Function in PHP:

कुनै पनि कार्य गर्नका लागि प्रोग्रामिङ भाषाका आविस्कारकले बनाएका निर्देशनहरुको समूहलाई Fuction भनिन्छ जसले processing भई सके पछि मान return गर्दछ । Function लाई नेपालीमा काम या कार्य भनिन्छ । Programming भाषामा Function भनेको कुनै पनि प्रोग्राम बनाउदा खेरी Programmer ले पहिले नै परिभाषित गरेको या पहिले नै बनाएको एउटा build-in प्रोग्राम हो जसले कुने विशेष कार्य गर्दछ यसले कुनै कार्य सम्पन्न गरि सके पछि एउटा मानलाई फर्काउने गर्दछ ।

जस्तै : fopen(), fread() etc.

PHP मा दुई प्रकारको Function को प्रयोग गरिन्छ । जुन निम्न छन् ।

Types of function:

1. Build- in Function
2. User Define function

Build-in Function:

PHP बनाउँदा programmer ले पहिले नै लेखेको वा बनाएको तथा पहिले नै परिभाषित गरेको formula वा प्रोग्रामलाई build-in function भनिन्छ । Build-in function लाई library function पनि भनिन्छ ।

User Define function: Programmer ले आफ्नो आवश्यकता अनुसार विभिन्न किसिमको कार्यहरुलाई सम्पन्न गर्नको लागि बनाइएको function लाई user-defined function भनिन्छ । यसरी बनाइएको user-defined function लाई आफ्नो आवश्यकता अनुसार विभिन्न ठाउँहरुमा call गर्न सकिन्छ ।

Creating PHP Function:

PHP मा user defined function बनाउन धेरै नै सजिलो छ । PHP मा function बनाउनको लागि function keyword को प्रयोग गरिन्छ । जस्तै तलको उदाहरणमा एउटा user-defined function बनाउन सिकाइएको छ -

Syntax

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

e.g., user-defined function:

```
<?php
/* Defining a PHP function */
function writeMessage()
{
echo" You are really a nice person, Have a nice time!";
}
/* Calling a PHP function */
writeMessage();
?>
```

PHP Function Arguments:

PHP function मा प्रयोग भएको सानो bracket () लाई parenthesis भनिन्छ र parenthesis भित्र भएका values लाई arguments वा parameter पनि भनिन्छ । कुनै पनि function लाई काम गर्नका लागि चाहिने आवश्यक data/value जसलाई प्रयोगकर्ता ले keyboard को सहायताले function लाई दिन्छन भने त्यस्ता values हरुलाई arguments भनिन्छ । एउटा मात्र value दिएको छ भने एउटा argument, दुई या दुईभन्दा बढि values दिएको छ भने त्यसलाई parameter list भनिन्छ जसलाई comma (,) को सहायताले छुट्याइएको हुन्छ ।

Example

```
<?php
function familyName($fname) {
  echo "$fname Adhikari.<br>";
}

familyName("Ram");
```

```
familyName("Krishna");
familyName("Shyam");
familyName("Kamal Bahadur");
familyName("Birkha");
?>
```

तल दिएको उदाहरणमा दुईवटा arguments दिइएको छ : ($fname and $year):

Example

```
<?php
function familyName($fname, $year) {
  echo "$fname Adhikari. Born in $year <br>";
}
familyName("Maya", "1975");
familyName("Padma", "1978");
familyName("Kim Jong", "1983");
?>
```

PHP Default Argument Value

तल दिएको उदाहरणमा default parameter लाई कसरी प्रयोग गर्ने भन्ने बारे सिकाइएको छ । यदि function setHeight() लाई argument बिना call गरियो भने यसले default parameter लाई लिन्छ ।

Example

```
<?php declare(strict_types=1); // strict requirement
function setHeight(int $minheight = 50) {
  echo "The height is : $minheight <br>";
}
setHeight(350);
setHeight(); // will use the default value of 50
setHeight(135);
setHeight(80);
?>
```

PHP Functions - Returning values

कुनै पनि programming भाषामा function कार्यन्वयन भैसके पछि त्यसले कुनै न कुनै मानलाई return गर्नु पर्दछ । Function को प्रयोग गरि value return गर्नको लागि return statement को प्रयोग गरिन्छ ।

Example

```
<?php declare(strict_types=1); // strict requirement
function sum(int $x, int $y) {
  $z = $x + $y;
  return $z;
}
echo "5 + 10 = " . sum(5, 10) . "<br>";
echo "7 + 13 = " . sum(7, 13) . "<br>";
echo "2 + 4 = " . sum(2, 4);
?>
```

PHP Return Type Declarations

PHP 7 ले return statement को लागि Type Declarations लाई supports गर्दछ. function return को लागि type declare गर्दा सेमिकोलन को प्रयोग गरिन्छ colon (:) र opening curly ({)bracket भन्दा अगाडि type लेख्नु पर्दछ ।

तलको उदाहरणमा function को लागि type declare गरिएको छ:

Example

```
<?php declare(strict_types=1); // strict requirement
function addNumbers(float $a, float $b) : float {
  return $a + $b;
}
echo addNumbers(1.2, 5.2);
?>
```

Example

```
<?php declare(strict_types=1); // strict requirement
function addNumbers(float $a, float $b) : int {
  return (int)($a + $b);
}
echo addNumbers(1.2, 5.2);
?>
```

Passing Arguments by Reference

PHP मा प्राय arguments, value द्वारा passed गरिन्छ, जसको मतलब value को copy लाई function मा प्रयोग गरिएको हो र function मा passed गरिएको variable लाई परिवर्तन गर्न सकिदैन ।

