SỬ DỤNG PHP & MYSQL THIẾT KẾ WEB ĐỘNG (PDF)

TÀI LIỆU ĐƯỢC TRÍCH TỪ NGUỒN BẢN IN GIẤY "SỬ DỤNG PHP & MYSQL THIẾT KẾ WEB ĐỘNG" DO TÁC GIẢ NGUYỄN TRƯỜNG SINH BIÊN SOẠN
NHÀ SÁCH MINH KHAI PHÁT HÀNH

CHUYỂN ĐỔI NGUYÊN GỐC SANG BẢN PDF BỞI

BEEHOST VIỆT NAM – NHÀ CUNG CẤP HOSTING & DOMAIN CHUYÊN NGHIỆP

Địa chỉ: 31C/1 Phú Mỹ P.22 Q.Bình Thạnh TP.HCM Việt Nam

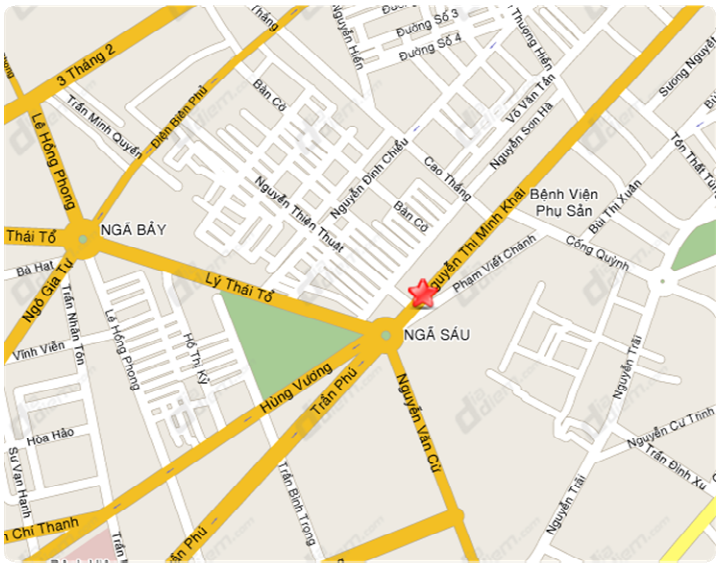
ĐT: (84-8) 5144843 - (84-8) 2168171 Fax: (84-8) 5144842

Email: info@beehost.vn | sales@beehost.vn | support@beehost.vn

Hotline: 0979 554556 – 0909 639586

Giờ làm việc: Sáng 8h – 12h | Chiều 13h – 17h (Chủ Nhật và Lễ nghỉ)

Để có chất lượng tham khảo tốt nhất, bạn hãy mua bản in giấy cuốn sách này tại



Lưu ý: Xem tốt nhất ở độ Zoom 100 % và Resolution 96 pixel/inch

Còn rất nhiều tài liệu và tutorials hỗ trợ khách hàng tại www.beehost.vn

NGUYỄN TRƯỜNG SINH *(Chủ biên)*
LÊ MINH HOÀNG - HOÀNG ĐỨC HẢI

SỬ DỤNG PHP & MySQL THIẾT KẾ WEB ĐỘNG

NHÀ XUẤT BẢN THỐNG KÊ

LỜI NÓI ĐẦU

Ngày nay, các Web site tĩnh với các trang HTML đơn giản không còn đáp ứng được nhu cầu của người dùng. Các Web site động kết hợp với cơ sở dữ liệu đã trở thành xu thế phát triển Web.

Cuốn sách này sẽ giới thiệu với các bạn hai công nghệ phổ biến nhất được dùng để tạo các Web site động: *ngôn ngữ kịch bản PHP và chương trình quản lý cơ sở dữ liệu MySQL*. Thông qua các bước thực hành được hướng dẫn cụ thể, các mã kịch bản rõ ràng với nhiều gợi ý chuyên môn, cuốn sách sẽ giúp bạn tìm hiểu tổng quát về công nghệ trước khi đề cập tới các vấn đề cụ thể như: an toàn, session và cookie, sử dụng các công cụ Web... Ở phần cuối cuốn sách, chúng ta sẽ áp dụng các kỹ thuật đã được giới thiệu để xây dựng hoàn chỉnh một số ứng dụng mẫu. Để thuận tiện cho phần thực hành, bạn có thể sao chép các mã lệnh tham khảo dùng cho cuốn sách từ địa chỉ: *http://minhkhai.com.vn/download.html*.

MK.PUB

Ghi chú: Trong cuốn sách này, ở phần các mã lệnh, biểu tượng (→) cho biết dòng lệnh này nối tiếp với dòng lệnh trên.

LỜI NGỎ

Kính thưa quý Bạn đọc gần xa, Ban xuất bản MK.PUB trước hết xin bày tỏ lòng biết ơn và niềm vinh hạnh trước nhiệt tình của đông đảo Bạn đọc đối với tủ sách MK.PUB trong thời gian qua.

Khẩu hiệu của chúng tôi là:

* Lao động khoa học nghiêm túc.

* Chất lượng và ngày càng chất lượng hơn.

* Tất cả vì Bạn đọc.

Rất nhiều Bạn đọc đã gửi *mail* cho chúng tôi đóng góp nhiều ý kiến quý báu cho tủ sách.

Ban xuất bản MK.PUB xin được kính mời quý Bạn đọc tham gia cùng nâng cao chất lượng tủ sách của chúng ta.

Trong quá trình đọc, xin các Bạn ghi chú lại các sai sót (dù nhỏ, lớn) của cuốn sách hoặc các nhận xét của riêng Bạn. Sau đó xin gửi về địa chỉ:

E-mail: *mk.book@minhkhai.com.vn - mk.pub@minhkhai.com.vn*

Hoặc gửi về: Nhà sách Minh Khai

249 Nguyễn Thị Minh Khai, Q.I, Tp. Hồ Chí Minh

Nếu Bạn ghi chú trực tiếp lên cuốn sách, rồi gửi cuốn sách đó cho chúng tôi thì chúng tôi sẽ xin hoàn lại cước phí bưu điện và gửi lại cho Bạn cuốn sách khác.

Chúng tôi xin gửi tặng một cuốn sách của tủ sách MK.PUB tùy chọn lựa của Bạn theo một danh mục thích hợp sẽ được gửi tới Bạn.

Với mục đích ngày càng nâng cao chất lượng của tủ sách MK.PUB, chúng tôi rất mong nhận được sự hợp tác của quý Bạn đọc gần xa.

"MK.PUB và Bạn đọc cùng làm !"

MK.PUB

MỤC LỤC

Chương mở đầu

Thời đại của các trang Web tĩnh đã qua. Trong nhiều năm trước đây, Web tồn tại như một miền với các trang HTML đơn giản, liên kết với nhau để tạo một site. Ngày nay, người dùng mong muốn các trang Web phải sinh động hơn, được cập nhật thường xuyên và nhiều tùy biến. Đồng thời, người quản trị Website cũng muốn việc duy trì và cập nhật các site được thuận tiện hơn. Vì những lý do này và một số yếu tố khác, việc thiết lập site chỉ với các tập tin HTML tĩnh đã không còn được chấp nhận. Web phải là nơi dành cho các ứng dụng động và kết hợp với cơ sở dữ liệu.

Thông qua các mã lệnh ví dụ, cuốn sách này sẽ hướng dẫn bạn cách bắt đầu phát triển các Website động.

Website động là gì?

Các Website động là những thực thể mạnh và linh hoạt, thường được thể hiện dưới dạng các ứng dụng hơn là các Website đơn thuần. Các ưu điểm của Website động:

Đáp ứng nhiều tham số khác nhau (ví dụ như thời gian của ngày hoặc phiên bản trình duyệt của người dùng).

- Thường có các giao diện cho phép người quản trị có thể quản lý nội dung của site.
- Có "bộ nhớ", cho phép người dùng đăng ký và đăng nhập, thực hiện thương mại điện tử và các quá trình tương tự.
- Dễ dàng duy trì, cập nhật và phát triển.

Có nhiều công nghệ để tạo ra Website động. Các công nghệ thường được sử dụng là ASP (Active Server Pages, sản phẩm của Microsoft), JSP (Java Server Pages), ColdFusion và PHP. Website động không nhất thiết phải phụ thuộc vào cơ sở dữ liệu, nhưng càng ngày sự phụ thuộc đó càng nhiều, đặc biệt khi các ứng dụng cơ sở dữ liệu có sẵn như MySQL... với chi phí rất thấp hoặc miễn phí.

PHP là gì?

PHP là chữ viết tắt của "Personal Home Page" do Rasmus Lerdorf tạo ra năm 1994, để theo dõi người dùng truy cập lý lịch trực tuyến của ông. Vì tính hữu dụng, khả năng phát triển, PHP đã bắt đầu được sử dụng trong môi trường chuyên nghiệp và nó đã trở thành "PHP: Hypertext Preprocessor".

Theo Website chính thức của PHP ở địa chỉ *www.php.net* (xem hình 0-1) thì PHP là "một ngôn ngữ kịch bản nhúng trong HTML".

"PHP nhúng trong HTML", có nghĩa là PHP có thể được đặt rải rác trong HTML, giúp cho việc phát triển các Website động được dễ dàng. PHP là một *ngôn ngữ kịch bản (scripting language)*. Khác với ngôn ngữ lập trình, PHP được thiết kế để chỉ thực hiện điều gì đó sau khi một sự kiện xảy ra (ví dụ, khi người dùng gửi một biểu mẫu hoặc chuyển tới một URL).

Hình 0-1: *Trang chủ của PHP.*

PHP là một *công nghệ phía máy chủ* (server-side) và *không phụ thuộc môi trường* (cross-platform). Cả hai yếu tố này đều rất quan trọng. Khái niệm *công nghệ phía máy chủ* nói đến việc mọi thứ trong PHP đều xảy ra trên máy chủ (ngược với máy khách là máy của người dùng). Tính chất không phụ thuộc môi trường cho phép PHP chạy trên hầu hết các hệ điều hành như Windows, Unix (và nhiều biến thể của nó), Macintosh... Một điều cũng rất quan trọng là các mã kịch bản PHP viết trên máy chủ này sẽ làm việc bình thường trên các máy chủ khác mà không cần chỉnh sửa hoặc chỉnh sửa rất ít.

Cuốn sách này chủ yếu nói về PHP 4, nhưng những thay đổi trong phiên bản 5 (chủ yếu tập trung vào lập trình hướng đối tượng) sẽ không ảnh hưởng đến nội dung hoặc ví dụ trong cuốn sách. Nói chung, PHP nói luôn phát triển theo hướng bảo đảm tính tương thích với các phiên bản trước.

Thông thường, làm việc trên máy chủ với phiên bản PHP mới nhất sẽ thích hợp hơn, nhưng không phải lúc nào bạn cũng kiểm soát được vấn đề này, nên cuốn sách khuyến khích sử dụng mã lệnh không phụ thuộc phiên bản. Trong trường hợp sử dụng các hàm và các biến mới được bổ sung gần đây nhất, sẽ có các ghi chú về giải pháp thay thế.

Tại sao cần dùng PHP?

PHP được sử dụng để phát triển Website động vì nó tốt, nhanh và dễ dàng nghiên cứu hơn các giải pháp khác. PHP có khả năng thực hiện và tích hợp chặt chẽ với hầu hết các cơ sở dữ liệu có sẵn, tính bền vững, linh động và khả năng

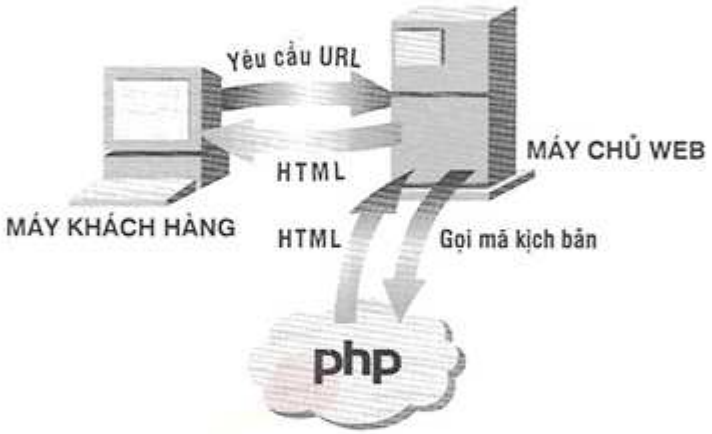
phát triển không giới hạn. Tất cả các đặc tính trên đều miễn phí vì PHP là mã nguồn mở. PHP vừa dễ với người mới sử dụng và vừa có khả năng làm được mọi thứ, đáp ứng yêu cầu của lập trình viên chuyên nghiệp.

PHP được sử dụng càng ngày càng nhiều và mới đây đã bắt kịp ASP (vốn được xem là ngôn ngữ kịch bản phổ biến nhất hiện nay). PHP là môđun thông dụng cho Apache (máy chủ Web phổ biến nhất) và nó đã có mặt trên 12 triệu Website.

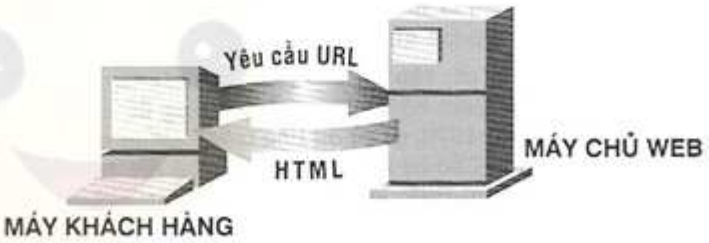
Cách làm việc của PHP

Như đã giới thiệu, PHP là một ngôn ngữ máy chủ, mã lệnh PHP sẽ tập trung trên máy chủ để phục vụ các trang Web theo yêu cầu của người dùng thông qua trình duyệt.

Khi người dùng truy cập Website viết bằng PHP, máy chủ đọc mã lệnh PHP và xử lý chúng theo các hướng dẫn đã được mã hóa. Trong ví dụ ở hình 0-2, mã lệnh PHP yêu cầu máy chủ gửi một dữ liệu thích hợp (mã lệnh HTML) đến trình duyệt Web. Trình duyệt sẽ xem nó như một trang HTML tiêu chuẩn.



Hình 0-2: *Cách làm việc của PHP phù hợp với mô hình khách-chủ khi người dùng yêu cầu một trang Web.*



Hình 0-3: *Quá trình xử lý yêu cầu đối với một trang HTML tĩnh.*

Điều này khác với site HTML tĩnh ở chỗ: Khi có một yêu cầu, máy chủ chỉ đơn thuần gửi dữ liệu HTML đến trình duyệt Web và không xảy ra một sự biên dịch nào từ phía máy chủ (xem hình 0-3). Đối với người dùng cuối và trên trình duyệt

Web, các trang *home.html* và *home.php* trông cũng tương tự nhau, nhưng thực chất nội dung của trang được tạo ra theo cách khác nhau.

MySQL là gì?

MySQL (*www.mysql.com* - xem hình 0-4) là cơ sở dữ liệu phổ biến nhất thế giới, một số người còn cho rằng đây là cơ sở dữ liệu mã nguồn mở tốt nhất. Thật vậy, từ khi phiên bản 4 bổ sung thêm một vài đặc điểm mới thì MySQL đã trở thành đối thủ của những người khổng lồ đất giá như Oracle và SQL Server của Microsoft. Giống như PHP, MySQL có một khả năng thực thi hoàn hảo, rất linh động, đáng tin cậy, dễ nắm bắt và ít chi phí hoặc miễn phí.

