INTRODUCTION :

PHP started out as a small open source project that evolved as more and more people found out how useful it was. Rasmus Lerdorf unleashed the first version of PHP way back in 1994.

- PHP is a recursive acronym for "PHP: Hypertext Preprocessor".
- PHP is a server side scripting language that is embedded in HTML. It is used to manage dynamic content, databases, session tracking, even build entire e-commerce sites.
- It is integrated with a number of popular databases, including MySQL, PostgreSQL, Oracle, Sybase, Informix, and Microsoft SQL Server.
- PHP is pleasingly zippy in its execution, especially when compiled as an Apache module on the Unix side. The MySQL server, once started, executes even very complex queries with huge result sets in record-setting time.
- PHP supports a large number of major protocols such as POP3, IMAP, and LDAP. PHP4 added support for Java and distributed object architectures (COM and CORBA), making n-tier development a possibility for the first time.
- PHP is forgiving: PHP language tries to be as forgiving as possible.
- PHP Syntax is C-Like.

Common uses of PHP

- PHP performs system functions, i.e. from files on a system it can create, open, read, write, and close them.
- PHP can handle forms, i.e. gather data from files, save data to a file, through email you can send data, return data to the user.
- You add, delete, modify elements within your database through PHP.
- Access cookies variables and set cookies.
- Using PHP, you can restrict users to access some pages of your website.
- It can encrypt data.

Characteristics of PHP

Five important characteristics make PHP's practical nature possible −

- Simplicity
- Efficiency
- Security
- Flexibility
- Familiarity

"Hello World" Script in PHP

To get a feel for PHP, first start with simple PHP scripts. Since "Hello, World!" is an essential example, first we will create a friendly little "Hello, World!" script.

As mentioned earlier, PHP is embedded in HTML. That means that in amongst your normal HTML (or XHTML if you're cutting-edge) you'll have PHP statements like this −

```
<html>

   <head>
      <title>Hello World</title>
   </head>

   <body>
      <?php echo "Hello, World!";?>
   </body>

</html>
```

It will produce following result −

```
Hello, World!
```

If you examine the HTML output of the above example, you'll notice that the PHP code is not present in the file sent from the server to your Web browser. All of the PHP present in the Web page is processed and stripped from the page; the only thing returned to the client from the Web server is pure HTML output.

All PHP code must be included inside one of the three special markup tags ATE are recognised by the PHP Parser.

```
<?php PHP code goes here ?>

<?    PHP code goes here ?>

<script language = "php"> PHP code goes here </script>
```

Syntax

PHP syntax

A PHP script starts with **<?php** and ends with **?>**.

```
<?php
// write your PHP code here
?>
```

The file extension for PHP files is **".php"**.

A PHP file contains PHP scripting code, HTML tags, css and JavaScript.

Below, we have an example of a simple PHP file.

```
<!DOCTYPE html>
<html>
<body>
<h1>My first PHP Script</h1>
<?php
echo "Hello World!";
? >
```

Note:

PHP statements end with a semicolon (;).

Comments in PHP

There are various way of commenting PHP

```
<!DOCTYPE html>
<html>
<body>
<?php
// This is a single-line comment
# This is also a single-line comment
/*
This is a multiple-lines comment
*/
? >
</body>
</html>
```

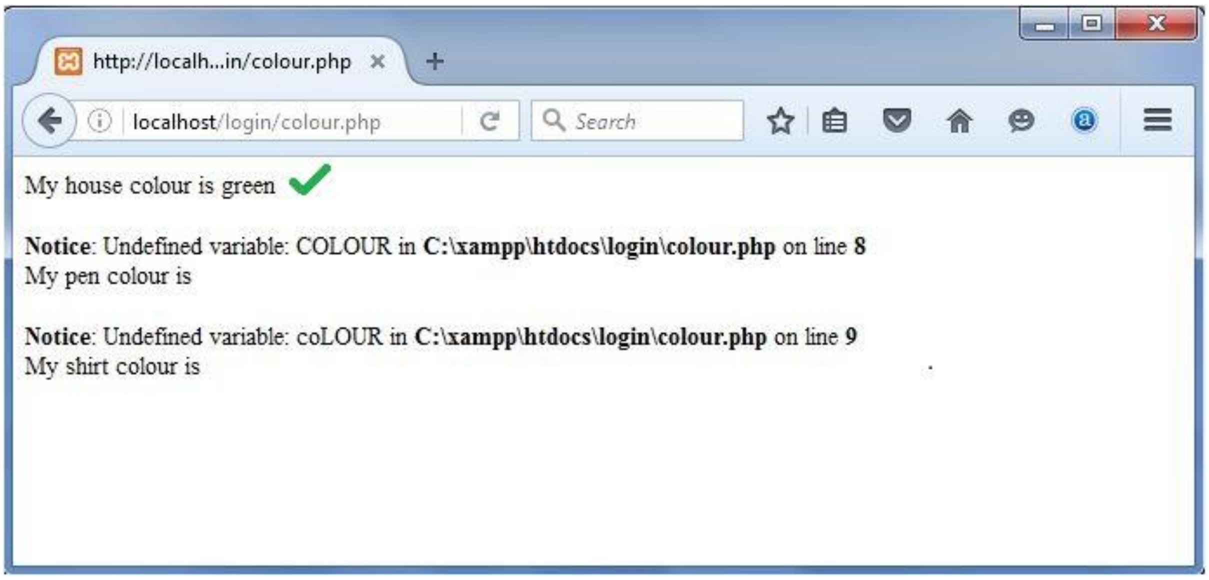
PHP Case Sensitivity

In PHP, all keywords (e.g. if, else, while, echo, etc.), classes, functions, and user-defined functions are NOT case-sensitive.

```
<!DOCTYPE html>
<html>
<body>
<?php
ECHO "Hello World! <br>";
eCHO "Hello World! <br>";
echo "Hello World! <br>";
EcHo "Hello World! <br>";
? >
</body>
</html>
```

But in PHP all variable names are case-sensitive.

```
<!DOCTYPE html>
<html>
<body>
<?php
$colour = "green";
echo "My house colour is " . $colour . "<br>";
echo "My pen colour is " . $COLOUR . "<br>";
echo "My shirt colour is" . $coLOUR . "<br>";
? >
</body>
</html>
```



```
My house colour is green ✓

Notice: Undefined variable: COLOUR in C:\xampp\htdocs\login\colour.php on line 8
My pen colour is

Notice: Undefined variable: coLOUR in C:\xampp\htdocs\login\colour.php on line 9
My shirt colour is
```

PHP echo and print statement

If there is an input then it must have an output. So in PHP also there is two statements which are used for displaying output data to the screen

- echo
- print

Both the statement are used for displaying data on the screen.

