PHP and MySQL
Insert

Intro PHP

Download sample code from Nexus

The Insert Form

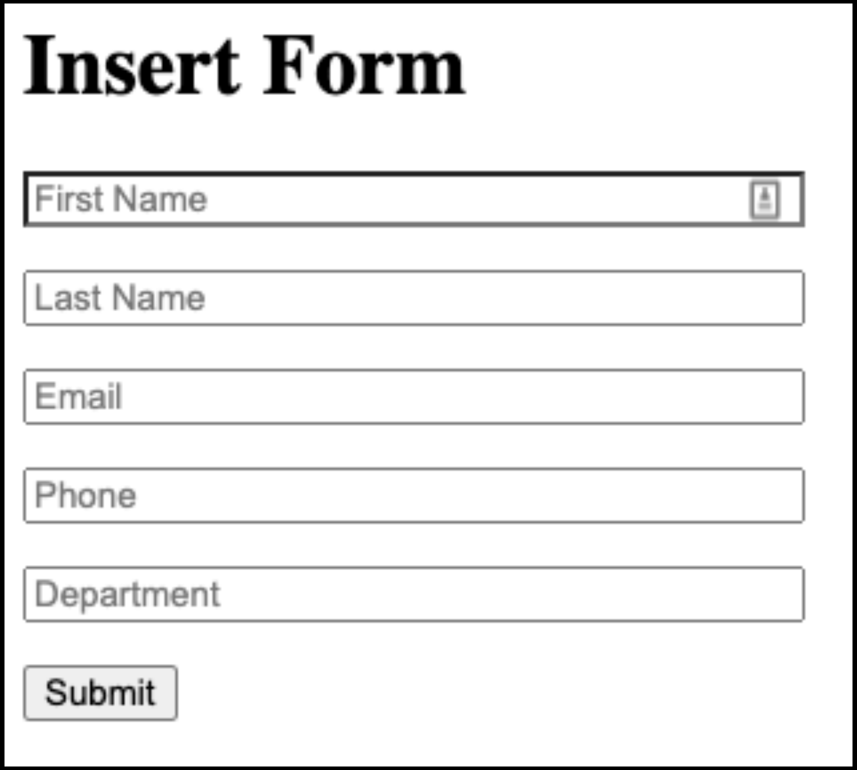
The first step in the process of inserting a new record into the database, is to provide a form for the user.

```
<form method="post">

<p><input type="text" name="first_name" size="40" placeholder="First Name"></p>

<p><input type="text" name="last_name" size="40" placeholder="Last Name"></p>

<p><input type="text" name="email" size="40" placeholder="Email"></p>

<p><input type="text" name="phone" size="40" placeholder="Phone"></p>

<p><input type="text" name="department" size="40" placeholder="Department">

<p><input type="submit" name="add_employee"></p>

</form>
```

01_insert_form.php

The Insert Form

The first step in the process of inserting a new record into the database, is to provide a form for the user.

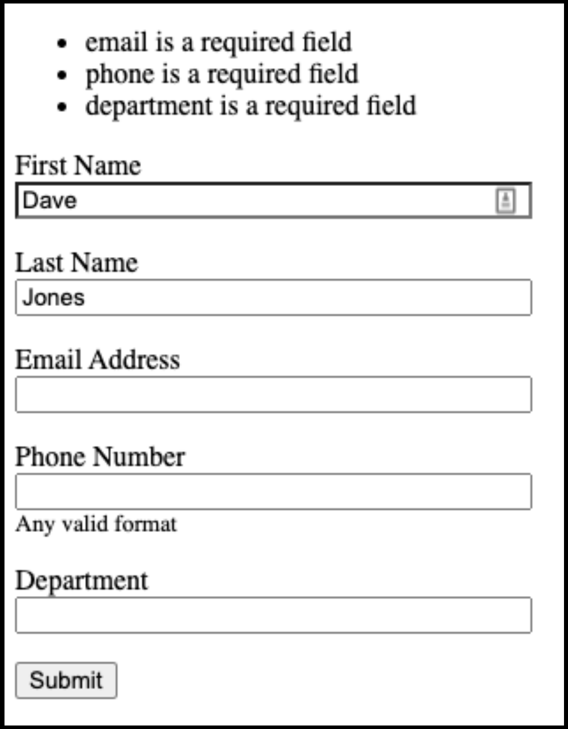


When creating a form when the desired end result is to insert or update data in a database, it is a best practice to ensure that the field names used in the form are exactly the same as the field names used in the database.