Hình 0-4: *Trang chủ ứng dụng cơ sở dữ liệu MySQL.*

MySQL được phát triển và hỗ trợ bởi công ty MySQL AB của Thụy Điển. Nó là hệ thống quản trị cơ sở dữ liệu (DBMS) cho các *cơ sở dữ liệu quan hệ* (vì vậy, MySQL là một RDBMS). Cơ sở dữ liệu là một tập hợp các dữ liệu có liên quan với nhau, có thể là văn bản, số hoặc các tập tin nhị phân được lưu trữ có tổ chức bởi DBMS.

Có nhiều loại cơ sở dữ liệu, từ các tập tin đơn giản đến các tập tin quan hệ và hướng đối tượng. Một cơ sở dữ liệu quan hệ sử dụng nhiều bảng để lưu trữ thông tin trong những phần riêng biệt. Trước 1970, cơ sở dữ liệu trông giống như các bảng tính lớn, đơn giản và lưu trữ mọi thứ. Các cơ sở dữ liệu quan hệ đòi hỏi phải tập trung suy nghĩ nhiều hơn trong giai đoạn thiết kế và lập trình, nhưng chúng có độ tin cậy và tính toàn vẹn dữ liệu tốt hơn. Ngoài ra, các cơ sở dữ liệu quan hệ có thể thực hiện việc tìm kiếm và cho phép nhiều người sử dụng cùng một lúc.

Với việc kết hợp một cơ sở dữ liệu vào ứng dụng Web, nhiều dữ liệu do PHP sinh ra có thể được lấy từ MySQL (xem hình 0-5). Điều này sẽ chuyển nội dung của site từ trạng thái tĩnh (mã hóa cứng) sang trạng thái động và độ linh hoạt chính là chìa khóa cho Website động.

Hình 0-5: *Cách hoạt động của phần lớn các ứng dụng Web động sử dụng PHP và MySQL.*

MySQL là một ứng dụng mã nguồn mở giống như PHP hoặc giống như một vài biến thể của Unix... Nó được dùng miễn phí hoặc có thể sửa đổi (có thể tải mã nguồn). Tuy nhiên, có trường hợp phải trả phí bản quyền MySQL, đặc biệt nếu bạn kiếm tiền từ việc buôn bán hoặc kết hợp với sản phẩm MySQL.

Phần mềm MySQL gồm nhiều phần, trong đó có máy chủ MySQL (mysqld để chạy và quản lý các cơ sở dữ liệu), máy khách MySQL (mysql cung cấp giao diện với máy chủ), các tiện ích để duy trì và dùng cho các mục đích khác. PHP luôn luôn hỗ trợ tốt cho MySQL và điều này càng đúng ở các phiên bản gần đây nhất.

MySQL có thể xử lý những cơ sở dữ liệu rất lớn gần 60.000 bảng với hơn năm tỷ mẩu tin, làm việc với những bảng lớn đến 8 triệu terabytes (từ phiên bản 3.23) trên một số hệ điều hành. Nói chung, nó thường làm việc với những bảng có kích thước 4GB. Cuốn sách này sử dụng MySQL 3.23, nếu máy chủ của bạn phải vận hành phiên bản 4.0 hoặc mới hơn thì các ví dụ vẫn có thể thực hiện mà không gặp trở ngại.

Bạn cần gì?

Để thực hiện ví dụ trong cuốn sách này, bạn cần những công cụ sau:

- Một ứng dụng máy chủ Web (ví dụ Apache, Xitami hoặc IIS).
- PHP.
- MySQL.
- Trình duyệt Web (như Internet Explorer của Microsoft, Navigator của Netscape, Safari của Apple, Mozilla, Opera...).
- Trình soạn thảo văn bản, ứng dụng WYSIWYG có hỗ trợ PHP (Dreamweaver Qualifies của Macromedia) hoặc IDE (Integrated Development Environment: Môi trường phát triển tích hợp).
- Ứng dụng FTP nếu dùng máy chủ ở xa.

Điều hấp dẫn của việc phát triển các Website động dùng PHP và MySQL là tất cả các yêu cầu có thể được đáp ứng mà không tốn tiền, bất kể hệ điều hành đang sử dụng. Apache, PHP và MySQL là những mã nguồn mở, một số trình duyệt

Web là ứng dụng thương mại (ngoại trừ Omni Web, Opera và một số khác) và rất nhiều trình soạn thảo văn bản tốt, có sẵn, miễn phí.

Phụ lục A "Cài đặt", nói về quá trình cài đặt trên hệ điều hành Windows. Nếu máy tính có kết nối Internet, bạn chỉ cần tải xuống một số phần mềm là đã có thể tạo ra các Website động (trong trường hợp này, máy tính của bạn sẽ đại diện cho cả máy chủ lẫn máy khách như trong hình 0-2 và 0-5). Ngược lại, bạn có thể mua dịch vụ ký gửi để tạo ra môi trường trực tuyến có khả năng vận hành với PHP và MySQL (có nhiều công ty làm dịch vụ ký gửi PHP và MySQL miễn phí, nhưng họ thường không thực hiện nhiều yêu cầu của khách hàng).

Nói về cuốn sách

Cuốn sách này hướng dẫn cách phát triển Website động bằng PHP và MySQL, với những kiến thức mà phần lớn các nhà phát triển yêu cầu. Thông tin được đề cập theo từng bước với các mô tả tương ứng và ví dụ thực tế.

Sách được bắt đầu bằng ba chương cơ bản về PHP (qua Chương 2, bạn đã có thể tạo trang Web động đầu tiên của mình). Tiếp theo là hai chương nói về SQL (Structure Query Language, dùng để tương tác với mọi cơ sở dữ liệu) và MySQL để cập đến những vấn đề cơ bản của SQL, thiết kế cơ sở dữ liệu và đặc biệt là ứng dụng MySQL. Kế tiếp là chương đặc biệt nói về sự kết hợp giữa PHP và MySQL (kể từ chương này, PHP sẽ luôn được sử dụng kết hợp với MySQL).

Ba chương tiếp theo sẽ tập trung giới thiệu các công nghệ ứng dụng, làm phong phú thêm kiến thức của bạn. Chương 10 "Các chủ đề mở rộng", đưa ra nhiều ý tưởng đáng để xem xét nhưng không đòi hỏi phải áp dụng trong mọi ứng dụng. Cuối cùng là ba chương ví dụ, hướng dẫn việc phát triển các ứng dụng Web. Các phụ lục bàn về việc cài đặt, các ứng dụng nhóm thứ ba (*third-party application*), cung cấp các nguồn thông tin và tài liệu tham khảo.

Cuốn sách này dành cho bạn?

Cuốn sách này có thể dùng cho nhiều đối tượng, từ người bắt đầu cho đến lập trình viên nâng cao. Trong sách có sử dụng XHTML để tương thích trong tương lai, sẽ rất thuận tiện nếu bạn có một số kiến thức tối thiểu về XHTML hoặc HTML. Mặc dù rất nhiều vấn đề được đề cập, nhưng đây không phải là cuốn sách học về HTML và thiết kế trang Web.

Ngoài ra, sẽ rất thuận lợi cho việc tham khảo cuốn sách này nếu bạn có một trong các yếu tố sau:

- Kiến thức sơ bộ về PHP.
- Đã từng sử dụng một ngôn ngữ lập trình khác (ví dụ, có kỹ năng vững chắc về JavaScript sẽ là một điều thuận lợi).
- Có khả năng tự học và thực hành.

Cuốn sách đề cập đến PHP và MySQL từ A đến Z, hướng dẫn mọi điều cần thiết để phát triển Website thực sự. Vì những chương đầu tiên giới thiệu PHP rất nhanh, nên bạn cũng cần có một số kinh nghiệm lập trình, có sự tìm tòi và suy nghĩ độc lập trong khi nghiên cứu. Cuốn sách không đòi hỏi bạn phải có kinh nghiệm về cơ sở dữ liệu, vì SQL và MySQL được giới thiệu ở mức căn bản.

Chương 1:

GIỚI THIỆU VỀ PHP

Mọi cuộc hành trình đều bắt đầu từ những bước nhỏ và bước đầu tiên trong việc phát triển các ứng dụng Web bằng PHP và MySQL là tìm hiểu các vấn đề cơ bản của ngôn ngữ lập trình PHP.

Mặc dù cuốn sách giới thiệu kết hợp cả MySQL và PHP, nhưng chúng ta sẽ xây dựng nhiều Website chỉ bằng PHP. Trong chương này và chương sau, bạn sẽ tìm hiểu các vấn đề cơ bản của ngôn ngữ lập trình, từ cú pháp, các biến, toán tử cho đến kết cấu ngôn ngữ (điều kiện, vòng lặp...). Đồng thời, các mã lệnh sẽ được phát triển theo từng bước và có thể sử dụng lại chúng cho các ứng dụng lớn hơn ở phần sau của cuốn sách.

Chương này sẽ giới thiệu các khái niệm cơ bản nhất của ngôn ngữ kịch bản Web PHP. Bạn sẽ tìm hiểu về cú pháp viết mã lệnh PHP, cách gửi dữ liệu đến trình duyệt Web, cách sử dụng hai dạng biến (chuỗi và số) và hằng số. Một số ví dụ có vẻ như không có trình tự, nhưng chúng sẽ cung cấp các ý tưởng giúp bạn viết các mã kịch bản cao cấp hơn. Sau khi nắm vững lý thuyết giới thiệu trong chương này, bạn sẽ bắt đầu tạo các Website động của riêng mình.

Cú pháp cơ bản

Như đã giới thiệu, PHP là một ngôn ngữ kịch bản được nhúng trong HTML, nghĩa là có thể trộn mã lệnh PHP và mã lệnh HTML trong cùng mã kịch bản (script).

Đoạn mã 1.1 cung cấp ví dụ về hồ sơ XHTML Transitional, dùng làm nền cho mọi trang Web trong cuốn sách. Để đặt mã PHP trong hồ sơ này, cần bao bọc mã lệnh với các thẻ PHP (*PHP tags*) theo định dạng chính thức (kiểu XML):

```
<?php

?>
```

hoặc theo định dạng không chính thức:

```
<?

?>
```

Đoạn mã 1.1: *Một hồ sơ XHTML 1.0 Transition.*

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
→ Transitional//EN" "http://www.w3.org/TR/2000/REC-xhtml1-
```

```
→ 20000126/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">

<head>

<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />

<title>Tieu de trang</title>

</head>

<body>

</body>

</html>
```

Những gì được đặt trong các thẻ này sẽ được máy chủ Web xử lý như các mã lệnh PHP (trình biên dịch PHP sẽ xử lý mã lệnh thay vì gửi nó trực tiếp cho trình duyệt).

Một điều cần quan tâm cuối cùng cho các mã kịch bản PHP là tập tin phải sử dụng một phần mở rộng mà máy chủ biết để xử lý như là một trang PHP. Phần lớn máy chủ Web sử dụng *.html* hoặc *.htm* cho các trang HTML và *.php* cho mã kịch bản PHP.

Một điều cần quan tâm cuối cùng là tên tập tin chứa các mã kịch bản PHP phải sử dụng phần mở rộng thích hợp để máy chủ có thể nhận biết và xử lý như một trang PHP. Phần lớn máy chủ Web sử dụng phần mở rộng *.html* hoặc *.htm* cho các trang HTML và *.php* cho mã kịch bản PHP.

Tạo mã kịch bản PHP cơ bản theo các bước sau:

1. Tạo một hồ sơ mới trong trình soạn thảo văn bản (xem đoạn mã 1.2).

Bạn có thể sử dụng một trình soạn thảo bất kỳ, như *BBEdit* trên máy Macintosh, *Homesite* hoặc *Notepad* trên máy Windows hoặc *vi* trên Linux.

2. Bắt đầu với hồ sơ HTML:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
→ "http://www.w3.org/TR/2000/REC-xhtml1-20000126/DTD/xhtml1-
→ transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">

<head>

<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />
```

```
<title>Ma kich ban PHP trong</title>
</head>
<body>
</body>
</html>
```

Những cú pháp này sẽ được sử dụng trong toàn bộ cuốn sách, nhưng bạn vẫn có thể thay đổi để phù hợp với chuẩn muốn sử dụng (chẳng hạn như HTML 4.0 Strict).

3. Giữa thẻ body mở và đóng, chèn vào các thẻ PHP:

```
<?php
?>
```

Theo chúng tôi, bạn nên sử dụng các thẻ PHP chính thức (kiểu XML) như trong cuốn sách này.

4. Lưu tập tin với tên **first.php**. Mã lệnh đầy đủ được thể hiện trong đoạn mã 1.2.

Chú ý, nếu không lưu tập tin với phần mở rộng PHP thích hợp, có thể mã kịch bản sẽ không thực hiện đúng.

5. Tải tập tin lên máy chủ Web và mở nó trong trình duyệt (xem hình 1-1).

Tuy là một trang trắng, nhưng đây là trang Web hoàn toàn hợp lệ. Nếu thấy xuất hiện các mã lệnh PHP, có thể là do bạn chưa sử dụng đúng phần mở rộng của tên tập tin hoặc PHP chưa được cài đặt trên máy chủ.

Hình 1-1: *Trông như một trang trắng, nhưng thực tế nó là một mã kịch bản PHP cơ bản và là cơ sở cho tất cả các mã lệnh còn lại trong cuốn sách này.*

Đoạn mã 1.2: *Mã kịch bản PHP đầu tiên này chưa thực hiện bất kỳ điều gì, nó dùng để minh họa cú pháp.*

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
→ Transitional//EN" "http://www.w3.org/TR/2000/REC-xhtml1-
→20000126/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">
<head>
<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />
<title>Ma kich ban PHP trong</title>
</head>
<body>
<?php # Doan ma 1.2 first.php
?>
</body>
</html>
```

Xem Phụ lục D để biết thêm chi tiết về HTML và XHTML.

Có bốn cặp thẻ PHP, ngoài hai cặp thẻ được giới thiệu ở trên, còn có cặp thẻ kiểu ASP (`<%` và `%>`) và kiểu Script (`<script language="php">` và `</script>`), nhưng chúng ít được sử dụng.

Trước khi thực hiện các bước tiếp theo, nếu không chắc về trạng thái của PHP trên máy chủ của mình, bạn có thể sử dụng hàm `phpinfo()` (ví dụ này được minh họa trong Phụ lục A).

Có thể nhúng nhiều đoạn mã PHP trong một hồ sơ HTML. Bạn sẽ thấy các ví dụ về điều này trong toàn bộ cuốn sách.

Vì mã kịch bản PHP cần được biên dịch bởi máy chủ, nên bạn phải truy xuất mã kịch bản PHP thông qua URL (ví dụ, http://localhost/first.php), chứ không đơn giản mở nó trong cửa sổ trình duyệt.

Nếu đã cài đặt một máy chủ Web và PHP có trong máy của mình, bạn có thể thử các mã kịch bản PHP mà không cần sử dụng máy chủ trên mạng.

Gửi dữ liệu đến trình duyệt Web

Để bắt đầu xây dựng Website động với PHP, chúng ta sẽ tìm hiểu cách gửi dữ liệu đến trình duyệt Web. PHP có một số hàm cài đặt sẵn để thực hiện công việc này. Thông dụng nhất là các hàm `echo()` và `print()`.