Difference between 'echo' and 'print'

echo	print
This statement can pass multiple string separated by ','.	It cannot pass multiple strings.
It doesnot return any values.	It always return 1.
It is faster than print.	It is slower than echo.

Examples of 'echo' statement

```
<html>
<body>
<?PHP
        echo "<h1>studentstutorial.com</h1>";
        echo "welcome to PHP world <br>";
?>
</body>
</html>
```

Example for multiple string input

```
<html>

<body>

<?PHP

      echo "PHP", "is", "one ", "of", "the","easiest","internet","technology.";
```

Input through variable

```
<html>

        <body>

<?PHP

        $a = "welcome to";

        $b = "studentstutorial.com";

        echo "<h1>$a</h1>";

        echo " $b<br>";

?>

</body>

</html>
```

Examples of 'print' statement

```
<html>
<body>
        <?PHP
        print "<h1>studentstutorial.com</h1>";
        print "welcome to PHP world <br>";
        ?>
</body>
</html>
```

PHP Basics

In PHP Variables are used for storing values such as numeric values, characters, character strings, or memory addresses.

In PHP, a variable starts with the $ sign, followed by the name of the variable.

Rules for PHP variables:

- A variable name must start with a letter or the underscore character.
- A variable name cannot start with a number.
- A variable name can only contain alpha-numeric characters and underscores (A-z, 0-9, and _)
- Variable names are case-sensitive ($age and $AGE are two different variables)

```
<!DOCTYPE html>
<html>
<body>
<?php
$a = "www.studentstutorial.com";
echo " $a is a PHP Tutorial";
? >
</body>
</html>
```

```
<!DOCTYPE html>
<html>
<body>
<?php
$a = 20;
$b= 10;
echo $a + $b;
? >
</body>
</html>
```

Output

```
30
```

Note- Variable name are user defined. You can take any name as your choice.

PHP has three different variable scopes:

- local
- global
- static

Local Variable scope

A variable declared within a function is called as LOCAL SCOPE and can only be accessed within that function.

```
<!DOCTYPE html>
<html>
<body>
<?php
function myTest() {
$a = 10; // local scope
echo "<p>Variable a inside function is: $a</p>";
}
myTest();
// using x outside the function will generate an error
echo "<p>Variable a outside function is: $a</p>";
? >
</body>
</html>
```

Global Variable scope

A variable declared outside a function is called as GLOBAL SCOPE variable and can only be accessed outside a function:

```

<!DOCTYPE html>
<html>
<body>
<?php $a = 10; // global scope
function myTest() {
// using a inside this function will generate an error
echo "<p>Variable a inside function is: $a</p>";
}
myTest();
echo "<p>Variable a outside function is: $a</p>";
? >
</html>
```

Static Variable Scope

A static variable exists only in a local function scope, but it does not lose its value when program execution leaves this scope.

PHP Superglobal variable with example

In PHP there is 9 Superglobals variable available these are:

1. $GLOBALS
2. $_SERVER

3. $_REQUEST
4. $_POST
5. $_GET
6. $_FILES
7. $_ENV
8. $_COOKIE
9. $_SESSION

$_REQUEST

In PHP $_REQUEST is used to collect form data.

$_SERVER

$_SERVER is an super globalvariable containing information such as headers, paths, and script locations.

$_GET

In PHP $_GET is used to collect form data.

In GET you send information to server in two way :

1. Through URL
2. Through from

$_POST

In PHP $_POST is used to collect form data.

PHP Data type with example

We can assign different type of data to a variable. Data type means the various type of data that we assign to a variable.

PHP support various type of data type and all are also different from each other.

Below are the list of data type :

1. String
2. Integer
3. Float (floating point numbers - also called double)
4. Boolean
5. Array
6. Object
7. NULL
8. Resource

PHP String

A string can have letters, numbers, special characters and arithmetic values or combination of all.

```
<!DOCTYPE html> <html> <body> <?php  $a = "Hello world!"; $b = 'Hello world!';
echo $a; echo "<br>";  echo $b; ?> </body> </html>
```

You can use both single quotation or double quotation

PHP Integer

All the number without decimal point is called as integer.

Example: 100, 2000 etc.

```
<!DOCTYPE html> <html> <body> <?php   $a = 100; var_dump($a); ?>   </body>
</html>
```

PHP Float or Double

The Float or Double data type are number with decimal point.

Example: 29.6, 30.03 etc..

```
<!DOCTYPE html> <html> <body> <?php   $a = 29.6; var_dump($a); ?>   </body>
</html>
```

PHP null

A variable of type null is a variable without any data.

```
<!DOCTYPE html> <html> <body> <?php $a= NULL; var_dump($a); ?> </body>
</html>
```

PHP String with example

A string is can have letters, numbers, special characters and arithmetic values or combination of all.

Example: Hello World !

Example: Hello World ! 123

Above both example are string.

To assign a string to a variable is there is two way.

```
$string='Hello World !';
$string_two="Hello World !";
```

You can use both single quotation or double quotation.But there is a small difference between single quotation and double quotation.

single quotation

1. Single quoted strings are the easiest way to specify string.
2. It is faster than double quotation.

double quotation

1. Double quotes force PHP to specify the string.
2. It is less faster than single quotation.

So now in our mind a big confusion come that where to use single quotation and where to use double quotation.

Best example to under stand.

```
<?php
$a='Hello';
echo '$a World';
?>
```

Output: $a World

```
<?php
$a='Hello';
echo "$a World";
?>
```

Output: Hello World

PHP Operators with example

Operator in PHP is a symbol that is used to perform operations.For example: **+, -, *,** / etc.

