Module 4: Server Side Programming (PHP)

PHP stands for Hypertext Preprocessor (PHP). It is a server side scripting language that is used to create dynamic webpages. It runs on a Web Server that is designed to take input in the form of PHP code and display output in the form of webpage content.

- Features of PHP -

① Simple - PHP is very simple and easy to use scripting language. The rules and regulation in PHP are simple to understand and follow.

② Interpreted - PHP code is interpreted, there is no need of a compiler.

③ Faster - The execution of PHP code is very fast as compared to other scripting languages like ASP and JSP.

④ OpenSource - PHP is open source and easily available.

⑤ Platform Independent - once created, PHP script can be executed on any machine irrespective of its platform.

⑥ Efficiency - PHP is efficient.

⑦ Flexibility - PHP is very flexible language. It can be easily embedded with HTML. It can also integrate with other scripting language like JavaScript or VBScript.

Syntax of PHP:

PHP code starts and ends with ?

```
<? php

echo "Hello";
?>
```

Datatypes in PHP -

PHP Data Types.

- Scalar Types
  - Boolean
  - Integer
  - Float
  - String.
- Compound Types
  - Array
  - Object
- Special Types.
  - Resource
  - Null.

In PHP it is not necessary to specify the data types of a variable. It automatically decides the data type according to the value assigned to a variable.

A) Scalar Data types -

① Boolean - Represents a true or false value. is_bool() fun is used to check if value is Boolean or not.

② Integer - An integer data type is a non-decimal number is_int() or is_integer() function checks whether a value is an integer.

③ Float - Represents real numbers that include decimal plac is_float() or is_real() function checks whether a value is floating point or not.

① String - A string is sequence of characters, i.e. string() function checks whether a value is string or not. In PHP string can be represented using single or double quotes.

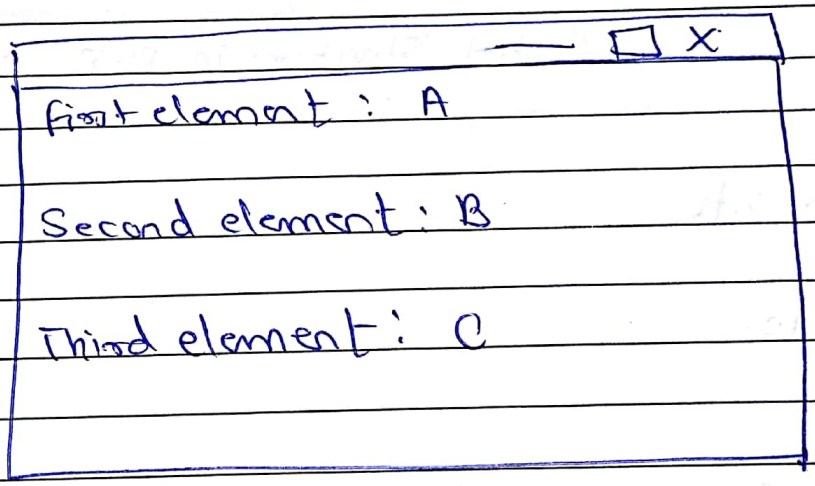
for eg - $name = "Prachi";
echo "Hi" $name"; // Hi Prachi
echo 'Hi $name'; Hi $name.

B) Compound Data types -

① Array - Array is a group of elements having same data type.

eg - <?php>
$demoArray = array ('A', 'B', 'C');
echo "First element : $demoArray[0]
";
echo "Second element : $demoArray[1]
";
echo "Third element : $demoArray[2]
";
?>

Output



② Object - Objects are defined as instances of user defined classes that can hold both values and functions.

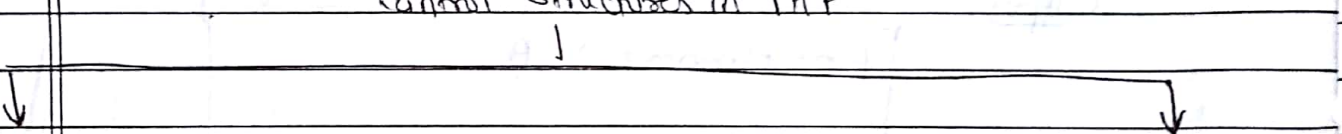
c) Special Types - There are two special data types in PHP -

① Resource - A resource is a special variable which has a reference to an external resource which allows us to connect to other applications. A resource variable has special function or handlers to open the file and database connection.