```
echo 'Hello, World!';
```

```
print "It's nice to see you";
```

Trong ví dụ trên, các dấu nháy đơn hoặc nháy kép đều có thể sử dụng được (sự khác biệt giữa chúng sẽ nói tới ở cuối chương). Chú ý, tất cả các câu lệnh PHP (dòng mã lệnh) phải được kết thúc bằng dấu chấm phẩy.

Tìm kiếm lối thoát

Một vấn đề phiền phức trong việc gửi dữ liệu xuống trình duyệt Web là việc muốn gửi cả dấu nháy đơn và dấu nháy kép. Cả hai mã lệnh dưới đây đều gây ra lỗi (xem hình 1-2):

```
echo  "She said, "How are you?"";

print 'I'm just ducky.';
```

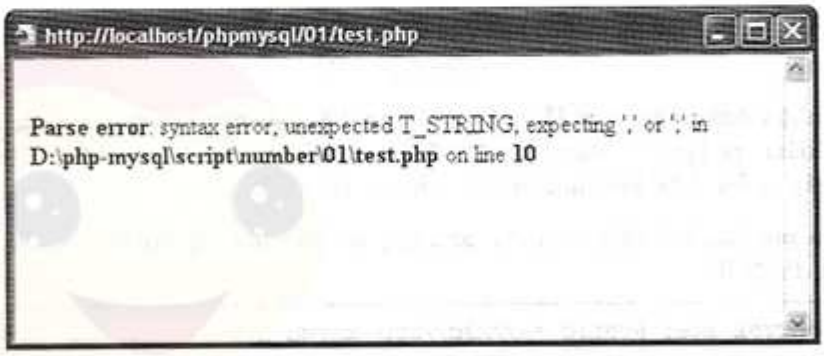
Có hai giải pháp cho vấn đề này. Trước hết, hãy sử dụng dấu nháy đơn khi muốn in dấu nháy kép và ngược lại.

```
echo  'She said, "How are you?"';

print "I'm just ducky.";
```

Hoặc, có thể mã hóa escape các ký tự có vấn đề bằng cách đặt trước nó một dấu chéo ngược:

```
echo  "She said, \"How are you?\"";

print 'I\'m just ducky.';
```

Cách sử dụng dấu chéo ngược để mã hóa ký tự đặc biệt là một khái niệm quan trọng và điều này sẽ được đề cập chi tiết ở cuối chương.



Hình 1-2: *Đây có thể là lỗi biên dịch đầu tiên mà bạn gặp phải khi viết mã PHP.*

Thực hành gửi dữ liệu đến trình duyệt Web thông qua các bước sau:

1. Mở tập tin **first.php** (xem lại đoạn mã 1.2) trong trình soạn thảo.

2. Giữa các thẻ PHP, bổ sung một thông điệp đơn giản. Ví dụ:

```
echo 'Day la ma kich ban PHP thu hai.';
```

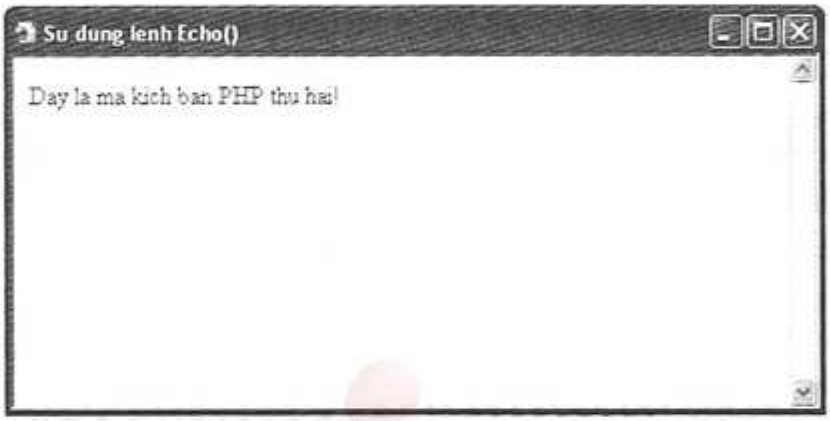
Khi nhập thông điệp, hàm hoặc loại dấu nháy nào được sử dụng đều không quan trọng, chỉ nên cẩn thận nếu muốn in dấu nháy đơn hoặc nháy kép trong thông điệp (xem lại mục *"Tìm kiếm lối thoát"* ở phần trên).

3. Nếu muốn, hãy thay đổi tiêu đề trang thành:

```
<title>Su dung lenh()</title>
```

Việc thay đổi này chỉ là một tùy chọn có tính trình bày.

4. Lưu tập tin với tên **second.php**, tải nó lên máy chủ Web và thử trong một trình duyệt Web (xem hình 1-3). Mã lệnh đầy đủ của tập tin này được thể hiện trong đoạn mã 1.3.



Hình 1-3: *Tuy kết quả vẫn chưa thú vị, nhưng đây là trang Web được tạo từ mã kịch bản PHP.*

Nếu nhận được thông báo lỗi thay vì thông điệp của mình (xem lại hình 1-2), bạn nên kiểm tra lại các dấu nháy mở, đóng và mã hóa tất cả các ký tự đặc biệt có thể gây ra lỗi. Nhớ kết thúc câu lệnh bằng dấu chấm phẩy.

Đoạn mã 1.3: *Với việc sử dụng `print()`, `echo()`, PHP có thể truyền dữ liệu đến trình duyệt Web.*

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
-> Transitional//EN" "http://www.w3.org/TR/2000/REC-xhtml1-
->20000126/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
-> lang="en">

<head>
```

```
<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />
<title>Su dung lenh Echo()</title>
</head>
<body>
<?php # Doan ma 1.3 second.php
echo 'Day la ma kich ban PHP thu hai!';
?>
</body>
</html>
```

*Về mặt kỹ thuật, **echo()** và **print()** là thành phần kết cấu của ngôn ngữ chứ không phải là hàm. Tuy nhiên, để thuận tiện, chúng ta vẫn gọi nó là "hàm".*

*Có thể sử dụng hàm **echo()** và **print()** để gửi mã lệnh HTML đến trình duyệt Web (xem hình 1-4):*

```
echo '<b>Hello<font size="+2"> world</font></b>';
```

Hình 1-4: *PHP có thể gửi mã lệnh HTML hoặc văn bản đơn giản đến trình duyệt Web.*

*PHP là ngôn ngữ không phân biệt chữ hoa và chữ thường, nên sử dụng mã lệnh **ECHO()**, **echo()** hoặc **Echo()** ... đều được.*

*Cuốn sách này còn giới thiệu một số hàm khác cũng được dùng để in là: **print_r()**, **var_dump()** để in các giá trị của một biến và **printf()** để định dạng những gì muốn in ra.*

*Khác với hàm **print()**, có thể dùng hàm **echo()** với dấu phẩy phân cách để gửi nhiều mẩu dữ liệu tới trình duyệt Web. Ví dụ:*

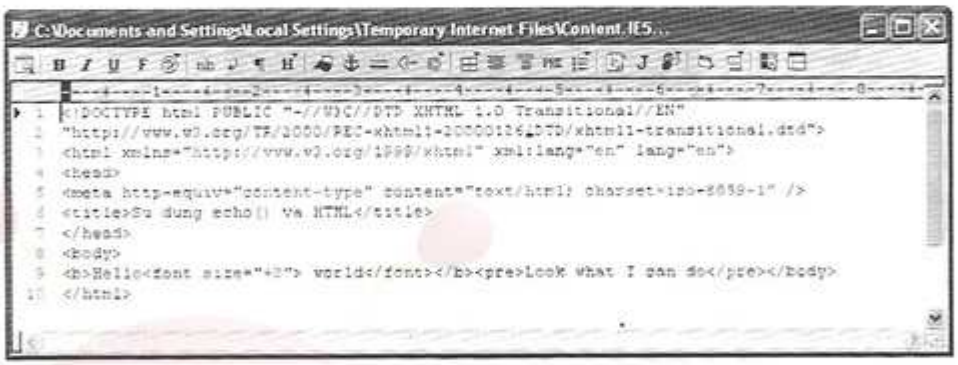
```
echo 'Hello, ', 'world!';
```

*Hàm **echo()** và **print()** đều có thể in ra văn bản nhiều dòng. Bạn sẽ thấy điều này ở phần sau.*

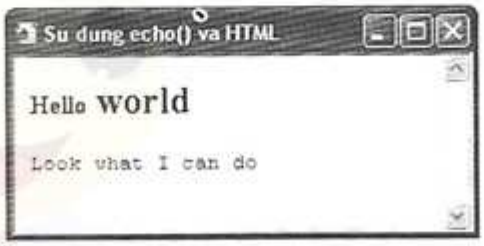
Tìm hiểu PHP, HTML và khoảng trắng

Trước khi đi sâu vào việc sử dụng PHP và dữ liệu khác để gửi cho trình duyệt Web, một điều quan trọng là cần nắm được những gì sẽ xảy ra. Với PHP, bạn có thể truyền dữ liệu (tổ hợp của các thẻ HTML và văn bản) đến trình duyệt Web và đến lượt mình, trình duyệt Web sẽ thể hiện nó dưới dạng trang Web mà người dùng sẽ nhìn thấy. Như vậy, điều bạn đang thực hiện với PHP là tạo ra mã nguồn HTML của trang Web. Về mặt hình ảnh, điều này nghĩa là PHP sẽ phát sinh ra mã nguồn như trong hình 1-5 (truy xuất bằng cách chọn View > Source hoặc View > Page Source, tùy theo trình duyệt đang dùng) và trình duyệt sẽ thể hiện kết quả này như trong hình 1-6.

Trước khi đi sâu vào việc sử dụng PHP để gửi HTML và các dạng dữ liệu khác cho trình duyệt Web, bạn phải biết chính xác những gì sẽ xảy ra. Với PHP, bạn có thể truyền dữ liệu (các thẻ HTML và văn bản) đến trình duyệt Web và trình duyệt sẽ thể hiện chúng dưới dạng trang Web để người dùng có thể xem được. Như vậy, chúng ta đã dùng PHP để tạo mã nguồn HTML cho trang Web. PHP phát sinh mã nguồn như hình 1-5 (truy xuất bằng cách chọn View > Source hoặc View > Page Source, tùy theo trình duyệt) và kết quả thể hiện trong trình duyệt như hình 1-6.



Hình 1-5: *Dùng PHP để tạo mã nguồn HTML cho trang Web.*



Hình 1-6: *Kết quả thể hiện trong trình duyệt.*

Nói chung, PHP và HTML là các ngôn ngữ không "nhạy cảm" với khoảng trắng, khoảng trắng có thể được đặt xung quanh để các mã lệnh trở nên rõ ràng và dễ

đọc hơn. Chỉ có khoảng trắng đơn có ảnh hưởng đến sự thể hiện của trang Web (nhiều khoảng trắng liên tiếp sẽ chỉ thể hiện dưới dạng một khoảng trắng đơn). Kết quả là mã lệnh trong hình 1-7 sẽ tạo ra trang Web giống như hình 1-6.

Hình 1-7: *Các khoảng trắng dư thừa sẽ không ảnh hưởng đến sự thể hiện của trang Web (xem lại hình 1-6), nhưng chúng làm cho mã nguồn dễ đọc hơn.*

Có thể thay đổi khoảng trống của trang Web hoàn tất bằng các thẻ HTML **
** (ngắt dòng) và ** ** (đoạn văn bản). Sử dụng một trong hai cách sau để thay đổi khoảng trống của mã nguồn HTML được tạo ra từ PHP:

- Sử dụng **echo()** hoặc **print()** trên một số dòng.
- Sử dụng ký tự xuống dòng **(\n)** trong phạm vi dấu nháy kép.

Thực hành tạo khoảng trắng theo các bước sau:

1. Mở tập tin **second.php** (xem lại mã kịch bản 1.3) trong trình soạn thảo văn bản.
2. Nếu muốn, có thể thay đổi tiêu đề trang thành **<title>Khoang trang</title>**.
3. Sau thẻ PHP đầu tiên, nhấn phím Enter hoặc Return.

Một dòng trắng sẽ được tạo bởi phím Enter (hoặc Return) và điều này không ảnh hưởng tới mã kịch bản PHP, mã nguồn HTML và trang Web được tạo. Nó chỉ làm cho phần mã kịch bản PHP trở nên rõ ràng và dễ đọc hơn.

4. Thay đổi trạng thái câu lệnh **echo()** hiện có để nó nằm trên nhiều dòng:

```
echo 'Lệnh echo() này
nằm trên hai dòng!';
```

Khi mã kịch bản này được thực hiện, phím Enter/Return ở giữa câu (làm cho từ "nằm" bắt đầu một dòng mới) sẽ tạo mã nguồn HTML trên hai dòng.

5. Nhập một lệnh **echo()** khác, sử dụng ngắt dòng và ký tự xuống dòng:

```
echo "<br />Dòng này sẽ xuất hiện trên một hàng riêng trong
→ trang web.\n\n";
```

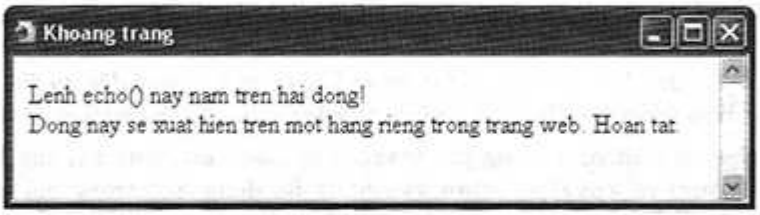
Có hai mục tiêu được thực hiện trong câu lệnh trên. Đầu tiên, gửi một ký tự ngắt dòng tới trình duyệt để hiển thị câu này trên một dòng mới. Thứ nhì, kết thúc chuỗi lệnh với hai ký tự xuống dòng. Chú ý, các lệnh bổ sung này chỉ ảnh hưởng tới mã nguồn HTML (để sử dụng ký tự xuống dòng, bạn phải sử dụng dấu nháy kép để bao chuỗi cần thể hiện).

6. Bổ sung câu lệnh **echo()** cuối cùng:

```
echo 'Hoàn tất.';
```

Ở câu lệnh trên, chúng ta đã sử dụng dấu nháy đơn để bao chuỗi văn bản. Trong trường hợp này, nếu muốn in ký tự nháy đơn trong chuỗi, bạn phải mã hóa Escape ký tự nháy đơn.

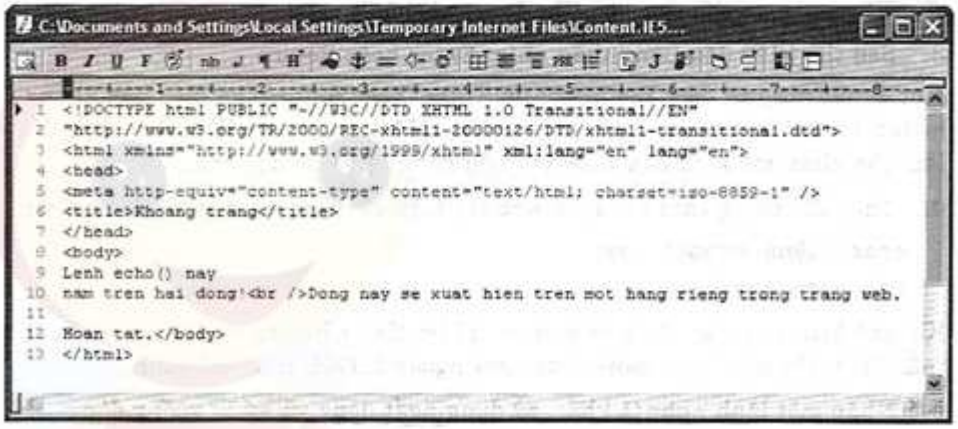
7 Lưu tập tin theo tên **whitespace.php**, tải nó lên máy chủ Web và thử trong trình duyệt (xem hình 1-8). Mã lệnh đầy đủ của tập tin này được thể hiện trong đoạn mã 1.4.



