CHƯƠNG

Lập trình ứng dụng WEB với PHP

Các chủ đề chính

Mục tiêu

Sau khi hoàn thành chương này, chúng ta sẽ có thể:

- Phân tích được sự khác nhau cơ bản của hai phương thức POST và GET cũng như vận dụng chúng để lấy dữ liệu từ Form.
- Biết cách khai báo và tạo cookie.
- Trình bày được tầm quan trọng của cookie trong các ứng dụng thực tế.
- Trình bày được các bước cơ bản để kết nối csdl MySQL và hiển thị dữ liệu trên trang web.
- Vận dụng được các thao tác thêm, xóa, sửa dữ liệu.

Câu hỏi kiểm tra mở đầu

Trả lời các câu hỏi sau:

1. MySQL là?
   a. Cơ sở dữ liệu
   b. Hệ cơ sở dữ liệu
   c. Hệ quản trị cơ sở dữ liệu
2. Trong ASP.NET để lấy dữ liệu từ Form dùng phương thức GET chúng ta dùng lệnh?
   a. Request.QueryString.Get
   b. Request.Form.Get
   c. Request.Get
   d. Request.GetValue
3. Theo bạn giải pháp nào là tốt nhất để phân quyền người dùng trong các ứng dụng web?

a. Dùng lệnh rẽ nhánh để phân quyền
b. Lưu dấu người dùng và chỉ định bằng lệnh rẽ nhánh
c. Lưu dấu người dùng và chỉ định bằng URL

4. Nếu quá trình hiển thị dữ liệu của chúng ta là rất lớn (500 bản ghi), chúng ta sẽ nghĩ đến giải pháp gì?
   a. Hạn chế dữ liệu hiển thị
   b. Phân trang
   c. Truy vấn có điều kiện để hạn chế dữ liệu hiển thị
   d. Không vấn đề gì cả.
5. Theo bạn, các trang web HTML có kết nối được tới cơ sở dữ liệu nào không?
   a. Có
   b. Không

1.Trao đổi thông tin với trình duyệt

Dữ liệu của người dùng từ trình duyệt sẽ được gửi lên máy chủ dưới dạng từng cặp `biến=giá_trị` và có thể đi theo 3 con đường khác nhau. Tuỳ theo từng con đường cụ thể, trên máy chủ ta cũng có các cách khác nhau để lấy dữ liệu được gửi lên.. 3 con đường đó là: `GET, POST` và `COOKIES`. Trong phần này, chúng ta sẽ tìm hiểu về `GET` và `POST`. Chúng ta sẽ tìm hiểu về `Cookie` trong phần sau:

1.1 Truyền dữ liệu thông qua phương thức GET

Dữ liệu gửi từ trình duyệt lên qua phương thức `GET` là phần dữ liệu được nhập trực tiếp theo sau địa chỉ `URL` do trình duyệt gửi lên, được phân biệt với tên file script bằng dấu hỏi chấm (?). Ví dụ, khi ta gõ vào trình duyệt địa chỉ URL sau:

`http://codienhanoi.edu.vn/diendan/topic.php?TOPIC_ID=161`

Khi đó, trình duyệt sẽ gửi theo địa chỉ trên một cặp biến = giá trị, trong đó biến có tên là `TOPIC_ID` và giá trị là `161 (TOPIC_ID=161).` Chúng ta cũng có thể đưa lên nhiều cặp `biến=giá_trị` bằng cách phân cách chúng bởi dấu &:

`http://codienhanoi.edu.vn/diendan/index.php?`
`method=Reply&TOPIC_ID=161&FORUM_ID=20`

Địa chỉ `URL` trên, chúng ta sẽ gửi lên 3 cặp `biến=giá_trị` theo phương thức `GET`, đó là: `method=Reply, TOPIC_ID=161` và `FORUM_ID=20`. Khi trình duyệt gửi các thông tin này lên máy chủ, PHP sẽ tự động sinh ra một mảng có tên là `$_GET[]` để nắm giữ tất cả các cặp biến và giá trị đó, trong đó, chỉ số của mảng chính là một chuỗi mang tên của tên biến và giá

trị của chỉ số đó chính là giá trị của biến do trình duyệt gửi lên. Ví dụ, với địa chỉ URL sau:

http://codienhanoi.edu.vn/diendan/post.php?method=Reply&TOPIC_ID=161&FORUM_ID=20

Thì PHP sẽ tự động sinh ra một mảng $_GET có nội dung sau:

$_GET["method"] = "Reply" // tương ứng với cặp method=Reply

$_GET["TOPIC_ID"] = 161 // tương ứng với cặp TOPIC_ID=161

$_GET["FORUM_ID"] = 20 // tương ứng với cặp FORUM_ID=20

Ví dụ, chúng ta tạo ra hai file: wellcome.html và wellcome.php.

```
// wellcome.html
<form action="welcome.php" method="get">
    Name: <input type="text" name="fname" />
    Age: <input type="text" name="age" />
<input type="submit" />
</form>
```

Khi người dùng nhấp vào nút submit, URL gửi tới server có thể trông giống như thế này:

```
http://codienhanoi.edu.vn/welcome.php?fname=Peter&age=37
```

File "wellcome.php" bây giờ có thể sử dụng hàm $_GET để thu thập dữ liệu (Các tên của các trường trong Form sẽ tự động là các khóa trong mảng $_GET):

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

Khi chạy trên trình duyệt, nó sẽ hiển thị:

```
Welcome Peter!
You are 37 years old.
```

Chú ý: Phương thức này không nên sử dụng khi gửi password hoặc các thông tin nhạy cảm khác. Tuy nhiên, bởi vì các biến được hiển thị trên URL, nó có thể đánh dấu trang. Điều này có thể là hữu ích trong một số trường hợp. Một lưu ý nữa là phương thức GET không thích hợp cho các giá trị biến lớn (giá trị không thể vượt quá 100 ký tự)

1.2 Phương thức POST

Post là phần dữ liệu được gửi qua các form HTML có method ="POST" (xin xem lại bài về HTML).

Để lấy các biến theo kiểu POST, PHP sẽ tự động sinh ra mảng có tên là $_POST[]. Mảng này có chỉ số chính là tên của các phần tử trong form (các thẻ input, select... có thuộc tính name) và giá trị là nội dung giá trị do người sử dụng nhập vào các phần tử có tên tương ứng. Chẳng hạn với mẫu biểu HTML sau:

```
<form method="POST">
<p>
  User Name:<input type="text" name="T1" size="20"> </p>
<p>
  Password:
  <input type="password" name="T2" size="20"></p>
<p>Sex: <Select name ="sex">
           <option value =1>Male </option>
           <option value =0>Female </option>
        </select>
</p>
<input type="submit" value="Gui di" name="B1">
</form>
```

Khi người dùng nhập user name (giả sử là hieulv68), password (giả sử là 123456) và chọn sex là Male, khi đó, mảng $_POST sẽ có các phần tử sau:

```
$_POST["T1"] = hieulv68
$_POST["T2"] = 123456
$_POST["sex"] = 1
```

Đây là ví dụ một chương trình giải phương trình bậc nhất

```
<form method="POST">
<p style="margin-top: 0; margin-bottom: 0">
  Nhập a:<input type="text" name="a" size="20"></p>
  <p style="margin-top: 0; margin-bottom: 0">Nhập
b:<input type="text" name="b" size="20"></p>
  <p style="margin-top: 0; margin-bottom: 0">
  <input type="submit" value="Tính" name="B1"></p>
</form>
<?
      $a=0;$b=0;
      if (isset ($_POST["a"]))
      {
            $a =$_POST["a"];
      }
       if (isset ($_POST["b"]))
      {
            $b =$_POST["b"];
      }
      if ($a<>0)
      {
       echo "<BR>Nghiem la: " . $b/$a;Chỗ này là -$b/$a
      }
      else
      {
            if ($b==0)
                  {
                     echo "<BR>Vo so nghiem";
                  }
            else
            {
                  echo "<BR>Vo nghiem";
            }
      }
?>
```

2.Làm việc với Cookies

Khi các trang web tĩnh được phát triển trong các ứng dụng web động, điều cần thiết cho các ứng dụng này là duy trì trạng thái, đó là khả năng giữ lại các giá trị của biến và giữ lại các các thông tin của người dùng (Người hiện giờ đã đăng nhập vào hệ thống). Với các công nghệ trước kia chẳng hạn như CGI khi một client đưa ra một yêu cầu, server chỉ tạo ra một phản hồi và gửi trả nó về. Khi yêu cầu khác được nhận từ người dùng đó, server không có kế hoạch hành động nếu có một yêu cầu trước đó. Điều này bởi vì giao thức HTTP là chưa được công nhận.

2.1 Cookie là gì?

Cookie đã được phát triển để giải quyết vấn đề duy trì trạng thái giữa những người ghé thăm sau này đến một trang web hoặc giữa các lần truy cập vào các trang khác nhau trong một trang web. Cookie cho phép các máy chủ lưu trữ và truy xuất dữ liệu trên ổ cứng của client. Điều này tạo ra một tên miền mới của các ứng dụng mà có thể theo dõi đường dẫn của một client thông qua một trang web: ví dụ, các ứng dụng thương mại điện tử có thể lưu trữ bản ghi được lựa chọn bởi một khách hàng, một trang web thành viên có thể nhớ một ID cho mỗi người dùng và một web server có thể tạo các hồ sơ khách truy cập. Trong tất cả các trường hợp này, các cookie có thể được dùng để lưu trữ dữ liệu trên client.

Có những hạn chế cụ thể để tránh sự lạm dụng cookie. Đầu tiên, một trình duyệt được giới hạn đến 300 cookie và 20 cookie cho mỗi server. Nếu một ứng dụng cần để duy trì nhiều dữ liệu hơn, nó cần giữ dữ liệu ở phía server (điều này có thể được thực hiện với sự hỗ trợ của PHP 4.0, hoặc lưu trữ trong cơ sở dữ liệu). Thứ hai, cookie chỉ được gửi đến các

server được phép nhận chúng. Khi một server thiết lập một cookie, nó có thể hạn chế phạm vi của các máy chủ cookie được gửi tới. Bởi vì cookie có thể chứa dữ liệu nhạy cảm, bị rò rỉ dữ liệu này có thể dẫn đến một lỗ thủng bảo mật.

2.2 Các hạn chế của Cookie.

Phạm vi của cookie được xác định trong việc gửi phản hồi HTTP bởi web server. Phản hồi này bao gồm thông tin sau:

- Thông tin hết hạn (ví dụ: `01/01/2000, 03:00:00`)
- Thông tin đường dẫn (ví dụ: `/cgi-bin/php`)
- Thông tin tên miền (ví dụ: `codienhanoi.edu.vn`)
- Một tham số an toàn

Thông tin hết hạn được sử dụng để kiểm tra cookie vẫn còn hợp lệ đúng hay không. Một khi cookie đã hết hạn, client sẽ không gửi nó tới web server nữa. điều này được xác định trong GMT. Nếu ngày hết hạn không được xác định, client sẽ giải phóng cookie khi trình duyệt bị đóng. Thông tin đường dẫn xác định đường dẫn trên web server để cookie trong đó hợp lệ. Nếu thông tin đường dẫn của cookie và URL được yêu cầu không thỏa, client sẽ không gửi cookie.

