Web Programming Using PHP and MYSQL

UNIT – I

Fundamentals of PHP: About PHP - Tools needed for PHP: Preparing a PHP workstation - PHP Introduction: Integrating PHP with HTML - Writing and Testing PHP - PHP Basics - Parts of PHP: Types of Information - Variables and Constants – Operators - Statements and Expressions - Functions.

UNIT – I

1.1 Fundamentals of PHP: About PHP

- ✓ "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.

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

1.2 Tools needed for PHP:

Need tools for

- ✓ **PHP Script writing** with code assistance
- ✓ **PHP Script Testing** on your development computer
- ✓ **Development support** in a browser

Choosing Code Editor

The PHP language code represented just by plain text that can be edited with simple text editors like:

- ✓ Gedit (opensource code editor for Linux)
- ✓ Notepad++ (opensource code editor for Windows)
- ✓ Apple textEditor (Apple's basic text editor)
- ✓ Visual Studio Code

Choosing Web Server

To test and develop PHP projects we need a web server which can interpret PHP code and send the output to our browsers.

Open source options include:

- ✓ LAMP (AMP stack on Linux)
- ✓ XAMPP (Cross platform AMP stack implementation)
- ✓ WAMP (AMP stack for Windows

1.3 Preparing a PHP workstation

- ✓ Install web server- setting up web server
- ✓ After completing the installation of web server open getting started link
- ✓ Web server component that provides the local host in your browser
- ✓ Test the local host in a browser by opening a browser and typing local host in address bar
- ✓ Browse and run your program

1.4 Integrating PHP with HTML

Example 1 : **Assume we have a standard footer file called "footer.php", that looks like this:**

```
<?php
echo "<p>Copyright &copy; 1999-" . date("Y") . " W3Schools.com</p>";
?>
```

To include the footer file in a page, use the include statement:

Example

```
<html>
<body>
```

<h1>Welcome to my home page!</h1>

Some text.

Some more text.

<?php include 'footer.php';?>

</body>

</html>

1.5 Writing and Testing PHP

- ✓ Writing a PHP Program in text editor like notepad or Dream viewer
- ✓ To do testing install webserver
- ✓ Create and store php
- ✓ Run the program in browser

1.6 PHP Basics

- ✓ PHP script is executed on the server, and the plain HTML result is sent back to the brower.

Basic PHP Syntax

A PHP script can be placed anywhere in the document.

A PHP script starts with <?php and ends with ?>:

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

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

Sample program

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

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

PHP Case Sensitivity

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

In the example below, all three echo statements below are equal and legal:

Example

```
<?php
ECHO "Hello World!<br>";
echo "Hello World!<br>";
EcHo "Hello World!<br>";
?>
```

Comments in PHP

A comment in PHP code is a line that is not executed as a part of the program.

Single-line comments:

```
<?php
// This is a single-line comment
# This is also a single-line comment
?>
```

Multiple-line comments:

```
<?php
/*
This is a multiple-lines comment block
that spans over multiple
lines
*/
?>
```

Hello World" Script in 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)

```
<html>
    <head>
    <title>Hello World</title>
  </head>
    <body>
    <?php echo "WELCOME";
?>
  </body>
</html>
```

1.7 Parts of PHP: Types of Information

- ✓ PHP is not case sensitive
- ✓ In PHP there is no need to specify data type such as integer or string
- ✓ PHP determine the data type based on the value

Datatypes

PHP supports the following data types:

- String
- Integer
- Float (floating point numbers - also called double)
- Boolean

PHP String

- ✓ A string is a sequence of characters, like "Hello world!".

- ✓ A string can be any text inside quotes. You can use single or double quotes:

Example

```
<?php
$x = "Hello world!";
$y = 'Hello world!';
echo $x;
echo "<br>";
echo $y;
?>
```

PHP Integer

An integer data type is a non-decimal number between -2,147,483,648 and 2,147,483,647.

Rules for integers:

- ✓ An integer must have at least one digit
- ✓ An integer must not have a decimal point
- ✓ An integer can be either positive or negative

Example

```
<?php
$x = 5985;
var_dump($x);
?>
```

PHP Float

A float (floating point number) is a number with a decimal point or a number in exponential form.

Example

```
<?php
$x = 10.365;
var_dump($x);
?>
```

PHP Boolean

A Boolean represents two possible states: TRUE or FALSE.

Example

```
$x = true;
$y = false;
```

1.8 Variables and Constants

PHP Variables: Variables are "containers" for storing information.

Creating (Declaring) PHP Variables

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

Example

```
<?php
$txt = "Hello world!";
$x = 5;
$y = 10.5;
?>
```

OUTPUT

```
Hello world!
5
10.5
```

Rules for PHP variables:

- ✓ A variable starts with the $ sign, followed by the name of the variable
- ✓ 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)

Output Variables

The PHP echo statement is often used to output data to the screen.

Example

```
<?php
$txt = "W3Schools.com";
echo "I love $txt!";
?>
```

OUTPUT:

I love W3Schools.com!

PHP Constants

- ✓ A constant is an identifier (name) for a simple value. The value cannot be changed during the script.
- ✓ A valid constant name starts with a letter or underscore (no $ sign before the constant name).

Create a PHP Constant

To create a constant, use the define() function.

Syntax
define(*name, value, case-insensitive*)

Parameters:

- ✓ name: Specifies the name of the constant
- ✓ value: Specifies the value of the constant
- ✓ Case-insensitive: Specifies whether the constant name should be case-insensitive. Default is false

Example

Create a constant with a **case-sensitive** name:

```
<?php
define("GREETING", "Welcome to W3Schools.com!");
echo GREETING;
?>
```

Welcome to W3Schools.com!

1.9 PHP Operators

Operators are used to perform operations on variables and values.

PHP divides the operators in the following groups:

- Arithmetic operators
- Assignment operators
- Comparison operators
- Increment/Decrement operators
- Logical operators
- String operators
- Array operators
- Conditional assignment operators

PHP Arithmetic Operators

The PHP arithmetic operators are used with numeric values to perform common arithmetical operations, such as addition, subtraction, multiplication etc.

```
<!DOCTYPE html>
<html>
<body>
<?php
$x = 10;
$y = 6;
echo $x + $y;
echo $x - $y;
echo $x * $y;
echo $x / $y;
echo $x % $y;
echo $x ** $y;
?>
</body>
</html>
```

OUTPUT

16

4

60

1.6666666666667

4

1000

PHP Assignment Operators

The PHP assignment operators are used with numeric values to write a value to a variable.

Example 1

```
<?php
$x = 100;
$y = "100";
var_dump($x == $y); // returns true because values are equal
var_dump($x <> $y); // returns false because values are equal
?>
```

Output

Bool(true)

Bool(false)

Example 1

```
<?php
$x = 100;
$y = 50;
var_dump($x > $y); // returns true because $x is greater than $y
var_dump($x < $y); // returns true because $x is less than $y
?>
```

Output

Bool(true)

Bool(false)

PHP Comparison Operators

The PHP comparison operators are used to compare two values (number or string)

Example	Name	Result
$a == $b	Equal	**true** if *$a* is equal to *$b* after type juggling.
$a <> $b	Not equal	**true** if *$a* is not equal to *$b* after type juggling.
$a < $b	Less than	**true** if *$a* is strictly less than *$b*.
$a > $b	Greater than	**true** if *$a* is strictly greater than *$b*.
$a <= $b	Less than or equal to	**true** if *$a* is less than or equal to *$b*.
$a >= $b	Greater than or equal to	**true** if *$a* is greater than or equal to *$b*.

PHP Increment / Decrement Operators

- ✓ PHP supports C-style pre- and post-increment and decrement operators.

- ✓ The PHP increment operators are used to increment a variable's value.
- ✓ The PHP decrement operators are used to decrement a variable's value.

Example	Name	Effect
++$a	Pre-increment	Increments *$a* by one, then returns *$a*.
$a++	Post-increment	Returns *$a*, then increments *$a* by one.
--$a	Pre-decrement	Decrements *$a* by one, then returns *$a*.
$a--	Post-decrement	Returns *$a*, then decrements *$a* by one

EXAMPLE

```
<?php
$x = 10;
echo ++$x;
echo $x++;
```

```
echo --$x;
echo $x--;
?>
```

OUTPUT

11

10

9

10

PHP Logical Operators

The PHP logical operators are used to combine conditional statements.

Example	Name	Result
$a and $b	And	true if both $a and $b are true.
$a or $b	Or	true if either $a or $b is true.
$a xor $b	Xor	true if either $a or $b is true, but not both.
! $a	Not	true if $a is not true.
$a && $b	And	true if both $a and $b are true.
$a || $b	Or	true if either $a or $b is true.

PHP String Operators

There are two string operators.