Hình 1-8: *Cách duy nhất để thay đổi khoảng trống của một trang Web là sử dụng các thẻ HTML như* **
**.

Bạn sẽ thấy rằng chỉ có mã lệnh ngắt dòng **
** tạo ra sự thay đổi trong trang Web.

8. Xem mã nguồn của trang (xem hình 1-9).



Hình 1-9: *Mã kịch bản PHP đã tạo ra mã nguồn HTML này và trình duyệt sẽ chuyển nó thành trang Web như hình 1-8.*

Tùy theo trình duyệt, bạn có thể xem mã nguồn bằng một trong các cách sau:

- Chọn menu View và chọn Source hoặc Page Source.
- Nhấp nút chuột phải vào trang Web và chọn View Source hoặc View Page Source (Windows).
- Nhấn phím Control và nhấp chuột vào trang Web và chọn View Source hoặc View Page Source (Macintosh).

Trong mã nguồn HTML, bạn sẽ thấy các khoảng trắng trong mã lệnh PHP không được truyền tới trình duyệt. Việc sử dụng các ký tự xuống dòng (tham khảo đoạn mã 1.4) hoặc tiếp tục câu lệnh trên nhiều dòng sẽ ảnh hưởng đến mã nguồn HTML được tạo.

Đoạn mã 1.4: *Mã kịch bản này minh họa nhiều dạng khoảng trắng trong PHP, HTML và trang Web.*

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
→ Transitional//EN" "http://www.w3.org/TR/2000/REC-xhtml1-
→ 20000126/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">

<head>

<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />

<title>Khoang trang</title>

</head>

<body>

<?php # Doan ma 1.4 whitespace.php

echo 'Lenh echo() nay

nam tren hai dong!';

echo "<br />Dong nay se xuat hien tren mot hang rieng trong
trang web.\n\n";

echo 'Hoan tat.';

?>

</body>

</html>
```

Dạng ghi chú thứ ba bắt nguồn từ ngôn ngữ lập trình C, cho phép ghi chú trên nhiều dòng:

```
/* Đây là một ghi chú dài
xuất hiện trên hai dòng */
```

Thực hành ghi chú mã lệnh theo các bước sau:

1. Mở tập tin **whitespace.php** (tham khảo đoạn mã 1.4) trong trình soạn thảo văn bản.
2. Sau thẻ PHP khởi đầu, nhập vào ghi chú sau:

```
# Tạo ngày 10-04-2005
# Mã lệnh này chưa thực hiện gì nhiều.
```

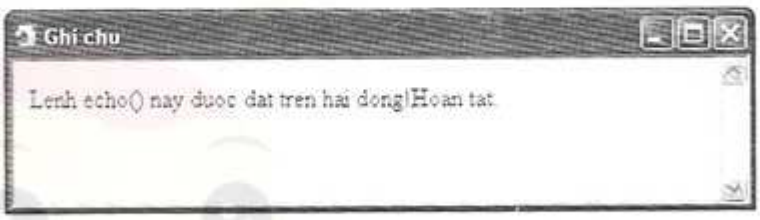
Việc bao bọc một khối lệnh bất kỳ với dạng ghi chú của ngôn ngữ C làm cho khối lệnh mất tác dụng mà không phải xóa nó. Khi cần, bạn có thể xóa ký hiệu ghi chú để khối lệnh có hiệu lực trở lại.

3. Bổ sung ghi chú cuối cùng ở dòng lệnh **echo()** sau cùng:

```
echo 'Hoan tat.'; // Ket thuc ma lenh PHP
```

Đây là một ví dụ về cách bổ sung ghi chú vào cuối dòng lệnh.

4. Nếu muốn, bạn có thể thay đổi tiêu đề của trang Web.
5. Lưu tập tin với tên **comments.php**, tải lên máy chủ Web và thử nó trong trình duyệt (xem hình 1-11). Mã lệnh đầy đủ của tập tin này được thể hiện trong đoạn mã 1.5.



Hình 1-11: *Các ghi chú PHP trong mã kịch bản 1.5 không xuất hiện trong trang Web, chúng có tác dụng làm bỏ qua một dòng trong mã nguồn.*

Đoạn mã 1.5: *Các ghi chú sẽ rất cần thiết khi mã kịch bản của bạn trở nên phức tạp.*

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
→ Transitional//EN" "http://www.w3.org/TR/2000/REC-xhtml1-
→20000126/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
```

```
→ lang="en">
<head>
<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />
<title>Ghi chu</title>
</head>
<body>
<?php # Doan ma 1.5 comments.php
# Tao ngay 10-04-2005
# Ma kich ban nay hien chua lam gi nhieu.
echo 'Lenh echo() nay duoc
dat tren hai dong!';
/*
echo "<br />Dong nay se xuat hien tren mot hang rieng trong
→ trang web.\n\n";
*/
echo 'Hoan tat.'; // Ket thuc ma lenh PHP.
?>
</body>
</html>
```

Với các kiểu ghi chú dạng ngôn ngữ C, không nên tạo các ghi chú lồng nhau (đặt một ghi chú bên trong một ghi chú khác) vì có thể gây ra lỗi.

Bạn có thể đặt một ghi chú PHP bất kỳ ở cuối câu lệnh để ghi chú cho câu lệnh này, ví dụ:

```
echo 'Howdy.'; /*In ra "Howdy"*/
```

Không nên quá lạm dụng việc ghi chú trong mã lệnh.

Vì trang sách có giới hạn nên chúng tôi không thực hiện được đầy đủ việc ghi chú cho tất cả các dòng lệnh.

Biến là gì?

Biến được xem như một thành phần dùng để lưu trữ tạm thời các giá trị. Các giá trị này có thể là số, văn bản hoặc các dạng dữ liệu phức tạp khác. Biến tồn tại

như "linh hồn" của nhiều ngôn ngữ lập trình và nắm vững về nó là chìa khóa cho việc sử dụng PHP.

Theo tài liệu của PHP, có tám dạng biến trong ngôn ngữ này: Bốn kiểu dữ liệu đơn là *Boolean* (*true* và *false*), *số nguyên*, *chấm phẩy động* (thập phân) và *chuỗi* (văn bản). Hai kiểu dữ liệu đa giá trị là *mảng* và *đối tượng*. Cuối cùng là kiểu dữ liệu *tài nguyên* (được sử dụng khi tương tác với cơ sở dữ liệu) và *NULL* (một biến đặc biệt không có giá trị).

Cho dù kiểu dữ liệu mà bạn tạo ra là gì thì các biến trong PHP cũng tuân theo các quy tắc nhất định:

Bất chấp kiểu dữ liệu, các biến trong PHP luôn tuân theo các quy tắc nhất định sau:

- Tên biến phải bắt đầu với một dấu đô la ($), ví dụ: **$name**.
- Tên biến có thể chứa các chuỗi, con số và dấu gạch dưới, ví dụ: **$my_report1**.
- Ký tự đầu tiên sau dấu **$** không thể là số (phải là chữ cái hoặc dấu gạch dưới).
- Các tên biến trong PHP có phân biệt chữ hoa và chữ thường. Ví dụ: **$name** và **$Name** là hai biến hoàn toàn khác nhau.
- Các biến có thể được gán giá trị bằng cách sử dụng dấu bằng (=).

Phần trên chỉ là một số giới thiệu ngắn gọn về biến. Chúng ta sẽ cùng tìm hiểu và thao tác với các kiểu cụ thể (chẳng hạn như chuỗi và số) ở phần sau của chương này.

Để bắt đầu làm việc với biến, bạn sẽ sử dụng một số biến định nghĩa sẵn với giá trị được thiết lập tự động khi thực hiện mã kịch bản PHP.

Thực hành việc in ra giá trị của các biến định nghĩa sẵn theo các bước sau:

1. Tạo một hồ sơ HTML mới trong trình soạn thảo văn bản:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
→ Transitional//EN""http://www.w3.org/TR/2000/REC-xhtml1-
→20000126/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">
<head>
<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />
<title>Cac bien dinh nghia san</title>
</head>
<body>
</body>
</html>
```

2. Trong phạm vi của thẻ `<body></body>`, bổ sung cặp thẻ `PHP` và ghi chú đầu tiên của bạn:

```
<?php # Doan ma 1.6 - predefined.php

?>
```

Từ bây giờ, mã kịch bản sẽ không có các ghi chú về người tạo, ngày tạo... nhưng bạn vẫn nên bổ sung các ghi chú này trong mã lệnh thực tế của mình.

3. In ra giá trị của mã kịch bản đang chạy:

```
echo "Ban dang chay tap tin
→ <b>$PHP_SELF</b>.<br/><br/>\n";
```

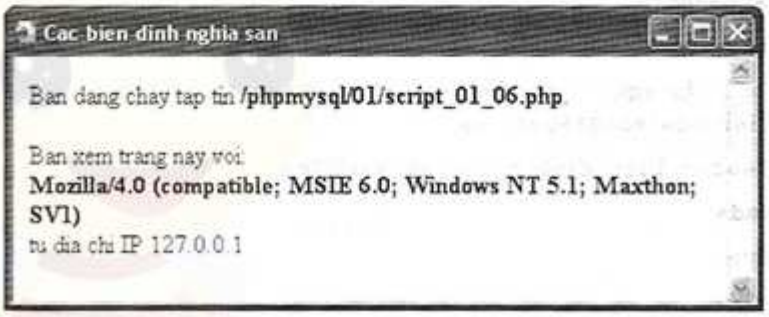
Biến định nghĩa sẵn đầu tiên được sử dụng trong ví dụ này là `$PHP_SELF`. Nó tham chiếu đến mã kịch bản đang chạy. Phụ thuộc vào máy chủ Web, giá trị của biến này có thể là tên của tập tin mã kịch bản (ví dụ: `scriptname.php`) hoặc đường dẫn đầy đủ tới tập tin mã kịch bản (ví dụ: `/path/to/scriptname.php`). Chú ý, biến này phải được in ra phía trong dấu nháy kép và chúng ta cũng sử dụng một ngắt dòng HTML và một ký tự xuống dòng của PHP.

4. In ra thông tin về người dùng truy cập mã kịch bản:

```
echo 'Bạn xem trang này với:<br/><b>', $HTTP_USER_AGENT,
→ '</b><br/>từ địa chỉ IP ', $REMOTE_ADDR;
```

Ở dòng lệnh trên, hai biến định nghĩa sẵn đã được sử dụng. Biến đầu tiên là `$HTTP_USER_AGENT`, tham chiếu đến hệ điều hành, loại trình duyệt và phiên bản trình duyệt đang dùng để truy cập trang Web. Biến thứ hai là `$REMOTE_ADDR`, tham chiếu đến địa chỉ IP của người dùng đang truy xuất trang Web. Để thuận tiện, mã kịch bản này tận dụng khả năng của câu lệnh `echo()` để in ra một loạt các giá trị cách nhau bằng dấu phẩy (như đã đề cập ở phần trước). Cũng có thể đạt được kết quả tương tự nếu đặt toàn bộ các giá trị này bên trong cặp dấu nháy kép.

5. Lưu lại tập tin theo tên **predefined.php**, tải lên máy chủ Web và thử nó trong trình duyệt (xem hình 1-12). Mã lệnh đầy đủ của tập tin này được thể hiện trong đoạn mã 1.6.



Hình 1-12: *Mã kịch bản* ***predefined.php*** *in ra các thông tin về mã kịch bản và trình duyệt đang sử dụng để xem nó.*

Đoạn mã 1.6: *Mã kịch bản này in ra các biến PHP được định nghĩa sẵn.*

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
→ Transitional//EN" "http://www.w3.org/TR/2000/REC-xhtml1-
→20000126/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">
<head>
<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />
<title>Cac bien dinh nghia san</title>
</head>
<body>
<?php # Doan ma 1.6 - predefined.php
// Hien thi ten ma kich ban.
echo "Ban dang chay tap tin <b>$PHP_SELF</b>.<br/><br/>\n";
// Hien thi thong tin nguoi dung
echo 'Ban xem trang nay voi:<br/><b>', $HTTP_USER_AGENT,
→ '</b><br/>tu dia chi IP ', $REMOTE_ADDR;
?>
</body>
</html>
```

Nếu có thể, hãy thử mã kịch bản này trong một trình duyệt khác (xem hình 1-13).

Hình 1-13: *Mã kịch bản này có tính "động" thực sự. Nội dung của trang Web được thay đổi tùy theo trình duyệt mà người dùng sử dụng.*

Nếu mã kịch bản của bạn không in ra bất kỳ giá trị nào ứng với các biến đã được định nghĩa, hãy tham khảo mục "Thiết lập register_globals" trong Chương 2 "Lập trình với PHP" để biết cách khắc phục.

Điều quan trọng cần nhớ khi tạo biến là sử dụng quy tắc đặt tên nhất quán. Bạn có thể tham khảo điều này tại trang web: http://alltasks.net/code/php_coding_standard.html.

PHP rất "dễ dãi" trong việc xử lý biến, nghĩa là không cần phải khởi tạo chúng (không phải gán giá trị khi khai báo) hoặc khai báo chúng (xác định kiểu dữ liệu cho biến) và có thể chuyển đổi từ dạng này sang dạng khác mà không gặp phải vấn đề gì.