Thông tin tên miền xác định tên miền cookie có giá trị. Chúng ta có thể hạn chế các web server đến một máy chủ cụ thể (ví dụ: `" .edu.vn"`), hoặc một tên miền hoàn toàn (ví dụ: `"codienhanoi.edu.vn".;` Lưu ý dấu chấm đầu tiên (.)). Điều này cho phép các cookie được chia sẻ giữa nhiều máy chủ. Ví dụ, một trang web lớn có thể sử dụng `hostnames www1.site.com,`

`www2.site.com`, `vv` Nếu thông tin tên miền được thiết lập để `".site.com"`, Cookie sẽ được truy cập từ tất cả các host này.

Nếu các thông số an toàn được kích hoạt, cookie sẽ chỉ được gửi trên các kênh an toàn (tức là qua giao thức HTTPS). Một kênh an toàn không thể được đọc bởi các bên thứ ba, do đó, dữ liệu không thể bị đánh cắp. Nếu tham số này không được thiết lập, cookie sẽ được gửi qua tất cả các kênh, bao gồm các kênh an toàn.

Mặc định cho các tham số này là:

Tên tham số	Giá trị mặc định
`Path`	"/" (Tất cả các đường dẫn trên server)
`Domain`	Tên miền của server để thiết lập cookie
`Expire information`	Cho đến khi đóng trình duyệt
`Secure`	Vô hiệu hóa (disable)

2.3 Cookie trong PHP

Cookie được hỗ trợ trong PHP, vì vậy người lập trình PHP có thể lấy được đầy đủ các tính năng của công nghệ này. Đọc cookie trong PHP đơn giản như truy cập biến. Trong lúc bắt đầu kịch bản của chúng ta, cookie tự động tạo ra biến đó là biến toàn cục.Ví dụ, nếu chúng ta thiết lập một cookie có tên `username` với nội dung `hieulv68`, thì biến `$username` sẽ bao gồm `"hieulv68"`.

Chú ý rằng cookie và biến dẫn xuất chỉ sẵn sàng khi client chấp nhận cookie và gửi nó lại cho server.

2.4 Khai báo và tạo cookie

Chúng ta hãy bắt đầu với một ví dụ đơn giản nơi mà chúng ta muốn đếm khách ghé thăm đã xem site của chúng ta như thế nào. Để làm điều này, chúng ta sử dụng một cookie có tên "count" để chứa số người ghé thăm. PHP sẽ tự động làm có hiệu lực biến $count vào kịch bản của chúng ta nếu cookie được gửi bởi hành động người dùng (Trình duyệt). Chúng ta sử dụng hàm setcookie() để gửi yêu cầu tới trình duyệt để thiết lập một cookie. Yêu cầu này cập nhật hoặc tạo một cookie trên client. Mã lệnh này phải xuất hiện tại lúc bắt đầu của trang, bất kỳ nội dung nào (kể cả khoảng trống) xuất hiện trước lúc mở thẻ PHP sẽ phát sinh lỗi.

```
<?php
   $count++;
   setcookie("count", $count);
?>
Welcome! You have seen this site
<? echo($count . ($count == 1 ? " time!" : "
times!")); ?>
```

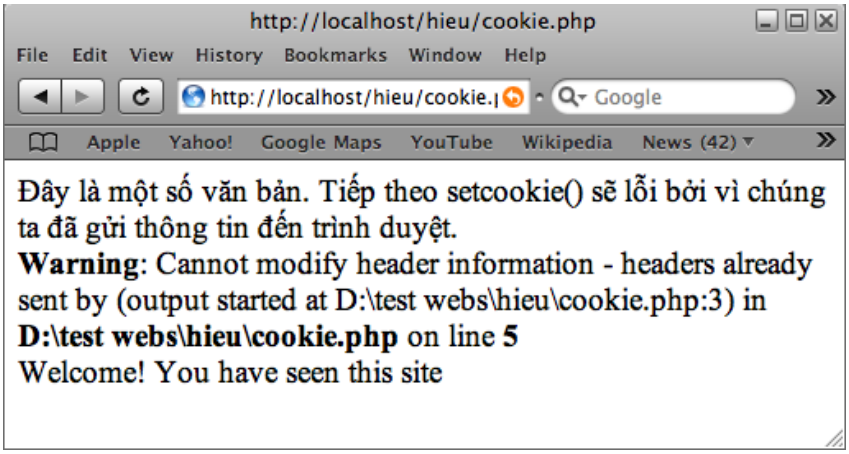
Kịch bản tăng biến $count và gửi giá trị đã tăng tới tác nhân người dùng sử dụng setcookie. Nếu tác nhân người dùng không gửi cookie cho chúng ta, PHP sẽ khởi tạo biến thành 0 khi chúng ta bắt đầu sử dụng nó. Tham số đầu tiên setcookie() là tên của cookie và tham số thứ hai là giá trị chúng ta muốn thiết lập cho nó. Phần cuối của mã lệnh trên đơn giản là chúng ta hiển thị ra một thông báo.

Một lỗi thường xảy ra là setcookie được gọi sau khi nội dung đã được gửi tới tác nhân người dùng. Điều này xảy ra khi chúng ta hiển thị dữ liệu bất kỳ trước khi setcookie được gọi. Một ký tự khoảng trống đơn (một dòng mới) có thể đủ để tạo cho setcookie() lỗi.

Đây là một số văn bản. Tiếp theo `setcookie()` sẽ lỗi bởi vì chúng ta đã gửi thông tin đến trình duyệt.

```
<?php
   $count++;
   setcookie("count", $count);
?>
Welcome! You have seen this site
<? echo($count . ($count == 1 ? " time!" : "
times!")); ?>
```

PHP sẽ tự động tạo ra một thông điệp cảnh báo nếu setcookie được gọi sau khi hồi âm HTTP chắc chắn đã được gửi.



Bây giờ chúng ta sẽ mở rộng ví dụ đầu tiên của chúng ta. Cookie được thiết lập mặc định chỉ cho phiên làm việc hiện tại của tác nhân người dùng và hết hạn một khi người dùng đóng trình duyệt. Nếu chúng ta không muốn điều đó xảy ra, chúng ta cần phải thiết lập một thời gian và ngày hết thời hạn rõ ràng trong lời gọi `setcookie ().` Thời gian hết hạn được quy định như một dấu thời gian (số giây kể từ kỷ nguyên (01 Tháng một 1970)). Dấu thời gian này có thể được tính toán trong PHP bằng cách sử dụng time() và hàm `mktime().` Hàm `time()` trả về mốc thời gian cho thời gian hiện tại và hàm `mktime()` chuyển đổi một ngày "thân thiện với con người" thành một mốc thời gian. Các tham số cho hàm này là

giờ, phút, giây, tháng, ngày và năm cho ngày để được chuyển đổi (trong thứ tự đó).

```
<?php
   // Hết hạn trong 3600 giây (1 giờ)
   setcookie("name", $value, time() + 3600);

   // Hết hạn vào 01/01/2002
   setcookie("name", $value, mktime(0,0,0,1,1,2002));
   // Hết hạn lúc 6:30 PM ngày 05/12/2020
   setcookie("name", $value, mktime(18, 30, 0, 5, 12,
2020));
?>
```

2.5 Bạn tên gì?

Chúng ta hãy xem một ví dụ khác. Trang này thông báo người dùng nhập tên của họ, tên của họ sau đó được submit tới server. Server sẽ gửi một yêu cầu "setcookie" tới client và trên vùng người dùng ghé thăm sẽ được chào đón bởi tên.

```
<?php

   if($action == "setcookie") {
     setcookie("visitorname",$visitorname,time()+90*86400);
      // Hết hạn trong 90 ngày
   }
   if(isset($visitorname)):
?>
Welcome <B><? echo $visitorname ?></B>!
<? else: ?>
<FORM>
   <INPUT TYPE="HIDDEN" NAME="action" VALUE="setcookie">
   Welcome, please tell us your name:
       <INPUT TYPE="TEXT" NAME="visitorname"><BR>
       <INPUT TYPE="SUBMIT" VALUE="   OK   ">
```

```
</FORM>
<? endif; ?>
```

Khi người dùng để lại dấu vết đến trang này, mã lệnh kiểm tra xem biến `$visitorname` thiết lập đúng hay sai. Nếu nó đúng, một thông điệp lời chào sẽ được hiển thị. Mặt khác chúng ta sẽ hiển thị một form nhỏ mời người dùng nhập tên của họ.

Khi người dùng nhập tên, trang sẽ nhận yêu cầu và kiểm tra biến $action từ phần tử `<ELEMENT>` ẩn thiết lập tới `"setcookie"` đúng hay sai. Nếu nó đúng, kịch bản cố gắng thiết lập một cookie trên client sử dụng `setcookie().` Chúng ta xác định vòng đời của cookie là 90 ngày (một ngày có 86400 giây) với hiệu lực rằng cookie sẽ hết hạn trong 3 tháng sau, nếu nó được chấp nhận bởi client.

2.6 Truy cập một cookie

Nếu yêu cầu HTTP được gửi bởi tác nhân người dùng bao gồm thông tin cookie, PHP sẽ tự động truyền dữ liệu này vào trong biến để kịch bản của chúng ta có thể truy cập vào biến đó. Ví dụ, nếu tác nhân người dùng gửi một cookie có tên `"username",` kịch bản có thể truy cập giá trị của cookie bằng cách sử dụng một trong hai phương thức sau:

- `$username` – biến được lưu trữ trong biến toàn cục với tên giống như cookie.
- `$_COOKIE["username"]` – Mảng kết hợp toàn cục bao gồm duy nhất các biến từ các cookie. Điều này giúp cho thấy được sự khác biệt giữa các biến mà nó tạo ra từ các nguồn dữ liệu khác nhau (Xem thêm `$_GET` và `$_POST`). Nếu chúng ta truy cập mảng này, thông tin về nguồn gốc là có thể tin cậy.