The first is the concatenation operator ('.'), which returns the concatenation of its right and left arguments.

The second is the concatenating assignment operator ('.='), which appends the argument on the right side to the argument on the left side.

```
<?php
$a = "Hello ";
$b = $a . "World!"; // now $b contains "Hello World!"
$a = "Hello ";
```

```
$a .= "World!";     // now $a contains "Hello World!"
?>
```

1.10 Statements and Expressions

- ✓ PHP Statements contain either comments or statements
- ✓ A Statement is anything between opening and closing tags
- ✓ Statement is single line end with semicolon
- ✓ Statement contains expressions
- ✓ Expressions are anything that has value or evaluates to a value
- ✓ Values are assigned to a variable

1.11 Functions

- ✓ A function is a block of statements that can be used repeatedly in a program.
- ✓ A function will not execute automatically when a page loads.
- ✓ A function will be executed by a call to the function.
- ✓ A user-defined function declaration starts with the word function:

Syntax

```
function functionName ( )
{
        code to be executed;
}
```

Example

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

OUTPUT

Hello world !

Array functions

Create an array

The array() function is used to create an array:

```
array();
```

In PHP, there are three types of arrays:

- **Indexed arrays** - Arrays with a numeric index
- **Associative arrays** - Arrays with named keys
- **Multidimensional arrays** - Arrays containing one or more arrays

Get The Length of an Array - The count() Function

The count() function is used to return the length (the number of elements) of an array:

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

Output

3

Date/Time functions

PHP date() Function--Formats a local date and time

```
<?php
// Prints the day
echo date("l") . "<br>";
// Prints the day, date, month, year, time, AM or PM
echo date("l jS \of F Y h:i:s A") . "<br>";

?>
```

Output

Tuesday

Tuesday 23rd of November 2021 08:15:25 AM

PHP getdate() Function

-Returns date/time information of a timestamp or the current local date/time

Example

```
<?php
print_r(getdate());
?>
```

Output

Tuesday, November 23, 2021

PHP time() Function

Return the current time as a Unix timestamp, then format it to a date:

```
<?php
$t=time();
echo($t . "<br>");
echo(date("Y-m-d",$t));
?>
```

Output

```
1637655433
2021-11-23
```

PHP Math Functions

PHP abs() Function

The abs() function returns the absolute (positive) value of a number.

```
 <?php
echo(abs(6.7) . "<br>");
echo(abs(-6.7) . "<br>");
echo(abs(-3) . "<br>");
echo(abs(3));
?>
```

Output

```
6.7
6.7
3
3
```

PHP rand() Function

The rand() function generates a random integer.

```
<?php
echo(rand() . "<br>");
echo(rand() . "<br>");
echo(rand(10,100));
?>
```

Output

```
512549293
1132363175
79
```

PHP round() Function

The round() function rounds a floating-point number.

```
<?php
echo(round(0.60) . "<br>");
echo(round(0.50) . "<br>");
echo(round(0.49) . "<br>");
echo(round(-4.40) . "<br>");
echo(round(-4.60));
?>
```

Output

1

1

0

-4

-5

PHP String Functions

PHP strrev() Function

The strrev() function reverses a string.

```
<?php
echo strrev("Hello World!");
?>
```

Output

!dlroW olleH

PHP substr() Function

The substr() function returns a part of a string.

```
<?php
echo substr("Hello world",6);
?>
```

World

PHP strlen() Function

The strlen() function returns the length of a string.

```
<?php
echo strlen("Hello");
?>
```

Output

5

PHP strcmp() Function

The strcmp() function compares two strings.

```
<?php
echo strcmp("Hello world!","Hello world!");
?>
```

Output

0

If this function returns 0, the two strings are equal.

UNIT – II

PHP Control and File Handling: Control Structures: if/else statements - while and do - while statements - for and for each statement -switch statements.

PHP File and Directory Management: Basic File Functions - Additional File Functions .

Cookies and Session and Server Variables: Session Variables-Cookies - Server Variables

2.1 PHP Control and File Handling:Control Structures

PHP - The if Statement

The if statement executes some code if one condition is true.

Syntax

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

Example

```
<!DOCTYPE html>

<html>

<body>

<?php

$X = 5;

if ($X < "200") {

echo "Have a good day!";

}

?>

</body>

</html>
```

OUTPUT:

Have a good day!

PHP - The if...else Statement

The if...else statement executes some code if a condition is true and another code if that condition is false.

Syntax

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

Example

```
<!DOCTYPE html>

<html>

<body>

<?php

$X = 7;

if ($X < "01") {

 echo "Have a good day!";

} else {

  echo "Have a good night!";

}

?>

</body>

</html>
```

OUTPUT:

Have a good night!

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

The if...elseif...else statement executes different codes for more than two conditions.

Syntax

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

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

Example

```
<!DOCTYPE html>

<html>

<body>

<?php

$X = 200;

echo "<p>The hour (of the server) is " . $X;

echo ", and will give the following message:</p>";

if ($X < "10") {

 echo "Have a good morning!";

} elseif ($X < "201") {

  echo "Have a good day!";

} else {

  echo "Have a good night!";

}

?>

</body>

</html>
```

OUTPUT:

The hour (of the server) is 200, and will give the following message:

Have a good day!

The PHP switch Statement

Use the switch statement to select one of many blocks of code to be executed**.**

Syntax

```
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;
}
```

Example

```
<!DOCTYPE html>

<html>

<body>

<?php

$favcolor = "green";

switch ($favcolor) {

  case "red":

    echo "Your favorite color is red!";

    break;

  case "blue":

    echo "Your favorite color is blue!";
```

```
    break;
  case "green":
    echo "Your favorite color is green!";
    break;
  default:
    echo "Your favorite color is neither red, blue, nor green!";
}
?>
</body>
</html>
```

OUTPUT:

Your favorite color is green!

The PHP while Loop

The while loop executes a block of code as long as the specified condition is true.

Syntax

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

Example

```
<!DOCTYPE html>
<html>
<body>
<?php
$x = 1;
 while($x <= 5) {
  echo "The number is: $x <br>";
  $x++;
```

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

OUTPUT:

The number is: 1

The number is: 2

The number is: 3

The number is: 4

The number is: 5

The PHP do...while Loop

The do...while loop will always execute the block of code once, it will then check the condition, and repeat the loop while the specified condition is true.

Syntax

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

Example

```
<!DOCTYPE html>
<html>
<body>
<?php
$x = 10;
do {
  echo "The number is: $x <br>";
  $x++;
} while ($x <= 20);
?>
```

</body>

</html>

OUTPUT:

The number is: 10

The number is: 11

The number is: 12

The number is: 13

The number is: 14

The number is: 15

The number is: 16

The number is: 17

The number is: 18

The number is: 19

The number is: 20

The PHP for Loop

The for loop is used when you know in advance how many times the script should run.

Syntax

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

Example

```
<!DOCTYPE html>
<html>
<body>
<?php
for ($x = 0; $x <= 3; $x++) {
  echo "The number is: $x <br>";
}
?>
```

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

OUTPUT:

The number is: 0

The number is: 1

The number is: 2

The number is: 3

The PHP foreach Loop

The foreach loop works only on arrays, and is used to loop through each key/value pair in an array.

Syntax

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

Example

```
<!DOCTYPE html>
<html>
<body>
<?php
$colors = array("red", "green", "blue", "yellow");
foreach ($colors as $value) {
  echo "$value <br>";
}
?>
</body>
</html>
```

OUTPUT:

red

green

blue

yellow

2.2.PHP File and Directory Management:

- ✓ Benefits of server side programming is that you can read and write information on storage devices.
- ✓ Predefined Function allow you to check file attributes like copy, move delete etc.

Basic File Functions

- ✓ Fopen() function establish a file connection
- ✓ The fwrite() function write a data string to an opened file
- ✓ The fread() function reads from an open file.
- ✓ The fclose() function is used to close an open file.

The file may be opened in one of the following modes:

Modes	**Description**
r	Open a file for read only.
w	Open a file for write only.
a	Open a file for write only.
x	Creates a new file for write only.
r+	Open a file for read/write.
w+	Open a file for read/write.
a+	Open a file for read/write.
x+	Creates a new file for read/write.

Example

```
<?php
$myfile = fopen("webdictionary.txt", "r") or die("Unable to open file!");
echo fread($myfile,filesize("webdictionary.txt"));
fclose($myfile);
?>
```

Additional File Functions .

- ✓ The fgets() function is used to read a single line from a file.
- ✓ The feof() function checks if the "end-of-file" (EOF) has been reached.
- ✓ The fgetc() function is used to read a single character from a file.