Eg - $resource = database_connect();

② NULL - The NULL value is used to represent a variable without value. A variable is assumed to be null if it has been assigned the constant NULL or it has not set to any value.

Control structures in PHP -



Conditional Statements
- if-else
- else if ladder
- switch case.

Loop statements
- while
- do-while
- for
- for-each.

Eg-1
```
<? php
$x = 1;
if ($x == 1)
echo '$x is equal to 1';
?>
```

Eg. if -
```
<?php
$a = 10;
$b = 5;
if ($a > $b)
echo "$a is greatr than $b";
else
echo " $b is greater than $a";
```

- For each loop - The foreach loop allows you to iterate over elements in an array.

```
foreach (array as value)
{
statement;
}
```

Eg -
```
<?php
$name = array ('Prachi', 'Priya', 'Kiran');
foreach ($name as $value)
echo " My name is:" .$value. "<br>";
?>
```

Built-in functions in PHP -

PHP provides various types of built-in functions -

① String functions -

① PHP strtolower() function - The strtolower() function converts the string into lowercase format and return it.

```
<?php
$string1 = "PRACHI"
$string1 = strtolower($string1);
```

```
echo $string // output - prachi
?>
```

② PHP strtoupper() - Converts the string to uppercase.

```
<?php
$str = "prachi";
$str = strtoupper($str);
echo $str; // output - PRACHI
?>
```

③ PHP ucfirst() - Converts first character of string in uppercase and then returns the string.

```
<?php
$str = "prachi";
$str = ucfirst($str); // Prachi
echo $str;
?>
```

④ lcfirst() - Converts first character of string is lowercase and then return the string.

```
<?php
$str = "Prachi";
$str = lcfirst($str);
echo $str; // prachi
?>
```

⑤ PHP strrev() - strrev() function reverses the string and return it.

```
<?php
$str = "Prachi";
$str = strrev($str);
```

```
echo $str; // output - charP
?>
```

⑥ PHP strlen() funct^n - strlen() funct^n counts the length of string and returns it.

```
<?php
$str = "Pradi";
$len = strlen($str);
echo "Length of string : ".$len; // Length of string : 6
?>
```

⑦ PHP strpos() funct^n - This funct^n searches substring in the main string. If a substring is found, the funct^n return position of character of the first match. If not found, it will return false.

```
<?php
echo strpos("Hello friends!", "friends"); // 6
?>
```

⑧ PHP str_replace() funct^n - It replaces the existing characters with new given characters in a string.

```
<?php
echo str_replace("friends", "world", "Hello friends");
?>                                  // Hello world
```

⑨ PHP strcmp() funct^n - It compares two strings. It returns 0 if two strings are equal. It is case sensitive.

```
<?php
echo strcmp("Hello", "Hello"); // 0
?>
```

② Numeric / Math functions -

abs() - Returns the absolute value of a number.
acos() - Returns the cosine of a number
asin() - Returns the sine of a number
ceil() - Rounds a number up to the nearest integer
exp() - Calculates exponent of e.
floor() - Rounds the number down to the nearest integer
fmod() - Returns the remainder of x/y.
max() - finds highest value
min() - finds minimum value.
pow() - calculates x^y value.
sqrt() - Returns the square root of a number.

③ Array functions -

① array() to create an array.
eg - $cars = array("Volvo", "BMW", "Toyota");
$student = array("rollno"=>"101", "name"=>"piyu");

② count() function - It is used to count length of array.

```
<?php
$cars = array("Volvo", "BMW", "Honda");
echo count($cars);  //3
?>
```

③ extract() fnctn - It converts the array into variables
The array keys become variable names and array elements become the values of variables.

④ compact() func - It works opposite to extract(). It creates a group of variables to an array.

⑤ is_array() - This function is used to check whether a particular variable is an array or not. It returns true if the variable is an array or else returns false.

⑥ shuffle() - It shuffles the array elements means change their sequence randomly.

⑦ Sorting arrays -
sort() - It sorts the array elements in ascending order.
rsort() - It sorts the array elements in descending order.

Date/Time functions:

~~⑧ PHP checkdate() func~~

date() - formats a local date and time.
getdate() - Returns the current date/time.
gettimeofday() - Returns current time.
date_diff() - Returns the difference btw two dates.
date_create() - Returns a new DateTime object.
localtime() - Returns local time.
mktime() - Returns unix timestamp for a date.
time() - Returns the current time as a unix timestamp.