2.7 Thiết lập một cookie

Cách cơ bản nhất để thiết lập một cookie là sử dụng hàm `setcookie()`. Chúng ta đã thấy form đơn giản nhất của hàm này – một cách đơn giản là chúng ta gọi `setcookie()` với tên cookie và giá trị để nó thiết lập. Ví dụ, để lưu trữ giá trị `“value”` trong cookie `“cookiename”`, chúng ta sẽ sử dụng trong kịch bản của chúng ta:

```
    setcookie("cookiename", "value");
```

2.8 Các cookie nhiều giá trị

Tuy nhiên, giả sử chúng ta muốn lưu trữ cả tên của khách ghé thăm và số lần người dùng dã ghé thăm trang của chúng ta. Chúng ta có thể sử dụng hai cookie tách rời nhau, nhưng khi có sự hạn chế 20 cookie trên mỗi server, chúng ta có thể không muốn làm điều này. May thay, chúng ta có thể lưu trữ nhiều giá trị trong một cookie đơn. Để làm điều này, chúng ta coi cookie là một mảng và gán giá trị tới mỗi phần tử trong mảng đó:

```
<?php
   if (!isset($mycookie[0])) {
      setcookie("mycookie[0]", $visitorname);
   }
   $mycookie[1]++;
   setcookie("mycookie[1]", $mycookie[1]);
   echo("Hello $mycookie[0], you've seen this page " .
     $mycookie[1] . ($mycookie[1] == 1 ? " time!" : "
times!"));
?>
```

2.9 Thiết lập ngày hết hạn

Khichúng ta gửi một yêu cầu `“set cookie”` tới client (không thiết lập ngày hết hạn). Nếu client chấp nhận cookie, nó sẽ cố gắng duy

trì cho đến khi trình duyệt bị đóng. Để kéo dài vòng đời của cookie, chúng ta có thể gửi một thời gian hết hạn với yêu cầu "set cookie". Thời gian này phải được xác định trong một mốc thời gian. Chúng ta có thể tính toán mốc thời gian này bằng cách sử dụng hai hàm. Hàm đầu tiên là mktime(), nó có thể được sử dụng để tính toán ngày thuần túy. Hàm thứ hai là time(), nó trả về thời gian hiện tại theo giây. Bằng việc thao tác số chúng ta có thể xác định vòng đời của một cookie có liên quan tới thời gian hiện tại. Hãy nhớ rằng thời gian hết hạn là thời gian trên máy của client, không phải trên server – do vậy nó có thể là vùng thời gian khác nhau.

```
// Các ngày thuần túy
// Tham số: giờ:phút:giây:tháng:ngày:năm
$lifetime = mktime(0, 0, 0, 12, 1, 1999);
// giữa đêm 01.12.1999
$lifetime= mktime(12, 50, 30, 6, 20, 2010);
//12:50:30 20.06.2010

// Các ngày tương đối
$lifetime = time() + 3600;            // Vòng đời 1 giờ
$lifetime = time() + 86400;          // Vòng đời 1 ngày
$lifetime=time()+86400*30;//Vòng đời một tháng(30 ngày)
```

Sau khi chúng ta đã tính toán vòng đời sống, chúng ta có thể truyền nó đến setcookie() là tham số thứ 3:

```
    setcookie("cookiename", "value", $lifetime);
```

Trình duyệt sẽ duy trì cookie chạy phía sau và sẽ tự động hủy cookie tại thời điểm xác định.

2.10 Giới hạn phạm vi của cookie

Tùy chọn hữu ích khác là chỉ định các trang trên web server của chúng ta để cookie sẽ được gửi tới nó. Hãy tưởng tượng rằng một web server nơi mà một số người dùng đã lưu trữ các trang của họ trong `/customer1, /customer2 .v.v.` Nếu trình duyệt luôn gửi cookie tới web server, cookie thiết lập bằng cách một kịch bản đang thuộc về người dùng đầu tiên cũng sẽ thấy được tất cả các trang của người dùng khác trên cùng server. Phụ thuộc vào nội dung của các cookie, điều này có thể có khả năng ra tạo một vấn đề bảo mật.

Do vậy, các tác nhân người dùng có thể hạn chế vòng đời của một cookie. Giới hạn đầu tiên xác định tập hợp con của URL trong một tên miền để cookie có hiệu lực. Chú ý rằng tất cả các đường dẫn bắt đầu với chuỗi xác định sẽ được thỏa, ví dụ: `“/cust”` thỏa cả `“/customer1/test.php”` và `“/cust.php”`. Do vậy, nếu chúng ta muốn chỉ định một đường dẫn, chúng ta sẽ nối thêm một vạch xiên `(/)`. Đường dẫn mức đỉnh là `“/”;` chúng ta có thể chỉ định điều này nếu chúng ta muốn cookie có hiệu lực cho toàn web server. Giá trị mặc định cho tham số này là đường dẫn của tài liệu mà nó gọi `setcookie().` Đối với các ví dụ phía trên của chúng ta, chúng ta sẽ chỉ định theo sau việc giới hạn cookie bằng đường dẫn `/customer1`:

```
setcookie("cookiename","value",$lifetime,"/customer1/";
```

Giới hạn thứ hai điều khiển các tên miền để cookie hợp lệ. Cookie chỉ gửi tới web server nếu tên miền của host từ URL được thỏa thuộc tính

miền. Cookie là hợp lệ nếu có đuôi được thỏa. Ví dụ: “.server.com” sẽ thỏa nhưng “webserver.com” thì không thỏa.

```
setcookie("cookiename","value",$lifetime,"/customer1/",
".server.com");
```

Điều này sẽ cho biết tác nhân người dùng gửi cookie tới tất cả server, các hostname của server này trong tên miền server.com.

Bây giờ, giả sử rằng cookie của chúng ta bao gồm các thông tin nhạy cảm mà chúng ta muốn bảo vệ từ các con mắt khác. Sử dụng một HTTP (HTTPS) kết nối tới web server mã hóa dữ liệu và do vậy làm cho nó khó khăn hơn để xem trộm. Để bỏ qua sự rủi ro của việc gửi cookie trên một kết nối văn bản rõ ràng (không mã hóa), tham số thứ 6 có thể được truyền. Nếu tham số này được thiết lập là 1, tác nhân người dùng sẽ không gửi cookie trừ khi kết nối là được bảo vệ.

Cú pháp đầy đủ cho hàm `setcookie()` như sau:

```
int setcookie(string cookiename, string [value], integer
[lifetime], string [path], string [domain], integer
[secure]);
```

Tổng kết các tham số trên:

- `Cookiename` – tên cookie, giá trị sau đó có thể truy cập là `$cookiename`
- `Value` – Đây là giá trị để lưu trữ trong `$cookiename`. Nó tự động mã hóa và giải mã bởi PHP.
- `Lifetime` – thời gian khi cookie sẽ hết hạn, có thể được tính toán bởi `mktime()` và `time()`.
- `Path` – Tâph hợp con của đường dẫn để cookie hợp lệ. Một dấu gạch chéo nên được thêm vào nếu chúng ta muốn chỉ định một đường dẫn.

- `Domain` – quyết định loại server cookie sẽ gửi. Tên miền của host phải thỏa với miền được chỉ định nếu cookie được gửi.
- `Secure` – được sử dụng để ngăn chặn cookie gửi trên một kết nối không an toàn (HTTP chuẩn).

Tất cả các tham số này là tùy chọn, ngoại trừ `cookiename`. Giá trị mặc định cho mỗi tham số tùy chọn là chuỗi rỗng (`value, path, domain`) hoặc 0 (`lifetime, secure`). Ví dụ, nếu chúng ta muốn chỉ định miền nhưng không vòng đời sống hoặc đường dẫn, chúng ta sẽ sử dụng:

```
setcookie("cookiename", "value", 0, "", ".server.com");
```

2.11 Xóa một cookie

Một cookie có thể được xóa bằng cách sử hùng hàm `setcookie()` với duy nhất một tham số.

```
    setcookie("cookiename")
```

Điều này sẽ gây cho cookie `“cookiename”` bị xóa trên client. Điều này không ảnh hưởng đến thiết lập các biến cookie hiện thời. Nó cũng không thay đổi `$_COOKIE`.

3. Thao tác với cơ sở dữ liệu MySQL

MySQL là cơ sở dữ liệu được sử dụng cho các ứng dụng Web có quy mô vừa và nhỏ. Tuy không phải là một cơ sở dữ liệu lớn nhưng chúng cũng có trình giao diện trên Windows hay Linux, cho phép người dùng có thể thao tác các hành động liên quan đến cơ sở dữ liệu.

Cũng giống như các cơ sở dữ liệu, khi làm việc với cơ sở dữ liệu MySQL, chúng ta đăng ký kết nối, tạo cơ sở dữ liệu, quản lý người dùng, phần quyền sử dụng, thiết kế đối tượng Table của cơ sở dữ liệu và xử lý dữ liệu.

Tuy nhiên, trong bất kỳ ứng dụng cơ sở dữ liệu nào cũng vậy, nếu bản thân chúng có hỗ trợ một trình giao diện đồ hoạ, chúng ta có thể sử dụng chúng tiện lợi hơn các sử dụng Command line. Bởi vì, cho dù chúng ta điều khiển MySQL dưới bất kỳ hình thức nào, mục đích cũng quản lý và thao tác cơ sở dữ liệu.

3.1 SQL là gì?

SQL có thể được định nghĩa là ngôn ngữ chuẩn được sử dụng để tương tác với cơ sở dữ liệu quan hệ. Tuy nhiên, SQL không phải là ngôn ngữ máy tính giống như C, C++ hoặc PHP. Thực tế, nó là công cụ tương tác để thực hiện các nhiệm vụ quản lý cơ sở dữ liệu khác nhau, thường là một tập các câu lệnh được định nghĩa tới người dùng. Chính xác hơn nó là một ngôn ngữ truy vấn, SQL cung cấp một dãy các công cụ để tương tác với cơ sở dữ liệu, bao gồm những phần sau đây:

- Ngôn ngữ định nghĩa dữ liệu (Data Structure definition): SQL có thể định nghĩa các cấu trúc khác nhau mà cơ sở dữ liệu sử dụng để lưu trữ dữ liệu.

- Truy vấn dữ liệu (Data querying): SQL có thể phục hồi dữ liệu trong cơ sở dự liệu và đưa ra một định dạng có thể đọc được một cách đơn giản.
- Thao tác dữ liệu (Data manipulaion): SQl có thể chèn, cập nhật và xóa dữ liệu cơ sở dữ liệu.
- ...

3.2 Định nghĩa dữ liệu

Các kiểu miền trong SQL

- CHAR(n): Xâu kí tự có độ dài cố định n.
- VARCHAR(n): Xâu kí tự có độ dài tối đa n.
- INT, SMALLINT: Kiểu số nguyên.
- NUMBER(p,d): Số thập phân gồm p chữ số và một dấu chấm và d chữ số bên phải dấu chấm.
- REAL, DOUBLE, PRECISION: Số dấu phẩy động.
- FLOAT(n): Số dấu phẩy động với độ chính xác ít nhất n chữ số.
- DATE: Kiểu ngày, tháng, năm.
- TIME: Kiểu giờ trong ngày.