There are many types of operators in PHP which are given below:

1. Arithmetic operators
2. Assignment operators
3. Comparison operators
4. Increment/Decrement operators
5. Logical operators
6. String operators
7. Array operators

1. Arithmetic operators

+	Addition	`$x + $y`	Sum of $x and $y
-	Subtraction	`$x - $y`	Difference of $x and $y.
*	Multiplication	`$x * $y`	Product of $x and $y.
/	Division	`$x / $y`	Quotient of $x and $y
%	Modulus	`$x % $y`	Remainder of $x divided by $y

2. Assignment operators

=	Assign	`$x = $y`	`$x = $y`
+=	Add and assign	`$x += $y`	`$x = $x + $y`
-=	Subtract and assign	`$x -= $y`	`$x = $x - $y`
*=	Multiply and assign	`$x *= $y`	`$x = $x * $y`
/=	Divide and assign quotient	`$x /= $y`	`$x = $x / $y`
%=	Divide and assign modulus	`$x %= $y`	`$x = $x % $y`

3. Assignment operators

==	Equal	`$x == $y`	True if $x is equal to $y
===	Identical	`$x === $y`	True if $x is equal to $y, and they are of the same type
!=	Not equal	`$x != $y`	True if $x is not equal to $y
<>	Not equal	`$x <> $y`	True if $x is not equal to $y
!==	Not identical	`$x !== $y`	True if $x is not equal to $y, or they are not of the same type
<	Less than	`$x < $y`	True if $x is less than $y
>	Greater than	`$x > $y`	True if $x is greater than $y
>=	Greater than or equal to	`$x >= $y`	True if $x is greater than or equal to $y
<=	Less than or equal to	`$x <= $y`	True if $x is less than or equal to $y

4. Increment/Decrement operators

`++$x`	Pre-increment	Increments $x by one, then returns $x

`$x++`	Post-increment	Returns $x, then increments $x by one
`--$x`	Pre-decrement	Decrements $x by one, then returns $x
`$x--`	Post-decrement	Returns $x, then decrements $x by one

5. Logical operators

`and`	And	`$x and $y`	True if both $x and $y are true
`or`	Or	`$x or $y`	True if either $x or $y is true
`xor`	Xor	`$x xor $y`	True if either $x or $y is true, but not both
`&&`	And	`$x && $y`	True if both $x and $y are true
`||`	Or	`$x || $y`	True if either $$x or $y is true
`!`	Not	`!$x`	True if $x is not true

6. String operators

`.`	Concatenation	`$str1 . $str2`	Concatenation of $str1 and $str2
`.=`	Concatenation assignment	`$str1 .= $str2`	Appends the $str2 to the $str1

7. Array Operator

`+`	Union	`$x + $y`	Union of $x and $y
`==`	Equality	`$x == $y`	True if $x and $y have the same key/value pairs
`===`	Identity	`$x === $y`	True if $x and $y have the same key/value pairs in the same order and of the same types
`!=`	Inequality	`$x != $y`	True if $x is not equal to $y
`<>`	Inequality	`$x <> $y`	True if $x is not equal to $y
`!==`	Non-identity	`$x !== $y`	True if $x is not identical to $y

PHP 5 if...else...elseif Statements

PHP - The if Statement

If statement is a control statement. The body of the if statement is executed when condition is true or nonzero. In PHP we can pass either Boolean value or any other value.

```
if (condition) {
code to be executed if condition is true;
}
```

```
<?php
$a=10;
$b=10;
if($a==$b)
{
echo"successful";
}
?>
```

PHP - The if...else Statement

It is also a control statement. It is very useful in programming. If condition is not true the else part executed. Else has no condition.

```
if (condition) {
code to be executed if condition is true;
} else {
code to be executed if condition is false;
}
```

```
<?php
$a=10;
$b=10;
if($a==$b)
{
echo"successful";
}
else
{
echo"failure";
}
?>
```

PHP - The if...elseif....else Statement

```
if (condition) {
code to be executed if this condition is true;
} elseif (condition) {
code to be executed if this condition is true;
} else {
code to be executed if all conditions are false;
}
```

```
<?php
$a=10;
$b=10;
$c=10;
if($a > $b and $a > $c)
{
echo"a is largest.";
}
elseif($b > $a and $b > $c)
{
echo"b is largest";
}
elseif($c > $a and $c > $b)
{
echo" c is largest";
}
else
{
echo"a=b=c";
}
?>
```

PHP Switch Statement with example

The switch-case statement is an alternative to the if-elseif-else statement. It does the same thing as if-elseif-else statement.

How it Works

- The switch expression is evaluated once.
- The value of the expression is compared with the values of each Case.
- If there is a match, the associated block of code is executed other wise the default block is executed.

```
switch (n) {
case label1:
code to be executed if n=label1;
break;
case label2:
code to be executed if n=label2;
break;
case label3:
code to be executed if n=label3;
break;
...
default:
code to be executed if n is different from all labels;
}
```

```
<!DOCTYPE html>
<html>
<body>
<?php
$d = date("D");
switch ($d){
case "Mon":
echo "Today is Monday";
break;
case "Tue":
echo "Today is Tuesday";
break;
case "Wed":
echo "Today is Wednesday";
break;
case "Thu":
echo "Today is Thursday";
break;
case "Fri":
echo "Today is Friday";
break;
case "Sat":
echo "Today is Saturday";
break;
case "Sun":
echo "Today is Sunday";
break;
default:
echo "Wonder which day is this ?";
}
?>
</body>
</html>
```

```
<!DOCTYPE html>
<html>
<body>
<?php
$favsport = "cricket";
switch ($favsport) {
case "hockey":
echo "Your favorite sport is Hocky!";
break;
case "tennis":
echo "Your favorite sport is Cricket!";
break;
case "football":
echo "Your favorite sport is Football!";
break;
case "badminton":
echo "Your favorite sport is Badminton!";
break;
default:
echo "Your favorite sport is neither Hockey, Cricket, Football nor Golf!";
}
?>
</body>
</html>
```

PHP while loops

In PHP while loops is used to execute a block of code while the specified condition is true.

Some time we want the same block of code to run over and over again.

Example: Print 1 to 10 number.

Example: Fetch multiple students data.In that case we can use while loop.

```
while (condition is true) {
        code to be executed;
}
```

```
<?php
$a=1;
while ($a<=10)
{
echo $a;
$a++;
}
?>
```

Example(Print all even and odd number between 0 to 10)

```
<?php
$i=0;
        while($i <= 10){
                if($i % 2 == 0){
                        echo $i." - Even,  ";
                }else{
                        echo $i." - Odd,  ";
                }
                $i++;
        }
?>
```

PHP do-while loops

The do...while loop will always execute the block of code once In PHP do..while loop is similar to while loop but one key difference between them.The do...while loop will always execute the block of code once. If the the condition is true then code of block executed repeatedly, until the condition becomes false.