जव function argument लाई reference द्वारा passed गरिन्छ , त्यस्तो वेला यदि argument मा परिवर्तन भयो भने passed गरिएको variable मा पनि परिवर्तन आउदछ । Function argument लाई reference मा लैजानको लागि ampersand (&) operator को प्रयोग गरिन्छ ।

Example

Use a pass-by-reference argument to update a variable:

```
<?php
function add_five(&$value) {
  $value += 5;
}
$num = 2;
add_five($num);
echo $num;
?>
```

Array In PHP:

Array एउटा विशेष प्रकारको variable हो जसले एउटै प्रकारका data हरुको समूहलाई एउटै variable मा भण्डारण गरि राख्न सक्दछ त्यसलाई array भनिन्छ । जस्तै उदाहरणको लागि, यदि १०० ओटा नम्बर भण्डारण गर्नु परेमा , 100 variable defining गर्नु भन्दा array variable को प्रयोग गर्नु राम्रो हुन्छ ।

Types of PHP array:

PHP मा जम्मा तीन प्रकारको array को प्रयोग गरिन्छ । जुन निम्न प्रकारको रहेका छन् :

1. Numeric Array
2. Associative Array
3. Multi-dimensional array

Numeric Array

Numeric array मा कुनै पनि नम्बर, string, वा object भण्डारण गर्न सकिन्छ तर तिनीहरुको index लाई number बाट प्रस्तुत गरिन्छ भने त्यस्तो array लाई numeric array भनिन्छ । सामान्यतया array को index जहिले पनि शुन्य बाट सुरु हुन्छ । Numeric array लाई दुई वटा तरिका बाट defining गर्न सकिन्छ ।

1st way:

```
$season=array("summer","winter","spring","autumn");
```

2nd way:

```
$season[0]="summer";
$season[1]="winter";
$season[2]="spring";
$season[3]="autumn";
```

Example

```
<?php
$season=array("summer","winter","spring","autumn");
echo "Season are: $season[0], $season[1], $season[2] and $season[3]";
?>
```

e.g.

```
<?php
$season[0]="summer";
$season[1]="winter";
$season[2]="spring";
$season[3]="autumn";
echo "Season are: $season[0], $season[1], $season[2] and $season[3]";
?>
```

Loop Through an Indexed Array

Loop को प्रयोग गरेर index array मा रहेको सबै मानहरु प्रिन्ट गर्नको लागि for loop को प्रयोग गरिन्छ।

Example

```
<?php
$seanon = array("summer", "winter", "spring","autumn");
$arrlength = count($season);
for($x = 0; $x < $arrlength; $x++) {
  echo $season[$x];
  echo "<br>";
}
?>
```

Associative Array:

Associative array कामको आधारमा हेर्दा numeric array जस्तै मिल्दोजुल्दो छ तर index को आधारमा भने फरक रहेको छ। Associative array ले values assign गर्नको लागि named keys को प्रयोग गर्दछ।

Associative array लाई दुई वटा तरिका बाट निर्माण गर्न सकिन्छ।

1st way:

```
$salary=array("Sonoo"=>"350000","John"=>"450000","Kartik"=>"200000");
```

2nd way:

```
$salary["Sonoo"]="350000";
$salary["John"]="450000";
$salary["Kartik"]="200000";
```

Example

File: arrayassociative1.php

```
<?php
$salary=array("Badri"=>"350000","John"=>"450000","Nim"=>"200000");
```

```
echo "Badri salary: ".$salary["Badri"]."<br/>";
echo "John salary: ".$salary["John"]."<br/>";
echo "Nim salary: ".$salary["Nim"]."<br/>";
?>
```

File: arrayassociative2.php

```
<?php
$salary["Badri"]="350000";
$salary["John"]="450000";
$salary["Nim"]="200000";
echo "Badri salary: ".$salary["Badri"]."<br/>";
echo "John salary: ".$salary["John"]."<br/>";
echo "Nim salary: ".$salary["Nim"]."<br/>";
?>
```

Loop Through an Associative Array

Loop को प्रयोग गरेर associative array मा रहेको सबै मानहरु प्रिन्ट गर्नको लागि foreach loop को प्रयोग गरिन्छ।

Example

```
<?php
$salary=array("Badri"=>"350000","John"=>"450000","Nim"=>"200000");
foreach($salary as $x => $x_value) {
  echo "Key=" . $x . ", Value=" . $x_value;
  echo "<br>";
}
?>
```

Multi-dimensional array:

यदि एउटा array भित्र अन्य एक वा एक भन्दा वढि array लाई समावेश गरिन्छ भने त्यस्तो array लाई multi-dimensional array भनिन्छ। PHP ले दुई, तीन, चार वा अरु बढि level deep को multi-dimensional array लाई supports गर्दछ। कहिले काँही three level deep भन्दा माथिको array लाई समावेश गर्न अफ्ठ्यारो हुने गर्दछ। Multi-dimensional array मा भएको मानहरुलाई access गर्नको लागि multiple index को प्रयोग गरिन्छ। PHP multidimensional array लाई matric को रुपमा प्रस्तुत गरिन्छ जसलाई रो र कोलुमले प्रतिनिधित्व गरिन्छ।

Example:

तलको टेबल अनुसार multidimensional array को एउटा साधारण उदाहरण हेरौ जस्मा ३ ओटा रो र ३ ओटा कोलुम रहेका छन्।

Id	Name	Salary
1	shyam	400000
2	ram	500000
3	hari	300000

File: multiarray.php

```
<?php
$emp = array
(
array(1,"shyam",400000),
array(2,"ram",500000),
```

```
array(3,"hari",300000)
);
for ($row = 0; $row < 3; $row++)
{
for ($col = 0; $col < 3; $col++)
{
echo $emp[$row][$col]." ";
}
echo "<br/>";
}
?>
```