Định nghĩa lược đồ trong SQL

Dạng đơn giản nhất của câu lệnh tạo một bảng có cú pháp như sau:

```
CREATE TABLE <Tên bảng>(
    <Tên cột 1><Kiểu dữ liệu 1>(<Kích thước 1>),
```

```
        <Tên cột 2><Kiểu dữ liệu 2>(<Kích thước 2>),
        ..................
        <Tên cột n><Kiểu dữ liệu n>(<Kích thước n>)
        );
```

Ví dụ: Để tạo bảng ***nhan_vien*** có thể dùng câu lệnh sau:

```
    CREATE TABLE nhan_vien (
        Manv      NUMBER(2),
        Ho_ten    VARCHAR(25) Ng_sinh    DATE,
        Gioi_tinh VARCHAR(3),
        Ma_dv     CHAR(2),
        Luong     NUMBER(9)
        );
```

3.3 Thao tác dữ liệu

Các câu lệnh cơ bản được tập chung nói đến trong phần thao tác dữ liệu này là:

- **Cấu trúc** **SELECT** dùng để truy vấn dữ liệu trong CSDL
  FROM
  WHERE
- **INSERT**: Thêm các bộ mới vào một bảng.
- **UPDATE**: Sửa đổi thông tin đã có trong một bảng.
- **DELETE**: Xoá bỏ một số bộ trong một bảng.

Cấu trúc cơ sở để truy vấn

Cú pháp điển hình của một biểu thức truy vấn trong SQL là:

```
SELECT [ DISTINCT | ALL] { * | [<biểu thức cột> AS [<Tên mới>]] [,...]}
FROM <tên bảng>[<bí danh>] [,...]
[WHERE <điều kiện>]
[GROUP BY <danh sách tên cột>] [HAVING<điều kiện>]
[ORDER BY <Danh sách tên cột>]
```

Trong dạng trên <biểu thức cột> là tên của một cột hoặc một biểu thức, <tên bảng> là tên của một bảng trong CSDL hay một khung nhìn (View) mà ta có thể truy cập vào, <bí danh> là một tên viết tắt của tên bảng. Ngoài ra:

- ***GROUP BY*** dùng để gộp nhóm các bộ cùng giá trị tương ứng ở các cột xuất hiện trong <danh sách tên cột>.
- ***HAVING*** dùng để lọc các nhóm thỏa điều kiện.
- ***ORDER BY*** quy định thứ tự các cột trong kết quả trả ra.

Thứ tự các câu trong dạng biểu thức truy vấn trên không thể thay đổi. Kết quả của biểu thức truy vấn đó là một bảng. Dưới đây là một số ví dụ minh họa.

Ví dụ

Để tìm tên các dự án và mã các đơn vị (Các phòng) QL dự án tương ứng có thể dùng truy vấn sau:

```
SELECT ten_da, ma_dv
FROM du_an
```

Bảng kết quả của truy vấn trên (đối với bảng DU_AN) sẽ là:

ten_da	ma_dv
Phần mềm A	P4
Mạng B	P3

Agent C	P2
Phần mềm B	P1

Muốn bảng kết quả không chỉ có 2 cột `ten_da` và `ma_da` mà là tất cả các cột trong bảng `du_an`, có 2 cách.

```
SELECT ma_da, ten_da, dia_diem_da, ma_dv
FROM du_an
```

hoặc

```
SELECT *
FROM du_an
```

Ví dụ

Câu truy vấn sau cho danh sách nhân viên có `lương` trên `2300000`, với mỗi nhân viên như vậy các thông tin đưa ra gồm: `mã nhân viên, họ tên, mã đơn vị và lương`.

```
SELECT ma_nv, ho_ten, ma_dv, luong
FROM nhan_vien
WHERE luong > 2 300 000
```

Ví dụ

Liệt kê lương của các nhân viên theo thứ tự tăng dần.

```
SELECT ma_nv, ho_ten, ma_dv, luong
FROM nhan_vien
ORDER BY luong;
```

Bảng kết quả sẽ là:

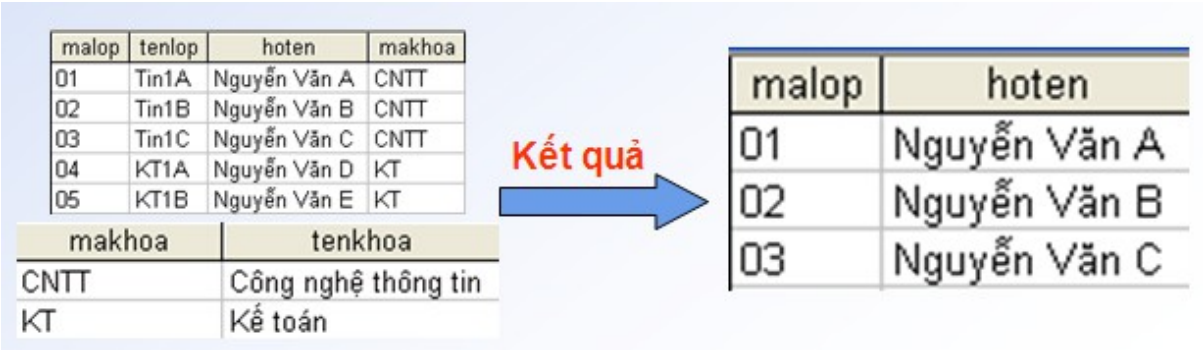
ma_nv	ho_ten	ma_dv	luong
004	Nguyễn Văn D	P4	2000000
001	Nguyễn Văn A	P1	2500000
003	Nguyễn Văn C	P3	2800000
002	Nguyễn Văn B	P2	3500000

Ví dụ

Để tìm mã số và họ tên những học sinh khoa ‘Công nghệ thông tin’ có thể dùng câu lệnh truy vấn sau:

```
SELECT malop, hoten
FROM lop
WHERE makhoa =
          SELECT makhoa
          FROM khoa
          WHERE tenkhoa = 'Công nghệ thông tin');
```

Bảng kết quả sẽ là:



Cập nhật cơ sở dữ liệu.

Trong SQL có 3 câu lệnh có thể biến đổi cơ sở dữ liệu (Thêm, bớt, thay đổi thông tin).

- **INSERT**: Thêm các bộ mới vào một bảng.
- **UPDATE**: Sửa đổi thông tin đã có trong một bảng.
- **DELETE**: Xoá bỏ một số bộ trong một bảng.

- **Dạng INSERT.**

Câu lệnh Insert cho phép một bộ được thêm vào một bảng, cú pháp như sau:

```
INSERT INTO <Tên bảng>[(Danh sách cột)]
VALUES (Danh sách các giá trị)
```

Ví dụ

Câu lệnh sau thêm vào một bộ vào bảng **nhan_vien**.

```
INSERT INTO nhan_vien
VALUES ('11','Trần Hữu Việt', 'Nam', 'Hà Nội');
```

- **Dạng UPDATE.**

Trong một số trường hợp chúng ta cần thay đổi một số giá trị trong một bộ chứ không phải là thay đổi tất cả các giá trị của bộ. Lệnh **UPDATE** cho phép làm điều đó và cú pháp như sau:

```
UPDATE <Tên bảng>
SET <Tên cột i>=<Giá trị i>[,<Tên cột j>=<Giá trị j>,...]
[WHERE <điều kiện>]
```

Khi câu **WHERE** không xuất hiện trong câu lệnh cập nhật này thì mọi bộ trong bảng sẽ được cập nhật trên những cột xác định bởi **SET**, ngược lại thì chỉ những bộ thoả điều kiện đặt sau **WHERE** mới bị sửa đổi.

Ví dụ

Tăng lương 5% cho mọi nhân viên, câu lệnh được viết như sau:

```
UPDATE nhan_vien
```

```
SET luong = luong*1.05;
```

Ví dụ

Tăng lương 5% cho nhân viên làm ở đơn vị có mã số "**P2**", câu lệnh được viết như sau:

```
UPDATE nhan_vien
SET luong = luong*1.05
WHERE ma_dv = 'P2';
```

- **Dạng DELETE**

Cú pháp của câu lệnh xoá một số bộ khỏi một bảng là:

```
DELETE FROM <Tên bảng>
[WHERE <Điều kiện>]
```

Trong câu lệnh trên, nếu câu **WHERE** không xuất hiện thì tất cả các bộ trong bảng sẽ bị xoá, ngược lại thì chỉ những bộ thoả điều kiện bị loại bỏ khỏi bảng.

Câu lệnh sau sẽ xoá tất cả các bộ trong bảng **cong_thang**.

```
DELETE FROM cong_thang
```

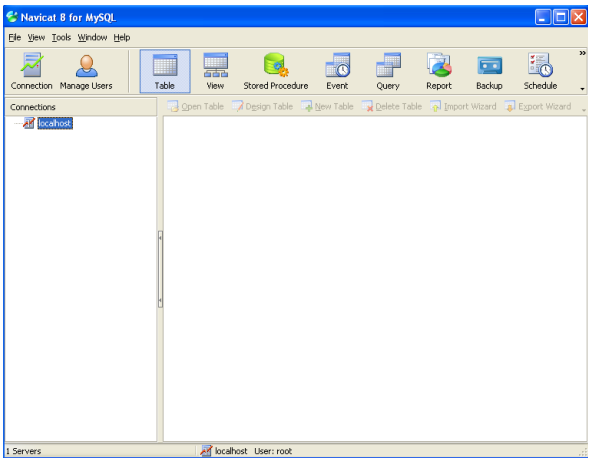
Để xoá bộ nói về nhân viên có mã số 5 trong bảng **nhan_vien**, có thể viết.

```
DELETE FROM nhan_vien
WHERE manv = '5';
```

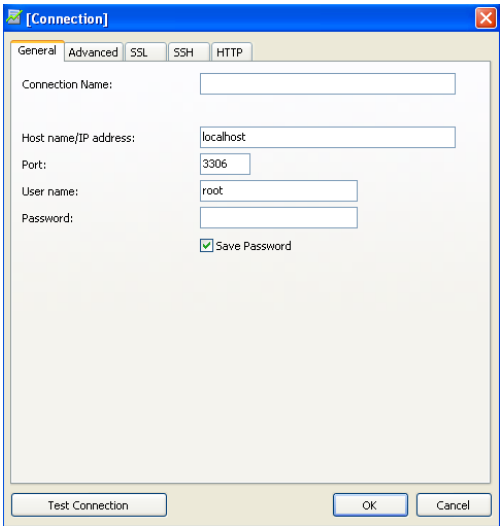
3.4 Tạo một cơ sở dự liệu MySQL

Trong phần này, để tiện cho việc tạo một cơ sở dữ liệu đơn giản, chúng ta sử dụng phần mềm **navicat** đã được giới thiệu trong chương 1.

❖ Mở màn hình chính của navicat.