But in while loops execute a block of code while the specified condition is true.

```
do {

    code to be executed;
```

```
<?php
$a=1;
do
{
echo $a;
$a++;
}
while($a<=10)
?>
```

PHP for Loop

For loop is a control Statement.

```
for (init counter; test counter; increment counter) {
code to be executed;
}
```

In for loop there are 3 section.

- init counter
- test counter
- increment counter

The **init counter** is Used for Intialize Because it is Executed only once.

The **test counter** Check the Condition.Here we Write Expression which can be Executed Multiple Times.

The **increment counter** is Used for Increasing or Decreasing the value of a variable or we can Write any Expression.It is also Executed Several Times.

```
<?php
for($a=1;$a<=10;$a++)
echo "$a <br>";
?>
```

PHP foreach loop

foreach loop is a special type of control statement which is used for array .

it access all the elements in an array .

```
foreach ($array as $value) {
code to be executed;
}
```

```
<?php
$data=array(1,2,3,4,5);
foreach($data as $element)
{
echo "$element";
}
?>
```

PHP Array

Array is a special kind of variable which is able to store more than one variable simultaneously.

```
<?php
$data=array("30","50","55");
echo $data[0]."<br>";
echo $data[1]."<br>";
echo $data[2];
?>
```

Output

```
30
50
55
```

Get The Length of an Array

```
<?php
$data=array("30","50","55");
echo count($data);
?>
```

Output

```
3
```

How to assign element in array?

```
<?php
$data=array();
for($i=0;$i<=10;$i++)
{
$data[$i]=$i;
}
foreach($data as $element)
{
echo "$element";
}
?>
```

Output

```
12345678910
```

PHP Function

PHP function with real time example

A function is a statement or block of code use for perform certain task or operation.

A function is not execute when we run a program. A function is execute only when we call the function.

```
function functionName(){
// Code to be executed
}
```

There is two type of function in PHP:

1. Built in function
2. User defined function

Built in function

In php there is almost 1000 built in function exists.We only have to use this as per our requirements.

Example:

print_r(), strlen(), fopen(),mail() function etc.

User Defined function

A user defined function is create by the user.

```
<?php
function writeMsg() {
echo "Hello world!";
}
writeMsg();
?>
```

Here writeMsg() is a user defined function.

Advantages

1. Code reusability
2. Less code
3. Easy to understand
4. Easy to find error

Code reusability

As the function is separate from other script of function we can easily reused the function for other application.

Less code

Less code means because you don't need to write the logic many times. Once you write a function we have to only call to that function where it needed.

Easy to understand

Easy to understand in the sense that as the logic inside the function we have to only know the function name . Once we get the function name we can find the logic and easily modify it.

Easy to find error

We can easily find the error if we use function because it is separate from php programming.By find where the function call we get the function name and easily solve the error.

```
<!DOCTYPE html>
<html>
<body>
<?php
function writeMsg() {
echo "Hello world!";
}
writeMsg();
?>
</body>
</html>
```

Function with parameter

```
function myFunc($oneParameter, $anotherParameter){
// Code to be executed
}
```

```
<!DOCTYPE html>
<html>
<body>
<?php
function getSum($num1, $num2){
$sum = $num1 + $num2;
echo "Sum of the two numbers $num1 and $num2 is : $sum";
}
getSum(20, 30);
?>
</body>
</html>
```

Difference between GET and POST PHP

In PHP there are two ways to send information to server.

These are GET and POST. $_GET and $_POST both are super global variable.

The variable which is always accessible is called as super global variable.

Difference

$_GET :

1. $_GET is super global variable.
2. Data send in GET is visible in URL.
3. Not preferable for send sensitive data.
4. Limitation of send data.(About 2000 character)
5. In GET you send information to server in two way :
   i. Through URL
   ii. Through from

Through URL

```
<!DOCTYPE html>
<html>
<body>
<a href="http://www.example.com/action.php?name=john&age=24">Send</a>
</body>
</html>
```

Through From

```
<!DOCTYPE html>
<html>
<body>
<form action="action_page.php" method="get">
First name:<br>
<input type="text" name="firstname">
<br>
Last name:<br>
<input type="text" name="lastname">
<br><br>
<input type="submit" value="Submit">
</form>
</body>
</html>
```

In action_page.php we gather the information using $_GET super global variable.

$_POST

1. $_GET is super global variable.
2. Data send in POST is not visible in URL.
3. Preferable for send sensitive data.
4. No limitation for send data.
5. In POST you send information to server in one way that is form.

```
<!DOCTYPE html>
<html>
<body>
<form action="action_page.php" method="post">
First name:<br>
<input type="text" name="firstname">
<br>
Last name:<br>
<input type="text" name="lastname">
<br><br>
<input type="submit" value="Submit">
</form>
</body>
</html>
```

In action_page.php we gather the information using $_POST super global variable.

MY SQL

How to create a table in PhpMyAdmin XAMPP or WAMP server

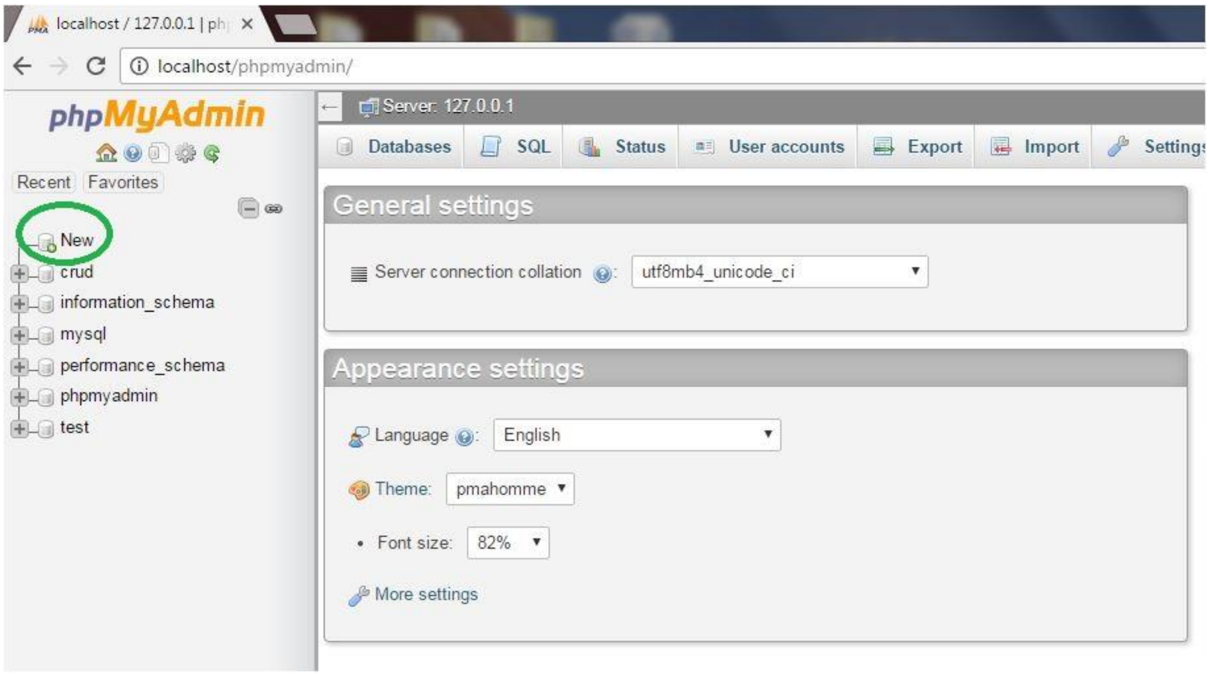
To start any kind of project in PHP you need to create a table in PhpMyAdmin.