Example

```
<?php
$myfile = fopen("webdictionary.txt", "r") or die("Unable to open file!");
// Output one character until end-of-file
while(!feof($myfile)) {
  echo fgetc($myfile);
}
fclose($myfile);
?>
```

2.3 Cookies and Session and Server Variables:

- ✓ Cookies and Sessions are **used to store information**.
- ✓ Cookies are only stored on the client-side machine, while sessions get stored on the client as well as a server.
- ✓ A session creates a file in a temporary directory on the server where registered session variables and their values are stored.

Set Cookie in PHP

The **setcookie( )** function is used to set a cookie.

Syntax

setcookie(cookiename, value, expire, path, domain);

Example

```
<?php
Setcookie(“cookie-name”,”abhi”,time()60*60);
?>
```

In the example above cookie_name variale creates and assign value inside variable is "abhi" which work for 1 hour.

PHP get Cookie

The **$_COOKIE[ ]** super global variable is used to retrieve a cookie value.

<?php

Echo “welcome”.$_COOKIE[“cookie_name”];

?>

Output Welcome abhi

Set Multiple Cookie and get all Cookie

```
<?php

//set cookie
setcookie("user", $_POST['n'] , time()+60*60);

setcookie("age", $_POST['a'], time()+60*60);

setcookie("profile", $_POST['prof'], time()+60*60);

?>

<html>

<body>

<form method="post">

 Enter your name <input type="text" name="n"/><hr/>

 Enter your age <input type="text" name="a"/><hr/>

Enter your profile <input type="text" name="prof"/><hr/>

 <input type="submit" value="SET COOKIE"/&gt

</form>

</body>

</html>
```

Output :

Enter your name abhi

Enter your age 25

Enter your profile developer

SET COOKIE

Retrieve all cookies

```
<?php

print_r($_COOKIE);

?>
```

Output Array ([user] => abhi [age] => 25 [profile] => developer)

PHP delete Cookie

When deleting a cookie you should assure that the expiration date is in the past.

```
<?php

// set the expiration date to one hour ago
setcookie("user", "abhi", time()-60*60);

?>
```

Session Variables

A session is a way to store information (in variables) to be used across multiple pages.

Unlike a cookie, the information is not stored on the users

Creating Session Variables

The **session_start( )** function is used to **start session environment**.

```
<?php
```

```
session_start();

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

Storing a Session Variable

To store and retrieve session variables use the **$_SESSION [ ]**. **save it stroreSession.php**

```
<?php

//first start session environment

session_start();

// store data in session variable through user

$_SESSION['user']= $_POST['un'];

$_SESSION['profile']= $_POST['prof'];

?>

<html>

<body>

        <form method="post">

                Enter your user name <input type="text" name="un"/><hr/>

                Enter your profile  <input type="text" name="prof"/><hr/>

                <input type="submit" value="Store in session variable"/><

        </form>

 </body>

</html>
```

Output :
check output on retrieveSession.php

Enter your user name abhi

Enter your profile developer

Store in session variable

Retrieving a Session Variable

To retrieve session variables first start session environment. **save it retrieveSession.php**

```
<?php

//first start session environment
 session_start();

// retrieve session data

echo "Welcome ".$_SESSION['user']."<br/>";

echo "Your profile is ".$_SESSION['profile'];

?>
```

Output :

Welocme abhi
Your profile is developer

Destroying a Session Variable

If you want to delete some session data, you can use the **unset( )**, **session_unregister( )** or the **session_destroy( )** function.

Server Variables.

> $_SERVER is a PHP super global variable which holds information about headers, paths, and script locations.

Example

```
<?php
echo $_SERVER['PHP_SELF'];
echo "<br>";
```

```
echo $_SERVER['SERVER_NAME'];
echo "<br>";
echo $_SERVER['HTTP_HOST'];
echo "<br>";
echo $_SERVER['HTTP_REFERER'];
echo "<br>";
echo $_SERVER['HTTP_USER_AGENT'];
echo "<br>";
echo $_SERVER['SCRIPT_NAME'];
?
```

most important elements that can go inside $_SERVER:

Element/Code	**Description**
$_SERVER['PHP_SELF']	Returns the filename of the currently executing script
$_SERVER['SERVER_ADDR']	Returns the IP address of the host server
$_SERVER['SERVER_NAME']	Returns the name of the host server (such as www.w3schools.com)
$_SERVER['SERVER_SOFTWARE']	Returns the server identification string (such as Apache/2.2.24)

UNIT – III

PHP Arrays and Forms : Using Arrays : Creating Arrays - Working with Arrays - Looping through Arrays - Sorting Arrays - Navigating in Arrays -Converting Arrays to and from String - Joining and Splitting Arrays - Joining and Splitting Arrays - Comparing Arrays - Handling Multidimensional Arrays - Building and Handling Forms : Create a Form in HTML – Add Java script to a Form - Accepting and Filing Data in PHP – Introduction to Relational Databases : Databases and Relational Databases : Understanding Databases - Understanding a Relational Database - Using phpMy Admin : Get and Install phpMyAdmin - Opening and Exploring phpMyAdmin - Create and Use a Database in phpMyadmin.

PHP Arrays and Forms :

Using Arrays

- ✓ An array stores multiple values in one single variable.
- ✓ An array is a special variable, which can hold more than one value at a time.
- ✓ If you have a list of items (a list of car names, for example), storing the cars in single variables could look like this:

```
$cars1 = "Volvo";
$cars2 = "BMW";
$cars3 = "Toyota";
```

Creating Arrays

Array can be created either through assignment or array() function .

Example

Four methods of creating a four-element array of cars

Method 1: individual integer assignment

$cars 0] = "Ford";

$cars [1] = "Chevy".

$cars [2] = "Honda";

$cars [3] =x "BMW" ;

Method 2: array () function integér assignment

$cars = array(0 => "Ford", 1 =; "Chevy", 2 <5 =>"Honda",3=> "BMW");

Method 3: array () function-implied integer assignment

$cars = array ("Ford", "Cheyy" "Honda", "BMW) .

Method 4: array () function-associative assignment

$cars = array ("Ed" => "Ford", "Sue" => "Chevy", "Kate" => Honda", "Bob = , "BMW");

Example

```
<?php
$cars = array("Volvo", "BMW", "Toyota");
echo "I like " . $cars[0] . ", " . $cars[1] . " and " . $cars[2] . ".";
?>
```

Output

I like Volvo, BMW and Toyota.

Working with Arrays

PHP allows ou to add replace and delete values in array

PHP array_pop() Function

The array_pop() function deletes the last element of an array.

Syntax

array_pop(*array*)

Example

```
<?php
$a=array("red","green","blue");
array_pop($a);
print_r($a);
?>
```

Output

Array ([0] => red [1] => green)

PHP array_push() Function

The array_push() function inserts one or more elements to the end of an array.

Syntax

array_push(*array*, *value1*, *value2*, ...)

example

```
<?php
$a=array("a"=>"red","b"=>"green");
array_push($a,"blue","yellow");
print_r($a);
?>
```

Output

Array ([a] => red [b] => green [0] => blue [1] => yellow)

PHP array_replace() Function

Replace the values of the first array ($a1) with the values from the second array ($a2):

Example

```
<?php
$a1=array("red","green");
$a2=array("blue","yellow");
print_r(array_replace($a1,$a2));
?>
```

Output

Array ([0] => blue [1] => yellow)

Loop Through an Array

To loop through and print all the values of an associative array, you could use a foreach loop, like this:

```
<?php
$age = array("Peter"=>"35", "Ben"=>"37", "Joe"=>"43");
foreach($age as $x => $x_value) {
  echo "Key=" . $x . ", Value=" . $x_value;
  echo "<br>";
}
?>
```

Sorting Arrays

The elements in an array can be sorted in alphabetical or numerical order, descending or ascending.

Sort Functions For Arrays

- sort() - sort arrays in ascending order
- rsort() - sort arrays in descending order
- asort() - sort associative arrays in ascending order, according to the value
- ksort() - sort associative arrays in ascending order, according to the key
- arsort() - sort associative arrays in descending order, according to the value
- krsort() - sort associative arrays in descending order, according to the key

Sort Array in Ascending Order - sort()

Example

```
<?php
$cars = array("Volvo", "BMW", "Toyota");
sort($cars);
?>
```

Output

```
 BMW
Toyota
Volvo
```

Sort Array in Descending Order - rsort()

Example

```
<?php
$cars = array("Volvo", "BMW", "Toyota");
rsort($cars);
?>
```

Output

```
Volvo
Toyota
BMW
```

Sort Array (Ascending Order), According to Value - asort()

Example

```
<?php
$age = array("Peter"=>"35", "Ben"=>"37", "Joe"=>"43");
asort($age);
?>
```

Output

```
Key=Peter, Value=35
Key=Ben, Value=37
Key=Joe, Value=43
```

Sort Array (Ascending Order), According to Key - ksort()

Example