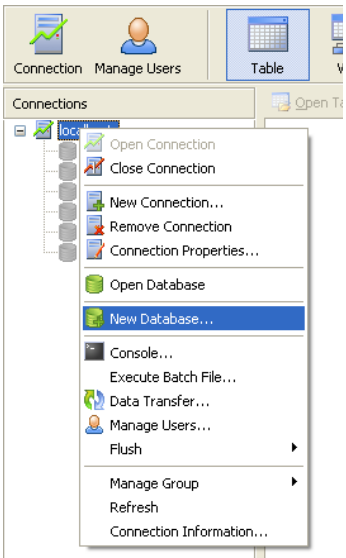


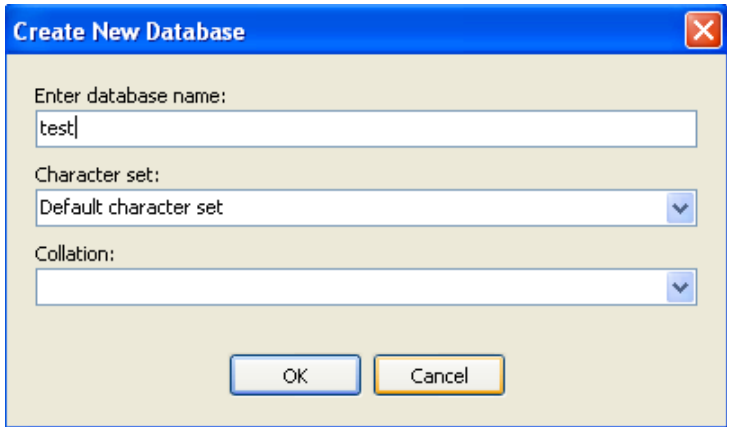
❖ Tạo 1 connection với MySQL bằng cách click vào nút Connection . Cửa sổ Connection hiện ra , chúng ta điền các thông tin vào , rồi nhấn Test Connection , nếu báo Connection Successful thì chúng ta đã thành công kết nối với MySQL . Bấm OK để hoàn tất:



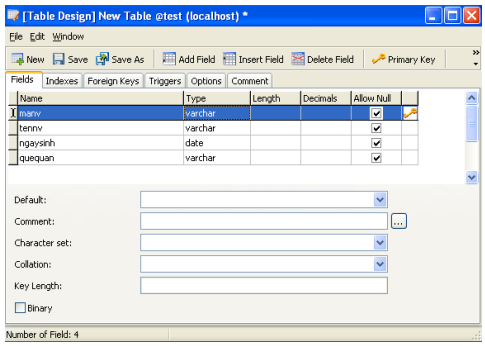
❖ Danh sách các database mà user root có thể quản lý sẽ được hiện thị ở cột phía bên trái :

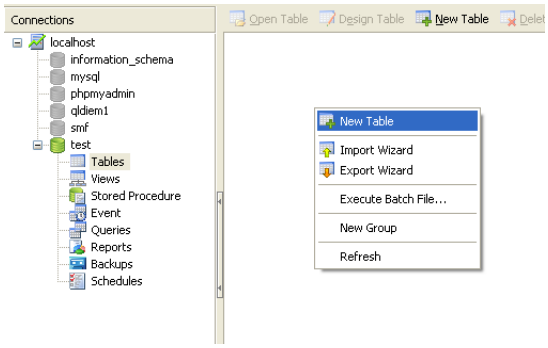
❖ Bây giờ chúng ta sẽ tạo 1 database mới . Right Click vào tên Connection (có thể ở bất cứ chỗ nào trong cột Connection) chọn New Database … , một cửa sổ mới hiện ra , bạn điền tên Database muốn tạo vào rồi click OK





- Bây giờ bạn đã có 1 database mới , nó cũng được liệt kê ở cột Connection . Bây giờ, chúng ta tiến hành tạo các bảng cho csdl.
- Tại cửa sổ bên phải, nhấp chuột phải -> chọn New Table -> xuất hiện cửa sổ để thiết kế.





❖ Các bước tiếp theo rất đơn giản vì chúng ta sẽ thao tác tương tự như với Access và SQL Server.

3.5 Các bước truy cập cơ sở dự liệu MySQL trong PHP

Để truy cập cơ sở dự liệu MySQL trong PHP, chúng ta cần thực hiện 5 bước cơ bản sau:

- Tạo kết nối đến Database Server.
- Lựa chọn CSDL.
- Xây dựng truy vấn và thực hiện truy vấn.
- Xử lý kết quả trả về.
- Đóng kết nối đến Server.

Với 5 bước cơ bản trên, sau đây chúng ta sẽ tìm hiểu chi tiết về câu lệnh của từng bước trên.

Tạo kết nối đến Database Server

Hàm `mysql_connect()` được sử dụng để thiết lập một kết nối với máy chủ MySQL. Khi kết nối được thiết lập thành công, một cơ sở dữ liệu đang lưu trú trên server đó có thể được lựa chọn. Cú pháp là:

```
int      mysql_connect(string      [hostname      [:port]
[:/path_to_socket]],     string     [username],     string
[password]);
```

Các đối số cho hàm này được đưa ra trong bảng sau và tất cả những điều này là tùy chọn:

Tham số	Mô tả	Mặc định
`Hostname`	Tên của host đang chạy database server. Không cần xác định điều này nếu nếu	"localhost"

	cơ sở dữ liệu và web server đang chạy trên máy giống nhau.	
:port	Cổng mà database server đang sử dụng để lắng nghe các yêu cầu. Chỉ thực hiện điều này nếu cài đặt của chúng ta sử dụng một cổng khác với cổng mặc định của MySQL.	": 3306"
: /path_to_socket	Unix socket để server sử dụng để lắng nghe các yêu cầu	": /tmp/mysql.sock"
Username	Tên của người dùng được phép kết nối đến database server.	
Password	Mật khẩu cho người dùng; nếu không có thì ta để rỗng.	

Một ví dụ kết nối được gọi như sau:

```
mysql_connect("localhost", "web", "4tf9zzzf") or die("Không thể kết nối tới MySQL server!");
```

Trong trường hợp này, `localhost` là host server, `web` là username và `4tf9zzzf` là password. Hàm `die("Chuỗi")`: Đưa ra thông báo và kết thúc. Với cách viết trên, die chỉ thực hiện khi lệnh trước nó không thành công.

Lựa chọn cơ sở dữ liệu

Khi một kết nối đã được thiết lập thành công với MySQLServer, một cơ sở dữ liệu đang lưu trú trên server đó có thể được lựa chọn. Điều này có thể được hoàn thành với `mysql_select_db()`. Cú pháp của nó là:

```
int mysql_select_db (string database_name [, int link_identifier])
```

Tham số	Mô tả	Mặc định
Database_name	Tên của cơ sở dữ liệu, nó là cơ sở dữ liệu có hiệu lực.	
Link_identifier	Tham chiếu của kết nối, trong đó những yêu cầu sẽ được gửi tới database server.	Liên kết nhận dạng kết nối trước được mở. Nếu kết nối không được mở thì hàm cố gắng mở một kết nối nới sử dụng mysql_connect với tham số mặc định.

Tham số `database_name` được yêu cầu, `link_identifier` là tùy chọn. hàm `mysql_select_db()` trả về true nếu thành công hoặc false nếu lỗi.

Ví dụ:

```
<?
@mysql_connect("localhost", "web", "4tf9zzzf")
or die("Không kết nối được tới MySQL server!");
@mysql_select_db("company") or die("Không lựa chọn được CSDL company!");
?>
```

Xây dựng truy vấn và thực hiện truy vấn

Hàm `mysql_query()` gửi câu lệnh SQL tới MySQL server thực thi. Cú pháp của nó là:

```
int mysql_query(string query, int [link_identifier]);
```

Tham số	Mô tả	Mặc định
query	Câu lệnh SQL được gửi tới MySQL server.	
Link_identifier	Tham chiếu của kết nối, trong đó câu lệnh SQL sẽ được gửi tới database server.	Liên kết nhận dạng kết nối trước được mở. Nếu kết nối không được mở thì hàm cố gắng mở một kết nối nới sử dụng mysql_connect với tham số mặc định.

Tham số query được yêu cầu, link_identifier là tùy chọn. hàm trả về kết quả hợp khuôn dạng (nguyên dương) nếu thành công hoặc false nếu lỗi. Kết quả hợp khuôn dạng bao gồm kết quả thực thi của câu lệnh SQL trên SQL server.

Ví dụ:

```
<?
mysql_connect("localhost", "web", "4tf9zzzf")
or die("Không kết nối được tới MySQL server!");
mysql_select_db("company") or die("Không lựa chọn được CSDL company!");
mysql_query(“SELECT * FROM products WHERE prod_name LIKE \"p%\"”) or die (“Không thực hiện được SQL”)
?>
```

Xử lý kết quả trả về

Mysql_num_rows() : Hàm này được sử dụng để xác định số lượng bản ghi được trả về từ câu lệnh truy vấn SELECT. Cú pháp của nó là:

```
int mysql_num_rows (int result)
```

Ví dụ:

```
<?
@mysql_connect("localhost", "web", "4tf9zzzf")
or die("Could not connect to MySQL server!");
@mysql_select_db("company") or die("Không thể lựa
chọn csdl company!");
// Lựa chọn tất cả các tên sản phẩm bắt đầu với ‘p’
$query = "SELECT prod_name FROM products WHERE
prod_name LIKE \"p%\"";
// Thực thi truy vấn
$result = mysql_query($query);
print "Total rows selected:
".mysql_num_rows($result);
mysql_close();
?>
```

Khi có duy nhất một sản phẩm có tên bắt đầu với p (pears), chỉ một bản ghi được lựa chọn. Đây là kết quả:

```
Total rows selected: 1
```

Tiếp theo chúng ta sử dụng hàm mysql_fetch_array() để đọc từng mẩu tin. Cú pháp của nó là:

```
array mysql_fetch_array (int result [, result_type])
```

Ví dụ:

```
<?
@mysql_connect("localhost", "web", "ffttss")
or die("Could not connect to MySQL server!");
```

```
@mysql_select_db("company") or die("Could not select
products database!");
$query = "SELECT * FROM products";
$result = mysql_query($query);
print "<table>\n";
print "<tr>\n<th>Product ID</th><th>Product Name</th>
<th>Product Price</th>\n</tr>\n";
//  No  result  type,  therefore  It  defaults  to
MYSQL_ASSOC
while ($row = mysql_fetch_array($result)) :
print "<tr>\n";
print "<td>".$row["prod_id"]."</td>\n
<td>".$row["prod_name"]."</td>\n
<td>".$row["prod_price"]."</td>\n";
print "</tr>\n";
endwhile;
print "</table>";
?>
```

Đóng kết nối đến server

Để kết thúc quá trình kết nối đến cơ sở dữ liệu, chúng ta có thể đóng kết nối theo cú pháp sau:

```
    mysql_close($biến_kết_nối)
```

3.6 Một số hàm hữu ích

Mysql_fetch_row()

Hàm trả về một mảng là giá trị của một bản ghi hiện tại với chỉ số là số thứ tự của các trường (chỉ số bắt đầu là 0). Sau đó hàm sẽ trỏ tới bản ghi tiếp theo cho tới khi gặp bản ghi cuối cùng hàm trả về giá

trị false. Để truy xuất tới giá trị của cột ta viết: Tên_mảng[số thứ tự]

Cú pháp:

```
    Array mysql_fetch_row(int result_identifier);
```

Trong đó: result_identifier là mã số trả về của hàm mysql_query() hoặc mysql_db_query().

