Mysql and Sql

Mysql is an application which let us store data in database in tables(in the form of row and column) and insert retrieve data from database using sql language

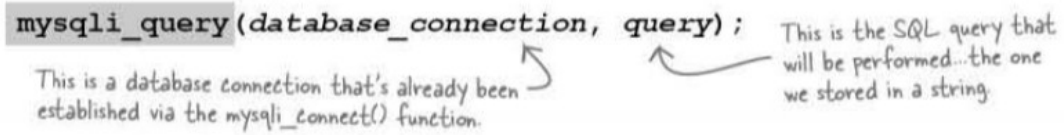
Sql is a structure query language using which we can interact with database application like Mysql.

PHPmyadmin is a graphical tool which is written in PHP which allow us to create database and tables through a web interface.

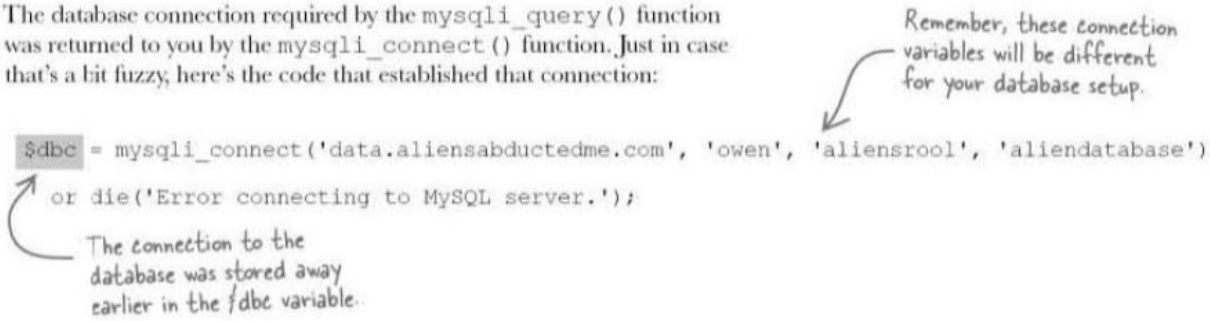
Mysqli_query()

Using Mysqli_query() we can add and retrieve data from database which need two parameter name of the connection and name of the variable which hold the query.

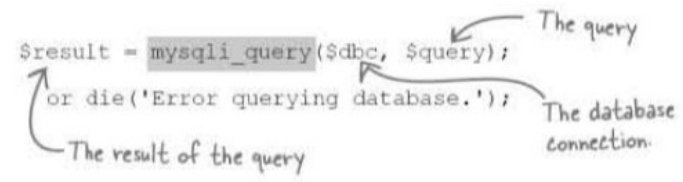
```
<?php

$conn=new mysqli_connect($servername, $username, $password, $dbname)
or die("error" ,mysqli_error);

$query="INSERT into table(student_name)values('$student_name')";

$result=mysali_query($conn,$query)
or die("error".mysqli_error);

?>
```



The database connection required by the `mysqli_query()` function was returned to you by the `mysqli_connect()` function. Just in case that's a bit fuzzy, here's the code that established that connection:



So you have a database connection (`$dbc`) and an SQL query (`$query`). All that's missing is passing them to the `mysqli_query()` function.