To create a table in PhpMyAdmin first you have to create a database.

So first of all i am going to show you how to create a database in PhpMyAdmin. Please follow the below step to easily create a database PhpMyAdmin.

Step 1

First of all start your XAMPP or WAMPP server. Then go to the link bar of your browser and type **localhost/phpmyadmin** and click enter.Then you find a page like this



Server connection collation: utf8mb4_unicode_ci

Language: English

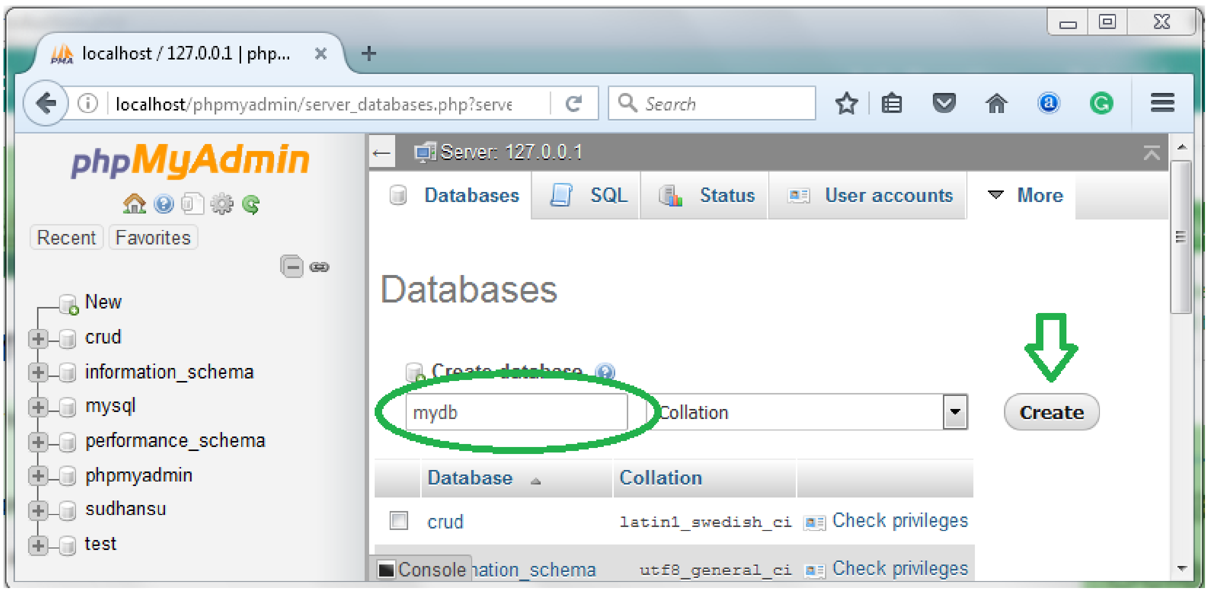
Theme: pmahomme

Font size: 82%

Then click on the new button that is available on the sidebar of that page.

Step 2

After click on the new button you find a page like that


localhost / 127.0.0.1 | php...
localhost/phpmyadmin/server_databases.php?serve
phpMyAdmin
Recent Favorites
New
crud
information_schema
mysql
performance_schema
phpmyadmin
sudhansu
test
Server: 127.0.0.1
Databases SQL Status User accounts More
Databases
Create database
mydb
Collation
Create
Database Collation
crud latin1_swedish_ci Check privileges
information_schema utf8_general_ci Check privileges
Console

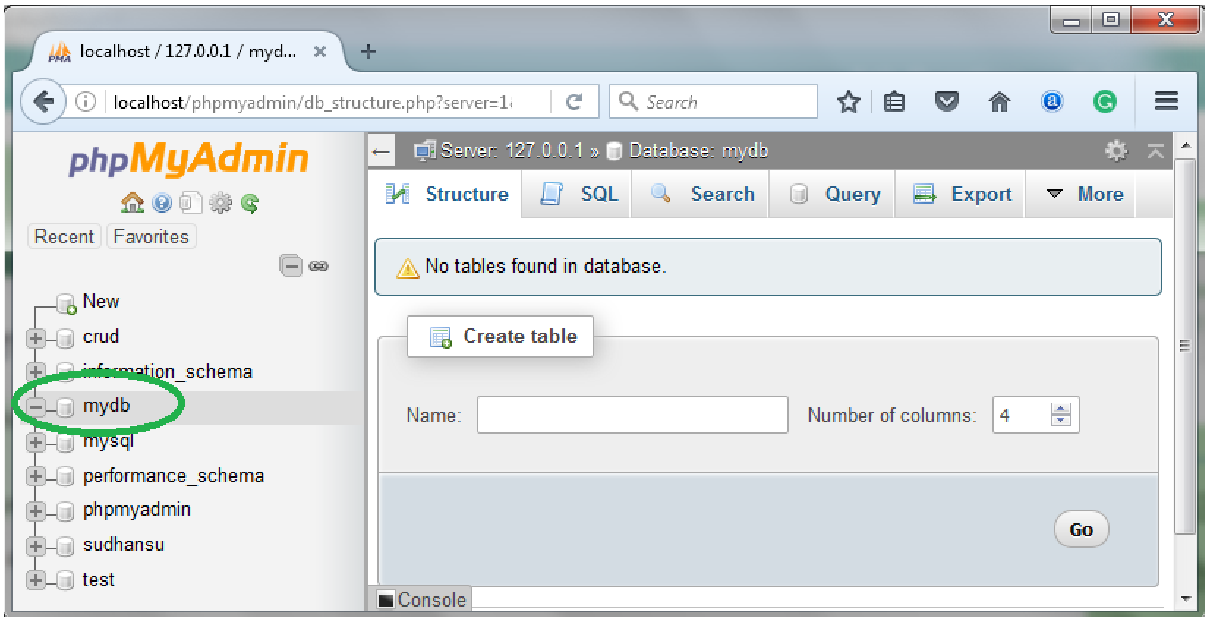
Here put a name of your own choice on the create database field and click on the **create** button and you get a successful message on the screen that you create database successfully and will appear in the database list.