BUILDING WEB APPLICATIONS USING PHP

Tracking Users: Cookie and Sessions

A) Cookies: Cookies are small text files which are used to store the session data. This data is related to user which is used to track different actions performed by the particular user. PHP supports HTTP cookies. There are three steps in tracking user through cookies:

1) Initially, the script running on the server sends a set of cookies to the client side browser. This information may be like name, age, id etc.

2) Browser accepts the cookies and stores it on local machine for further processing.

3) Next time whenever, the browser sends any request to the server, it sends the cookie data with request which is used by the server to identify the user.

- Setting Cookies with PHP

In PHP, setcookie() method is used to set a cookie. This function must be called before the <html> tag. It is called separately for each cookie.

```
setcookie(name, value ,expire, path, domain, security);
```

name- The name for cookie is set to this attribute. This data is stored in a variable called HTTP_COOKIE_VARS which is environment variable and used when cookie is accessed.

value- the value of cookie is set to this attribute. This is the actual session data.

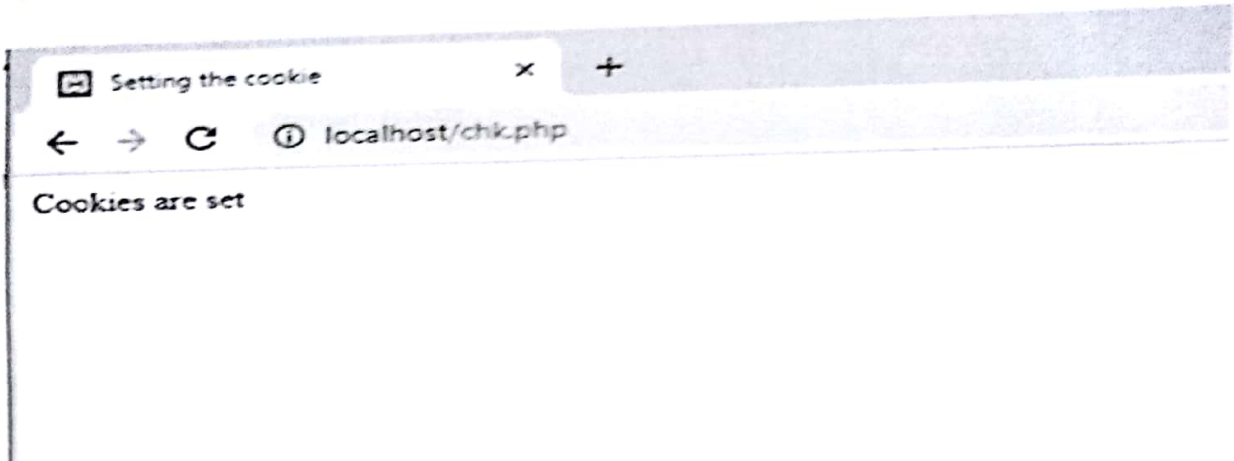
expire- cookies cannot be stored permanently. this attribute is used to set expiry time of cookie. If this attribute is not set then cookie is destroyed after the browser window is closed.

path- This is the URL of directories for which the cookie will be valid.

domain- This attribute is used to set to 1, it indicates that the cookie should be sent by secure transmission only using HTTPS. The value 0 indicates that regular HTTP send the cookie.

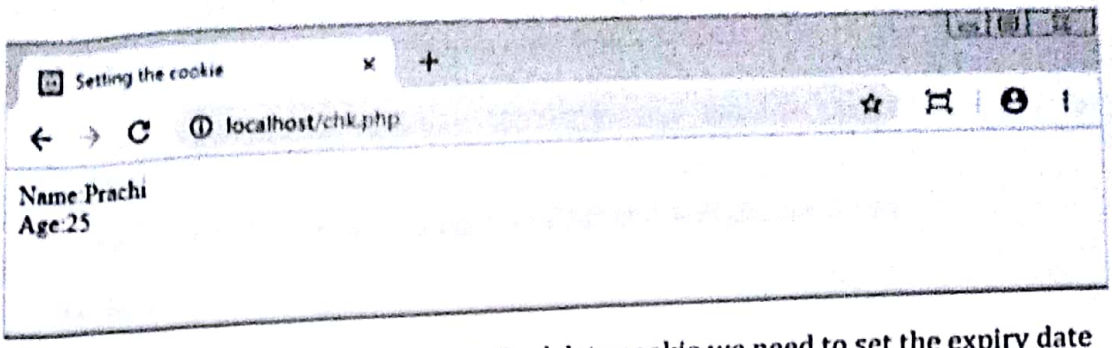
Write a code to set cookie: We will create two cookies name and age. The expiry date for cookie is set to one hour.

```
<?php
setcookie("name","Prachi",time()+3600,"/","",0);
setcookie("age","25",time()+3600,"/","",0);
?>
<html>
<head>
<title>Setting the cookie</title>
</head>
<body>
<?php echo "Cookies are set" ?>
</body>
</html>
```