```
<?php
$age = array("Peter"=>"35", "Ben"=>"37", "Joe"=>"43");
ksort($age);
?>
```

Output

```
Key=Ben, Value=37
Key=Joe, Value=43
Key=Peter, Value=35
```

Navigating in Arrays

- ✓ PHP provides functions to allow you to move one element at a time through an array
- ✓ The current () function returns the value of the current element in an array.
- ✓ end() - moves the internal pointer to, and outputs, the last element in the array
- ✓ next() - moves the internal pointer to, and outputs, the next element in the array
- ✓ prev() - moves the internal pointer to, and outputs, the previous element in the array
- ✓ reset() - moves the internal pointer to the first element of the array
- ✓ each() - returns the current element key and value, and moves the internal pointer forward

Converting Arrays to and from String

PHP Provides two functions to Converting Arrays to String and String to array

- ✓ Implode()
- ✓ Explode()

The implode() function returns a string from the elements of an array.

Syntax

implode(*separator*,*array*)

Example

```
<?php
$arr = array('Hello','World!','Beautiful','Day!');
echo implode(" ",$arr)."<br>";
echo implode("+",$arr)."<br>";
echo implode("-",$arr)."<br>";
```

```
echo implode("X",$arr);
?>
```

Output

Hello World! Beautiful Day!
Hello+World!+Beautiful+Day!
Hello-World!-Beautiful-Day!
HelloXWorld!XBeautifulXDay!

The explode() function breaks a string into an array.

Syntax

explode(*separator,string,limit*)

Example

```
<?php
$str = 'one,two,three,four';

// zero limit
print_r(explode(',',$str,0));

// positive limit
print_r(explode(',',$str,2));

// negative limit
print_r(explode(',',$str,-1));
?>
```

Output

Array ([0] => one,two,three,four)
Array ([0] => one [1] => two,three,four)
Array ([0] => one [1] => two [2] => three)

Joining and Splitting Arrays

PHP provides many functions and array operator to join two or more arrays

UNION $a+$b – joins two arrays based on their unique keys, keeping the original keys

Array_merge() Function

The array_merge() function merges one or more arrays into one array.

Example

Merge two arrays into one array:

```
<?php
$a1=array("red","green");
$a2=array("blue","yellow");
print_r(array_merge($a1,$a2));
?>
```

Output

Array ([0] => red [1] => green [2] => blue [3] => yellow)

array_combine() Function

The array_combine() function creates an array by using the elements from one "keys" array and one "values" array.

Example

```
<!DOCTYPE html>

<html>

<body>

<?php

$fname=array("Peter","Ben","Joe");

$age=array("35","37","43");

$c=array_combine($fname,$age);

print_r($c);

?>

</body>

</html>
```

Output

Array ([Peter] => 35 [Ben] => 37 [Joe] => 43)

PhP provides functions to breakup and re arrange arrays

array_slice() Function

The array_slice() function returns selected parts of an array.

Syntax

array_slice(*array, start, length, preserve*)

Example

```
<?php
$a=array("red","green","blue","yellow","brown");
print_r(array_slice($a,2));
?>
```

Output

Array ([0] => blue [1] => yellow [2] => brown)

array_splice() Function

The array_splice() function removes selected elements from an array and replaces it with new elements. The function also returns an array with the removed elements.

Syntax

array_splice(*array, start, length, array*)

example

```
<?php
$a1=array("a"=>"red","b"=>"green","c"=>"blue","d"=>"yellow");
$a2=array("a"=>"purple","b"=>"orange");
array_splice($a1,0,2,$a2);
print_r($a1);
?>
```

Output

array_slice(*array, start, length, preserve*)

array_chunk() Function

The array_chunk() function splits an array into chunks of new arrays.

Syntax

array_chunk(*array, size, preserve_key*)

Example

```
<?php
$cars=array("Volvo","BMW","Toyota","Honda","Mercedes","Opel");
```

```
print_r(array_chunk($cars,2));
?>
```

Output

Array ([0] => Array ([0] => Volvo [1] => BMW) [1] => Array ([0] => Toyota [1] => Honda) [2] => Array ([0] => Mercedes [1] => Opel))

Comparing Arrays

PHP array_diff() Function

The array_diff() function compares **the values** of two (or more) arrays, and returns the differences.

Syntax

array_diff(*array1, array2, array3, ...*)

Example

```
<?php
$a1=array("a"=>"red","b"=>"green","c"=>"blue","d"=>"yellow");
$a2=array("e"=>"red","f"=>"black","g"=>"purple");
$a3=array("a"=>"red","b"=>"black","h"=>"yellow");

$result=array_diff($a1,$a2,$a3);
print_r($result);
?>
```

Output

Array ([d] => yellow)

array_intersect() Function

The array_intersect() function compares **the values** of two (or more) arrays, and returns the matches.

example

```
<?php
$a1=array("a"=>"red","b"=>"green","c"=>"blue","d"=>"yellow");
$a2=array("e"=>"red","f"=>"green","g"=>"blue");

$result=array_intersect($a1,$a2);
print_r($result);
?>
```

Output

Array ([a] => red [b] => green [c] => blue)

PHP array_diff_assoc() Function

The array_diff_assoc() function compares **the keys and values** of two (or more) arrays, and returns the differences.

Syntax

array_diff_assoc(*array1,array2,array3...*)

Example

```
<?php
$a1=array("a"=>"red","b"=>"green","c"=>"blue","d"=>"yellow");
$a2=array("a"=>"red","b"=>"green","c"=>"blue");

$result=array_diff_assoc($a1,$a2);
print_r($result);
?>
```

Output

Array ([d] => yellow)

Handling Multidimensional Arrays

A multidimensional array is an array containing one or more arrays.

Example

```
<?php
echo $cars[0][0].":  In  stock:  ".$cars[0][1].",  sold:  ".$cars[0][2].".<br>";
echo $cars[1][0].":  In  stock:  ".$cars[1][1].",  sold:  ".$cars[1][2].".<br>";
echo $cars[2][0].":  In  stock:  ".$cars[2][1].",  sold:  ".$cars[2][2].".<br>";
echo $cars[3][0].":  In  stock:  ".$cars[3][1].",  sold:  ".$cars[3][2].".<br>";
?>
```

Building and Handling Forms : Create a Form in HTML

Forms are used to get input from the user and submit it to the web server for processing.

The form is defined using the <form>… </form> tags

A Simple HTML Form

```
<html>
<body>

<form action="welcome.php" method="post">
Name: <input type="text" name="name"><br>
E-mail: <input type="text" name="email"><br>
<input type="submit">
</form>

</body>
</html>
```

Output

Name:

E-mail:

Submit

Form validation

Validation means check the input submitted by the user. There are two types of validation are available in PHP. They are as follows −

Client-Side Validation − Validation is performed on the client machine web browsers.

Server Side Validation − After submitted by data, The data has sent to a server and perform validation checks in server machine.

Some of Validation rules for field

Field	Validation Rules
Name	Should required letters and white-spaces
Email	Should required @ and .
Website	Should required a valid URL
Radio	Must be selectable at least once
Check Box	Must be checkable at least once
Drop Down menu	Must be selectable at least once

Text Fields

Name: <input type="text" name="name">
E-mail: <input type="text" name="email">
Website: <input type="text" name="website">
Comment: <textarea name="comment" rows="5" cols="40"></textarea>

Radio Buttons

Gender:
<input type="radio" name="gender" value="female">Female
<input type="radio" name="gender" value="male">Male
<input type="radio" name="gender" value="other">Other

Introduction to Relational Databases:

Databases and Relational Databases:

- ✓ A database is an organized way of storing information
- ✓ The primary purpose of a database is to store information
- ✓ It can be easily read quickly found and retrieved.

Understanding Databases:

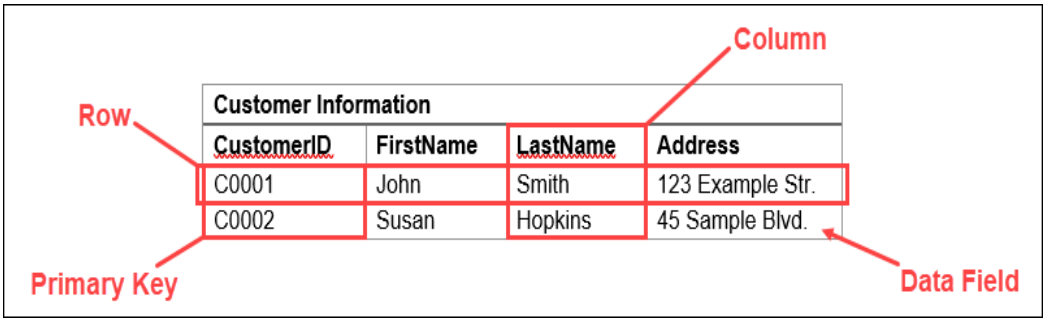
- ✓ A table is a collection of related data entries, and it consists of columns and rows.
- ✓ A column holds specific information about every record in the table.

- ✓ A record (or row) is each individual entry that exists in a table.
- ✓ A database is an organized collection of structured information, or data, typically stored electronically in a computer system.
- ✓ A database is usually controlled by a database management system (DBMS).
- ✓ Most databases use structured query language (SQL) for writing and querying data.
- ✓ Examples: Microsoft SQL Server, Oracle Database, MySQL, PostgreSQL and IBM Db2.

Understanding a Relational Database

- ✓ A relational database defines database relationships in the form of tables.
- ✓ The tables are related to each other - based on data common to each.
- ✓ A relational database is a type of database that stores and provides access to data points that are related to one another.
- ✓ Relational databases are based on the relational model, way of representing data in tables.
- ✓ In a relational database, each row in the table is a record with a unique ID called the key.
- ✓ The columns of the table hold attributes of the data, and each record usually has a value for each attribute, making it easy to establish the relationships among data points.
- ✓ Relational database systems use a model that organizes data into tables of rows (also called records or tuples) and columns (also called attributes or fields).
- ✓ Generally, columns represent categories of data, while rows represent individual instances.

Example



- ✓ These tables can be linked or related using **keys**. Each row in a table is identified using a unique key, called a **primary key.**
- ✓ This primary key can be added to another table, becoming a **foreign key.**

- ✓ The primary/foreign key relationship forms the basis of the way relational databases work.

Using phpMy Admin

- ✓ PHPMYADMIN is a free and open source administration tool for MySQL
- ✓ One of the most popular MySQL administration tools, especially for web hosting services

Get and Install phpMyAdmin:

1. Visit the PhpMyAdmin website and download a version equal to or higher than **4.8.4**.
2. Extract the .zip file to your local machine
3. Rename **config.sample.inc.php** to **config.inc.php**
4. Open **config.inc.php** in your favourite editor. Change the line,