Now i am going to show you how to **create a table in PhpMyAdmin** under the database that i have create currently.

Note: You can’t create table if you not have a database so first create database.

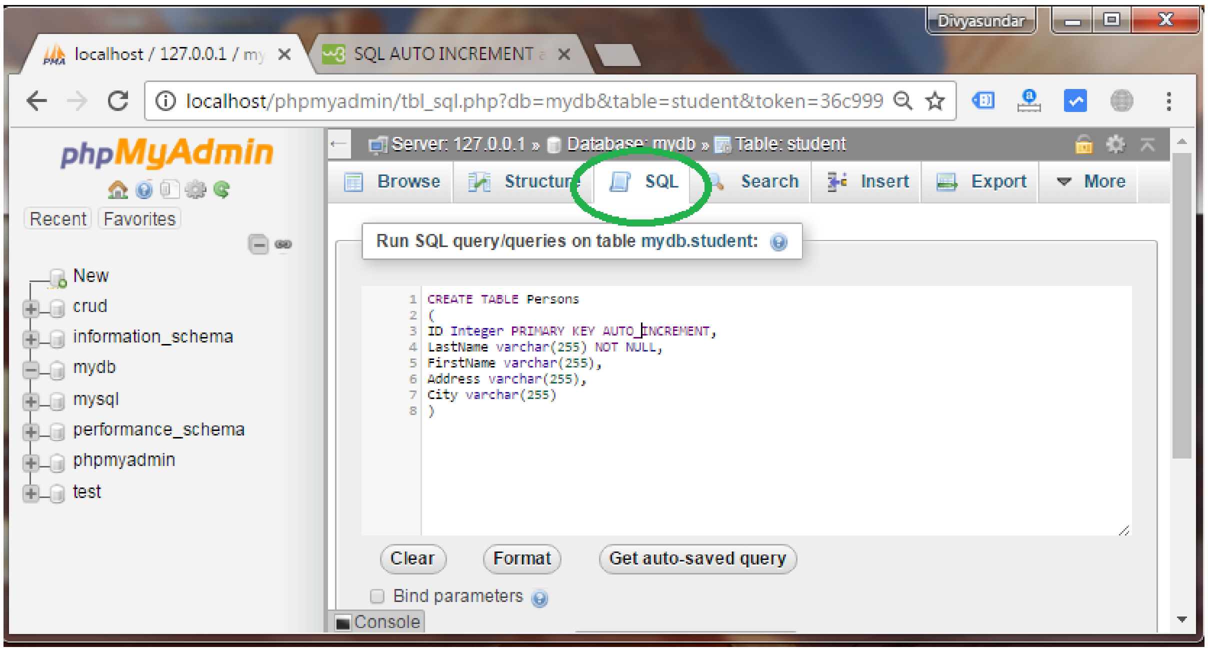
Step 3

Click on the database name in which under you create a table.After click on the database name you find a page like that.


localhost / 127.0.0.1 / myd...
localhost/phpmyadmin/db_structure.php?server=1
phpMyAdmin
Recent Favorites
New
crud
information_schema
mydb
mysql
performance_schema
phpmyadmin
sudhansu
test
Server: 127.0.0.1 » Database: mydb
Structure SQL Search Query Export More
No tables found in database.
Create table
Name:
Number of columns: 4
Go
Console

Here you have two option to create table first one is using **structure** and second one is using **SQL**.If you want to create table in structure option then put your table name on the create table name field and choose columns and click on the go button.

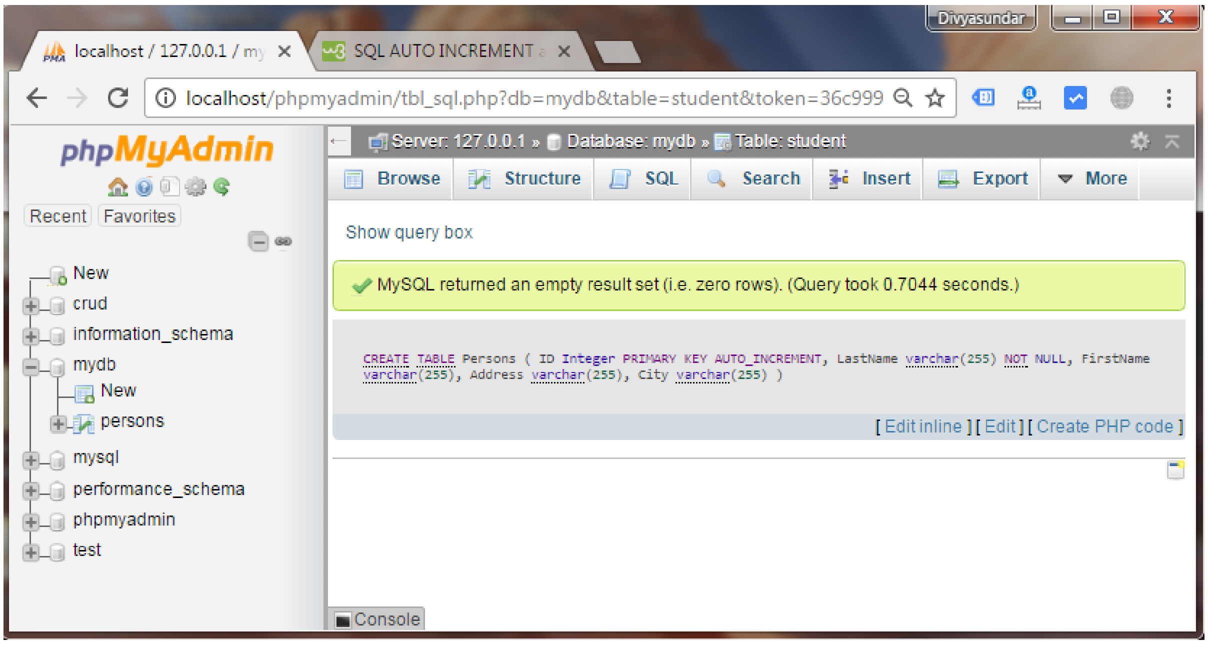
If you want to create a table by writing SQL Query simply click on the SQL button on the page and write your query and click on the go button.