Another E.g.: Two dimensional associative array:

```
<?php
   $marks = array(
      "ram" => array (
         "physics" => 35,
         "maths" => 30,
         "chemistry" => 39
      ),
      "shyam" => array (
         "physics" => 30,
         "maths" => 32,
         "chemistry" => 29
      ),
      "sita" => array (
         "physics" => 31,
         "maths" => 22,
         "chemistry" => 39
      )
   );
/* Accessing multi-dimensional array values */
   echo "Marks for ram in physics : " ;
   echo $marks['ram']['physics'] . "<br />";
   echo "Marks for shyam in maths : ";
   echo $marks['shyam']['maths'] . "<br />";
   echo "Marks for sita in chemistry : " ;
   echo $marks['sita']['chemistry'] . "<br />";
?>
```

PHP Sorting Arrays:

डाटालाई बढ्दो कम वा घट्दो कममा मिलाएर राख्ने प्रकियालाई sorting भनिन्छ। Data लाई मिलाएर राख्नका लागि array ले महत्वपूर्ण भूमिका निर्वाह गर्दछ। PHP मा data sort गर्नको लागि धेरै विधिहरु रहेको छन्। जुन निम्न छन् :

- **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()

तल दिएको उदाहरणमा $cars array elements लाई ascending alphabetical order मा मिलाउदछ ।

Example

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

तल दिएको उदाहरणमा $numbers array elements लाई ascending numerical order मा मिलाउदछ ।

Example

```
<?php
$numbers = array(4, 6, 2, 22, 11);
sort($numbers);
?>
```

Sort Array in Descending Order - rsort()

तल दिएको उदाहरणमा $cars array elements लाई descending alphabetical order मा मिलाउदछ ।

Example

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

तल दिएको उदाहरणमा $numbers array elements लाई descending numerical order मा मिलाउदछ ।

```
<?php
$numbers = array(4, 6, 2, 22, 11);
rsort($numbers);
?>
```

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

तल दिएको उदाहरणमा associative array लाई मान(value) को आधारमा ascending order मा मिलाउदछ ।

Example

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

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

तल दिएको उदाहरणमा associative array लाई key को आधारमा ascending order मा मिलाउदछ ।

Example

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

Sort Array (Descending Order), According to Value - arsort()

तल दिएको उदाहरणमा associative array लाई मान(value) को आधारमा descending order मा मिलाउदछ।

Example

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

Sort Array (Descending Order), According to Key - krsort()

तल दिएको उदाहरणमा associative array लाई key को आधारमा descending order मा मिलाउदछ।

Example

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

#PHP - GET & POST Methods:

Browser client ले web server मा information पठाउनको लागि प्रयोग गरिने दुई वटा methods हरु GET Method र POST Method हुन्। Client को browser बाट सुचनाहरुलाई server सम्म पुऱ्याउनको लागि प्रयोग गरिने method लाई नै GET method र POST method भनिन्छ। GET & POST mothod दुबैले array को निर्माण गर्दछ। जस्तैः (e.g. array(key1 => value1, key2 => value2, key3 => value3, ...)). यो array ले key र मान (value) लाई जोडिमा भण्डारण गरेर राख्दछ। जहाँ key भनेको form control को नाम हो भने values भनेको प्रयोगकर्ताले दिएको डाटा हो। GET & POST method लाई superglobals variable $_GET र $_POST मा लेखिन्छ। superglobals variable लाई जुनसुकै function, class वा file मा पनि प्रयोग गर्न सकिन्छ।

कम्प्युटर को प्रयोग गरि web browser बाट कनै server सम्म सुचनालाई पुऱ्याउन को लागि दुईवटा methods को प्रयोग गरिन्छ।

- **The GET Method**: $_GET method को प्रयोग गरेर कुनै पनि सुचनालाई server मा पठाउँदा प्रयोग गरिएको variable को नाम साथै यसको मान समेत URL मा सबैले देख्न सक्छन्। GET method को प्रयोग गरि सुचनालाई server मा पठाउँदा जम्मा 2000 character लाई मात्र पठाउँन सकिन्छ। GET method को प्रयोग गरि डाटा पठाउँदा सबैले URL को माध्यमबाट हेर्न सक्ने भएकोले यसलाई कुनै संवेदनसिल सुचना जस्तैः password , transaction pin, जस्ता data पठाउँनको लागि प्रयोग गर्नु हुदैन।

Try out following example by putting the source code in test.php script.

```
<?php
   if( $_GET["name"] || $_GET["age"] ) {
      echo "Welcome ". $_GET['name']. "<br />";
      echo "You are ". $_GET['age']. " years old.";
      exit();
   }
?>
<html>
   <body>
      <form action = "<?php $_PHP_SELF ?>" method = "GET">
         Name: <input type = "text" name = "name" />
         Age: <input type = "text" name = "age" />
         <input type = "submit" />
      </form>
   </body>
</html>
```

- **The POST Method**: $_POST method को प्रयोग गरेर कुनै पनि सुचनालाई server मा पठाउँदा HTTP request को माध्यमबाट पुग्ने भएकोले महत्वपूर्ण सुचना सबैले देख्न सक्दैनन्। POST method को प्रयोग गरि सुचनालाई server मा

पठाउँदा जस्तो सुकै ठूलाठूला आकारका फाइल तथा कुनैपनि डाटालाई सुरक्षित साथ पठाउँन सकिन्छ। यसको कुनै सिमा छैन। कुनै पनि डाटा सुरक्षित साथ server मा पठाउँनको लागि POST method को प्रयोग गर्नु राम्रो हुन्छ।

e.g.

```
<html>
<body>
<form method="post" action="<?php echo $_SERVER['PHP_SELF'];?>">
  Name: <input type="text" name="fname">
  <input type="submit">
</form>
<?php
if ($_SERVER["REQUEST_METHOD"] == "POST") {
  // collect value of input field
  $name = $_POST['fname'];
  if (empty($name)) {
    echo "Name is empty";
  } else {
    echo $name;
  }
}
?>
</body>
</html>
```