```
1    $cfg['Servers'][$i]['host'] = 'localhost';
```

5. to include your MySQL host, E.G change **localhost** to **mysql.pipeten.co.uk**.
   Note: *If you have a private server with local MySQL your hostname will stay as* **localhost**

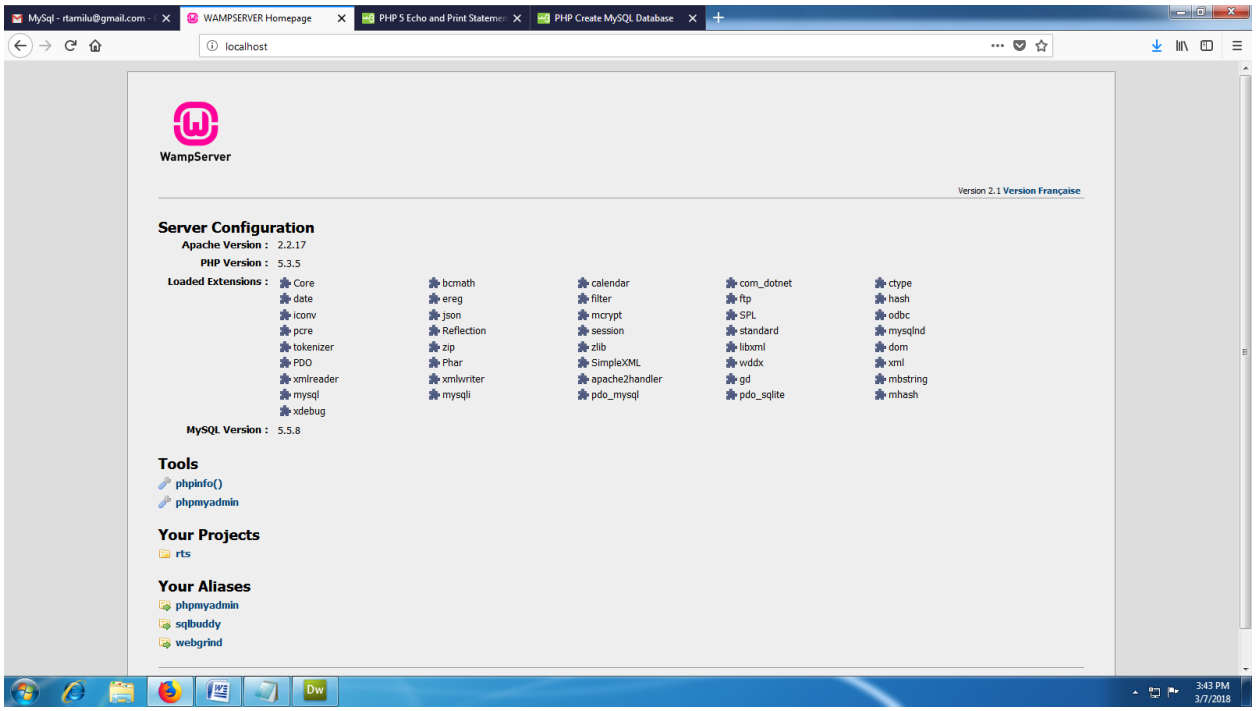
6. While the **config.inc.php** file is still open, you will also need to add what is called a "blowfish Secret" this is used for cookie Authentication. You should create a random string, which may contact numbers, letters and special characters and should be at least 32 characters in length. See the example below.