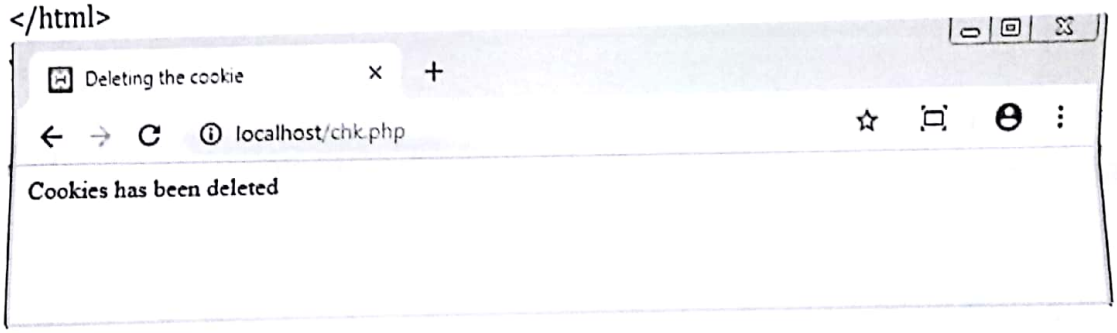
Write a code to access cookies in PHP: Variable $_COOKIE is used to access cookie.

```
<html>
<head>
<title>Setting the cookie</title>
</head>
<body>
<?php
$nm=$_COOKIE["name"];
$ag=$_COOKIE["age"];
echo "Name:".$nm;
echo "<br>Age:".$ag;
?>
</body>
</html>
```



Write a code to delete cookie in PHP: To delete cookie we need to set the expiry date for it. We can also use setcookie() with only name argument without setting value for it.

```
<?php
setcookie("name","",time()-120,"/","",0);
setcookie("age","2",time()-120,"/","",0);
?>
<html>
<head>
<title>Deleting the cookie</title>
</head>
<body>
<?php echo "Cookies has been deleted" ?>
</body>
</html>
```



B) Sessions: Cookies can be used to store the user data, but it has some issues regarding security. Cookies are stored on user's machine, anyone can modify the cookie content and insert harmful data to create problems in applications. Also with each browser request, the cookie is encapsulated with the request every time so it affects the performance of website. Both the above issues can be solved by using PHP session. Importantly the data of session is stored on the server rather the user's system.

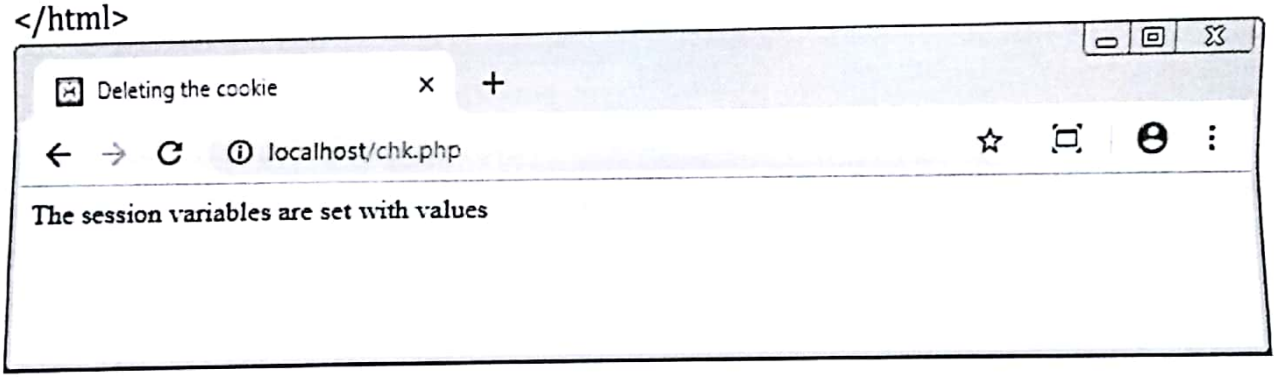
- Session variables store the user information which is to be used across multiple pages.