PHP $_REQUEST

PHP $_REQUEST एउटा super global variable हो जसको प्रयोग HTML form submit गरिसकेपछि फारमका डाटालाई संकलन गर्न प्रयोग गरिन्छ।

E.g.:

```
<form method="post" action="<?php echo $_SERVER['PHP_SELF'];?>">
  Name: <input type="text" name="fname">
  <input type="submit">
</form>
<?php
if ($_SERVER["REQUEST_METHOD"] == "POST") {
  // collect value of input field
  $name = $_REQUEST['fname'];
  if (empty($name)) {
    echo "Name is empty";
  } else {
    echo $name;
  }
}
?>
```

Unit-2: Access MySQL database

phpMyAdmin:

एउटा खुल्ला स्रोतको सफ्टवेयर टुल हो जुन सन् 1990 September 9 मा सर्व प्रथम विकास गरिएको थियो । यसलाई PHP प्रोग्रामिङ भाषामा लखिएको हो । PHP My Admin एउटा third party टुल हो, जसको प्रयोग गरेर डाटाबेसमा भएका टेबल र टेबलमा भएका डाटाहरुलाई व्यवस्थापन गर्न सकिन्छ । php my admin ले MariaDB र MySQLका धेरै जस्तो operation लाई स्विकार गर्दछ । PHP My Admin को मुख्य उद्धेश्य भनेको web को माध्यमबाट MySQL को administration लाई नियन्त्रण गर्नु हो ।यो एउटा MySQL database लाई ब्यवस्थापन गर्नको लागि प्रयोग गरिने बढी लोकप्रिय टुल हो । यस टुलको प्रयोग विशेष गरी MySQL database मा create, update, drop, alter, delete, import, र export जस्ता काम गर्न सनिन्छ । PHP My Admin एउटा GUI मा आधारित वेब टुल हो जसको सहयोग बाट MySQL का सम्पूर्ण database लाई व्यवस्थापन गर्न सकिन्छ ।

Database:

कुनैविशेष उद्देश्यका लागि एक स्थानमा सङ्गृहीत एकै प्रकारका डाटाहरूको सङ्ग्रहलाई Database भनिन्छ । डाटालाई व्यवस्थित गरी त्यसबाट विभिन्न सूचनाहरू प्राप्त गर्न सकिन्छ र त्यसलाई आवश्यकताअनुसार उपयोग पनि गर्न सकिन्छ । एउटै डाटाबेसको उपयोग एकभन्दा बढी उद्देश्यका लागि पनि गर्न सकिन्छ । डाटाबेसमा सूचनाहरूलाई सङ्गठित (Organised) रूपमा राखिन्छ जस्तै: टेलिफोन डाइरेक्टरी, लाइब्रेरीमा मिलाएर राखिएका किताबहरू, डिभिडी पसलमा मिलाएर राखिएका फिल्म र गीतका एल्बमहरू ।

What is SQL? *(SQL भनेको के हो ?)*

SQL को पूरा रुप Structured Query Language हो । यो एउटा प्रोग्रामिङ ल्याङ्ग्वेज हो, जसले रिलेसनल डाटावेसमा रहेका डाटालाई क्रमबद्ध गर्ने, परिवर्तन गर्ने, पुन: प्राप्त गर्ने कार्य गर्दछ । SQL एउटा रिलेसनल डाटावेस सिस्टमको लागि स्तरीय भाषा हो । धेरैजसो रिलेसनल डाटावेस सिस्टमहरु जस्तै : MySQL, MS Access, Oracle, Sybase, Informix, postgres, SQL Server ले SQL लाई स्तरीय भाषाको रुपमा प्रयोग गरेको पाइन्छ ।

Why SQL? *(SQL नै किन ?)*

- यसले प्रयोगकर्तालाई रिलेसनल डाटावेस सिस्टममा डाटाहरु प्रयोग गर्न मद्दत गर्दछ ।
- यसले डाटालाई वर्णन गर्न मद्दत गर्दछ ।
- यसले डाटालाई परिभाषित गर्न तथा उक्त डाटामा हेरफेर गर्न मद्दत गर्दछ ।
- यसले अन्य भाषाहरु जस्तै : SQL modules, libraries & pre-compilers संग मिलाएर राख्न मद्दत गर्दछ । साथै डाटावेस बनाउन, राख्न तथा टेवलहरु बनाउन मद्दत गर्दछ ।
- यसले प्रयोगकर्तालाई डाटावेसको प्रोसिडर तथा फङ्सनहरु हेर्न मद्दत गर्दछ ।
- यसले प्रयोगकर्तालाई कुनैपनि टेवल, प्रोसिडर तथा भ्यूमा अनुमति सेट गर्न मद्दत गर्दछ ।

The CREATE DATABASE command:

डाटाबेस बनाउनको लागि **SQL** मा **CREATE DATABASE COMMAND** को प्रयोग गरिन्छ । Syntax: **`CREATE DATABASE databasename;`**

e.g.: तलको SQL STATEMENT ले TestDB नामको database को निर्माण गर्दछ ।

```
CREATE DATABASE testDB;
```

Create table:

सामान्य टेवल बनाउँदा टेवलको नाम, त्यसमा प्रयोग हुने कोलमको नाम तथा कोलमको डाटा टाइप दिनुपर्ने हुन्छ । SQL मा CREATE TABLE statement नयाँ टेवल बनाउन प्रयोग गरिन्छ ।