Ví dụ:

```
<?php
    $mysql = “SELECT id, name FROM sinhvien”;
    $link = mysql_connect($host, $user, $pass);
    Mysql_select_db($database_name, $link);
    $result = mysql_query($mysql, $link);
    While($row = mysql_fetch_row($result))
    {
            echo $row[0];
            echo $row[1];
    }
?>
```

Mysql_fetch_object()

Hàm trả về một đối tượng là giá trị của một bản ghi hiện thời. Sau đó hàm sẽ trỏ tới bản ghi tiếp theo cho tới khi gặp bản ghi cuối cùng hàm trả về giá trị false. Để truy xuất tới các giá trị của cột ta viết tên_object -> tên_cột.

Cú pháp:

```
Object mysql_fetch_object(int result_identifier);
```

Trong đó: `result_identifier` là mã số trả về của hàm `mysql_query()` hoặc `mysql_db_query()`.

Ví dụ:

```
<?php
    $mysql = “SELECT id, name FROM sinhvien”;
    $link = mysql_connect($host, $user, $pass);
    Mysql_select_db($database_name, $link);
    $result = mysql_query($mysql, $link);
    While($row = mysql_fetch_object($result))
    {
            echo $row -> id;
            echo $row -> name;
    }
?>
```

Mysql_affected_rows()

Cú pháp:

```
    int mysql_affected_rows(int [link_identifier]);
```

Trong đó: `int link_identifier` là mã số nhận dạng, nó phải được thực hiện trong hàm `mysql_select_db()` trước đó.

Hàm trả về số dòng đã bị tác động bởi một câu truy vấn SQL: `INSERT, UPDATE, DELETE` trước đó theo tham số `link_identifier`. Nếu `link_identifier` không được chỉ định thì mã kết nối trước đó sẽ được chỉ định.

Chú ý:

- Nếu câu lệnh SQL trước đó là `DELETE` mà không có mệnh đề `WHERE` thì toàn bộ các bản ghi đã bị xóa nhưng hàm `mysql_affected_rows()` sẽ trả về giá trị 0.
- Hàm này không có tác dụng đối với câu lệnh truy vấn `SELECT`. Để lấy được số dòng trả về (số bản ghi bị tác động) bởi câu lệnh `SELECT` ta dùng hàm `mysql_num_rows()`.

3.7 Thêm mẩu tin

Để thêm mẩu tin, bạn sử dụng hàm mysql_query(chuỗi Insert). Chẳng hạn, chúng ta khai báo trang insert.php để thêm mẩu tin vào bảng tblships có hai cột dữ liệu là ShipID và ShipName như ví dụ trong trang insert.php.

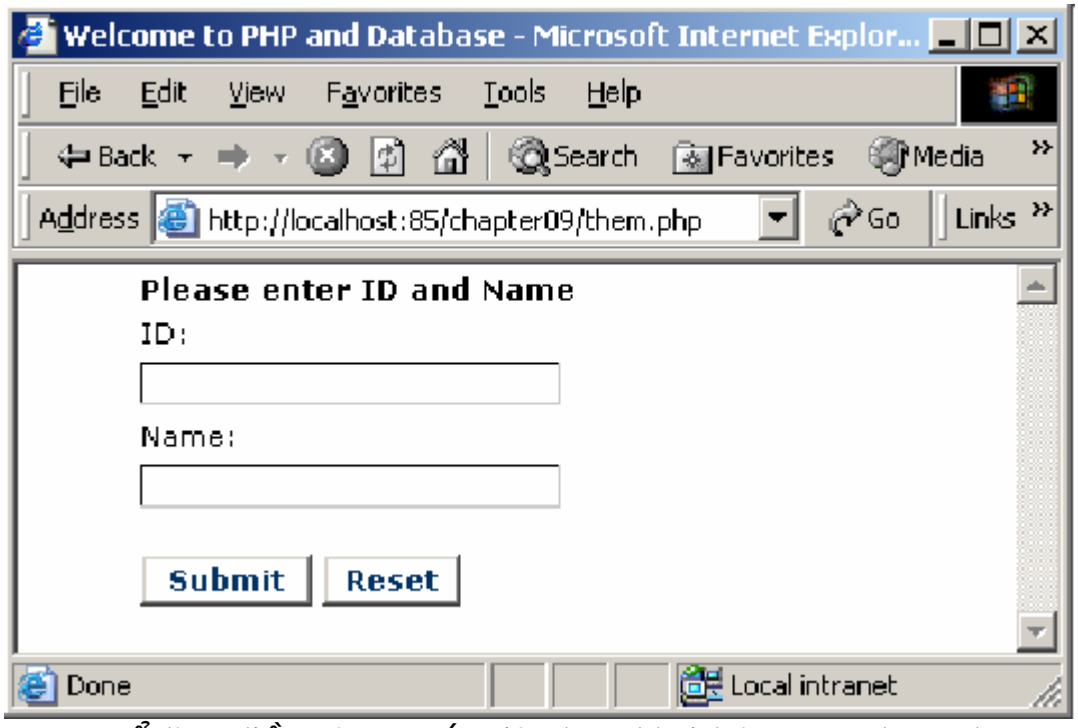
```
<HTML>
  <HEAD>
    <TITLE>::Welcome to PHP and mySQL</TITLE>
  </HEAD>
  <BODY>
    <h3>Them mau tin</h3>
<?php

  require("dbcon.php");
  $sql="insert into tblships
values('A01','Testing')";
  $result = mysql_query($sql,$link);
  $affectrow=0;
  if($result)
     $affectrow=mysql_affected_rows();
  mysql_close($link);
?>
   So mau tin them vao<?= $affectrow?>
 </BODY>
</HTML>
```

Trong đó, bạn sử dụng hàm mysql_query với hai tham số là $sql và $link. Kết quả trả về là số mẩu tin thực thi. Ngoài ra, bạn có thể sử dụng đoạn kết nối cơ sở dữ liệu trong tập tin dbcon.php như ví dụ sau:

```
  <?php
     $link = mysql_connect ("localhost", "root", "")
     or die ("Could not connect to MySQL Database");
     mysql_select_db("Test", $link);
?>
```

Trong trường hợp cho phép người sử dụng thêm mẩu tin thì bạn thiết kế form yêu cầu người sử dụng nhập hai giá trị sau đó submit đến trang kế tiếp để thực thi việc thêm gt sau đó submit đến trang kế tiếp để thực thi việc thêm giá trị vừa nhập vào cơ sở dữ liệu như hình sau.



Để làm điều này, trước tiên bạn khai báo trang them.php, trong đó khai báo đoạn javascript để kiểm tra dữ liệu nhập như sau:

```
<SCRIPT language=JavaScript>
 function checkInput()
     {

       if (document.frmPHP.txtID.value=="")
  {
      alert("Invalid ID, Please enter ID");
   document.frmPHP.txtID.focus();
   return false;
  }
```

```
  if (document.frmPHP.txtName.value=="")
  {
      alert("Please enter Name");
   document.frmPHP.txtName.focus();
   return false;
  }
  return true;
     }
</script>
```

Kế đến khai báo thể form và hai thẻ input lại text yêu cầu người sử dụng nhập ID và Name như sau:

```
<form name="frmPHP" method="post"
action="doinsert.php"
  onsubmit="return checkInput();">
    <tr>
      <td align="left" class="content-sm"><b>
        Please enter ID and Name
        </b></td>
    </tr>
    <tr>
      <td align="left" >ID:</td>
    </tr>
    <tr>
      <td align="left">
        <input type="text" name="txtID" size="25"
                        maxlength="3" class="textbox">
      </td>
    </tr>
    <tr>
      <td align="left" >Name:</td>
    </tr>
    <tr>
      <td align="left" >
        <input type="text" name="txtName" size="25"
                        maxlength="50"
class="textbox">
      </td>
    </tr>
    <tr>
      <td align="left"  valign="top"> <br>
        <input type="submit" value="Submit"
                                    class="button">
        <input type="reset" value="Reset"
                                    class="button">
      </td>
```

```
    </tr>
  </form>
```

Lưu ý rằng, bạn khai báo số ký tự lớn nhất cho phép nhập bằng với kích thước đã khai báo trong cơ sở dữ liệu ứng với thuộc tính maxlength.

Khi người sử dụng nhập hai giá trị và nhấn nút submit, trang kế tiếp được triệu gọi. Trang này lấy giá trị nhập bằng cách sử dụng biến form hay $_POST. Đối với trường hợp này chúng ta sử dụng biến form như trang doinsert.php.

```
<HTML>
  <HEAD>
     <TITLE>::Welcome to PHP and mySQL</TITLE>
  </HEAD>
  <BODY>
     <h3>Them mau tin</h3>
  <?php
      $affectrow=0;
      require("dbcon.php");
      $sql="insert into tblships(ShipID,ShipName) ";
      $sql .=" values('".$txtID."','".$txtName."')";
      $result = mysql_query($sql,$link);
      if($result)
        $affectrow=mysql_affected_rows();
      mysql_close($link);
  ?>
      So mau tin them vao<?= $affectrow?>
   </BODY>
</HTML>
```

3.8 Cập nhật mẩu tin

Đối với trường hợp cập nhật mẩu tin, bạn cũng sử dụng hàm mysql_query với phát biểu Update thay ví Insert như trên, ví dụ chúng ta khai báo trang update.php để cập nhật mẩu tin trong bảng tblShips với tên là UpdateTesting khi mã có giá trị là A01.

```
 <HTML>
   <HEAD>
```