An SQL query is a request written in SQL code that is sent to the

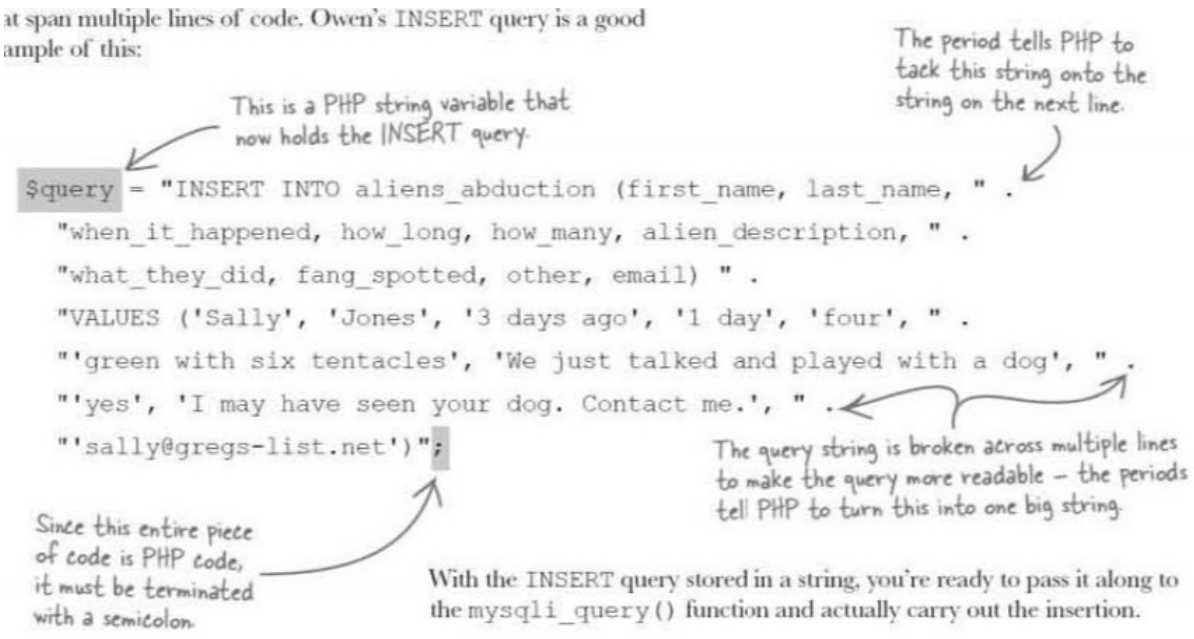
die() function

die() function provide us necessary feedback why the code fail. It terminate the rest of the php code if something goes wrong.

Query

Sql querys are passed to the PHP as a string format.

Using (.) we can break the query lines for reading purpose.



Query means asking the database to do something

We can leave off the column name from the insert or from the other query such as we can leave off (first_name, last_name from the INSERT query) in that case we have to provide the values of the column as they are appear in the database table
In short if you not provide column name then order of the values enter in the query must be same as the order of the column structured in the database.

$_POST

PHP $_POST is a PHP super global variable which is used to collect form data after submitting an HTML form with method="post". $_POST is also widely used to pass variables.

The $_POST variable is used to collect values from a form with method="post". Information sent from a form with the POST method is invisible to others and has no limits on the amount of information to send.

Example

```
<form action="welcome.php" method="post">
Enter your name: <input type="text" name="name" />
Enter your age: <input type="text" name="age" />
<input type="submit" />
</form>
```

When the user clicks the "Submit" button, the URL will not contain any form data, and will look something like this:

```
http://www.w3schools.com/welcome.php
```

The "welcome.php" file can now use the $_POST variable to catch the form data (notice that the names of the form fields will automatically be the ID keys in the $_POST array):

```
Welcome <?php echo $_POST["name"]; ?>.<br />
You are <?php echo $_POST["age"]; ?> years old!
```

Why use $_POST?

- Variables sent with HTTP POST are not shown in the URL
- Variables have no length limit

However, because the variables are not displayed in the URL, it is not possible to bookmark the page.

Remember that the name we use for $_POST must be same as the name in the HTML form field.

The good news is that the `report.php` script already has the form data stored away in variables thanks to the `$_POST` superglobal. Remember this PHP code?

```
$name = $_POST['firstname'] . ' ' . $_POST['lastname'];
$when_it_happened = $_POST['whenithappened'];
$how_long = $_POST['howlong'];
$how_many = $_POST['howmany'];
$alien_description = $_POST['aliendescription'];
$what_they_did = $_POST['whattheydid'];
$fang_spotted = $_POST['fangspotted'];
$email = $_POST['email'];
$other = $_POST['other'];
```

The $_POST superglobal's already being used to extract the data from each of Owen's form fields and store it in variables.

Remember, the name you use for $_POST needs to match up with the name of an HTML form field.

So you already have the form data in hand, you just need to incorporate it into the alien abduction `INSERT` statement. But you need to make a small change first. Now that you're no longer emailing the form data, you don't need the `$name` variable. You *do* still need the first and last name of the user so that they can be added to the database—but you need the names in separate variables.

```
$first_name = $_POST['firstname'];
$last_name = $_POST['lastname'];
```

The user's name is now stored in separate variables so that it can be inserted into distinct columns of the aliens_abduction table.