Syntax: CREATE TABLE statement को syntax तल दिइएको छ ।

CREATE TABLE table_name(column1 datatype,column2 datatype,column3 datatype, ... columnN datatype,PRIMARY KEY(one or more columns));

SQL CREATE TABLE Example

एउटा टेवल जसको नाम "Persons" र जसमा ५ वटा कोलमहरु : PersonID, LastName, FirstName, Address, and City बनाउनु परेमा निम्न कमाण्ड लेख्नुपर्ने हुन्छ । यस कार्यका लागि CREATE TABLE statement प्रयोग गरिन्छ ।

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

PHP Connect to MySQL:

PHP को प्रयोग गरेर mysql सम्बन्धि कुनै पनि कार्य वा query लाई execute गर्नु भन्दा पहिले PHP लाई MySQL सँग connect गर्नु पर्दछ । PHP लाई mysql सँग connect गर्ने निम्न तरिकाहरु छन् :

1. MySQLi (object-oriented)
2. MySQLi (procedural)
3. PDO

Example (MySQLi Object-Oriented)

```php
<?php
$servername = "localhost";
$username = "username"; // database user name
$password = "password"; // database password
// Create connection
$conn = new mysqli($servername, $username, $password);
// Check connection
if ($conn->connect_error) {
  die("Connection failed: " . $conn->connect_error);
}
echo "Connected successfully";
?>
```

Example (MySQLi Procedural)

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

```
echo "Connected successfully";
?>
```

Example (PDO)

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";
try {
  $conn = new PDO("mysql:host=$servername;dbname=myDB", $username,
$password);
  // set the PDO error mode to exception
  $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
  echo "Connected successfully";
}
catch(PDOException $e) {
  echo "Connection failed: " . $e->getMessage();
}
?>
```

PHP Create a MySQL Database:

Create a MySQL Database Using MySQLi and PDO

CREATE DATABASE statement को प्रयोग गरेर MySQL database बनाउन सकिन्छ। तल दिएको उदाहरणमा "myDB" नामको database बनाउन सिकाइएको छ।

Example (MySQLi Object-oriented)

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";
// Create connection
$conn = new mysqli($servername, $username, $password);
// Check connection
if ($conn->connect_error) {
  die("Connection failed: " . $conn->connect_error);
}
// Create database
$sql = "CREATE DATABASE myDB";
if ($conn->query($sql) === TRUE) {
  echo "Database created successfully";
} else {
  echo "Error creating database: " . $conn->error;
}
$conn->close();
?>
```

Note: जब हामीले mysql मा डाटाबेस बनाइन्छ , mysqli object मा जहिलेपनि सरुमा ुतीन वटा argument लाई राख्नु पर्दछ जुन (servername, username and password) हुन।

Tip: If you have to use a specific port, add an empty string for the database-name argument, like this: new mysqli("localhost", "username", "password", "", port)

Example (MySQLi Procedural)

```php
<?php
$servername = "localhost";
$username = "username";
$password = "password";
// Create connection
$conn = mysqli_connect($servername, $username, $password);
// Check connection
if (!$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);
?>
```

Example (PDO)

```php
<?php
$servername = "localhost";
$username = "username";
$password = "password";
try {
  $conn = new PDO("mysql:host=$servername", $username, $password);
  // set the PDO error mode to exception
  $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
  $sql = "CREATE DATABASE myDBPDO";
  // use exec() because no results are returned
  $conn->exec($sql);
  echo "Database created successfully<br>";
} catch(PDOException $e) {
  echo $sql . "<br>" . $e->getMessage();
}
$conn = null;
?>
```

PHP MySQL Create Table

प्रत्येक डाटाबेस टेबलमा आ:आफ्नै यनिक नाम र कोलुम र रोहरु रहेका हुन्छन् ।

Create a MySQL Table Using MySQLi and PDO

The CREATE TABLE statement is used to create a table in MySQL.

We will create a table named PHP को प्रयोग गरेर mysql मा database बनाउनको लागि create table statement को प्रयोग गरिन्छ । जस्तै तल दिएको उदाहरणमा "MyGuests", नामको टेबलमा पाँचवटा columns: "id", "firstname", "lastname", "email" and "reg_date" आदि राखिएको छ ।

```
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 DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP
)
```

Notes on the table above:

The data type specifies what type of data the column can hold. For a complete After the data type, you can specify other optional attributes for each column:

- NOT NULL - Each row must contain a value for that column, null values are not allowed
- DEFAULT value - Set a default value that is added when no other value is passed
- UNSIGNED - Used for number types, limits the stored data to positive numbers and zero
- AUTO INCREMENT - MySQL automatically increases the value of the field by 1 each time a new record is added
- PRIMARY KEY - Used to uniquely identify the rows in a table. The column with PRIMARY KEY setting is often an ID number, and is often used with AUTO_INCREMENT

Each table should have a primary key column (in this case: the "id" column). Its value must be unique for each record in the table.