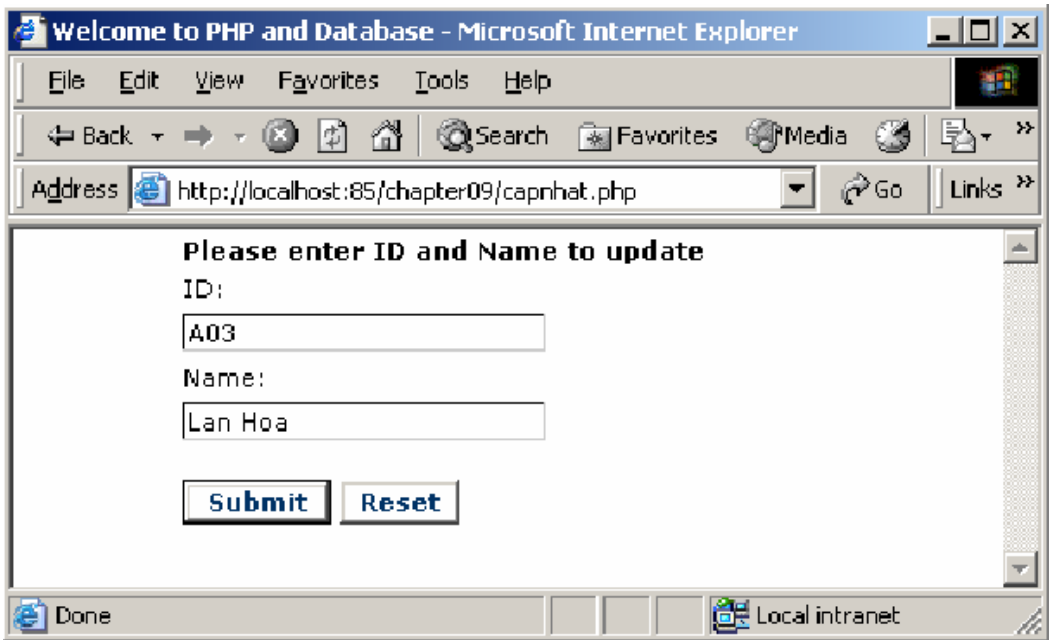
```
      <TITLE>::Welcome to PHP and mySQL</TITLE>
   </HEAD>
   <BODY>
      <h3>Cap nhat mau tin</h3>
<?php

   require("dbcon.php");
   $sql="Update tblships set
ShipName='UpdateTesting'";
   $sql.=" where ShipID='A01'";
   $result = mysql_query($sql,$link);
   $affectrow=0;
   if($result)
     $affectrow=mysql_affected_rows();
   mysql_close($link);
?>
     So mau tin cap nhat <?= $affectrow?>
  </BODY>
</HTML>
```

Lưu ý rằng, để biết số mẩu tin đã thực thi bởi phát biểu SQL bạn sử dụng hàm mysql_affected_rows.

```
if($result)
   $affectrow=mysql_affected_rows();
```

Tương tự như trên, bạn có thể thiết kế form cho phép người sử dụng cập nhật dữ liệu bằng cách thiết kế form yêu cầu người sử dụng nhập mã và tên cập nhận. Trước tiên thiết kế from cho phép nhập dữ liệu để cập nhật như ví dụ trang capnhat.php, sau khi học phần truy vấn xong, thay vì nhập mã bạn cho phép người sử dụng chọn trong danh sách đã có như hình sau.



Sau khi người sử dụng nhấn nút submit, trang doupdate.php sẽ triệu gọi, kết quả trả về 1 hay 0 mẩu tin.

```
<HTML>
  <HEAD>
    <TITLE>::Welcome to PHP and mySQL</TITLE>
  </HEAD>
  <BODY>
    <h3>Cap nhat mau tin</h3>
 <?php
     $affectrow=0;
     require("dbcon.php");
     $sql="update tblships set ShipName='";
     $sql .=$txtName."' where ShipID='".$txtID."'";
     $result = mysql_query($sql,$link);
     if($result)
        $affectrow=mysql_affected_rows();
     mysql_close($link);
 ?>
     So mau tin cap nhat <?= $affectrow?>
  </BODY>
</HTML>
```

3.9 Xóa mẩu tin

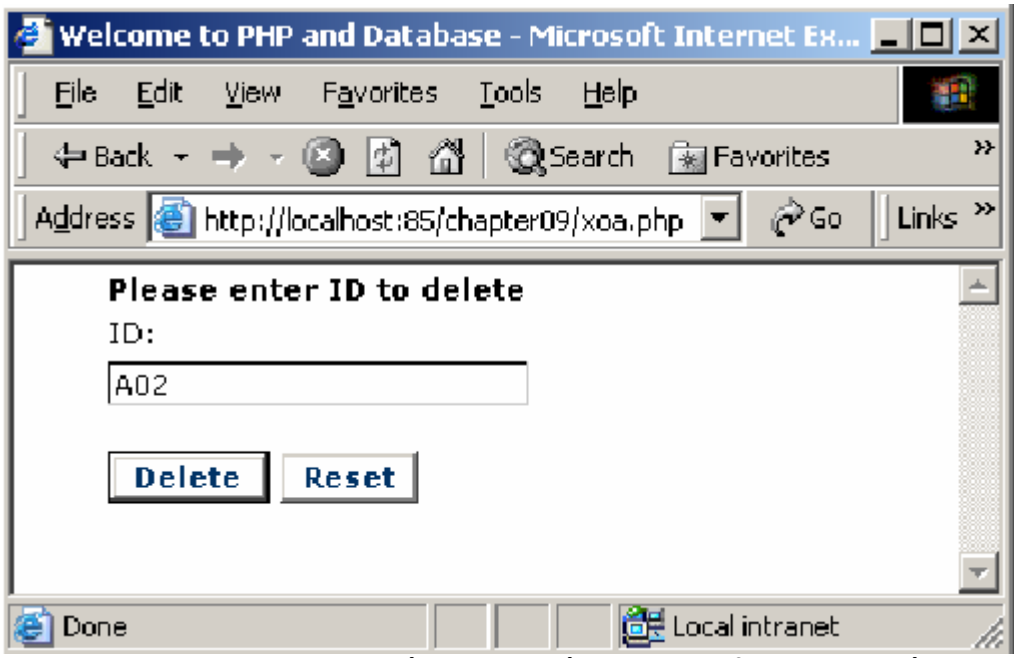
Tương tự như vậy khi xoá mẩu tin, bạn chỉ thay đổi phát biểu SQL dạng Delete như ví dụ trong tập tin delete.php.

```
<HTML>
  <HEAD>
```

```
      <TITLE>::Welcome to PHP and mySQL</TITLE>
   </HEAD>
   <BODY>
      <h3>Xoa mau tin</h3>
 <?php

    require("dbcon.php");
    $sql="Delete From tblships where ShipID='A01'";
    $result = mysql_query($sql,$link);
    $affectrow=0;
    if($result)
       $affectrow=mysql_affected_rows();
    mysql_close($link);
 ?>
      So mau tin da xoa <?= $affectrow?>
   </BODY>
</HTML>
```

Đối với trường hợp xoá thì đơn giản hơn, bạn cỉh cần biết được mã cần xoá, chính vì vậy trong trường hợp này chúng ta chỉ cần thiết kế trang cho phép nhập mã như hình sau.



Sau khi nhập mã cần xoá, nếu người sử dụng nhấn nút Delete lập tức trang dodelete.php sẽ triệu gọi và xoá mẩu tin tương ứng.

```
<HTML>
   <HEAD>
      <TITLE>::Welcome to PHP and mySQL</TITLE>
   </HEAD>
```

```
   <BODY>
      <h3>Xoa mau tin</h3>
 <?php
      $affectrow=0;
      require("dbcon.php");
      $sql="delete from tblships ";
      $sql .=" where ShipID='".$txtID."'";
      $result = mysql_query($sql,$link);
      if($result)
         $affectrow=mysql_affected_rows();
      mysql_close($link);
  ?>
       So mau tin xoa <?= $affectrow?>
   </BODY>
</HTML>
```

4. Phân trang kết quả truy vấn MySQL

4.1 Thiết lập các biến

Khi cơ sở dữ liệu của chúng ta tăng lên, hiển thị tất cả các kết quả truy vấn trên một trang là không thiết thực nữa. Đây là nơi phân trang trở thành có ích. Chúng ta có thể hiển thị các kết quả của chúng ta trên một số trang, mỗi liên kết chuyển tiếp cho phép người dùng của chúng ta duyệt nội dung của chúng ta trong một vùng xác định.

```
<?php
// Kết nối tới cơ sở dữ liệu
mysql_connect("your.hostaddress.com", "username",
"password") or die(mysql_error());
mysql_select_db("address") or die(mysql_error());

//Kiểm tra trang có tồn tại không. Nếu không, nó sẽ
thiết lập số trang bằng 1.
if (!(isset($pagenum)))
{
  $pagenum = 1;
}

//Đếm số kết quả
//Sửa $data thành truy vấn của chúng ta
$data = mysql_query("SELECT * FROM topsites") or
die(mysql_error());
$rows = mysql_num_rows($data);

//Đây là số kết quả hiển thị trên mỗi trang
$page_rows = 4;

//Điều này nói cho chúng ta số trang liền kề sau
cùng
$last = ceil($rows/$page_rows);

//Điều này để chắc chắn số trang không thấp hơn 1
hoặc lớn hơn số trang lớn nhất.
if ($pagenum < 1)
{
   $pagenum = 1;
}
elseif ($pagenum > $last)
{
```

```
    $pagenum = $last;
}

//Thiết lập vùng hiển thị của kết quả truy vấn.
$max = 'limit ' .($pagenum - 1) * $page_rows .',' .
$page_rows;
```

Ở đoạn mã trên, sau khi chúng ta kết nối tới cơ sở dữ liệu, chúng ta cần chắc chắn rằng chúng ta biết các kết quả của trang hiển thị cái gì. Mã `if (!(isset($pagenum)))` để kiểm tra nếu số trang `($pagenum)` không được thiết lập và nếu thiết lập thì nó là 1. Nếu số trang chắc chắn đã được thiết lập, mã này sẽ được lờ đi.

Tiếp theo chúng ta chạy truy vấn của chúng ta. Dòng `$data` nên được sửa để áp dụng cho trang web của chúng ta và trả về cái chúng ta cần đếm các kết quả. Dòng `$rows` thì đếm số kết quả truy vấn của chúng ta.

Tiếp theo chúng ta định nghĩa `$page_rows`. Đây là số các kết quả chúng ta muốn hiển thị trên mỗi trang trước khi chuyển sang các kết quả của trang tiếp theo. Sau đó chúng ta có thể tính tổng số trang chúng ta có `($last)` bằng cách chia giữa tổng các kết quả `($rows)` với số kết quả chúng ta muốn trên một trang. Ở đây chúng ta sử dụng `CEIL` để làm tròn tất cả các số theo cận trên bởi vì nếu có một số thập phân sẽ có nghĩa là chúng ta cần một trang khác.

Tiếp theo chúng ta chạy kiểm tra để chắc chắn rằng số trang thực tế là hợp lệ. Nếu số trang ít hơn 1 hoặc lớn hơn tổng số trang, chúng ta sẽ tái khởi động nó để số trang gần với nội dung.

Cuối cùng, chúng ta thiết lập vùng `($max)` hiển thị các kết quả của chúng ta bằng cách sử dụng hàm `LIMIT`.