Run SQL query/queries on table mydb.student:

```
CREATE TABLE Persons
(
ID Integer PRIMARY KEY AUTO_INCREMENT,
LastName varchar(255) NOT NULL,
FirstName varchar(255),
Address varchar(255),
City varchar(255)
)
```

If your SQL query is correct you find a page like that.


localhost / 127.0.0.1 / my
SQL AUTO INCREMENT
localhost/phpmyadmin/tbl_sql.php?db=mydb&table=student&token=36c999
phpMyAdmin
Recent
Favorites
New
crud
information_schema
mydb
New
persons
mysql
performance_schema
phpmyadmin
test
Server: 127.0.0.1 » Database: mydb » Table: student
Browse
Structure
SQL
Search
Insert
Export
More
Show query box
[Edit inline] [Edit] [Create PHP code]
Console

MySQL returned an empty result set (i.e. zero rows). (Query took 0.7044 seconds.)

```
CREATE TABLE Persons ( ID Integer PRIMARY KEY AUTO_INCREMENT, LastName varchar(255) NOT NULL, FirstName varchar(255), Address varchar(255), City varchar(255) )
```

MySQLi database Connection in PHP

```
<?php
// Create connection
$conn = mysqli_connect("localhost","root","admin","mydb");
?>
```

This code also use as database connection but it will not show the output message that it is successfully connect or not.

Here

- localhost=servername
- root=username
- admin=password
- mydb=database name

How to create a database in MySQL using PHP script

For create database in MySql the CREATE DATABASE statement is used.

The following example is about how to create a database in mysql without open the phpmyadmin or mysql.

```
<?php
$servername = "localhost";
$username = "root";
$password = "password";/* Put your password here*/

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

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

mysqli_close($conn);
?>
```

How to create table in MySQL using php script

For create table in MySQL the CREATE TABLE statement is used.Before run the script for create table in MySQL first create a database in MySQL.

The following example is about how to create a table in mysql without open the phpmyadmin or MySQL.

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "admin";/* put your database name here */

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

/* sql query to create table */
$sql = "CREATE TABLE Student
(
ID int NOT NULL AUTO_INCREMENT,
LastName varchar(50),
FirstName varchar(50),
RollNo varchar(50),
City varchar(50),
PRIMARY KEY (ID)
);

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

mysqli_close($conn);
?>
```

Insert Data Into MySQL Using PHP

For insert data in MySQL first we have to create a table in data base.

Here we using 3 file for insert data in MySQL:

- **database.php**:For connecting data base
- **insert.php**:for getting the values from the user
- **process.php**:A PHP file that process the request

```
CREATE TABLE `employee` (
        `userid` int(8) NOT NULL,
        `first_name` varchar(55) NOT NULL,
        `last_name` varchar(55) NOT NULL,
        `city_name` varchar(55) NOT NULL,
        `email` varchar(50) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

database.php

```
<?php
$servername='localhost';
$username='root';
$password='';
$dbname = "crud";
$conn=mysqli_connect($servername,$username,$password,"$dbname");
if(!$conn){
   die('Could not Connect My Sql:' .mysql_error());
}
?>
```

insert.php

```
<!DOCTYPE html>
<html>
  <body>
        <form method="post" action="process.php">
                First name:<br>
                <input type="text" name="first_name">
                <br>
                Last name:<br>
                <input type="text" name="last_name">
                <br>
                City name:<br>
                <input type="text" name="city_name">
                <br>
                Email Id:<br>
                <input type="email" name="email">
                <br><br>
                <input type="submit" name="save" value="submit">
        </form>
  </body>
</html>
```

process.php

```
<?php
include_once 'database.php';
if(isset($_POST['save']))
{
	$first_name = $_POST['first_name'];
	$last_name = $_POST['last_name'];
	$city_name = $_POST['city_name'];
	$email = $_POST['email'];
	$sql = "INSERT INTO employee
(first_name,last_name,city_name,email)
	VALUES ('$first_name','$last_name','$city_name','$email')";
	if (mysqli_query($conn, $sql)) {
		echo "New record created successfully !";
	} else {
		echo "Error: " . $sql . "
" . mysqli_error($conn);
	}
	mysqli_close($conn);
}
?>
```

Retrieve Data From MySQL Using PHP

For retrieve data from MySQL the SELECT statement is used. We can retrieve data from specific column or all column of a table.

To retrieve selected column data from database the SQL query is

SELECT column_name,column_name FROM table_name;

To retrieve all the column data from a table the SQL query is

SELECT * FROM table_name;

In the below example we retrieve the data from MySQL database.

In this example we used 2 file for retrieve data

- database.php- To connecting database.
- retrieve.php- For retrive data from database

database.php

```
<?php
$url='127.0.0.1:3306';
$username='root';
$password='';
$conn=mysqli_connect($url,$username,$password,"crud");
if(!$conn){
 die('Could not Connect My Sql:' .mysql_error());
}
?>
```

retrieve.php

```
<?php
include_once 'database.php';
$result = mysqli_query($conn,"SELECT * FROM myusers");
?>
<!DOCTYPE html>
<html>
 <head>
 <title> Retrive data</title>
 </head>
<body>
<?php
if (mysqli_num_rows($result) > 0) {
?>
  <table>

  <tr>
    <td>First Name</td>
    <td>Last Name</td>
    <td>City</td>
    <td>Email id</td>
  </tr>
<?php
$i=0;
while($row = mysqli_fetch_array($result)) {
?>
<tr>
    <td><?php echo $row["first_name"]; ?></td>
    <td><?php echo $row["last_name"]; ?></td>
    <td><?php echo $row["city_name"]; ?></td>
    <td><?php echo $row["email"]; ?></td>
</tr>
<?php
$i++;
}
?>
</table>
 <?php
}
else{
    echo "No result found";
}
?>
 </body>
</html>
```

style.css

```
table {
    font-family: arial, sans-serif;
    border-collapse: collapse;
    width: 100%;
}

td, th {
    border: 1px solid #dddddd;
    text-align: left;
    padding: 8px;

}

tr:nth-child(even) {
    background-color: white;
}
```

Update Data in MySQL Using PHP

To update a data that already exist in the database, UPDATE statement is used.