01_insert_form.php

Validating the form

When the form is submitted, we must validate all the fields to ensure they meet our needs



- email is a required field
- phone is a required field
- department is a required field

First Name: Dave

Last Name: Jones

Email Address:

Phone Number:

Any valid format

Department:

In this case, we are only validating for required fields. We want to ensure the user provides a value for each.

If a field is left empty, the form will reload and present the errors above the form

02_validate.php

Validating the form

When the form is submitted, we must validate all the fields to ensure they meet our needs

```
$errors = [];

if(isset($_POST['add_employee'])) {

  foreach($_POST as $key => $value) {
    $_POST[$key] = trim($value);
  }

  foreach($_POST as $key => $value) {
    if(empty($_POST[$key])) {
      $errors[$key] = "$key is required.";
    }
  }


} // end if $_POST
```

In this case, we are only validating for required fields. We want to ensure the user provides a value for each.

If a field is left empty, the form will reload and present the errors above the form

02_validate.php

Outputting errors

If there are any errors, we output them in a loop above the form.

```
<?php if(count($errors)) : ?>
  <div class="errors">
    <ul>
      <?php foreach($errors as $error) : ?>
        <li><?=e($error)?></li>
      <?php endforeach; ?>
    </ul>
  </div>
<?php endif; ?>
```

If there are no errors, the $errors array will have a count of (sizeof) 0, and our conditional statement will resolve to false.

```
<input type="text" name="first_name" value="<?=e($_POST['first_name'] ?? '')?>">
```

Note also that for each field we output its sticky value in the value attribute.   The null coalsesce operator insures that if there is not sticky value for a field, we will simply output an empty string.

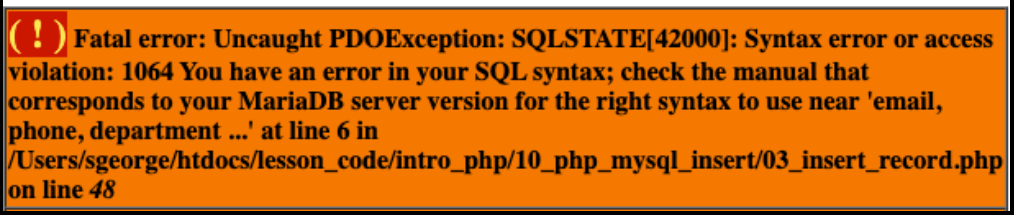
02_validate.php

If there are no errors

If there are no errors, we must insert the record in the database.

```
if(count($errors) == 0) {

$query = 'INSERT INTO employees
          (
               first_name,
               last_name,
               email
               phone,
               department
          )
          VALUES
          (
               :first_name,
               :last_name,
               :email,
               :phone,
               :department
          )';

$stmt = $dbh->prepare($query);
```

Building the query line by line, as you see here, makes it much easier to debug. If there are errors in the query, PHP will output the line number in the query that caused the error.



```
( ! ) Fatal error: Uncaught PDOException: SQLSTATE[42000]: Syntax error or access violation: 1064 You have an error in your SQL syntax; check the manual that corresponds to your MariaDB server version for the right syntax to use near 'email, phone, department ...' at line 6 in /Users/sgeorge/htdocs/lesson_code/intro_php/10_php_mysql_insert/03_insert_record.php on line 48
```

The exception above tells us that an error occurred at line 6 of our query. Hint: always look at the line before the line identified in the error message.

03_insert_record.php

Prepare the query, bind the values

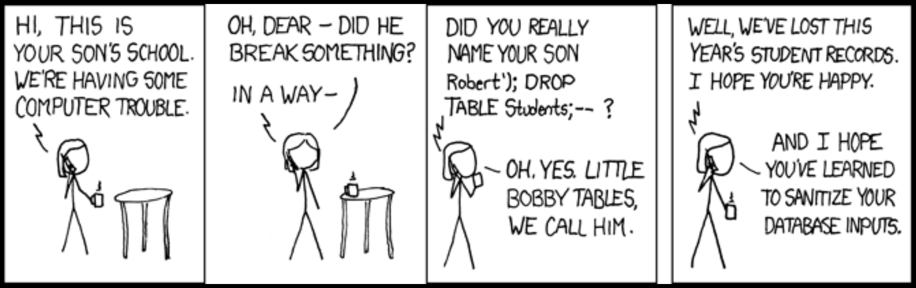
We must now take the values submitted by the form, and insert them into the database

```
$stmt = $dbh->prepare($query);

$stmt->bindValue(':first_name', $_POST['first_name'], PDO::PARAM_STR);
$stmt->bindValue(':last_name', $_POST['last_name'], PDO::PARAM_STR);
$stmt->bindValue(':email', $_POST['email'], PDO::PARAM_STR);
$stmt->bindValue(':phone', $_POST['phone'], PDO::PARAM_STR);
$stmt->bindValue(':department', $_POST['department'], PDO::PARAM_STR);

$stmt->execute();

$employee_id = $dbh->lastInsertId();
```

Using prepared prepared statements with placeholders in this way, and binding the values to those placeholders, properly escapes the values for use in MySQL.