Giới thiệu về biến kiểu chuỗi (string)

Kiểu biến quan trọng đầu tiên được đề cập là *chuỗi* (string). Chuỗi là một nhóm các ký tự, số, khoảng trắng, dấu ngắt... được đặt trong các dấu nháy. Ví dụ:

- `'Joe'`
- `"In watermelon sugar"`
- `'1,000'`
- `'January 9, 2004'`

Để tạo biến chuỗi, phải gán giá trị chuỗi cho một biến hợp lệ:

```
$first_name = "Joe";
$today = 'January 9, 2005';
```

Để in ra giá trị của chuỗi, sử dụng `echo()` hoặc `print()` với các dấu nháy kép hoặc không sử dụng dấu nháy:

```
echo "Hello, $first_name";
echo $first_name;
```

Trong ví dụ trước, bạn đã có dịp làm việc với các chuỗi trong khi sử dụng các biến định nghĩa sẵn. Bây giờ, chúng ta sẽ tiếp tục thực hành sử dụng chuỗi theo các bước dưới đây:

1. Tạo một hồ sơ HTML mới trong trình soạn thảo văn bản với các thẻ HTML như sau:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
→ Transitional//EN" "http://www.w3.org/TR/2000/REC-xhtml1-
→20000126/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">
```

```
<head>
<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />
<title>Chuoi</title>
</head>
<body>
<?php # Doan ma 1.7 - strings.php
?>
</body>
</html>
```

2. Trong phạm vi cặp thẻ PHP, tạo ba biến:

```
$first_name = 'Nguyen Van';
$last_name = 'A';
$book = 'ABC';
```

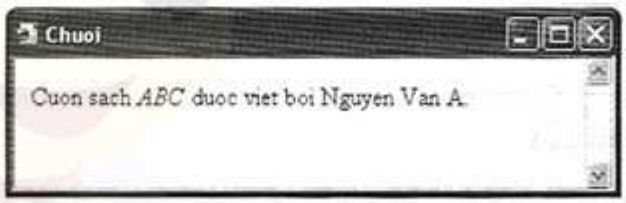
Ở ví dụ này, ba biến được tạo là: **$first_name**, **$last_name** và **$book**.

3. Tạo câu lệnh **echo()**:

```
echo "Cuon sach <i>$book</i> duoc viet boi $first_name
→ $last_name.";
```

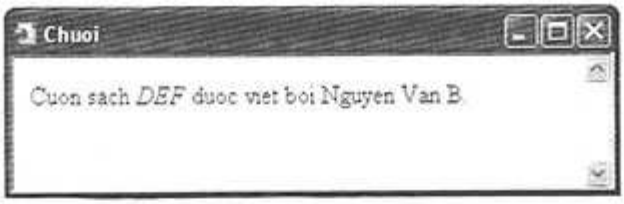
Mã kịch bản trên sẽ in ra một chuỗi với giá trị của các biến. Một định dạng HTML cũng được bổ sung để kết quả trở nên hấp dẫn hơn. Bạn nhớ sử dụng dấu nháy kép để giá trị các biến được in ra một cách chính xác (chi tiết về dấu nháy kép sẽ được đề cập ở cuối chương).

4. Lưu lại tập tin theo tên **strings.php**, tải lên máy chủ và chạy thử nó trong trình duyệt Web (xem hình 1-14). Mã lệnh đầy đủ của tập tin này được thể hiện trong đoạn mã 1.7.



Hình 1-14: *Giá trị của các biến được in ra trong trang Web kết quả.*

5. Nếu muốn, bạn có thể thay đổi giá trị của các biến và chạy thử lại mã kịch bản này (xem hình 1-15).



Hình 1-15: *Kết quả in ra sẽ thay đổi khi chúng ta thay đổi giá trị của các biến.*

Đoạn mã 1.7: *Trong mã kịch bản này, các biến chuỗi được tạo và các giá trị của chúng sẽ được gửi đến trình duyệt Web.*

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
→ Transitional//EN" "http://www.w3.org/TR/2000/REC-xhtml1-
→ 20000126/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">
<head>
<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />
<title>Chuoi</title>
</head>
<body>
<?php # Doan ma 1.7 - strings.php
// Tao cac bien.
$first_name = 'Nguyen Van';
$last_name = 'A';
$book = 'ABC';
// In ra cac gia tri.
echo "Cuon sach <i>$book</i> duoc viet boi $first_name
→ $last_name.";
?>
</body>
</html>
```

Kết hợp chuỗi

Kết hợp chuỗi (một tính năng quan trọng để tạo các Website động) cũng giống như việc chúng ta cộng các chuỗi và điều này được thực hiện bằng cách sử dụng dấu chấm (.). Ví dụ:

```
$first_name = 'Nguyen Van';
```

```
$last_name = 'A';
$name = $first_name . $last_name;
```

Trong ví dụ trên, biến **$name** có giá trị là "**Nguyen VanA**". Chỉnh sửa một chút để cải thiện kết quả của biến **$name** :

```
$name = $first_name . ' ' . $last_name;
```

Giờ đây chuỗi giá trị của biến **$name** sẽ có một khoảng trắng ở giữa (**Nguyen Van A**).

Việc kết hợp chuỗi sẽ được sử dụng thường xuyên trong khi xây dựng các câu hỏi truy vấn dữ liệu ở các chương sau. Bây giờ, chúng ta sẽ điều chỉnh mã kịch bản **strings.php** để thực hành công cụ kết hợp chuỗi:

1. Mở tập tin **strings.php** (tham khảo đoạn mã 1.7) trong trình soạn thảo văn bản.
2. Sau khi thiết lập các biến **$first_name** và **$last_name**, bổ sung dòng lệnh sau:

   ```
   $author = $first_name . ' ' . $last_name;
   ```

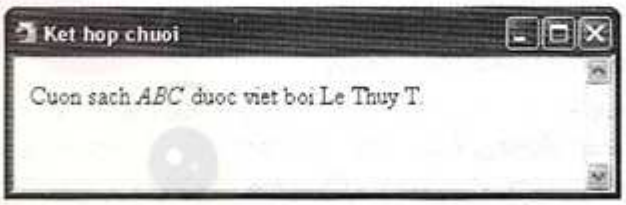
Biến **$author** được tạo dựa trên sự kết hợp giá trị của hai biến hiện có.

3. Thay đổi câu lệnh **echo()** để sử dụng biến mới:

   ```
   echo "Cuon sach <i>$book</i> duoc viet boi $author.";
   ```

Vì hai biến đã được kết hợp lại thành một, câu lệnh **echo()** sẽ thay đổi tương ứng.

4. Nếu muốn, bạn có thể thay đổi tiêu đề trang, thay đổi thông tin về tác giả và tên cuốn sách ở các biến.
5. Lưu lại tập tin theo tên **concat.php**, tải lên máy chủ Web và thử nghiệm nó trong trình duyệt Web (xem hình 1-16). Mã lệnh đầy đủ của tập tin này được thể hiện trong đoạn mã 1.8.



Hình 1-16: *Kết quả cuối cùng của việc kết hợp chuỗi hoàn toàn "vô hình" đối với người sử dụng (so sánh kết quả này với kết quả trong hình 1-14 và 1-15).*

Đoạn mã 1.8: *Kết hợp chuỗi cho phép dễ dàng thao tác với các chuỗi.*

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
→ Transitional//EN" "http://www.w3.org/TR/2000/REC-xhtml1-
→20000126/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
```

```
→ lang="en">
<head>
<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />
<title>Ket hop chuoi</title>
</head>
<body>
<?php # Doan ma 1.8 – concat.php
// Tao cac bien.
$first_name = 'Le Thuy';
$last_name = 'T';
$author = $first_name . ' ' . $last_name;
$book = 'ABC';
// In ra cac gia tri.
echo "Cuon sach <i>$book</i> duoc viet boi $author.";
?>
</body>
</html>
```

PHP có một số hàm chuyên về chuỗi rất hữu dụng (chúng sẽ được giới thiệu trong cuốn sách này). Ví dụ, hàm ***strlen()*** *sẽ cho kết quả là chiều dài của một chuỗi:*

```
$num = strlen($string);
```

Với một chuỗi, có thể sử dụng hàm ***strtolower()*** *để chuyển chuỗi sang dạng chữ thường, hàm* ***strtoupper()*** *để chuyển sang dạng chữ in hoa, hàm* ***ucfirst()*** *để chuyển sang dạng in hoa chữ đầu và hàm* ***ucwords()*** *để chuyển sang dạng in hoa các chữ đầu của mỗi từ.*

Nếu gán một giá trị khác cho một biến đã có (như ***$book*** *chẳng hạn), giá trị mới sẽ thay thế giá trị cũ.*

Để kết hợp một giá trị với một giá trị khác, có thể sử dụng phối hợp toán tử kết hợp và toán tử gán (.=). Ví dụ, hai câu lệnh sau là tương đương:

```
$title = $title . $subtitle;
$title .= $subtitle;
```

Ví dụ đầu tiên của phần này có thể viết lại bằng các cách sau:

```
$name = "$first_name $last_name";
```

hoặc:

```
$name = $first_name;
$name .= ' ';
$name .= $last_name;
```

Toán tử kết hợp có thể được sử dụng trong các hàm, ví dụ:

```
print $first_name . ' ' . $last_name;
```

Giới thiệu biến kiểu số (number)

Như đã giới thiệu trước đây, PHP có kiểu dữ liệu *số nguyên* và *số thập phân*. Tuy nhiên, cả hai loại dữ liệu này có thể gọi chung là kiểu dữ liệu số. Ví dụ về kiểu dữ liệu số hợp lệ trong PHP:

8

3.14

10980843958

-4.2398508

Chú ý, các giá trị trên không được đặt trong các dấu nháy, vì nếu đặt trong dấu này, chúng sẽ được xem là các chuỗi. Các giá trị này cũng không sử dụng ký tự phân tách hàng ngàn, hàng triệu... Các giá trị số được xem là các số dương trừ khi chúng bắt đầu bằng ký hiệu dấu trừ (-).

Ngoài các toán tử toán học chuẩn thông thường (xem bảng 1.1), bạn có thể sử dụng nhiều hàm toán học khác với các giá trị số. Chúng ta sẽ đề cập tới hai hàm: `round()` và `number_format()`.

Hàm `round()` sẽ làm tròn số thập phân thành số nguyên gần nhất, ví dụ:

```
$n = 3.14;
$n = round($n); // Kết quả là 3
```

Hoặc làm tròn tới số thập phân với số số lẻ xác định, ví dụ:

```
$n = 3.1415926;
$n = round($n, 3); // Kết quả là 3.142
```

Hàm `number_format()` chuyển số sang dạng thông thường, với các con số hàng ngàn, hàng triệu... được chia thành nhóm và phân cách bởi dấu phẩy, ví dụ:

```
$n = 20943;
$n = number_format($n); // Kết quả là 20,943
```

Hàm này cũng có thể thiết lập số số lẻ sau dấu chấm thập phân, ví dụ:

```
$n = 20943;
```

```
$n = number_format($n, 2); // Kết quả là 20,943.00
```

Bảng 1.1: *Các toán tử toán học chuẩn.*

Toán tử	Ý nghĩa
+	Cộng
-	Trừ
*	Nhân
/	Chia
%	Modulus
++	Tăng
--	Giảm

Để thực hành với các giá trị số, chúng ta sẽ viết một ví dụ để tính toán cho việc mua hàng trực tuyến. Bạn thực hiện theo các bước sau:

1. Tạo một hồ sơ PHP mới trong trình soạn thảo văn bản.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
→ Transitional//EN" "http://www.w3.org/TR/2000/REC-xhtml1-
→20000126/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">

<head>

<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />

<title>So</title>

</head>

<body>

<?php # Doan ma 1.9 - numbers.php
```

2. Thiết lập các giá trị sau:

```
$quantity = 30; // số lượng hàng

$price = 119.95; // đơn giá hàng

$taxrate = .05; // phần trăm thuế
```

Mã kịch bản sẽ sử dụng ba biến cố định trên để tính toán. Trong phần còn lại của cuốn sách, bạn sẽ thấy cách xác định "động" các giá trị này (ví dụ, khi người dùng tương tác với biểu mẫu HTML).

3. Thực hiện tính toán:

```
$total = $quantity * $price;

$total = $total + ($total * $taxrate);
```

Dòng lệnh thứ nhất thiết lập tổng giá trị của hóa đơn mua hàng theo công thức tính: số lượng hàng nhân với đơn giá. Dòng lệnh thứ nhì, cộng thêm phần thuế vào kết quả tính được ở dòng lệnh thứ nhất. Thuế được tính bằng cách nhân tỉ lệ thuế với tổng giá trị đã tính ở dòng lệnh thứ nhất.

4. Định dạng kết quả:

```
$total = number_format ($total, 2);
```

Hàm **number_format()** sẽ chia dữ liệu số theo nhóm hàng ngàn, hàng triệu... và làm tròn với hai số lẻ. Kết quả này sẽ thích hợp hơn cho người dùng.

5. In kết quả:

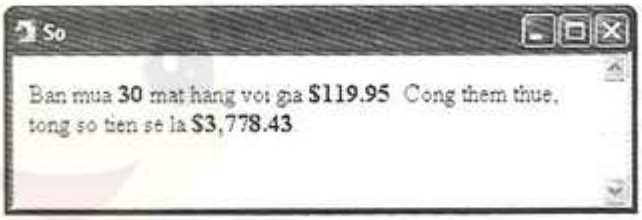
```
echo 'Ban mua <b>', $quantity, '</b> mat hang voi gia <b>$',
→ $price, '</b>. Cong them thue, tong so tien se la <b>$',
→ $total, '</b>.';
```

Bước cuối cùng trong mã kịch bản này là in ra kết quả. Để in ra các mã HTML, dấu đô la ($) và giá trị của các biến, chúng ta sử dụng câu lệnh **echo()** và cung cấp cho nó một loạt giá trị cách nhau bằng dấu phẩy.

6. Bổ sung các mã lệnh sau để hoàn tất mã PHP và trang HTML:

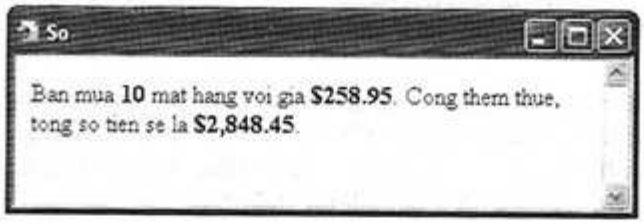
```
?>

</body>

</html>
```

7. Lưu tập tin theo tên **numbers.php**, tải lên máy chủ Web và thử nó trong trình duyệt (xem hình 1-17). Mã lệnh đầy đủ của tập tin này được thể hiện trong đoạn mã 1.9.