The following examples shows how to create the table in PHP:

Example (MySQLi Object-oriented)

```php
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDB";
// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
// Check connection
if ($conn->connect_error) {
  die("Connection failed: " . $conn->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 DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP
)";
if ($conn->query($sql) === TRUE) {
  echo "Table MyGuests created successfully";
} else {
  echo "Error creating table: " . $conn->error;
}
$conn->close();
?>
```

Example (MySQLi Procedural)

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

Example (PDO)

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";
```

```
$dbname = "myDBPDO";
try {
  $conn = new PDO("mysql:host=$servername;dbname=$dbname", $username,
$password);
  // set the PDO error mode to exception
  $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
  // 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 DEFAULT CURRENT_TIMESTAMP ON UPDATE
CURRENT_TIMESTAMP
  )";
  // use exec() because no results are returned
  $conn->exec($sql);
  echo "Table MyGuests created successfully";
} catch(PDOException $e) {
  echo $sql . "<br>" . $e->getMessage();
}
$conn = null;
?>
```

Insert Data Into MySQL Using MySQLi and PDO

Insert into statement को प्रयोग गरेर MySQL table मा नयाँ रेकर्डहरु राख्न सकिन्छ ।

Syntax:

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

Here are some syntax rules to follow:

- The SQL query must be quoted in PHP
- String values inside the SQL query must be quoted
- Numeric values must not be quoted
- The word NULL must not be quoted

The following examples add a new record to the "MyGuests" table:

Example (MySQLi Object-oriented)

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDB";
// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
// Check connection
if ($conn->connect_error) {
  die("Connection failed: " . $conn->connect_error);
```

```
}
$sql = "INSERT INTO MyGuests (firstname, lastname, email)
VALUES ('chakra', 'magar', 'chakramagar@gmail.com')";
if ($conn->query($sql) === TRUE) {
  echo "New record created successfully";
} else {
  echo "Error: " . $sql . "<br>" . $conn->error;
}
$conn->close();
?>
```

Example (MySQLi Procedural)

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDB";
// Create connection
$conn = mysqli_connect($servername, $username, $password, $dbname);
// Check connection
if (!$conn) {
  die("Connection failed: " . mysqli_connect_error());
}
$sql = "INSERT INTO MyGuests (firstname, lastname, email)
VALUES ('chakra', 'magar', 'chakramagar@gmail.com')";
if (mysqli_query($conn, $sql)) {
  echo "New record created successfully";
} else {
  echo "Error: " . $sql . "<br>" . mysqli_error($conn);
}
mysqli_close($conn);
?>
```

Example (PDO)

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDBPDO";
try {
  $conn = new PDO("mysql:host=$servername;dbname=$dbname", $username,
$password);
  // set the PDO error mode to exception
  $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
  $sql = "INSERT INTO MyGuests (firstname, lastname, email)
  VALUES ('chakra', 'magar', 'chakramagar@gmail.com')";
  // use exec() because no results are returned
  $conn->exec($sql);
```

```
  echo "New record created successfully";
} catch(PDOException $e) {
  echo $sql . "<br>" . $e->getMessage();
}
$conn = null;
?>
```

 # PHP MySQL Get Last Inserted ID:

Get ID of The Last Inserted Record

यदि हामीले कुनै पनि AUTO_INCREMETN field भएको टेबलमा INSERT र UPDATE को काम गर्‍यो भने हामिले अन्तिममा INSERT/UPDATE गरेको रेकर्डको ID तुरुन्तै प्राप्त गर्न सकिन्छ ।

तलको टेबल "MyGuests", मा "id" column AUTO_INCREMENT field रहेको छ ।

```
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 DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP
)
```

Example (MySQLi Object-oriented)

```
<?php
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "myDB";
// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
// Check connection
if ($conn->connect_error) {
  die("Connection failed: " . $conn->connect_error);
}
$sql = "INSERT INTO MyGuests (firstname, lastname, email)
VALUES ('chakra', 'magar', 'chakramagar.com')";
if ($conn->query($sql) === TRUE) {
  $last_id = $conn->insert_id;
  echo "New record created successfully. Last inserted ID is: " .
$last_id;
} else {
  echo "Error: " . $sql . "<br>" . $conn->error;
}
$conn->close();
?>
```

Example (MySQLi Procedural)

```php
<?php
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "myDB";

// Create connection
$conn = mysqli_connect($servername, $username, $password, $dbname);
// Check connection
if (!$conn) {
  die("Connection failed: " . mysqli_connect_error());
}
$sql = "INSERT INTO MyGuests (firstname, lastname, email)
VALUES ('chakra', 'magar', 'chakramagar.com')";
if (mysqli_query($conn, $sql)) {
  $last_id = mysqli_insert_id($conn);
  echo "New record created successfully. Last inserted ID is: " .
$last_id;
} else {
  echo "Error: " . $sql . "<br>" . mysqli_error($conn);
}
mysqli_close($conn);
?>
```

Example (PDO)

```php
<?php
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "myDBPDO";

try {
  $conn = new PDO("mysql:host=$servername;dbname=$dbname", $username,
$password);
  // set the PDO error mode to exception
  $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
  $sql = "INSERT INTO MyGuests (firstname, lastname, email)
  VALUES ('chakra', 'magar', 'chakramagar.com')";
  // use exec() because no results are returned
  $conn->exec($sql);
  $last_id = $conn->lastInsertId();
  echo "New record created successfully. Last inserted ID is: " .
$last_id;
} catch(PDOException $e) {
  echo $sql . "<br>" . $e->getMessage();
}
```

```
$conn = null;
?>
```

 # PHP MySQL Insert Multiple Records

Insert Multiple Records Into MySQL Using MySQLi and PDO

Multiple SQL statements लाई `mysqli_multi_query()` function को प्रयोग गरेर execute गर्न सकिन्छ ।

तलको उदाहरण "MyGuests" table मा तीनवटा नयाँ रेकर्डलाई insert गरिएको छ ।

Example (MySQLi Object-oriented)

```
<?php
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "myDB";
// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
// Check connection
if ($conn->connect_error) {
  die("Connection failed: " . $conn->connect_error);
}
$sql = "INSERT INTO MyGuests (firstname, lastname, email)
VALUES ('John', 'Doe', 'john@example.com');";
$sql .= "INSERT INTO MyGuests (firstname, lastname, email)
VALUES ('Mary', 'Moe', 'mary@example.com');";
$sql .= "INSERT INTO MyGuests (firstname, lastname, email)
VALUES ('Julie', 'Dooley', 'julie@example.com')";
if ($conn->multi_query($sql) === TRUE) {
  echo "New records created successfully";
} else {
  echo "Error: " . $sql . "<br>" . $conn->error;
}
$conn->close();
?>
```