```
$cfg['blowfish_secret'] = 'aRandomStringofCharacters!ThatisAtLeast32Characters';
```

7. Upload the contents of the folder to your web space.
8. Set the PHP version to PHP 5.6 or above.
9. Visit the URL within your web browser and login using the database user and associated password.
10. That's it, you now have your own PhpMyAdmin instance installed.

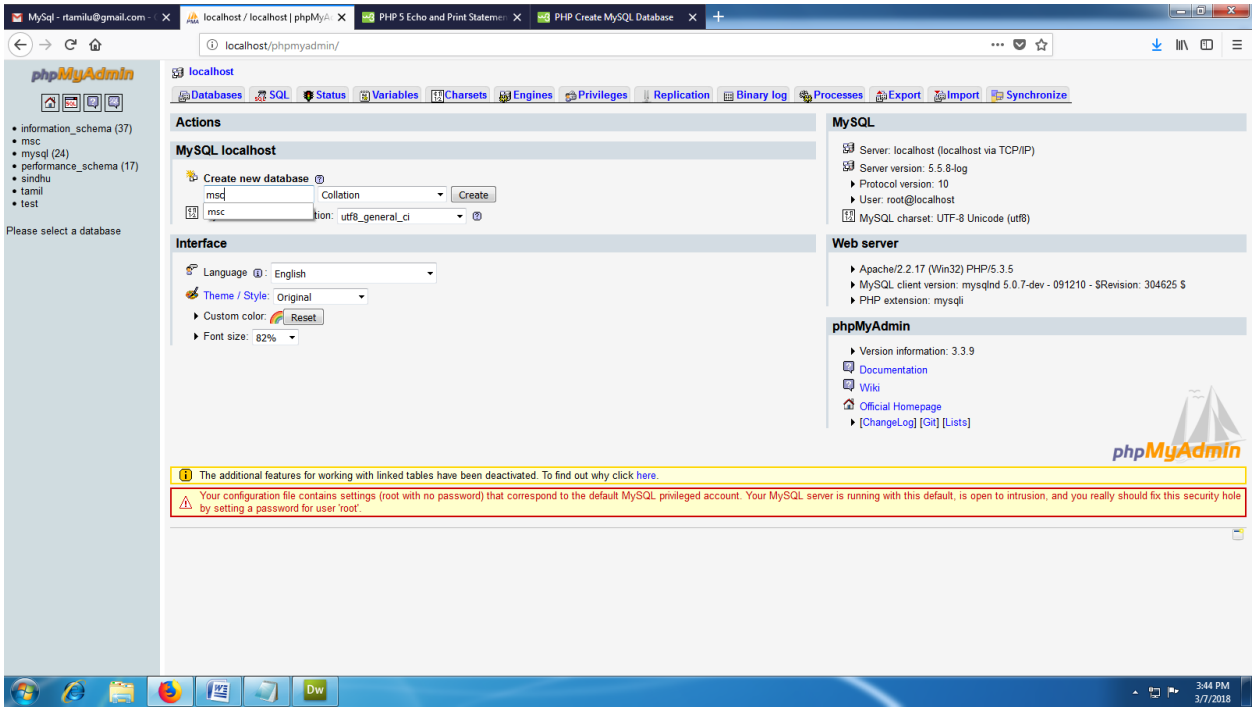
Opening and Exploring phpMyAdmin

- ✓ Wamp Server --- on
- ✓ Go to browser and type localhost

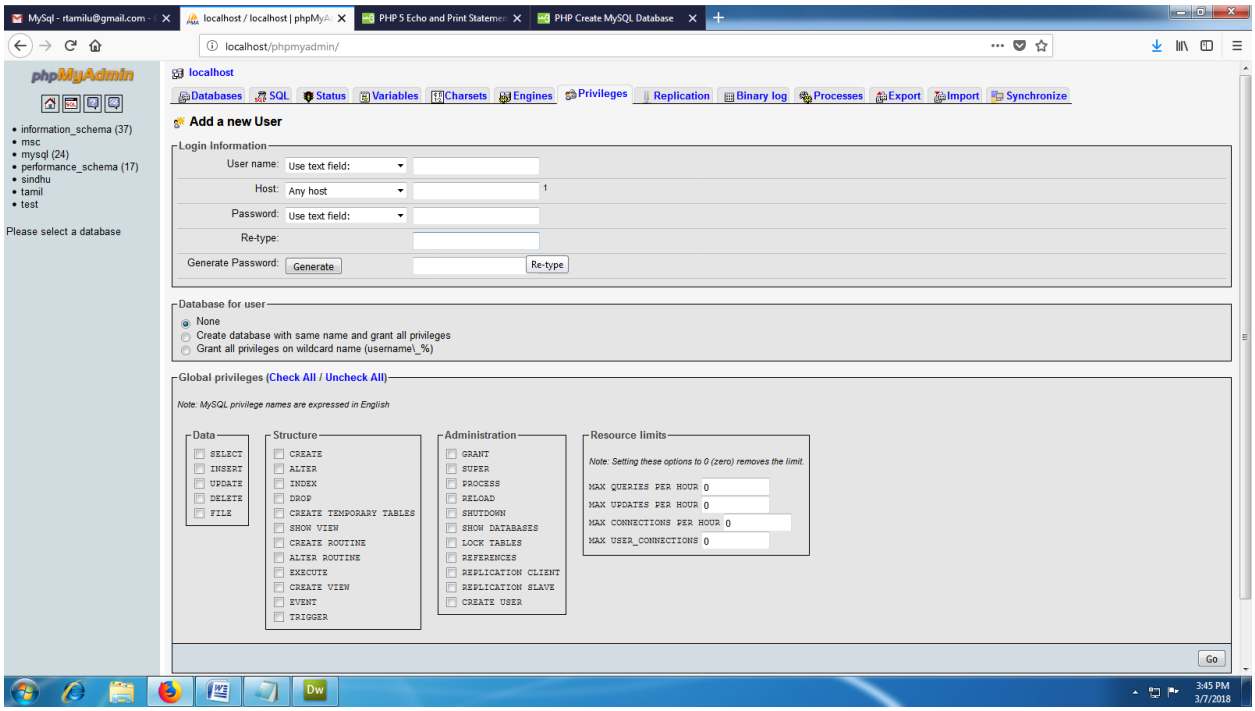


Create and Use a Database in phpMyadmin.

- ✓ Click phpmy admin
- ✓ Create new database name(msc)



- ✓ Click privileges – click add new user(tamil)



Open an Dreamweaver

Establishing and closing a connection

mysql_connect function is used for establishing connection.
first parameter is the server on which mysql is installed. normally its the same server and you can use 'localhost'.
second parameter is mysql database username and third parameter is its password.
return value of this function is a reference link to the connection and need to be passed to all further calls to database.

mysql_close function is used for closing connection.

```
<?php
$link = mysql_connect('localhost', 'mysql_user', 'mysql_password');
if (!$link) {
   die('Could not connect: ' . mysql_error());
}

echo 'Connected successfully';
mysql_close($link);
?>
```

-

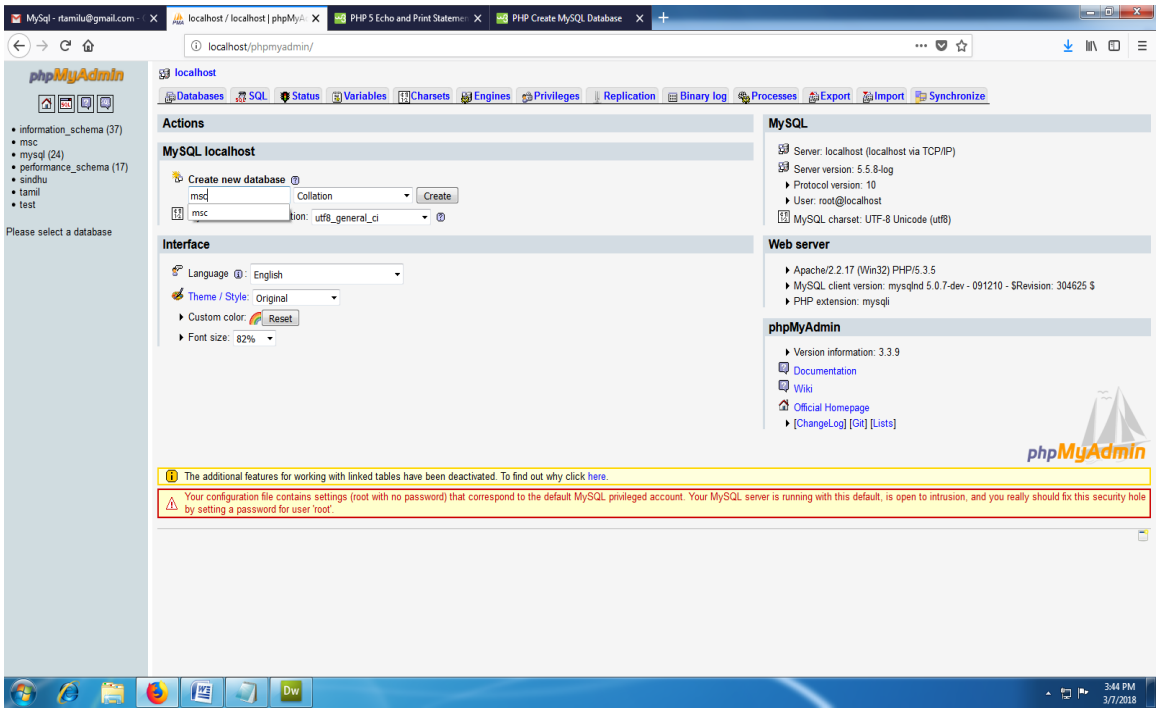
UNIT – IV

Fundamentals of MySQL and SQl : About MySQl

- ✓ MySQL is a relational database management system
- ✓ MySQL is open-source
- ✓ MySQL is free
- ✓ MySQL is ideal for both small and large applications
- ✓ MySQL is very fast, reliable, scalable, and easy to use
- ✓ MySQL is cross-platform
- ✓ MySQL is compliant with the ANSI SQL standard
- ✓ MySQL was first released in 1995
- ✓ MySQL is developed, distributed, and supported by Oracle Corporation
- ✓ MySQL is named after co-founder Monty Widenius's daughter: My

Installing and Using MySQL Workbench

- ✓ The MySQL Workbench is a recent graphic interface distributed and maintained by Oracle that allows users to create, manage, and administer MySQL databases.
- ✓ There are two versions of the Workbench: the free Community Edition that can be downloaded from mysqLcom, and the extended Standard Edition that can be purchased.



Install MySQL Workbench pwr load and install the MySQL Workbench Community Edition with the following steps

- ✓ Open & browser to mysql.com and click the Downloads tab
- ✓ Scroll down to MySQL Community Edition and click Download From MySQL Developer Zone
- ✓ Scroll down to MySQL Workbench and click Download.
- ✓ Enter your Oracle user ID and password or sign up for a new one, go through Oratle's questions and click Download for the third time.Click run
- ✓ Click Next three times and then click Install. Click Yes to allow the installation, and when it is done, click Finish. MySQL Workbench will appear

  Understanding SQl

- ✓ SQL stands for Structured Query Language
- ✓ SQL lets you access and manipulate databases
- ✓ SQL became a standard of the American National Standards Institute (ANSI) in 1986, and of the International Organization for Standardization (ISO) in 1987
- ✓ SQL Statements begin with declarative command such as Create, Delete, Update, , Select

Using MySQL

- ✓ MYSQL Server used in many ways
  1. Directly through the MYSQL Command –line client
  2. Online through the phpMyadmin user interface

Starting the MYSQL Command – Line Cient

- ✓ To start the **mysqld** server from the command line, you should start a console window (or "DOS window") and enter this command:

  C:\> "C:\Program Files\MySQL\MySQL Server 8.0\bin\mysqld"
- ✓ stop the MySQL server by executing this command:

  C:\> "C:\Program Files\MySQL\MySQL Server 8.0\bin\mysqladmin" -u root shutdown

Build a Table on Command line

- ✓ Showing existing databases. Use a SHOW DATABASES statement:

```
mysql> SHOW DATABASES;
```

- ✓ Creating a new database. Use a CREATE DATABASE statement:

```
mysql> CREATE DATABASE pets;
Query OK, 1 row affected (0.01 sec)
```

- ✓ Creating a table inside a database. First, pick the database in which you want to create the table with a USE statement:

```
mysql> USE pets
Database changed
```

- ✓ The USE statement tells MySQL to use pets as the default database for subsequent statements. Next, create a table with a CREATE TABLE statement:

```
CREATE TABLE cats
        (
id          INT unsigned NOT NULL AUTO_INCREMENT, # Unique ID for the record
name        VARCHAR(150) NOT NULL,          # Name of the cat
owner       VARCHAR(150) NOT NULL,          # Owner of the cat
birth       DATE NOT NULL,                  # Birthday of the cat
PRIMARY KEY (id)                            # Make the id the primary key
);
```

Exploring the MySQl Language: Reviewing MySQL word Usage

- ✓ MYSQL has number of commands , keywords , operators and functions
  - ✓ Commands are declarative words

    Create , Select , Update , Delete
  - ✓ Keywords are supporting words for commands

    Where , order by , Group By
  - ✓ Names or identifiers are used for specific elements such as database, tables
  - ✓ Literals are sets of alphabetic and numeric characters used as values in a statement
    - String Values are any combination of alphabetic and numeric characters enclosed in either single (') or double (") quotes.
    - Numeric values are sets of numeric characters that form an integer, decimal or floating point number.
    - Date and time values are either strings or numbers in a context where a date is expected.

- Boolean Values are constant expressed by true , false

Using MySQL Operators

- ✓ Operators are used to specifying a condition in a statement in MySQL.
- ✓ different types of operators used in MySQL.

Arithmetic Operators

- ✓ In MySQL, arithmetic operators are used to perform the arithmetic operations as described below.

Operator	Description	Example
+	Add	SELECT 30 + 20;
-	Subtract	SELECT 30 - 20;
*	Multiply	SELECT 30 * 20;
/	Divide	SELECT 30 / 20;
%	Modulo	SELECT 30 % 20;

Comparison Operators

- ✓ The comparison operators in MySql are used to compare values between operands and return true or false according to the condition specified in the statement.

Operator	Description	Example
=	Equal to	SELECT * FROM Products WHERE Price = 18;
>	Greater than	SELECT * FROM Products WHERE Price > 30;
<	Less than	SELECT * FROM Products WHERE Price < 30;
>=	Greater than or equal to	SELECT * FROM Products WHERE Price >= 30;

<=	Less than or equal to	SELECT * FROM Products WHERE Price <= 30;
<>	Not equal to	SELECT * FROM Products WHERE Price <> 18;

Logical Operators

Operator	Description	Example
AND or &&	Logical AND .Both values must be true for the result to be true	SELECT * FROM Customers WHERE City = "London" AND Country = "UK";
OR or ||	Logical OR .Both values may be true for the result to be true	SELECT * FROM Customers WHERE City = "London" OR Country = "UK";
NOT or !	Negative value. NOT true is FALSE	SELECT * FROM Customers WHERE City NOT LIKE 's%';

Exploring MySQL Functions

Arithmetic Functions

Function	Description	Example	Result
ABS()	Absolute (positive) value of a number.	SELECT ABS(-243.5);	243.5
AVG()	Returns the average value of an expression.	Select AVG(Salary) from Employee;	
CEIL()	Returns the smallest integer value that is bigger than or equal to a number.	SELECT CEIL(25.75);	26
COUNT()	Returns the number of records returned by a select query.	Select COUNT(Ename) from Employee;	
FLOOR()	Returns the largest integer value that is smaller than or equal to a number.	SELECT FLOOR(25.75);	25
MAX()	Returns the maximum value in a set of values.	SELECT MIN(Salary) From Employee	

MIN()	Returns the minimum value in a set of values.	Select MAX(Salary) From Employee)	
RAND()	Returns a random number between 0 (inclusive) and 1 (exclusive).	SELECT RAND(6);	0.656319084257184 7
ROUND(a,b)	Rounds a number to a specified number of decimal places.	SELECT ROUND(135.375, 2);	135.38
SUM()	Calculates the sum of a set of values.	Select Sum(Salary) From Employee;	
TRUNCATE(a,b)	Truncates a number to the specified number of decimal places.	SELECT TRUNCATE(345.156, 0);	345

Comparison Functions

Function	**Description**	**Example**	**Result**
COALESCE	return the first non-NULL arguments, which is very handy for substitution of NULL	SELECT COALESCE(NULL, 0); SELECT COALESCE(NULL, NULL);	0 NULL
GREATEST	return the greatest Value	SELECT GREATEST(10, 20, 30)	30
LEAST	Returns the smallest value	SELECT GREATEST(10, 20, 30)	10
STRCMP()	Compares two strings. • If string1 = string2, this function returns 0 • If string1 < string2, this function returns -1 • If string1 > string2, this function returns 1	SELECT STRCMP("SQL Tutorial", "SQL Tutorial");	0

Date and Time Functions

Function	**Description**	**Example**	**Result**
SYSDATE()	Return the current date and time	SELECT SYSDATE();	2021-09-

			21 06:47:43
ADDDATE()	Function adds a time/date interval to a date and then returns the date.	SELECT ADDDATE("2017-06-15", INTERVAL 10 DAY);	2017-06-25
CURDATE()	returns the current date as YYYYMMDD	SELECT CURDATE();	2021-09-21
DATE_FORMAT()	Returns formats a date as specified.	SELECT DATE_FORMAT("2017-06-15", "%Y");	2017

String Functions

Function	Description	Example	Result
CONCAT()	adds two or more expressions together.	SELECT CONCAT("SQL ", "Tutorial ", "is ", "fun!")	SQL Tutorial is fun!
LCASE	Converts a string to lower-case	SELECT LCASE("SQL Tutorial is FUN!");	sql tutorial is fun!
UCASE	Converts a string to upper-case	SELECT UCASE("SQL Tutorial is FUN!")	SQL TUTORIAL IS FUN!
REVERSE	Reverses a string and returns the result	SELECT REVERSE("SQL Tutorial");	lairotuT LQS

Implementing MySQl Command Statements: MySQL Commands

Data Definition Statements

- DDL changes the structure of the table like creating a table, deleting a table, altering a table, etc.

Here are some commands that come under DDL:

- CREATE

- ALTER
- DROP
- TRUNCATE

Data Manipulation Statements

- DML commands are used to modify the database. It is responsible for all form of changes in the database.
- INSERT
- UPDATE
- DELETE

Other Statements

Data Control Language

DCL commands are used to grant and take back authority from any database user.

- Grant
- Revoke

Transaction Control Language

TCL commands can only use with DML commands like INSERT, DELETE and UPDATE only.

- COMMIT
- ROLLBACK
- SAVEPOINT

Data Query Language

DQL is used to fetch the data from the database.

It uses only one command:

- SELECT

Understanding the MySQL Workbench

MySQL Workbench is a designing or a graphical tool, which is used for working with MySQL servers and databases. ...

The functionalities of MySQL Workbench are as follows:

- ✓ **SQL Development**: This functionality provides the capability to execute SQL queries, create and manage connections to database servers using the built-in SQL Editor.
- ✓ **Data Modeling (Design):** This functionality enables you to create models of your database schema graphically, perform reverse and forward engineer between a schema and a live database, and edit all aspects of your database using the comprehensive Table Editor.
- ✓ **Server Administration**: This functionality enables you to administer MySQL server instances by administering users, performing backup and recovery, inspecting audit data, viewing database health, and monitoring the MySQL server performance.
- ✓ **Data Migration**: This functionality allows you to migrate from Microsoft SQL Server, Microsoft Access, and other RDBMS tables, objects, and data to MySQL.
- ✓ **MySQL Enterprise Support**: This functionality provides support for Enterprise products such as MySQL Enterprise Backup, MySQL Firewall, and MySQL Audit.

Create Databases and Table : Creating and Using a Database

- ✓ CREATE DATABASE is the SQL command used for creating a database in MySQL.

  Syntax

  CREATE DATABASE *databasename*;

  Example

  CREATE DATABASE movies;
- ✓ **Use the command CREATE SCHEMA instead of CREATE DATABASE**

Creating a Table and Its Columns

CREATE TABLE command is used to create tables in a database

Syntax

CREATE TABLE table_name (Column1datatype,column2datatype,column3datatype,…);;

Example

