "<tr><th>Id</th><th>Firstname</th><th>Lastname</th></tr>";

class TableRows extends RecursiveIteratorIterator {
    function __construct($it) {
        parent::__construct($it, self::LEAVES_ONLY);
    }

    function current() {
        return "<td style='width:150px;border:1px solid black;'>" . parent::current(). "</td>";
    }

    function beginChildren() {
        echo "<tr>";
    }

    function endChildren() {
        echo "</tr>" . "\n";
    }
}
```

6.2 Технології MySQLi і PDO / MySQLi and PDO technologies

```
$stmt = $conn->prepare("SELECT id, firstname, lastname FROM MyGuests");
$stmt->execute();

// set the resulting array to associative
$result = $stmt->setFetchMode(PDO::FETCH_ASSOC);
foreach(new TableRows(new RecursiveArrayIterator($stmt->fetchAll())) as $k=>$v) {
    echo $v;
}
```



Id	Firstname	Lastname
1	John	Doe
2	Mary	Moe
3	Julie	Dooley

6.2 Технології MySQLi і PDO / MySQLi and PDO technologies

Підготовлені запити / Prepared statements

MySQLi

```
// prepare and bind
$stmt = $conn->prepare("INSERT INTO MyGuests (firstname, lastname, email) VALUES (?, ?, ?)");
$stmt->bind_param("sss", $firstname, $lastname, $email);

// set parameters and execute
$firstname = "John";
$lastname = "Doe";
$email = "john@example.com";
$stmt->execute();
```

Параметри, які заміняються (?) / Substituted parameters (?)

- i – Integer
- d – Double
- s – String
- b – BLOB

6.2 Технології MySQLi і PDO / MySQLi and PDO technologies

Підготовлені запити / Prepared statements

PDO

```
// prepare sql and bind parameters
$stmt = $conn->prepare("INSERT INTO MyGuests (firstname, lastname, email)
VALUES (:firstname, :lastname, :email)");
$stmt->bindParam(':firstname', $firstname);
$stmt->bindParam(':lastname', $lastname);
$stmt->bindParam(':email', $email);

// insert a row
$firstname = "John";
$lastname = "Doe";
$email = "john@example.com";
$stmt->execute();
```