This protects us against SQL Injection Attacks.

Once the insert is complete we can retrieve the Last Insert ID (primary key of the newly inserted record) by invoking the lastInsertId() method of the PDO object.


HI, THIS IS YOUR SON'S SCHOOL. WE'RE HAVING SOME COMPUTER TROUBLE.
OH, DEAR – DID HE BREAK SOMETHING?
IN A WAY–
DID YOU REALLY NAME YOUR SON Robert'); DROP TABLE Students;-- ?
OH. YES. LITTLE BOBBY TABLES, WE CALL HIM.
WELL, WE'VE LOST THIS YEAR'S STUDENT RECORDS. I HOPE YOU'RE HAPPY.
AND I HOPE YOU'VE LEARNED TO SANITIZE YOUR DATABASE INPUTS.

03_insert_record.php

Bind values alternative

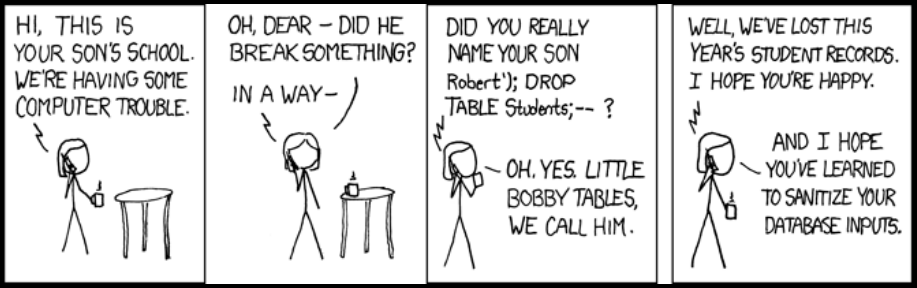
An alternative to binding the value of each field individually, we can do so in one fell swoop using a params array.

```
$stmt = $dbh->prepare($query);

$params = [
        ':first_name' => $_POST['first_name'],
        ':last_name' => $_POST['last_name'],
        ':email' => $_POST['email'],
        ':phone' => $_POST['phone'],
        ':department' => $_POST['department']
];

$stmt->execute($params);

$employee_id = $dbh->lastInsertId();
```

Using prepared prepared statements with placeholders in this way, and binding the values to those placeholders, properly escapes the values for use in MySQL.

This protects us against SQL Injection Attacks.

Once the insert is complete we can retrieve the Last Insert ID (primary key of the newly inserted record) by invoking the lastInsertId() method of the PDO object.


HI, THIS IS YOUR SON'S SCHOOL. WE'RE HAVING SOME COMPUTER TROUBLE.
OH, DEAR – DID HE BREAK SOMETHING?
IN A WAY–
DID YOU REALLY NAME YOUR SON Robert'); DROP TABLE Students;-- ?
OH. YES. LITTLE BOBBY TABLES, WE CALL HIM.
WELL, WE'VE LOST THIS YEAR'S STUDENT RECORDS. I HOPE YOU'RE HAPPY.
AND I HOPE YOU'VE LEARNED TO SANITIZE YOUR DATABASE INPUTS.

04_parameters.php

If the insert is successful

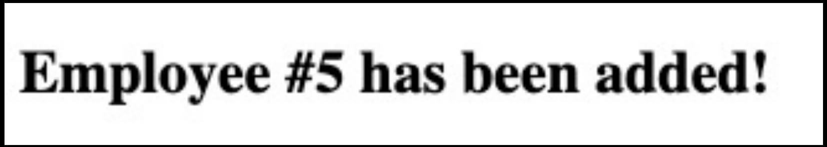
What happens if the insert is successful? In this case, we have a couple of options open to to us.

```
        $employee_id = $dbh->lastInsertId();

        if(!$employee_id) {
            die('There was a problem inserting the record.');
        }

        $_POST = []; // clear the form

    } // end if no errors
  } // end if $_POST
```

Option #1

If the insert worked (we have an $employee_id), we can empty all the form fields by setting $_POST to an empty array... load the form again, and output a success message... allowing the user to add additional records.

```
<?php if(isset($employee_id) && $employee_id > 0) : ?>
    <h2>Employee #<?=e($employee_id)?> has been added!</h2>
<?php endif; ?>
```



Employee #5 has been added!