```
CREATE TABLE Persons(PersonID      int,
                     LastName      varchar(255),
                     FirstName     varchar(255),
                     Address       varchar(255),
                     City          varchar(255)
);
```

Reviewing MySQL Data Types

Data types define the nature of the data that can be stored in a particular column of a table

MySQL has **3** main categories of data types namely

1. Numeric,
2. Text
3. Date/time.

Numeric Data types

Numeric data types are used to store numeric values. It is very important to make sure range of your data is between lower and upper boundaries of numeric data types.

TINYINT( )	-128 0 to 255 UNSIGNED.	to	127
SMALLINT( )	-32768 0 to 65535 UNSIGNED.	to	32767
MEDIUMINT( )	-8388608 0 to 16777215 UNSIGNED.	to	8388607
INT( )	-2147483648 0 to 4294967295 UNSIGNED.	to	2147483647
BIGINT( )	-9223372036854775808 0 to 18446744073709551615 UNSIGNED.	to	9223372036854775807
FLOAT	A small approximate number with a floating decimal point.		
DOUBLE(,)	A large number with a floating decimal point.		
DECIMAL(,)	A DOUBLE stored as a string , allowing for a fixed decimal point. Choice for storing currenc		

Text Data Types

As data type category name implies these are used to store text values. Always make sure you length of your textual data do not exceed maximum lengths.

CHAR( )	A fixed section from 0 to 255 characters long.
VARCHAR( )	A variable section from 0 to 255 characters long.

TINYTEXT	A string with a maximum length of 255 characters.
TEXT	A string with a maximum length of 65535 characters.
BLOB	A string with a maximum length of 65535 characters.
MEDIUMTEXT	A string with a maximum length of 16777215 characters.
MEDIUMBLOB	A string with a maximum length of 16777215 characters.
LONGTEXT	A string with a maximum length of 4294967295 characters.
LONGBLOB	A string with a maximum length of 4294967295 characters.

Date / Time

DATE	YYYY-MM-DD
DATETIME	YYYY-MM-DD HH:MM:SS
TIMESTAMP	YYYYMMDDHHMMSS
TIME	HH:MM:SS

Insert Data into Tables : Inserting Data:

The INSERT INTO statement is used to insert new records in a table.

Syntax

It is possible to write the INSERT INTO statement in two ways:

1. Specify both the column names and the values to be inserted:

```
INSERT INTO table_name (column1, column2, column3, ...)
VALUES (value1, value2, value3, ...);
```

2. If you are adding values for all the columns of the table, you do not need to specify the column names in the SQL query

```
INSERT INTO table_name
VALUES (value1, value2, value3, ...);
```

Example

```
INSERT INTO Customers (CustomerName, ContactName, Address, City, PostalCode, Country)
VALUES ('Cardinal', 'Tom B. Erichsen', 'Skagen 21', 'Stavanger', '4006', 'Norway');
```

UNIT – V

Select, Replace, Update, and Delete Data in Tables:

Selecting Data

The SELECT statement is used to select data from a database.

Syntax

SELECT *column1, column2,...*FROM *table_name*;

SELECT Columns Example

SELECT CustomerName, City, Country FROM Customers;

SELECT * Example

SELECT * FROM Customers;

Updating data

The UPDATE statement is used to modify the existing records in a table.

Syntax

UPDATE *table_name*

SET *column1 = value1, column2 = value2,..*

WHERE *condition*;

Example

UPDATE Customers SET ContactName = 'Alfred Schmidt', City = 'Frankfurt' WHERE CustomerID = 1;

Deleting Data

The DELETE statement is used to delete existing records in a table.

Syntax

DELETE FROM *table_name* WHERE *condition*;

Example

DELETE FROM Customers WHERE CustomerName='Alfreds Futterkiste';

Alter, Rename, and Drop Tables and Databases

Altering Tables and Databases :

- ✓ The ALTER TABLE statement is used to add, delete, or modify columns in an existing table.
- ✓ The ALTER TABLE statement is also used to add and drop various constraints on an existing table.

ALTER TABLE - ADD Column

To add a column in a table, use the following syntax:

ALTER TABLE *table_name* ADD *column_name datatype*;

Example

ALTER TABLE Customers ADD Email varchar(255);

ALTER TABLE - DROP COLUMN

To delete a column in a table, use the following syntax (notice that some database systems don't allow deleting a column):

ALTER TABLE *table_name* DROP COLUMN *column_name*;

Example

ALTER TABLE Customers DROP COLUMN Email;

ALTER TABLE - MODIFY COLUMN

To change the data type of a column in a table, use the following syntax:

ALTER TABLE *table_name* MODIFY COLUMN *column_name datatype*;

Renaming Tables

Syntax

The syntax to rename a table in MySQL is:

```
ALTER TABLE table_name
RENAME TO new_table_name;
```

Example

```
ALTER TABLE contacts
 RENAME TO people;
```

Dropping Tables and Databases

The DROP TABLE statement is used to drop an existing table in a database.

Syntax

DROP TABLE *table_name*;

Example

DROP TABLE Shippers;

The DROP DATABASE statement is used to drop an existing SQL database.

Syntax

DROP DATABASE *databasename*;

Example

DROP DATABASE testDB;

CREATE VIEW Statement

A view is a virtual table based on the result-set of an SQL statement.

A view contains rows and columns, just like a real table. The fields in a view are fields from one or more real table

A view is created with the CREATE VIEW statement.

CREATE VIEW Syntax

CREATE VIEW *view_name* AS
SELECT *column1*, *column2*, ...
FROM *table_name*
WHERE *condition*;

Example

```
CREATE VIEW [Brazil Customers] AS
SELECT CustomerName,ContactName
FROM Customers
WHERE Country = 'Brazil';
```

Create a Event

CREATE EVENT requires the EVENT privilege for the schema in which the event is to be created.

The minimum requirements for a valid CREATE EVENT statement are as follows:

- The keywords CREATE EVENT plus an event name, which uniquely identifies the event in a database schema.
- An ON SCHEDULE clause, which determines when and how often the event executes.
- A DO clause, which contains the SQL statement to be executed by an event.

This is an example of a minimal CREATE EVENT statement:

```
CREATE EVENT myevent
  ON SCHEDULE AT CURRENT_TIMESTAMP + INTERVAL 1 HOUR
  DO
    UPDATE myschema.mytable SET mycol = mycol + 1;
```

The previous statement creates an event named myevent. This event executes once—one hour following its creation—by running an SQL statement that increments the value of the myschema.mytable table's mycol column by 1.

Triggers

A trigger in MySQL is a set of SQL statements that reside in a system catalog. **It is a special type of stored procedure that is invoked automatically in response to an event**. Each trigger is associated with a table, which is activated on any DML statement such as **INSERT, UPDATE**, or **DELETE**.

triggers are of two types according to the SQL standard: row-level triggers and statement-level triggers.

Row-Level Trigger: It is a trigger, which is activated for each row by a triggering statement such as insert, update, or delete. For example, if a table has inserted, updated, or deleted multiple rows, the row trigger is fired automatically for each row affected by the insert, update, or delete statement.of Java

Statement-Level Trigger: It is a trigger, which is fired once for each event that occurs on a table regardless of how many rows are inserted, updated, or deleted.

Types of Triggers in MySQL?

We can define the maximum six types of actions or events in the form of triggers:

1. **Before Insert:** It is activated before the insertion of data into the table.
2. **After Insert:** It is activated after the insertion of data into the table.
3. **Before Update:** It is activated before the update of data in the table.
4. **After Update:** It is activated after the update of the data in the table.
5. **Before Delete:** It is activated before the data is removed from the table.
6. **After Delete:** It is activated after the deletion of data from the table.

Using a MySQL Database with PHP : Bringing PHP and MySQL Together :

A good practice when using databases is to set the username, the password and the database name values at the beginning of the script code. If you need to change them later, this way you will be able to perform the change easily.

```
$username="your_username";
$password="your_password";
$database="your_database";
```

You should replace *your_username*, *your_password* and *your_database* with the MySQL username, password and database that will be used by your script.

This will create three variables in PHP that will store the different MySQL connection details.

Next you should connect your PHP script to the database. This can be done with the **mysql_connect** PHP function:

```
$mysqli = new mysqli("localhost", $username, $password, $database);
```

With this line PHP connects to the MySQL database server at localhost with the provided username and password.

After the connection is established you should select the database you wish to use. This should be a database to which your username has access to. To select a database, you can use the following command:

```
$mysqli->select_db($database) or die( "Unable to select database");
```

With the above PHP uses the MySQL connection and with it – selects the database stored in the variable *$database* (in our case it will select the database "your_database"). If the script cannot connect it will stop executing and will show the error message "Unable to select database".

Another important PHP function is:

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

This is a very important function as it closes the connection to the database server.