Hình 1-17: *Mã kịch bản thực hiện việc tính toán dựa trên các giá trị đã được thiết lập.*

8. Nếu muốn, bạn có thể thay đổi giá trị của ba biến và chạy lại mã kịch bản này (xem hình 1-18).



Hình 1-18: *Để thay đổi trang Web được tạo, bạn thay đổi giá trị của ba biến trong mã kịch bản.*

Đoạn mã 1.9: *Mã kịch bản minh họa các phép tính cơ bản của một máy tính thương mại điện tử.*

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
→ Transitional//EN" "http://www.w3.org/TR/2000/REC-xhtml1-
→20000126/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">
<head>
<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />
<title>So</title>
</head>
<body>
<?php # Doan ma 1.9 - numbers.php
// Thiet lap bien.
$quantity = 30;
$price = 119.95;
$taxrate = .05;
// Tinh tong.
$total = $quantity * $price;
$total = $total + ($total * $taxrate); .
$total = number_format ($total, 2); // Dinh dang ket qua.
// In ket qua.
echo 'Ban mua <b>', $quantity, '</b> mat hang voi gia <b>$',
→ $price, '</b>. Cong them thue, tong so tien se la <b>$',
→ $total, '</b>.';
?>
</body>
</html>
```

PHP hỗ trợ giá trị số nguyên cực đại khoảng 2 tỷ trong hầu hết các môi trường.

Trong tập tin ***php.ini****, bạn có thể thiết lập số số lẻ cho số thập phân sẽ sử dụng.*

Khi làm việc với các phép tính sẽ phát sinh vấn đề về độ ưu tiên (thứ tự thực hiện các phép tính). Để xác định độ ưu tiên, bạn nên gom các mệnh đề trong một biểu thức bằng các dấu ngoặc.

Nếu ứng dụng Web đòi hỏi tính toán với độ chính xác cao, bạn có thể tham khảo phần nói về các con số và hàm trong các tài liệu của PHP để biết thêm chi tiết.

Nếu thiết lập giá mà không sử dụng hai số lẻ (ví dụ như 119.9 hoặc 34), có thể sử dụng hàm ***number_format()*** *để thống nhất định dạng trước khi in ra.*

Giới thiệu về hằng (constant)

Hằng là một kiểu dữ liệu đặc biệt trong PHP. Nó khác với biến ở chỗ giá trị khởi tạo của nó được giữ nguyên trong toàn bộ mã kịch bản. Trong thực tế, không thể thay đổi giá trị của một hằng khi nó đã được thiết lập.

Một hằng được khởi tạo bằng hàm **define()** (thay vì sử dụng toán tử gán (=) như khi khởi tạo giá trị biến). Ví dụ:

```
define('NAME', 'giá trị');
```

Chú ý, tuy không bắt buộc nhưng các tên hằng thường ở dạng chữ hoa. Một điều quan trọng cần ghi nhớ là hằng không có dấu $ ở đầu như các biến (vì về mặt kỹ thuật hằng không phải là biến).

Hằng có thể được in ra bằng một trong hai kỹ thuật sau:

```
echo 'Hello, ' . NAME;
```

hoặc:

```
echo 'Hello, ', NAME;
```

Không thể in hằng bằng cách sử dụng **echo "Hello, NAME"** vì khi đó PHP sẽ in ra giá trị là *Hello, NAME* chứ không in ra giá trị của hằng **NAME**.

PHP chạy với một số hằng được định nghĩa sẵn, cũng giống như các biến được định nghĩa sẵn đã sử dụng trước đây. Các hằng này là **PHP_VERSION** (cho biết phiên bản PHP đang sử dụng) và **PHP_OS** (cho biết hệ điều hành của máy chủ).

Thực hành sử dụng hằng theo các bước sau:

1. Tạo một hồ sơ PHP mới trong trình soạn thảo văn bản.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
→ Transitional//EN" "http://www.w3.org/TR/2000/REC-xhtml1-
→20000126/DTD/xhtml1-transitional.dtd">
```

```
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">
<head>
<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />
<title>Hang</title>
</head>
<body>
<?php # Doan ma 1.10 - constants.php
```

2. Tạo một hằng ngày tháng:

```
define ('TODAY', 'January 8, 2003');
```

Ví dụ này chỉ mang tính minh họa cách định nghĩa một hằng, vì hằng **TODAY** được định nghĩa như trên sẽ không có ứng dụng thực tế. Chương 6 "Sử dụng PHP và MySQL" sẽ đề cập kỹ hơn về cách sử dụng các hằng để lưu giữ thông tin truy xuất cơ sở dữ liệu được an toàn.

3. In ra ngày, phiên bản PHP và thông tin về hệ điều hành:

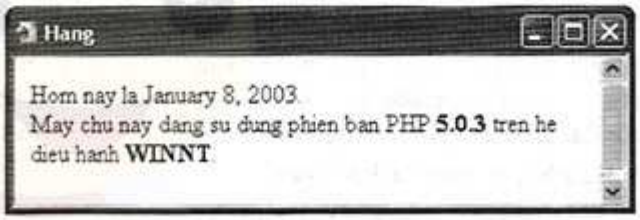
```
echo 'Hom nay la ' . TODAY . '.<br />May chu nay dang su
→ dung phien ban PHP <b>' . PHP_VERSION . '</b> tren he
→ dieu hanh <b>' . PHP_OS . '</b>.<br/><br/>';
```

Vì các hằng không thể in ra khi chúng được đặt trong phạm vi của các dấu nháy (không có dấu $ ở đầu sẽ làm cho chúng in ra ở dạng chữ in hoa), nên toán tử kết hợp đã được sử dụng trong câu lệnh **echo()**.

4. Hoàn tất mã PHP và trang HTML:

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

5. Lưu lại tập tin theo tên **constants.php**, tải lên máy chủ Web và thử nó trong trình duyệt Web (xem hình 1-19). Mã lệnh đầy đủ của tập tin này được thể hiện trong đoạn mã 1.10.

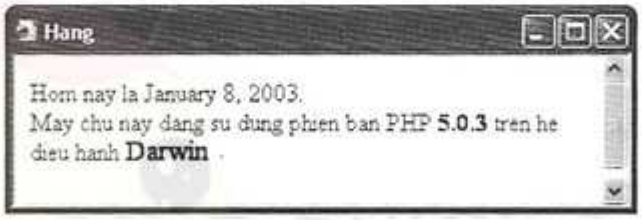


Hình 1-19: *Bằng cách sử dụng hằng trong PHP, bạn có thể tìm hiểu thêm nhiều thông tin về PHP đã cài đặt.*

Đoạn mã 1.10: *Hằng là kiểu dữ liệu bạn có thể sử dụng trong PHP.*

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
→ Transitional//EN" "http://www.w3.org/TR/2000/REC-xhtml1-
→ 20000126/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">
<head>
<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />
<title>Hang</title>
</head>
<body>
<?php # Doan ma 1.10 - constants.php
define ('TODAY', 'January 8, 2003');  // Thiet lap ngay
// Hien thi thong bao.
echo 'Hom nay la ' . TODAY . '.<br />May chu nay dang su dung
→ phien ban PHP <b>' . PHP_VERSION . '</b> tren he dieu hanh
→ <b>' . PHP_OS . '</b>.<br/><br/>';
?>
</body>
</html>
```

Nếu có thể, hãy chạy mã kịch bản này với một máy chủ Web khác (xem hình 1-20).



Hình 1-20: *Chạy cùng một mã kịch bản (tham khảo mã kịch bản 1.10) trên các máy chủ khác nhau sẽ cho các kết quả khác nhau.*

Chương 7 "Cookie và Session", sẽ đề cập đến một hằng khác, đó là ***SID*** *(Session ID).*

Chương 9 "Phát triển ứng dụng Web", sẽ nói đến hai hằng nữa là ***__FILE__*** *và* ***__LINE__****.*

Các dấu nháy kép và nháy đơn

Trong PHP, một điều quan trọng là phải phân biệt được sự khác nhau về công dụng của dấu nháy đơn và nháy kép. Như bạn đã thấy trong các ví dụ của chương này, câu lệnh **echo()** và **print()** có thể sử dụng cả hai kiểu dấu nháy. Vậy đâu là sự khác biệt giữa chúng và vì sao phải sử dụng cả hai kiểu? Bây giờ, chúng ta sẽ đề cập vấn đề này một cách chi tiết hơn.

Với PHP, các giá trị được đặt trong cặp nháy đơn sẽ được xem như *hằng chữ*, trong khi đó các giá trị được đặt trong cặp nháy kép sẽ được biến đổi. Nói cách khác, việc đặt các biến và các ký tự đặc biệt (xem bảng 1.2) trong cặp nháy kép sẽ làm cho giá trị của chúng được in ra thay vì các giá trị hằng chuỗi của chúng. Ví dụ, chúng ta có biến **$var** như sau:

```
$var = 'test';
```

Câu lệnh **echo "var is equal to $var";** sẽ in ra chuỗi *var is equal to test*, trong khi câu lệnh **echo 'var is equal to $var'** sẽ in ra chuỗi *var is equal to $var*. Bằng cách sử dụng dấu đô la (**$**) được mã hóa escape, câu lệnh **echo "\$var is equal to $var"** sẽ in ra chuỗi *$var is equal to test*, trong khi đó câu lệnh **echo '\$var is equal to $var'** sẽ in ra chuỗi *\$var is equal to $var*.

Trong ví dụ này, cặp nháy kép sẽ thay thế tên biến (**$var**) bằng giá trị của nó (**test**) và mã ký tự đặc biệt (**\$**) sẽ thay thế cho giá trị của biến. Các giá trị trong cặp nháy đơn luôn in ra chính xác những gì đã nhập, ngoại trừ dấu nháy đơn (**\'**) và dấu chéo ngược (****) được mã hóa escape.

Nguyên tắc trên được áp dụng cho các cặp dấu nháy đơn và nháy kép, cho dù nó được sử dụng trong các câu lệnh **echo()** hoặc **print()**, hoặc khi gán giá trị cho một chuỗi, hoặc khi truyền dữ liệu cho một hàm dưới dạng tham số. Để có một ví dụ nữa về sự khác biệt của hai kiểu dấu nháy, chúng ta sẽ chỉnh sửa lại mã kịch bản **numbers.php**.

Bảng 1.2: *Các ký tự này có ý nghĩa đặc biệt khi được sử dụng phía trong cặp nháy kép.*

Mã	Ý nghĩa
\"	Dấu nháy kép.
\'	Dấu nháy đơn.
\\	Dấu chéo ngược.
\n	Xuống dòng.
\r	Về đầu dòng.
\t	Tab.
\$	Dấu $.

Bây giờ, chúng ta sẽ thực hành sử dụng dấu nháy đơn và nháy kép theo các bước sau:

1. Mở tập tin **numbers.php** (tham khảo đoạn mã 1.9) trong trình soạn thảo văn bản.
2. Thay đổi câu lệnh **echo** để sử dụng dấu nháy kép thay cho dấu nháy đơn.

```
echo "Ban mua <b>$quantity</b> mat hang voi don gia
→ <b>\$$price</b>. Cong voi thue, tong so tien la
→ <b>\$$total</b>.\n";
```

Chúng ta vẫn có được kết quả mong muốn bằng cách sử dụng dấu nháy kép thay cho các dấu nháy đơn và toán tử kết hợp chuỗi. Chú ý, kỹ thuật **\$$variable** đã được sử dụng để tạo ra các kết quả chẳng hạn như **$3778.43** (dấu đô la đầu tiên được in ra và dấu đô la thứ hai cho biết bắt đầu một biến).

3. Tạo một số dòng trống trong trình duyệt Web:

```
echo '<br /><br />';
```

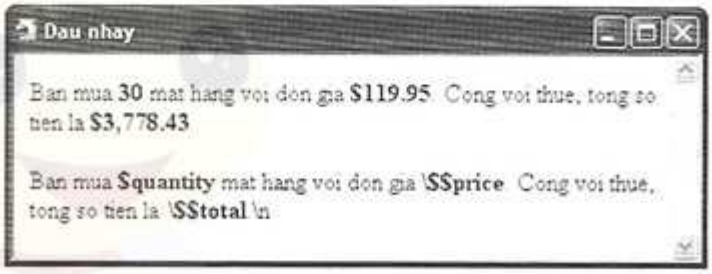
Để thuận tiện so sánh hai dòng lệnh **echo()**, chúng được phân cách bằng các ngắt dòng.

4. Lặp lại câu lệnh **echo** thứ nhất, nhưng lần này sử dụng dấu nháy đơn.

```
echo 'Ban mua <b>$quantity</b> mat hang voi don gia
→ <b>\$$price</b>. Cong voi thue, tong so tien la
→ <b>\$$total</b>.\n';
```

Các mã lệnh đã được in ra để thấy rõ sự khác biệt giữa việc sử dụng dấu nháy đơn và nháy kép.

5. Thay đổi tiêu đề trang, nếu muốn.
6. Lưu lại tập tin theo tên **quotes.php**, tải lên máy chủ Web và thử nó trong trình duyệt Web (xem hình 1-21). Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 1.11.



Hình 1-21: *Kết quả trên minh họa cho cách sử dụng và sự khác biệt trong việc sử dụng dấu nháy đơn và nháy kép của mã lệnh.*

Đoạn mã 1.11: *Mã kịch bản này cho thấy sự khác biệt trong việc sử dụng dấu nháy đơn và nháy kép.*

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
→ Transitional//EN" "http://www.w3.org/TR/2000/REC-xhtml1-
→ 20000126/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">
<head>
<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />
<title>Dau nhay</title>
</head>
<body>
<?php # Doan ma 1.11 - quotes.php
// Thiet lap bien
$quantity = 30;
$price = 119.95;
$taxrate = .05;
// Tinh tong.
$total = $quantity * $price;
$total = $total + ($total * $taxrate); // Bo sung thue.
$total = number_format ($total, 2); // Dinh dang ket qua.
// In ket qua.
echo "Ban mua <b>$quantity</b> mat hang voi don gia
→ <b>\$$price</b>. Cong voi thue, tong so tien la
→ <b>\$$total</b>.\n";
echo '<br /><br />';
echo 'Ban mua <b>$quantity</b> mat hang voi don gia
→ <b>\$$price</b>. Cong voi thue, tong so tien la
→ <b>\$$total</b>.\n';
?>
</body>
</html>
```

Vì PHP sẽ cố gắng biên dịch tất cả các giá trị bên trong dấu nháy kép, nên về mặt lý thuyết, việc sử dụng dấu nháy đơn sẽ nhanh hơn. Các dấu nháy đơn thường được sử dụng khi bạn không muốn biên dịch những gì có trong chuỗi.

PHP cũng hỗ trợ phương pháp heredoc để biểu diễn chuỗi. Nếu đã quen với khái niệm này, bạn có thể tham khảo các tài liệu về PHP để tìm hiểu thêm về cú pháp.

Chương 2:

LẬP TRÌNH VỚI PHP

Một số kiến thức cơ bản về ngôn ngữ lập trình PHP đã được giới thiệu trong các chương trước. Bây giờ, bạn sẽ sử dụng những kiến thức này để bắt đầu lập trình. Trong Chương 2, đồng thời với việc tạo các mã kịch bản phức tạp hơn, bạn sẽ tiếp tục tìm hiểu về thuật ngữ và cú pháp chuẩn của ngôn ngữ.

Chúng ta sẽ khởi đầu bằng việc viết các biểu mẫu HTML và dùng PHP để xử lý các giá trị gửi đi. Sau đó, bạn sẽ tìm hiểu về các câu lệnh điều kiện, các toán tử còn lại, mảng (kiểu dữ liệu cuối cùng được đề cập trong cuốn sách này) và vòng lặp (dạng kết cấu ngôn ngữ sau cùng).

Tạo biểu mẫu HTML

PHP quản lý biểu mẫu HTML theo một quá trình gồm hai bước. Đầu tiên, tạo biểu mẫu HTML bằng một trình soạn thảo văn bản (hoặc một trình soạn thảo WYSIWYG). Sau đó, tạo mã kịch bản PHP để nhận dữ liệu.

Tuy không đi sâu vào chi tiết, nhưng trong chương này, biểu mẫu HTML sẽ được minh họa bằng một ví dụ đơn giản.

Biểu mẫu HTML được tạo với các thẻ **<form>** và một số dạng nhập. Ví dụ về thẻ **form**:

```
<form action="script.php" method="post"></form>
```

Thuộc tính quan trọng nhất của thẻ **form** là **action**. Thuộc tính này cho biết dữ liệu của biểu mẫu sẽ được gửi đi đâu. Thuộc tính thứ hai là **method** (tham khảo mục *"Chọn phương pháp gửi dữ liệu"*), tùy chọn **post** của thuộc tính này thường được sử dụng nhất.