- The session variables stores data regarding a single user and are accessible to all pages in one application.
- Each session has a session ID to identify the user. SID links the particular user to his related data on server.

Write a PHP code to start a session and set the session variables

The session_start() begins the session and should be at beginning.

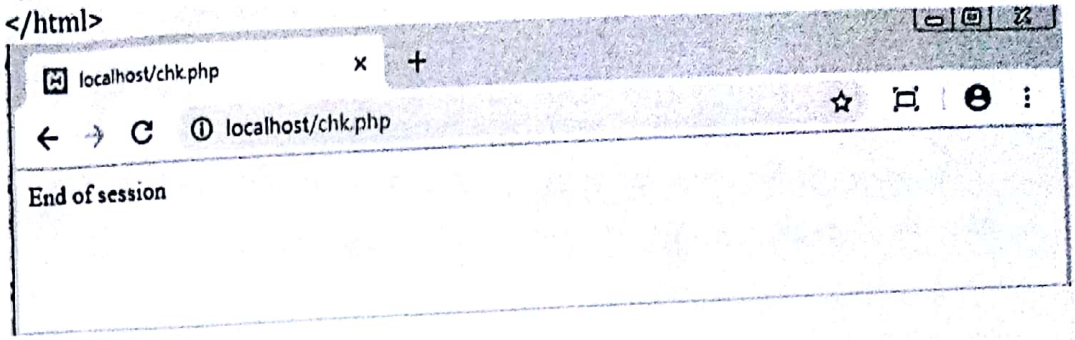
```
<?php
session_start();
?>
<html>
<head>
<title>Deleting the cookie</title>
</head>
<body>
<?php
$SESSION["uname"]="Prachi";
$SESSION["fcolor"]="Red";
echo "The session variables are set with values"
?>
</body>
</html>
```



Write a PHP code to destroy a session.

```
session_start();
?>
<html>
<head>
</head>
<body>
<?php
session_unset();
session_destroy();
echo "End of session";
```

```
?>
</body>
</html>
```



PHP and MySQL database connectivity with example

With PHP, you can connect to and manipulate databases.
MySQL is the most popular database system used with PHP

MySQL is a database system used on the web

MySQL is a database system that runs on a server

MySQL is ideal for both small and large applications

MySQL is very fast, reliable, and easy to use

MySQL uses standard SQL

MySQL compiles on a number of platforms

MySQL is free to download and use

MySQL is developed, distributed, and supported by Oracle Corporation

PHP 5 and later can work with a MySQL database using:

- **MySQLi extension** (the "i" stands for improved)
- **PDO (PHP Data Objects)**

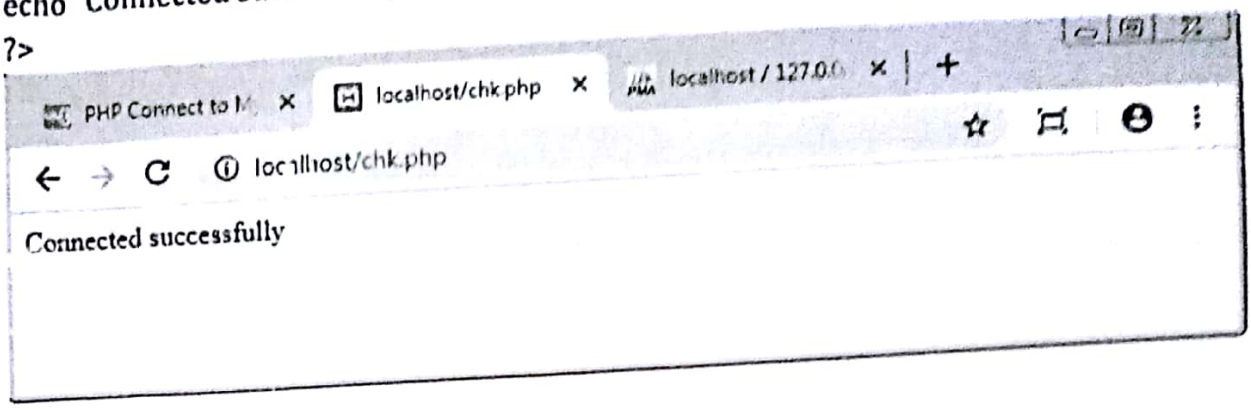
Earlier versions of PHP used the MySQL extension. However, this extension was deprecated in 2012.