4.2 Truy vấn và các kết quả

```
// Vẫn truy vấn giống phần trên, chỉ có sự khác biệt
duy nhất là chúng ta thêm $max vào đó.

$data_p = mysql_query("SELECT * FROM topsites $max")
or die(mysql_error());

//Phần này sữ hiển thị kết quả truy vấn của chúng ta.
while($info = mysql_fetch_array( $data_p ))
{
   Print $info['Name'];
   echo "<br>";
}
   echo "<p>";

//Phần này cho người dùng biết trang họ đang duyệt và
tổng số trang hiện có.
   echo " --Page $pagenum of $last-- <p>";

// Đầu tiên, chúng ta kiểm tra nếu chúng ta đang ở
trang 1 thì chúng ta không cần liên kết đến trang
liền kề trước nó hoặc trang đầu tiên. Nếu chúng ta
không ở trang 1 thì chúng ta tạo liên kết cho trang 1
và trang liền kề trước đó.
if ($pagenum == 1)
{
}
else
{
     echo " <a href='{$_SERVER['PHP_SELF']}?
pagenum=1'>
                  <<-First</a> ";
      echo " ";
      $previous = $pagenum-1;
      echo "<a
       href='{$_SERVER['PHP_SELF']}?
pagenum=$previous'>
          <-Previous</a> ";
}

//just a spacer
```

```
echo " ---- ";

//Phần này tương tự phần trên.
if ($pagenum == $last)
{
}
else {
   $next = $pagenum+1;
   echo " <a href='{$_SERVER['PHP_SELF']}?
pagenum=$next'>Next-></a> ";
   echo " ";
   echo " <a    href='{$_SERVER['PHP_SELF']}?
pagenum=$last'>Last ->> </a> ";
}
?>
```

Cái đầu tiên chúng ta làm là chạy lại truy vấn của chúng ta. Lần này, chúng ta nhúng biến `$max` để hạn chế các kết quả truy vấn để những kết quả này thuộc về trang hiện tại của chúng ta. Sau khi truy vấn chúng ta muốn hiển thị các kết quả của chúng ta (sử dụng định dạng bất kì mà chúng ta mong muốn).

Sau khi kết quả được hiển thị, chúng ta cho người dùng biết trang hiện tại của họ và tổng số trang tồn tại. Điều này là không cần thiết nhưng nó là thông tin tốt để người dùng biết.

Khi chúng ta tạo ra sự điều hướng. Chúng ta thừa nhận rằng nếu chúng ta ở trang 1thì chúng ta không cần liên kết đến trang đầu tiên. Và khi nó là kết quả đầu tiên, không tồn tại trang liền kề trước đó. Do vậy, chúng ta kiểm tra `(if($pagenum == 1))` để xem nếu chúng ta ở trang 1. Nếu ở trang 1 thì không có vấn đề gì xảy ra. Nếu chúng ta không ở trang 1thì chúng ta sử dụng `PHP_SELF` và số trang để tạo ra các liên kết cả trang đầu tiên và trang liền kề trước nó.

Chúng ta thực hiện mọi cái giống nhau để tạo các liên kết trên các site khác, tuy nhiên lần này chúng ta kiểm tra chắc chắn rằng chúng

ta không ở trang cuối cùng. Nếu chúng ta ở trang cuối cùng thì không cần liên kết tới trang cuối cùng hoặc không tồn tại một trang tiếp theo.

4.3 Phân trang là gì?

Nếu chúng ta có một form để cho phép người dùng duyệt các bản ghi trong một bảng cơ sở dữ liệu, chúng ta làm gì nếu bảng đó có hang trăm thậm chí hang nghìn bản ghi? Nó không phải là một ý tưởng tốt để hiển thị tất cả các bản ghi này trên một form. Trong thực tế, chúng ta nên cắt nhỏ cơ sở dữ liệu trên mỗi trang. Có hai công việc chúng ta phải làm:

1. Quyết định số bản ghi cơ sở dữ liệu lớn nhất có thể được hiển thị trên mỗi trang. Chúng ta có thể khó khăn để viết mã giá trị này hoặc chúng ta có thể định nghĩa một biến để giá trị có thể thay đổi lúc thực thi.
2. Sau đó, chúng ta cần cho người dung biết các trang khác là sẵn dung và cung cấp một cơ chế để người dung có thể lựa chọn nội dung của các trang khác nhau. Chúng ta sử dụng một tập các siêu liên kết trong vùng phân trang giống như thế này:

Vùng phân trang

«FIRST ‹PREV Page 2 of 5 NEXT› LAST»

Vùng này cho người dùng biết trang hiện thời đang được hiển thị, tổng số trang sẵn có và bao gồm các liên kết để quay lui hoặc tiến lên các trang sẵn có. Các liên kết được hiển thị là văn bản thuần nếu không có trang liền kề trước hoặc các trang liền kề sau. Chú ý rằng các siêu liên kết này không đặt từ `FIST`, `PREV` hoặc `LAST` là tham số

trong chuỗi url, tất cả chúng xác định số trang theo định dạng `pageno = n`.

Một cách phân trang đơn giản

Để nhúng tất cả các chức năng liên quan trong một kịch bản đơn, chúng ta nên thực hiện theo các bước cơ bản sau:

1. Kiểm tra sự tồn tại của số trang đã yêu cầu.

   Đoạn mã sau sẽ kiểm tra sự tồn tại của số trang đã yêu cầu từ mảng `$_GET`. Chú ý rằng nếu nó không tồn tại nó sẽ mặc định là 1:

```
if (isset($_GET['pageno'])) {
   $pageno = $_GET['pageno'];
} else {
   $pageno = 1;
} // if
```

2. Xác định số bản ghi sẵn có trong cơ sở dữ liệu

   Đoạn mã này sẽ đếm xem có bao nhiêu bản ghi thỏa mãn truy vấn hiện thời.

```
$query = "SELECT count(*) FROM table WHERE ...";
$result = mysql_query($query, $db) or
trigger_error("SQL", E_USER_ERROR);
$query_data = mysql_fetch_row($result);
$numrows = $query_data[0];
```

3. Tính sô của `$lastpage`

   Đoạn mã sau sử dụng các giá trị trong `$rows_per_page` và `$numrows` theo thứ tự và xác định số trang cuối cùng:

```
$rows_per_page = 15;
$lastpage      = ceil($numrows/$rows_per_page);
```

4. Chắc chắn rằng `$pageno` nằm trong vùng phân trang

Đoạn mã sau kiểm tra các giá trị của $pageno là số nguyên giữa 1 và $lastpage:

```
$pageno = (int)$pageno;
if ($pageno > $lastpage) {
   $pageno = $lastpage;
} // if
if ($pageno < 1) {
   $pageno = 1;
} // if
```

5. Xây dựng mệnh đề LIMIT

Đoạn mã sau xây dựng mệnh đề LIMIT cho câu lệnh SELECT của sql:

```
$limit = 'LIMIT ' .($pageno - 1) *
$rows_per_page .',' .$rows_per_page;
```

6. Đưa ra truy vấn cơ sở dữ liệu

Bây giờ chúng ta có thể đưa ra truy vấn cơ sở dữ liệu và xử lý kết quả:

```
$query = "SELECT * FROM table $limit";
$result = mysql_query($query, $db) or
trigger_error("SQL", E_USER_ERROR);
```

7. Xây dựng các siêu liên kết phân trang.

Cuối cùng, chúng ta phải xây dựng các siêu lien kết. Nó sẽ cho phép người dung lựa chọn các trang khác. Chúng ta sẽ bắt đầu với các lien kết cho các trang liền kề phía trước.

```
if ($pageno == 1) {
   echo " FIRST PREV ";
} else {
   echo " <a href='{$_SERVER['PHP_SELF']}?
pageno=1'>FIRST</a> ";
   $prevpage = $pageno-1;
   echo " <a href='{$_SERVER['PHP_SELF']}?
pageno=$prevpage'>PREV</a> ";
}
```

Tiếp theo chúng ta cho người dùng biết vị trí hiện thời của họ trong một dãy các trang sẵn có.

```
echo " ( Page $pageno of $lastpage ) ";
```

Đoạn mã này sẽ cung cấp các liên kết cho các trang liền kề sau.

```
if ($pageno == $lastpage) {
   echo " NEXT LAST ";
} else {
   $nextpage = $pageno+1;
   echo " <a href='{$_SERVER['PHP_SELF']}?
pageno=$nextpage'>NEXT</a> ";
   echo " <a href='{$_SERVER['PHP_SELF']}?
pageno=$lastpage'>LAST</a> ";
}
```

5. Tổng kết

Trao đổi thông tin cũng như lấy dữ liệu từ người dùng là một phần không thể thiếu trong lập trình web động. Chúng ta có hai phương thức để lấy dữ liệu từ người dùng đó là POST và GET. Phương thức GET truyền giá trị của biến lên URL còn phương thức POST thì không. Tùy vào từng hoàn cảnh cụ thể mà chúng ta sử dụng khéo léo hai phương thức này. Chẳng hạn để đăng nhập thì bắt buộc ta phải sử dụng phương thức POST, nhưng để phân trang thì ta lại phải sử dụng phương thức GET.

Cookie là hữu ích cho các ứng dụng cần duy trì trạng thái giữa các lần ghé thăm của người dùng. Các site thương mại điện tử sử dụng các cookie để xác định các khách hàng và thu thập thông tin về các khách hàng.

Chương này cũng giúp chúng ta nhớ lại một số kiến thức cơ bản của cơ sở dữ liệu MySQL. Chúng ta biết cách tạo ra một cơ sở dữ liệu bằng công cụ navicat và kết nôi cơ sở dữ liệu đó vào trang web PHP.

Một phần không thể thiếu trong việc kết nối cơ sở dữ liệu và hiển thị kết quả ra màn hình khi số bản ghi là quá lớn đó là kỹ thuật phân trang. Kỹ thuật phân trang tạo ra một sự thoải mái và dễ chịu cho người dùng khi họ không phải nhìn một đống dữ liệu đồ sộ trên web.

Câu hỏi trắc nghiệm kết chương

Trả lời các câu hỏi sau:

1. Làm thế nào để lấy được thông tin từ một Form sử dụng phương thức GET?
   a. Request.QueryString;
   b. Request.Form;
   c. $_GET[];
2. Khi sử dụng phương thức POST, biến được hiển thị trên URL?
   a. Đúng
   b. Sai
3. Câu lệnh chính xác để kết nối tới một cơ sở dữ liệu MySQL?
   a. dbopen("localhost");
   b. mysql_connect("localhost");
   c. connect_mysql("localhost");
   d. mysql_open("localhost");
4. Để đếm số bản ghi kết quả truy vấn chúng ta sử dụng hàm?
   a. Count();
   b. Mysql_num_rows();
   c. Mysql_fetch_array();
   d. Mysql_fetch_object();
5. Trong kỹ thuật phân trang, phương thức bắt buộc chúng ta phải sử dụng?
   a. GET
   b. POST
   c. Không bắt buộc