Chọn phương pháp gửi dữ liệu

*Thuộc tính **method** của một biểu mẫu quy định cách gửi dữ liệu đến trang xử lý chúng. Thuộc tính này có hai tùy chọn là **get** và **post**, để cập đến cách thức sử dụng. Phương pháp **get** truyền dữ liệu đến trang xử lý dưới dạng một chuỗi các cặp **tên=giá trị** nối tiếp sau URL. Ví dụ:*

```
http://localhost/script.php?name=Homer&gender=M
```

*Lợi điểm của việc sử dụng phương pháp **get** là trang kết quả có thể được ghi nhận lại trong trình duyệt của người dùng (vì nó là URL). Vì lý do đó, bạn có thể nhấp nút Back để quay trở lại trang **get** hoặc tải lại nó mà không gặp phải vấn đề gì (cả hai đặc điểm này đều không có trong*

*phương pháp **post**). Tuy nhiên, phương pháp **get** có nhược điểm là bị giới hạn về lượng dữ liệu được truyền và không được an toàn (vì mọi người đều có thể nhìn thấy dữ liệu).*

*Vì các lý do trên, chúng tôi chủ yếu sử dụng phương pháp **post**. Để biết thêm thông tin về hai phương pháp gửi dữ liệu, bạn có thể tham khảo các chuẩn về HTML tại Website http://www.w3.org.*

Các trường nhập sẽ được đặt trong phạm vi thẻ mở và đóng của thẻ **form**. Các trường này có thể là hộp văn bản, nút chọn lựa, menu lựa chọn, hộp kiểm..., mỗi dạng trường nhập trong biểu mẫu sẽ cần tới các mã PHP khác nhau để xử lý.

Thực hành tạo biểu mẫu HTML theo các bước sau:

1. Tạo một hồ sơ HTML mới trong trình soạn thảo văn bản:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
→ Transitional//EN""http://www.w3.org/TR/2000/REC-xhtml1-
→ 20000126/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">
<head>
<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />
<title>Bieu mau HTML</title>
</head>
<body>
<!-- Doan ma 2.1 - form.html -->
```

2. Chèn vào dòng khởi tạo thẻ **form**:

```
<form action="handle_form.php" method="post">
```

Vì thuộc tính **action** cho biết mã kịch bản mà dữ liệu sẽ được gửi đến, nên nó phải có tên thích hợp (**handle_form** để ứng với mã kịch bản **form.html**) với phần mở rộng **.php** (vì đây sẽ là một mã kịch bản PHP).

3. Tạo biểu mẫu HTML:

```
<fieldset><legend>Nhap vao thong tin cua ban:</legend>
<p><b>Ten:</b> <input type="text" name="name" size="20"
→ maxlength="40" /></p>
<p><b>Email:</b> <input type="text" name="email" size="40"
→ maxlength="60" /> </p>
<p><b>Gioi tinh:</b> <input type="radio" name="gender"
→ value="M" /> Nam <input type="radio" name="gender"
→ value="F" /> Nu</p>
<p><b>Tuoi:</b>
```

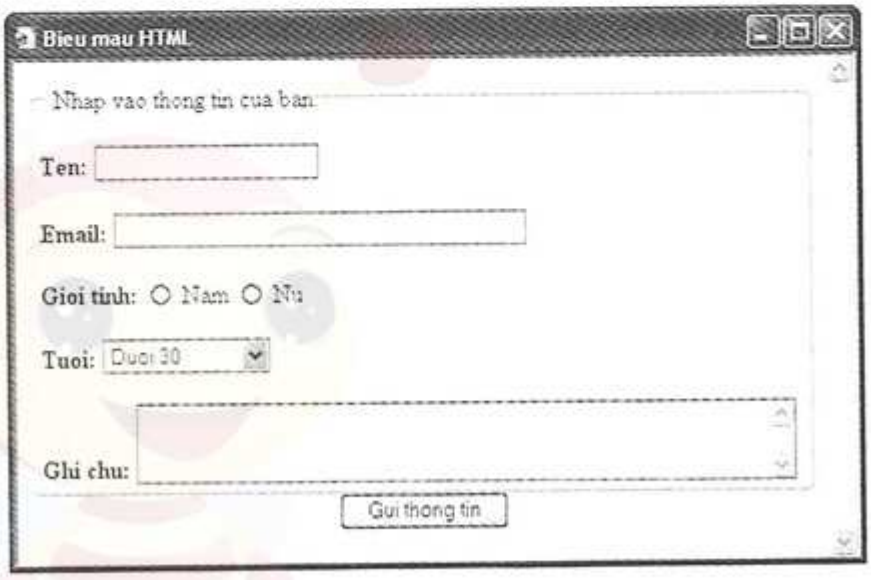
```
<select name="age">
<option value="0-30">Duoi 30</option>
<option value="30-60">Giua 30 va 60</option>
<option value="60+">Tren 60</option>
</select></p>
<p><b>Ghi chu:</b> <textarea name="comments" rows="3"
→ cols="50"></textarea></p>
</fieldset>
<div align="center"><input type="submit" name="submit"
→ value="Gui thong tin" /></div>
</form>
```

Biểu mẫu này có năm trường nhập với bốn dạng dữ liệu khác nhau là *name* (tên), *email* (văn bản), *gender* (nút chọn lựa), *age* (menu chọn) và *comments* (vùng văn bản). Ngoài ra, biểu mẫu còn có dạng nhập thứ năm là *submit* để tạo nút gửi biểu mẫu.

4. Hoàn tất trang HTML:

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

5. Lưu tập tin theo tên **form.html**, tải lên máy chủ và chạy thử nó trong trình duyệt (xem hình 2-1). Mã lệnh đầy đủ của tập tin này được thể hiện trong đoạn mã 2.1.



Hình 2-1: *Biểu mẫu này dùng để nhận thông tin từ người dùng.*

Đoạn mã 2.1: *Biểu mẫu HTML dùng cho một số ví dụ trong chương này.*

```
<!DOCTYPE html PUBLIC
→ "-//W3C//DTD XHTML 1.0 Transitional//EN"
→ "http://www.w3.org/TR/2000/REC-xhtml1-20000126/DTD/xhtml1-
→ transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">

<head>

<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />

<title>Bieu mau HTML</title>

</head>

<body>

<!-- Doan ma 2.1 - form.html -->

<form action="handle_form.php" method="post">

<fieldset><legend>Nhap vao thong tin cua ban:</legend>

<p><b>Ten:</b> <input type="text" name="name" size="20"
→ maxlength="40" /></p>

<p><b>Email:</b> <input type="text" name="email" size="40"
→ maxlength="60" /> </p>

<p><b>Gioi tinh:</b> <input type="radio" name="gender"
→ value="M" /> Nam <input type="radio" name="gender"
→ value="F" /> Nu</p>

<p><b>Tuoi:</b>

<select name="age">

<option value="0-30">Duoi 30</option>

<option value="30-60">Giua 30 va 60</option>

<option value="60+">Tren 60</option>

</select></p>

<p><b>Ghi chu:</b> <textarea name="comments" rows="3"
→ cols="50"></textarea></p>

</fieldset>

<div align="center"><input type="submit" name="submit"
```

```
→ value="Gui thong tin" /></div>
</form>
<!-- Ket thuc bieu mau -->
</body>
</html>
```

Vì trang trên chỉ chứa mã lệnh HTML nên phần mở rộng của tên tập tin thường là **.html**, *nhưng cũng có thể là* **.php** *(vì mã lệnh phía ngoài thẻ PHP được xem như HTML).*

Vì đây là một trang HTML chuẩn nên nó có thể được xem trong trình duyệt mà không cần đến máy chủ Web.

Xử lý biểu mẫu HTML

Sau khi đã có biểu mẫu HTML, chúng ta sẽ viết mã kịch bản PHP để xử lý nó (xử lý dữ liệu nhận được). Trong chương này, các mã kịch bản sẽ tái tạo dữ liệu và gửi trở lại trình duyệt, nhưng trong các ví dụ sau, chúng sẽ được đưa vào cơ sở dữ liệu MySQL, được kiểm tra với các giá trị được lưu trữ trước đó, được gửi đến một địa chỉ thư điện tử...

PHP dễ học và dễ sử dụng vì nó có sự tương tác với các biểu mẫu HTML. Nếu một hộp văn bản (một trường) trong biểu mẫu HTML có thuộc tính **name** là **email**, PHP sẽ chứa văn bản được nhập vào trường này trong biến có tên là **$email**.

Thay vì đòi hỏi một thủ tục biên dịch (như các mã kịch bản CGI thực hiện), để truy xuất dữ liệu của biểu mẫu, bạn chỉ cần đặt dấu đô la (**$**) trước tên trường nhập. Cách này không phụ thuộc vào dạng trường nhập, nên trong ví dụ của chúng ta, mã kịch bản **handle_form.php** có thể sử dụng các biến có tên là **$name**, **$email**, **$gender**, **$age**, **$comments** và **$submit**.

Thực hành xử lý biểu mẫu HTML theo các bước sau:

1. Tạo một hồ sơ PHP mới trong trình soạn thảo văn bản bắt đầu với HTML.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
→ Transitional//EN" "http://www.w3.org/TR/2000/REC-xhtml1-
→ 20000126/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">

<head>

<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />
```

```
<title>Phan hoi</title>

</head>

<body>
```

2. Tạo các thẻ PHP và in ra các giá trị **name**, **email**, **age**, **gender** và **comments** vừa nhận được.

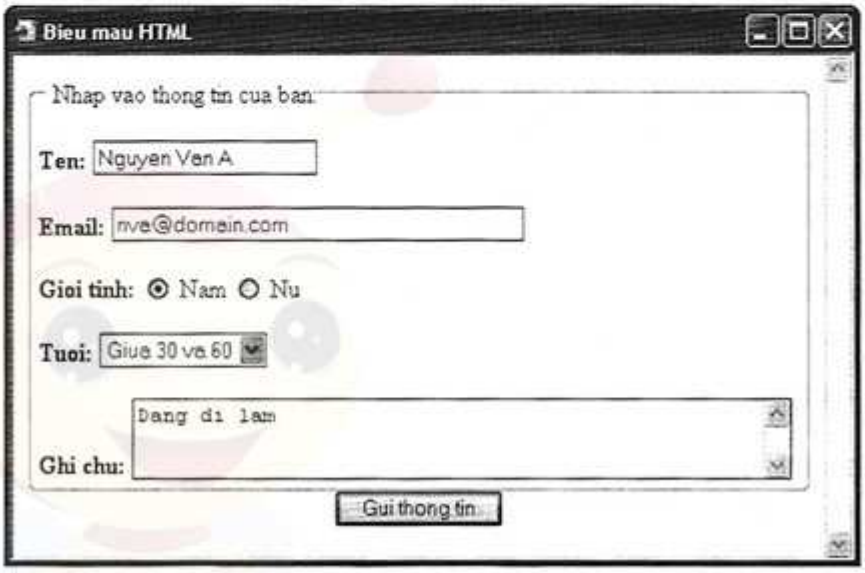
```
echo "Cam on <b>$name</b> da dong gop y kien sau:<br />
→ <tt>$comments</tt><p>Chung toi se hoi am ban theo dia chi
→ <i>$email</i>.</p>";
```

Ba phần tử biểu mẫu (tham khảo mã kịch bản 2.1) được đặt tên là **name**, **email** và **comments** (hai phần tử còn lại là **gender** và **age**). Các biến **$name**, **$email** và **$comments** được sử dụng để truy xuất giá trị mà người dùng nhập vào và chúng được in ra bằng lệnh **echo()** kết hợp với các dấu nháy kép.

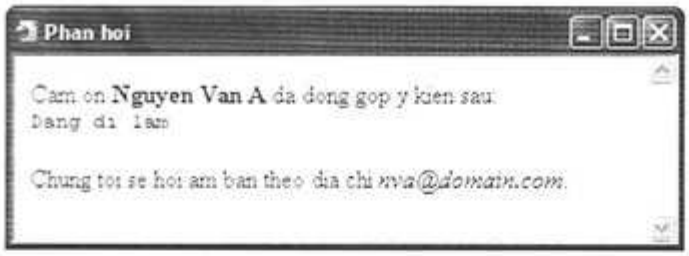
3. Hoàn tất trang HTML:

```
</body>

</html>
```

4. Lưu lại tập tin theo tên **handle_form.php**, tải lên máy chủ Web và chạy thử nó trong trình duyệt (xem hình 2-2 và 2-3). Mã lệnh đầy đủ của tập tin này được thể hiện trong đoạn mã 2.2.



Hình 2-2: *Để thử mã kịch bản* ***handle_form.php****, bạn phải điền các giá trị trong biểu mẫu.*



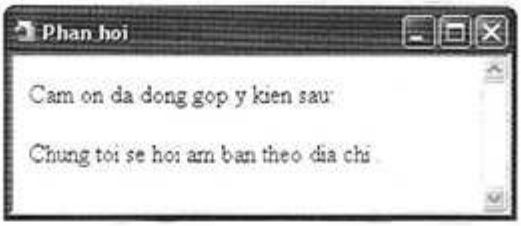
Hình 2-3: *Kết quả tạo ra từ mã kịch bản.*

Đoạn mã 2.2: *Mã kịch bản này cho thấy PHP xử lý dữ liệu của biểu mẫu HTML rất dễ dàng.*

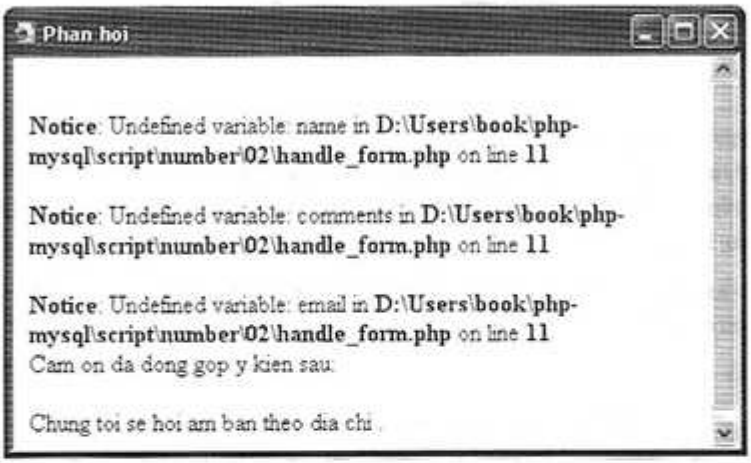
```
<!DOCTYPE html PUBLIC
→ "-//W3C//DTD XHTML 1.0 Transitional//EN"
→ "http://www.w3.org/TR/2000/REC-xhtml1-20000126/DTD/xhtml1-
→transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">

<head>

<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />

<title>Phan hoi</title>

</head>

<body>

<?php # Doan ma 2.2 - handle_form.php

echo "Cam on <b>$name</b> da dong gop y kien sau:<br />
→<tt>$comments</tt><p>Chung toi se hoi am ban theo dia chi
→ <i>$email</i>.</p>";

?>

</body>

</html>
```

Nếu thấy kết quả như trong hình 2-4 hoặc 2-5, bạn có thể tham khảo mục "Thiết lập register_globals".