- **Connection to server:**

PHP provides mysql_connect() to open a database connection.

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

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

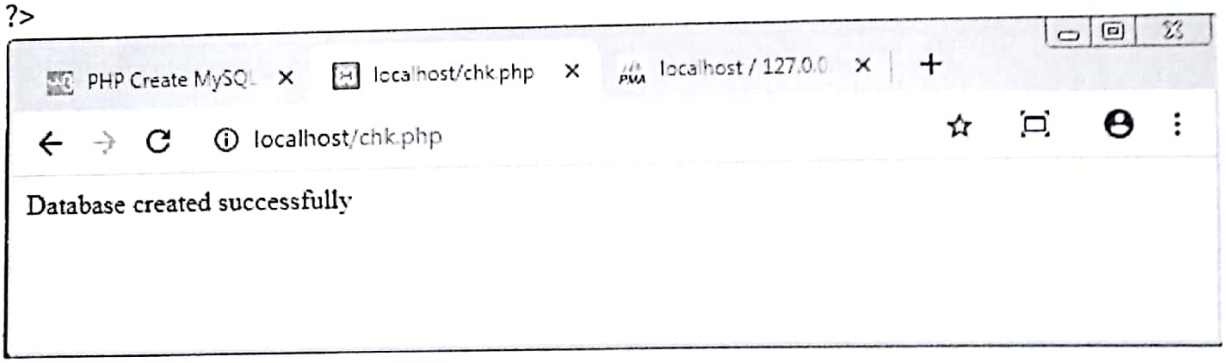


- **Creating database**

PHP provides mysqli_query() method to create a database.

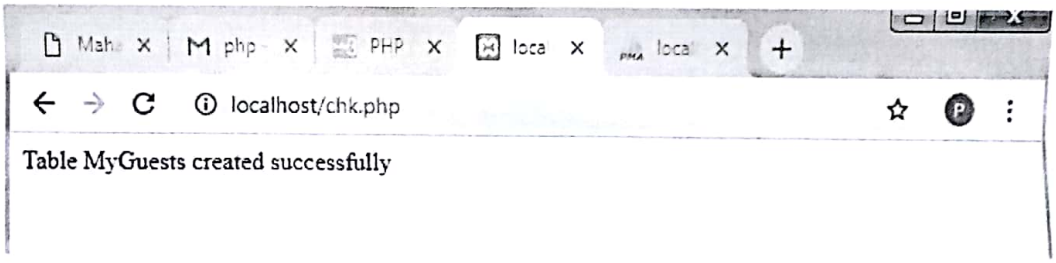
```
<?php
// Create connection
$conn = mysqli_connect("localhost","root","");
// Check connection
if (!$conn) {
   die("Connection failed: " . mysqli_connect_error());
}

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



- **Create MySQL table**

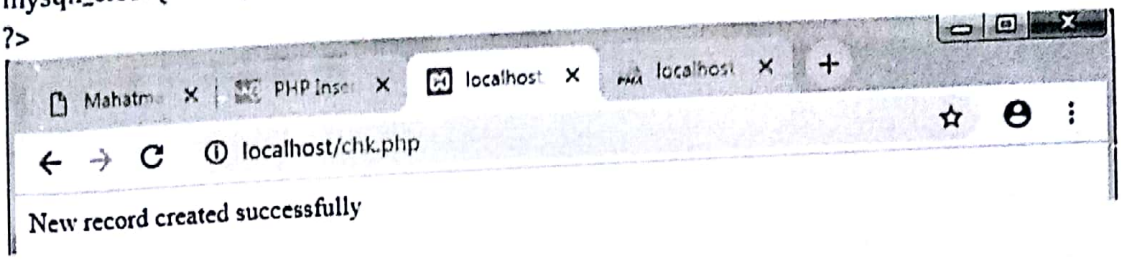
```
<?php
// Create connection
$conn = mysqli_connect("localhost","root","","mydb");
// Check connection
if (!$conn) {
   die("Connection failed: " . mysqli_connect_error());
}
// sql to create table
$sql = "CREATE TABLE MyGuests (
id INT(6) UNSIGNED AUTO_INCREMENT PRIMARY KEY,
firstname VARCHAR(30) NOT NULL,
lastname VARCHAR(30) NOT NULL,
email VARCHAR(50),
reg_date TIMESTAMP
)";
if (mysqli_query($conn, $sql)) {
   echo "Table MyGuests created successfully";
} else {
   echo "Error creating table: " . mysqli_error($conn);
}
mysqli_close($conn);
?>
```



- **Insert data into MySQL table**