Example (MySQLi Procedural)

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

```
$sql = "INSERT INTO MyGuests (firstname, lastname, email)
VALUES ('John', 'Doe', 'john@example.com');";
$sql .= "INSERT INTO MyGuests (firstname, lastname, email)
VALUES ('Mary', 'Moe', 'mary@example.com');";
$sql .= "INSERT INTO MyGuests (firstname, lastname, email)
VALUES ('Julie', 'Dooley', 'julie@example.com')";
if (mysqli_multi_query($conn, $sql)) {
  echo "New records created successfully";
} else {
  echo "Error: " . $sql . "<br>" . mysqli_error($conn);
}
mysqli_close($conn);
?>
```

PDO बाट अलिकति फरक छ ।

Example (PDO)

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDBPDO";
try {
  $conn = new PDO("mysql:host=$servername;dbname=$dbname", $username,
$password);
  // set the PDO error mode to exception
  $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
  // begin the transaction
  $conn->beginTransaction();
  // our SQL statements
  $conn->exec("INSERT INTO MyGuests (firstname, lastname, email)
  VALUES ('John', 'Doe', 'john@example.com')");
  $conn->exec("INSERT INTO MyGuests (firstname, lastname, email)
  VALUES ('Mary', 'Moe', 'mary@example.com')");
  $conn->exec("INSERT INTO MyGuests (firstname, lastname, email)
  VALUES ('Julie', 'Dooley', 'julie@example.com')");
  // commit the transaction
  $conn->commit();
  echo "New records created successfully";
} catch(PDOException $e) {
  // roll back the transaction if something failed
  $conn->rollback();
  echo "Error: " . $e->getMessage();
}
$conn = null;
?>
```

 # PHP MySQL Select Data

Select Data From a MySQL Database:

Table मा भएको रेकर्डहरुलाई देखाउनको लागि SELECT statement को प्रयोग गरिन्छ ।

```
SELECT column_name(s) FROM table_name
```

टेबलमा भएको सबै डाटाहरुलाई देखाउनको लागि * character को प्रयोग गरिन्छ ।

```
SELECT * FROM table_name
```

Select Data With MySQLi

तलको उदाहरणले MyGuests भन्ने table बाट id, firstname and lastname columns को डाटालाई देखाउदछ ।

Example (MySQLi Object-oriented)

```php
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDB";
// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
// Check connection
if ($conn->connect_error) {
  die("Connection failed: " . $conn->connect_error);
}
$sql = "SELECT id, firstname, lastname FROM MyGuests";
$result = $conn->query($sql);
if ($result->num_rows > 0) {
  // output data of each row
  while($row = $result->fetch_assoc()) {
    echo "id: " . $row["id"]. " - Name: " . $row["firstname"]. " " .
$row["lastname"]. "<br>";
  }
} else {
  echo "0 results";
}
$conn->close();
?>
```

Example (MySQLi Procedural)

```php
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDB";
// Create connection
$conn = mysqli_connect($servername, $username, $password, $dbname);
// Check connection
if (!$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);
?>
```

You can also put the result in an HTML table:

Example (MySQLi Object-oriented)

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDB";
// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
// Check connection
if ($conn->connect_error) {
  die("Connection failed: " . $conn->connect_error);
}
$sql = "SELECT id, firstname, lastname FROM MyGuests";
$result = $conn->query($sql);
if ($result->num_rows > 0) {
  echo "<table><tr><th>ID</th><th>Name</th></tr>";
  // output data of each row
  while($row = $result->fetch_assoc()) {
    echo "<tr><td>".$row["id"]."</td><td>".$row["firstname"]."
".$row["lastname"]."</td></tr>";
  }
  echo "</table>";
} else {
  echo "0 results";
}
$conn->close();
?>
```

Select Data With PDO (+ Prepared Statements)

The following example uses prepared statements.
It selects the id, firstname and lastname columns from the MyGuests table and displays it in an HTML table:

Example (PDO)

```
<?php
echo "<table style='border: solid 1px black;'>";
echo "<tr><th>Id</th><th>Firstname</th><th>Lastname</th></tr>";
class TableRows extends RecursiveIteratorIterator {
  function __construct($it) {
    parent::__construct($it, self::LEAVES_ONLY);
  }
  function current() {
    return "<td style='width:150px;border:1px solid black;'>" .
parent::current(). "</td>";
  }
  function beginChildren() {
    echo "<tr>";
  }
  function endChildren() {
    echo "</tr>" . "\n";
  }
}
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDBPDO";
try {
  $conn = new PDO("mysql:host=$servername;dbname=$dbname", $username,
$password);
  $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
  $stmt = $conn->prepare("SELECT id, firstname, lastname FROM
MyGuests");
  $stmt->execute();
  // set the resulting array to associative
  $result = $stmt->setFetchMode(PDO::FETCH_ASSOC);
  foreach(new TableRows(new RecursiveArrayIterator($stmt-
>fetchAll())) as $k=>$v) {
    echo $v;
  }
} catch(PDOException $e) {
  echo "Error: " . $e->getMessage();
}
$conn = null;
echo "</table>";
?>
```

 # Delete Data From a MySQL Table Using MySQLi and PDO:

The DELETE statement is used to delete records from a table: टेबलबाट रेकर्डलाई हटाउनको लागि DELETE statement को प्रयोग गरिन्छ । रेकर्डलाई हटाउदा delete statement सँगै where clause को प्रयोग गरिन्छ ।

```
DELETE FROM table_name
WHERE some_column = some_value
```