04_parameters.php

If the insert is successful

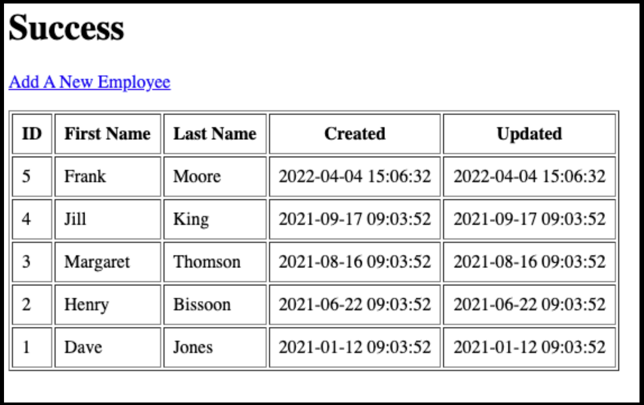
What happens if the insert is successful? In this case, we have a couple of options open to to us.

Option #2

```
        $employee_id = $dbh->lastInsertId();

        if(!$employee_id) {
          die('There was a problem inserting the record.');
        }

        header('Location: 06_success.php');
        die;

    } // end if no errors

  } // end if $_POST
```

If the insert worked (we have an $employee_id), we can redirect the user to a list view of the records, including the new record, using the header() function

Just added record is at top of list



Success

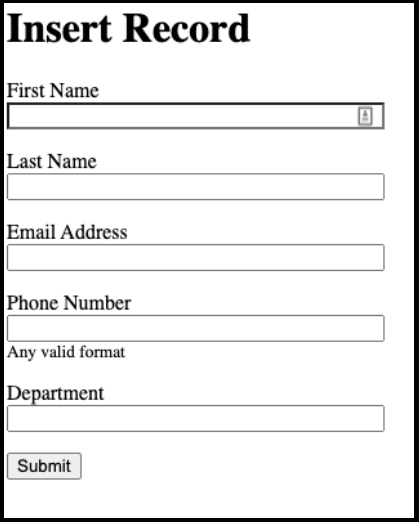
Add A New Employee

ID	First Name	Last Name	Created	Updated
5	Frank	Moore	2022-04-04 15:06:32	2022-04-04 15:06:32
4	Jill	King	2021-09-17 09:03:52	2021-09-17 09:03:52
3	Margaret	Thomson	2021-08-16 09:03:52	2021-08-16 09:03:52
2	Henry	Bissoon	2021-06-22 09:03:52	2021-06-22 09:03:52
1	Dave	Jones	2021-01-12 09:03:52	2021-01-12 09:03:52

05_add_redirect.php

The Insert workflow

Inserting records is such a common administrative task it has an established workflow.



Insert Record

First Name

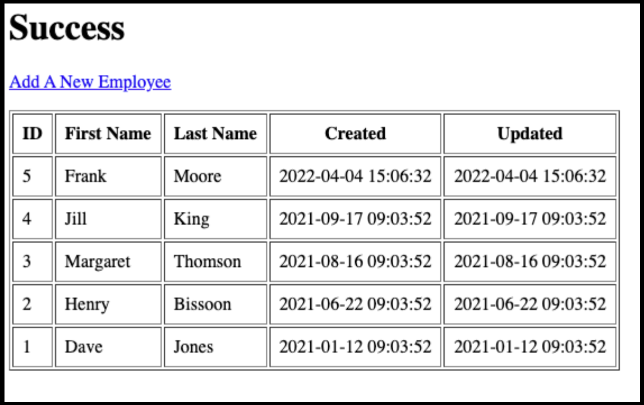
Last Name

Email Address

Phone Number

Any valid format

Department



Success

Add A New Employee

ID	First Name	Last Name	Created	Updated
5	Frank	Moore	2022-04-04 15:06:32	2022-04-04 15:06:32
4	Jill	King	2021-09-17 09:03:52	2021-09-17 09:03:52
3	Margaret	Thomson	2021-08-16 09:03:52	2021-08-16 09:03:52
2	Henry	Bisson	2021-06-22 09:03:52	2021-06-22 09:03:52
1	Dave	Jones	2021-01-12 09:03:52	2021-01-12 09:03:52

1. Start at the list view
2. Click the “Add New Record” link
3. Fill in the form and submit
4. Redirect back to list view
5. Repeat as necessary

05_add_redirect.php

Next:

PHP and MySQL Updates