In the below example we update the employee data from MySQL database.

we used 2 file for update data

- **database.php**- To connecting database.
- **update.php**- TO retrieve data from database with a update option.
- **update-process.php**- TO update data from database.

database.php

```
<?php
   $servername='localhost';
   $username='root';
   $password='';
   $dbname = "crud";
   $con=mysqli_connect($servername,$username,$password,"$dbname");
   if(!$conn){
      die('Could not Connect My Sql:' .mysql_error());
   }
?>
```

update.php

```
<?php
include_once 'database.php';
$result = mysqli_query($conn,"SELECT * FROM employee");
?>
<!DOCTYPE html>
<html>
 <head>
   <title> Retrive data</title>
   <link rel="stylesheet" href="style.css">
 </head>
<body>
<?php
if (mysqli_num_rows($result) > 0) {
?>
<table>
          <tr>
            <td>Sl No</td>
                <td>First Name</td>
                <td>Last Name</td>
                <td>City</td>
                <td>Email id</td>
                <td>Action</td>
          </tr>
                        <?php
                        $i=0;
                        while($row = mysqli_fetch_array($result)) {
                        ?>
          <tr>
            <td><?php echo $row["id"]; ?></td>
                <td><?php echo $row["first_name"]; ?></td>
                <td><?php echo $row["last_name"]; ?></td>
                <td><?php echo $row["city_name"]; ?></td>
                <td><?php echo $row["email"]; ?></td>
                <td><a href="update-process.php?id=<?php echo $row["id"];
?>">Update</a></td>
      </tr>
                        <?php
                        $i++;
                        }
                        ?>
</table>
 <?php
}
else
{
    echo "No result found";
}
?>
 </body>
</html>
```

update-process.php

```
<?php
include_once 'database.php';
if(count($_POST)>0) {
mysqli_query($conn,"UPDATE employee set userid='" . $_POST['userid'] .
"', first_name='" . $_POST['first_name'] . "', last_name='" .
$_POST['last_name'] . "', city_name='" . $_POST['city_name'] . "'
,email='" . $_POST['email'] . "' WHERE userid='" . $_POST['userid'] .
"'");
$message = "Record Modified Successfully";
}
$result = mysqli_query($conn,"SELECT * FROM employee WHERE userid='" .
$_GET['userid'] . "'");
$row= mysqli_fetch_array($result);
?>
<html>
<head>
<title>Update Employee Data</title>
</head>
<body>
<form name="frmUser" method="post" action="">
<div><?php if(isset($message)) { echo $message; } ?>
</div>
<div style="padding-bottom:5px;">
<a href="retrieve.php">Employee List</a>
</div>
Username: <br>
<input type="hidden" name="userid" class="txtField" value="<?php echo
$row['userid']; ?>">
<input type="text" name="userid"  value="<?php echo $row['userid']; ?>">
<br>
First Name: <br>
<input type="text" name="first_name" class="txtField" value="<?php echo
$row['first_name']; ?>">
<br>
Last Name :<br>
<input type="text" name="last_name" class="txtField" value="<?php echo
$row['last_name']; ?>">
<br>
City:<br>
<input type="text" name="city_name" class="txtField" value="<?php echo
$row['city_name']; ?>">
<br>
Email:<br>
<input type="text" name="email" class="txtField" value="<?php echo
$row['email']; ?>">
<br>
<input type="submit" name="submit" value="Submit" class="buttom">

</form>
</body>
</html>
```

Delete Data From MySQL Using PHP

To delete data from MySQL the **DELETE statement** is used. We can delete data from specific column or all column of a table.

To delete selected column data from database the SQL query is

DELETE FROM table_name WHERE some_column=some_value;

To delete all the column data from a table the SQL query is

DELETE FROM table_name;

or

DELETE * FROM table_name;

In the below example we delete the employee data from MySQL database.

In this example we used 2 file for retrieve data

- **database.php**- To connecting database.
- **delete.php**- For retrieve data from database with delete option
- **delete-process.php**- For dlete data from database or from your table

database.php

```
<?php
$url='127.0.0.1:3306';
$username='root';
$password='';
$conn=mysqli_connect($url,$username,$password,"crud");
if(!$conn){
 die('Could not Connect My Sql:' .mysql_error());
}
?>
```

delete.php

```
<?php
include_once 'database.php';
$result = mysqli_query($conn,"SELECT * FROM employee");
?>

<!DOCTYPE html>
<html>
<head>
<link rel="stylesheet" href="style.css">
<title>Delete employee data</title>
</head>
<body>
<table>
        <tr>
        <td>Employee Id</td>
        <td>First Name</td>
        <td>Last Name</td>
        <td>City</td>
        <td>Email id</td>
        <td>Action</td>
        </tr>
        <?php
        $i=0;
        while($row = mysqli_fetch_array($result)) {
        ?>
        <tr class="<?php if(isset($classname)) echo $classname;?>">
        <td><?php echo $row["userid"]; ?></td>
        <td><?php echo $row["first_name"]; ?></td>
        <td><?php echo $row["last_name"]; ?></td>
        <td><?php echo $row["city_name"]; ?></td>
        <td><?php echo $row["email"]; ?></td>
        <td><a href="delete-process.php?userid=<?php echo $row["userid"];
?>">Delete</a></td>
        </tr>
        <?php
        $i++;
        }
        ?>
</table>
</body>
</html>
```

delete-process.php

```
<?php
include_once 'database.php';
$sql = "DELETE FROM employee WHERE userid='" . $_GET["userid"] . "'";
if (mysqli_query($conn, $sql)) {
    echo "Record deleted successfully";
} else {
    echo "Error deleting record: " . mysqli_error($conn);
}
mysqli_close($conn);
?>
```