WHERE clause ले कुन records लाई हटाउने हो भनि निश्चित गर्ने गर्दछ । यदि WHERE clause राखिएन भने सबै रेकर्डहरु हटेर जान्छ ।

Let's look at the "MyGuests" table:

id	firstname	lastname	email	reg_date
1	Ram	BM	ram@example.com	2014-10-22 14:26:15
2	Mary	Moe	mary@example.com	2014-10-23 10:22:30
3	Julie	Dooley	julie@example.com	2014-10-26 10:48:23

Example (MySQLi Object-oriented)

```php
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDB";
// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
// Check connection
if ($conn->connect_error) {
  die("Connection failed: " . $conn->connect_error);
}
// sql to delete a record
$sql = "DELETE FROM MyGuests WHERE id=3";
if ($conn->query($sql) === TRUE) {
  echo "Record deleted successfully";
} else {
  echo "Error deleting record: " . $conn->error;
}
$conn->close();
?>
```

Example (MySQLi Procedural)

```php
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDB";
// Create connection
```

```php
$conn = mysqli_connect($servername, $username, $password, $dbname);
// Check connection
if (!$conn) {
  die("Connection failed: " . mysqli_connect_error());
}
// sql to delete a record
$sql = "DELETE FROM MyGuests WHERE id=3";
if (mysqli_query($conn, $sql)) {
  echo "Record deleted successfully";
} else {
  echo "Error deleting record: " . mysqli_error($conn);
}
mysqli_close($conn);
?>
```

Example (PDO)

```php
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDBPDO";
try {
  $conn = new PDO("mysql:host=$servername;dbname=$dbname", $username,
$password);
  // set the PDO error mode to exception
  $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
  // sql to delete a record
  $sql = "DELETE FROM MyGuests WHERE id=3";
  // use exec() because no results are returned
  $conn->exec($sql);
  echo "Record deleted successfully";
} catch(PDOException $e) {
  echo $sql . "<br>" . $e->getMessage();
}
$conn = null;
?>
```

Update Data In a MySQL Table Using MySQLi and PDO

टेबलमा भएको डाटालाई सच्याउनको लागि UPDATE statement को प्रयोग गरिन्छ ।

```
UPDATE table_name
SET column1=value, column2=value2,...
WHERE some_column=some_value
```

where clause को प्रयोग गरेर कुन रेकर्डलाई अद्यावधिक गर्ने हो त्यसलाई निश्चित गर्ने गर्दछ । यदि where clause लाई प्रयोग गरिएन भने सबै रेकर्डहरु अद्यावधिक भएर आउछन् ।

Let's look at the "MyGuests" table:

id	firstname	lastname	email	reg_date

1	John	Doe	john@example.com	2014-10-22 14:26:15
2	Mary	Moe	mary@example.com	2014-10-23 10:22:30

The following examples update the record with id=2 in the "MyGuests" table:

Example (MySQLi Object-oriented)

```php
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDB";
// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
// Check connection
if ($conn->connect_error) {
  die("Connection failed: " . $conn->connect_error);
}
$sql = "UPDATE MyGuests SET lastname='Doe' WHERE id=2";
if ($conn->query($sql) === TRUE) {
  echo "Record updated successfully";
} else {
  echo "Error updating record: " . $conn->error;
}
$conn->close();
?>
```

Example (MySQLi Procedural)

```php
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDB";
// Create connection
$conn = mysqli_connect($servername, $username, $password, $dbname);
// Check connection
if (!$conn) {
  die("Connection failed: " . mysqli_connect_error());
}
$sql = "UPDATE MyGuests SET lastname='Doe' WHERE id=2";
if (mysqli_query($conn, $sql)) {
  echo "Record updated successfully";
} else {
  echo "Error updating record: " . mysqli_error($conn);
}
mysqli_close($conn);
?>
```

Example (PDO)

```php
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDBPDO";
try {
  $conn = new PDO("mysql:host=$servername;dbname=$dbname", $username,
$password);
  // set the PDO error mode to exception
  $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
  $sql = "UPDATE MyGuests SET lastname='Doe' WHERE id=2";
  // Prepare statement
  $stmt = $conn->prepare($sql);
  // execute the query
  $stmt->execute();
  // echo a message to say the UPDATE succeeded
  echo $stmt->rowCount() . " records UPDATED successfully";
} catch(PDOException $e) {
  echo $sql . "<br>" . $e->getMessage();
}
$conn = null;
?>
```

What is an Operator in SQL?

Operator भनेको एउटा reserved word वा एउटा character हो जुन मुख्यतया SQL statement को WHERE clause सँग प्रयोग गरिन्छ जसको सहायताले विभिन्न कार्यहरु गर्न सकिन्छ जस्तै : comparisons र arithmetic operations. Operators को प्रयोग गरेर हामीले SQL statement मा condition दिन सक्छौँ वा एउटै statement मा धेरै conditions हरु जोडेर लेख्नको लागि पनि प्रयोग गर्न सक्छौँ ।

The SQL LIKE Operator:

SQL like operator को प्रयोग WHERE clause सँग कुनै निश्चित कोलुममा रहेको pattern लाई खोज्नको लागि प्रयोग गर्छ । SQL like operator सँग दुई वटा wildcards character को प्रयोग हुन्छ ।

- % - The percent sign represents zero, one, or multiple characters
- _ - The underscore represents a single character

SQL Joins:

SQL Joins clause को प्रयोग डाटावेसमा रहेका दुई वा दुईभन्दा बढी टेवलहरुको रेकर्डहरु मिलाउनको लागि प्रयोग गरिन्छ । JOIN एउटा माध्यम हो जसको सहायताले दुईवटा टेवलका फिल्डहरुमा रहेका समान values को आधारमा रेकर्डहरु मिलाउने कार्य गरिन्छ ।