```
<?php
// Create connection
$conn = mysqli_connect("localhost","root","","mydb");
// Check connection
if (!$conn) {
   die("Connection failed: " . mysqli_connect_error());
}
$sql = "INSERT INTO MyGuests (firstname, lastname, email)
VALUES ('John', 'Doe', 'john@example.com')";
if (mysqli_query($conn, $sql)) {
```

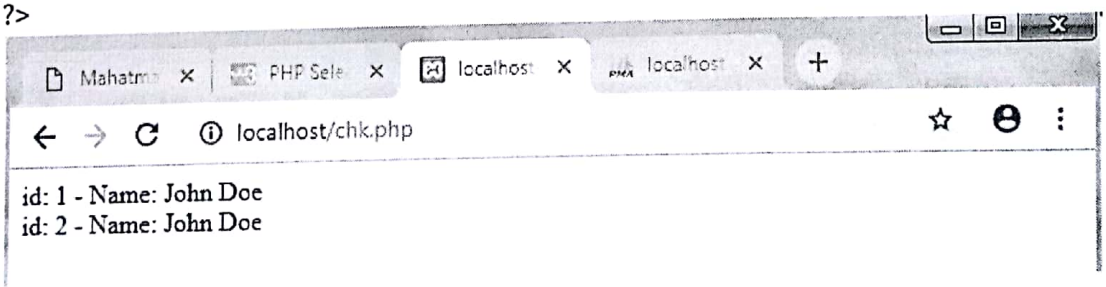
```
  echo "New record created successfully";
} else {
  echo "Error: " . $sql . "<br>" . mysqli_error($conn);
}
mysqli_close($conn);
?>
```



- **Select data from MySQL database**

```
<?php
// Create connection
$conn = mysqli_connect("localhost","root","","mydb");
// Check connection
if (!$conn) {
  die("Connection failed: " . mysqli_connect_error());
}
$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"]. " " . $row["lastname"].
"<br>";
  }
} else {
  echo "0 results";
}
mysqli_close($conn);
?>
```

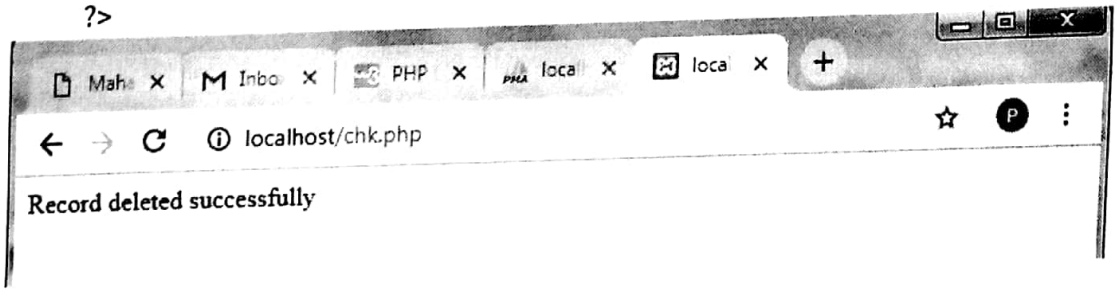


- **Delete data from MySQl**

```
<?php
// Create connection
$conn = mysqli_connect("localhost","root","","mydb");
// Check connection
if (!$conn) {
  die("Connection failed: " . mysqli_connect_error());
}

// sql to delete a record
$sql = "DELETE FROM MyGuests WHERE id=2";

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



- **Update data in MySQL**

```
<?php
// Create connection
$conn = mysqli_connect("localhost","root","","mydb");
// Check connection
if (!$conn) {
  die("Connection failed: " . mysqli_connect_error());
}

$sql = "UPDATE MyGuests SET lastname='Bravo' WHERE id=1";

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

```
mysqli_close($conn);
?>
```