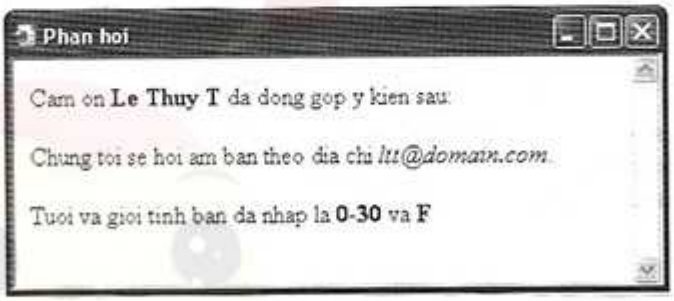
Để so sánh cách PHP xử lý các dạng biểu mẫu nhập khác nhau, bạn hãy in ra giá trị của ***$age*** *và* ***$gender*** *(xem hình 2-6).*



Hình 2-4: *Mã kịch bản của bạn có thể thể hiện kết quả như trong hình này.*



Hình 2-5: *Có hai vấn đề có thể tạo ra kết quả như trong hình này và chúng sẽ được đề cập ở mục "Thiết lập register_globals" và trong phần sau.*



Hình 2-6: *Các giá trị của* **`$gender`** *và* **`$age`** *tương ứng với các trường được định nghĩa trong biểu mẫu HTML.*

Thiết lập register_globals

Trong các phiên bản trước của PHP, **`register_globals`** *mặc định được thiết lập ở trạng thái bật (on). Đặc điểm này cho phép PHP tự động chuyển các trường nhập của biểu mẫu thành các biến có tên tương tự (như trong ví dụ trên).*

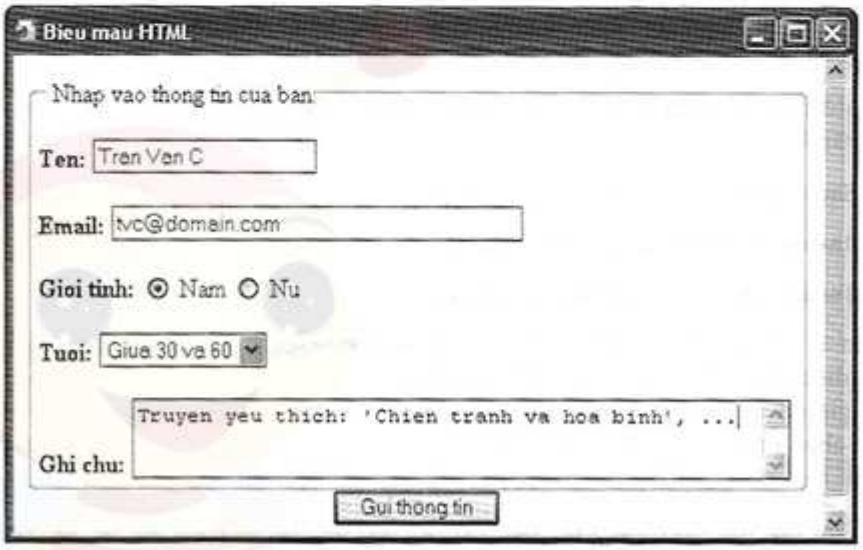
Từ phiên bản PHP 4.2, các nhà phát triển PHP lại thiết lập **`register_globals`** *mặc định ở trạng thái tắt (off) với suy nghĩ rằng sẽ làm tăng tính an toàn cho mã kịch bản. Thật không may, điều này cũng gây ra các hiệu ứng phụ, làm cho nhiều mã kịch bản đã có trước đây sẽ không làm việc được nữa và nhiều người lập trình đã lúng túng khi thấy các giá trị trống trong trang kết quả xử lý biểu mẫu (xem lại hình 2-4).*

Có hai tùy chọn để giải quyết vấn đề này. Đầu tiên, chuyển trạng thái của **`register_globals`** *thành bật, nếu bạn có quyền quản trị việc cài đặt PHP. Thứ hai, sử dụng các biến toàn cục, như* **`$_GET`** *và* **`$_POST`**. *Các biến này sẽ được giới thiệu trong phần "Mảng là gì?" ở phần sau của chương này.*

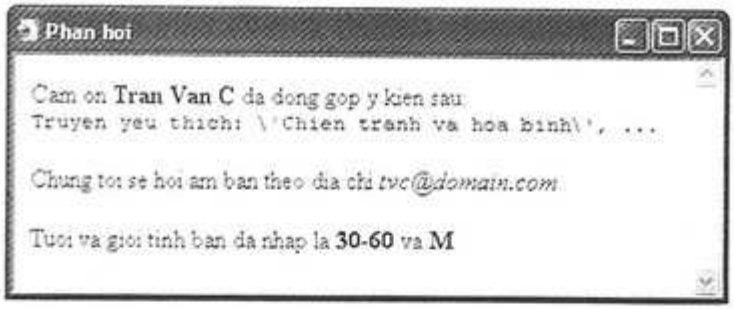
Nếu kết quả giống như hình 2-5 sẽ có hai nguyên nhân mà bạn cần xem xét. Trước hết, **`register_globals`** *đang được tắt và do đó các biến sẽ không có giá trị. Thứ hai,* **`error_reporting`** *được thiết lập với mức độ cao nhất và mã kịch bản thông báo là các biến không tồn tại đang được sử dụng. Vấn đề này được giải quyết bằng cách thay đổi cấp độ của* **`error_reporting`** *(xem Phụ lục A hoặc Chương 9 "Phát triển ứng dụng Web") hay sử dụng hàm* **`isset()`** *để kiểm tra sự tồn tại của biến. Hàm* **`isset()`** *sẽ được đề cập trong phần sau của chương này.*

Quản lý Magic Quotes

Magic Quotes là một đặc điểm tiện lợi được cài sẵn trong PHP. Khi được kích hoạt, Magic Quotes sẽ tự động mã hóa escape các dấu nháy đơn và nháy kép trong giá trị nhận được của các biến. Ví dụ, nếu nhập vào biểu mẫu một đoạn văn bản có kèm dấu nháy (xem hình 2-7), trang kết quả sẽ sẽ thể hiện bất thường như hình 2-8.



Hình 2-7: *Các dấu nháy nhập vào các trường trong biểu mẫu có thể làm hỏng ứng dụng Web (xem hình 2-8).*



Hình 2-8: *Dấu nháy được nhập vào trường trên biểu mẫu đã bị tự động mã hóa escape bởi PHP và gây ra kết quả bất thường.*

PHP có hai dạng Magic Quotes chính là **magic_quotes_gpc** được áp dụng cho biểu mẫu, URL và dữ liệu *cookie* (*gpc* là từ viết tắt của *get*, *post* và *cookie*) và **magic_quotes_runtime** áp dụng cho dữ liệu nhận vào từ các tập tin ngoài và từ cơ sở dữ liệu.

Nếu Magic Quotes được kích hoạt trên máy chủ, bạn có thể loại bỏ (undo) ảnh hưởng của nó bằng hàm **stripslashes()**, ví dụ:

```
$var = stripslashes($var);
```

Hàm **stripslashes()** sẽ loại bỏ các dấu chéo ngược có trong biến **$var**. Trong ví dụ về biểu mẫu, hàm này có tác dụng chuyển chuỗi đã được mã hóa escape bởi Magic Quotes trở lại giá trị ban đầu của chúng. Kết quả là chúng ta nhận được chuỗi không bị mã hóa.

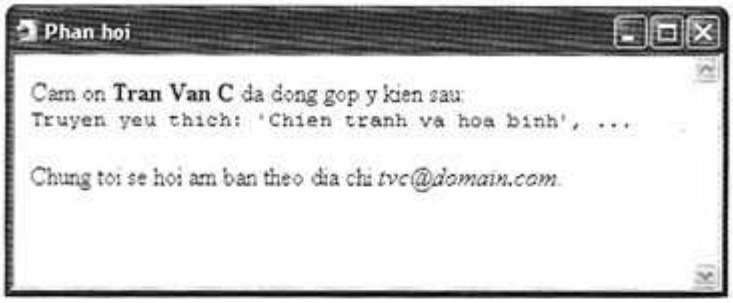
Thực hành sử dụng Magic Quotes theo các bước sau:

1. Mở tập tin **handle_form.php** (tham khảo đoạn mã 2.2) trong trình soạn thảo văn bản.
2. Sau thẻ PHP bắt đầu, bổ sung các dòng lệnh sau:

```
$name = stripslashes($name);
$comments = stripslashes($comments);
```

Hiện tại, đây là hai biến duy nhất cần làm sạch các dấu chéo ngược. Vì địa chỉ thư điện tử không chứa dấu chéo nào (nếu không nó sẽ trở nên không hợp lệ) và các giá trị nhập của age và gender được thiết lập sẵn cũng không chứa dấu chéo, nên chúng ta không cần xử lý các biến này. Nếu chúng có khả năng chứa các dấu chéo, bạn cần áp dụng hàm **stripslashes()** cho chúng.

3. Lưu tập tin, tải nó lên máy chủ Web và chạy thử trong trình duyệt (xem hình 2-9). Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 2.3.



Hình 2-9: *Việc áp dụng hàm* ***stripslashes()*** *cho các giá trị của biểu mẫu sẽ loại bỏ ảnh hưởng của Magic Quotes (so sánh với kết quả trong hình 2-8).*

Đoạn mã 2.3: *Hàm* ***stripslashes()*** *sẽ loại bỏ ảnh hưởng của Magic Quotes và rất hữu dụng khi muốn in lại các giá trị dữ liệu trong biểu mẫu.*

```
<!DOCTYPE html
→ PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
→ "http://www.w3.org/TR/2000/REC-xhtml1-20000126/DTD/xhtml1-
→transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">
<head>
<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />
<title>Phan hoi</title>
</head>
<body>
<?php # Doan ma 2.3 - handle_form.php
// Xu ly Magic Quotes...
$name = stripslashes($name);
$comments = stripslashes($comments);
echo "Cam on <b>$name</b> da dong gop y kien sau:<br />
→ <tt>$comments</tt><p>Chung toi se hoi am ban theo dia chi
→ <i>$email</i>.</p>";
?>
</body>
</html>
```

Câu lệnh điều kiện và toán tử

Cũng như các biến, câu lệnh điều kiện là một phần của lập trình và phần lớn mọi người đều quen thuộc với chúng ở dạng này hoặc dạng kia. Các trang Web động thường sử dụng câu lệnh điều kiện để thay đổi cách làm việc của mã kịch bản dựa trên một tập hợp các chỉ tiêu.

Ba thuật ngữ cơ bản của PHP dùng để tạo ra các câu lệnh điều kiện là **if**, **else** và **elseif**.

Mỗi câu lệnh điều kiện bao gồm một mệnh đề **if**:

```
if(điều kiện) {
  // thực hiện một điều gì đó
}
```

Điều kiện này có thể được mở rộng thành:

```
if(điều kiện) {
  // thực hiện một điều gì đó
} else {
  // thực hiện một điều khác
}
```

và:

```
if(điều kiện 1) {
  // thực hiện một điều gì đó
} elseif(điều kiện 2) {
  // thực hiện một điều khác
} else {
  // thực hiện một điều khác nữa
}
```

Nếu điều kiện là **true**, mã lệnh trong cặp dấu ngoặc móc (**{}**) tiếp theo sẽ được thực hiện. Nếu không, PHP sẽ tiếp tục tiến hành. Nếu có một điều kiện thứ hai (sau mệnh đề **elseif**), nó sẽ được kiểm tra tới. Quá trình cứ như vậy tiếp tục (bạn có thể sử dụng nhiều mệnh đề **elseif**) cho đến khi PHP bắt gặp mệnh đề **else** và thực hiện mã lệnh ứng với mệnh đề này hoặc khi câu lệnh điều kiện kết thúc mà không có mệnh đề **else**.

Vì lý do trên, một điều quan trọng cần nhớ là mệnh đề **else** phải là mệnh đề cuối cùng trong câu lệnh điều kiện và được xử lý mặc định, ngoại trừ khi một chỉ tiêu nào đó đã được thỏa mãn trong các mệnh đề **if** hoặc **elseif** ở trên.

Trong PHP, ở một số trường hợp, một điều kiện có thể trở thành **TRUE** với các dạng phổ biến là:

- **$var**, nếu **$var** có giá trị khác **0**, khác chuỗi rỗng hoặc khác **NULL**.
- **isset($var)**, nếu **$var** có giá trị khác **NULL**, kể cả giá trị **0** và chuỗi rỗng.
- **TRUE**.

Sử dụng các toán tử lôgic và so sánh (xem bảng 2.1) cùng với các dấu ngoặc để thực hiện các biểu thức phức tạp hơn.

Các toán tử lôgic và so sánh (xem bảng 2.1) được phối hợp với các dấu ngoặc để tạo ra các biểu thức phức tạp.

Bảng 2.1: *Các toán tử thường được sử dụng với câu lệnh điều kiện.*

Ký hiệu	Ý nghĩa	Dạng	Ví dụ
=	Được gán giá trị của	Gán	$n = 1
==	Bằng	So sánh	$x == $y
!=	Không bằng	So sánh	$x != $y
<	Nhỏ hơn	So sánh	$x < $y
>	Lớn hơn	So sánh	$x > $y
<=	Nhỏ hơn hoặc bằng	So sánh	$x <= $y
>=	Lớn hơn hoặc bằng	So sánh	$x >= $y
!	Phủ định	Lôgic	!$x
&&	AND	Lôgic	$x && $y
||	OR	Lôgic	$x || $y

Thực hành câu lệnh điều kiện theo các bước sau:

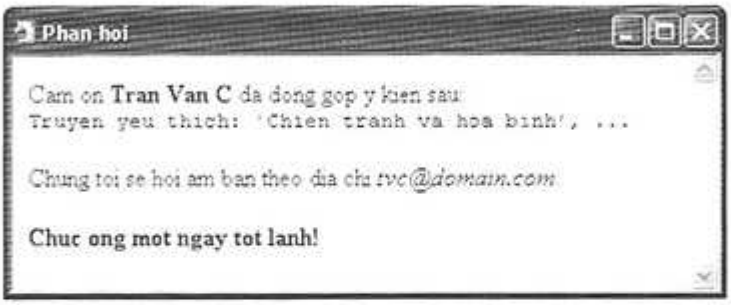
1. Mở tập tin **handle_form.php** (tham khảo đoạn mã 2.2) trong trình soạn thảo văn bản.
2. Sau câu lệnh **echo()**, bổ sung mã lệnh sau:

```
if ($gender == 'M') {
   echo '<b><p>Chuc ong mot ngay tot lanh!</p></b>';
} elseif ($gender == 'F') {
   echo '<b><p>Chuc ba mot ngay tot lanh!</p></b>';
} else {
```

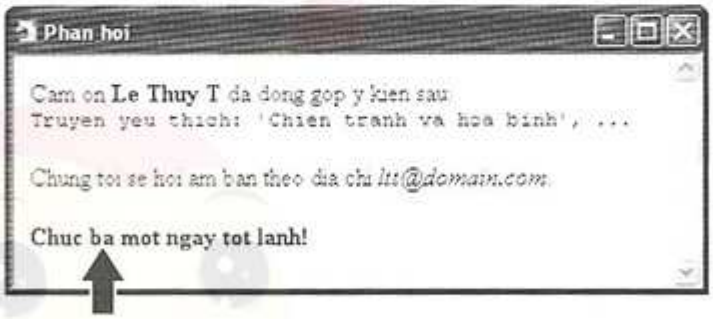
```
    echo '<p><b>Ban chua xac dinh gioi tinh!</b></p>';
}
```

Trong đoạn mã lệnh trên, câu lệnh điều kiện **if-elseif-else** sẽ xem xét giá trị của biến **$gender** và in ra các thông điệp khác nhau ứng với mỗi trường hợp. Một điều quan trọng cần nhớ là hai dấu bằng liên tiếp (==) là phép so sánh bằng, trong khi một dấu bằng (=) là phép gán một giá trị. Vì lý do này, **$gender == 'M'** có thể đúng hoặc không đúng, nhưng **$gender = 'M'** thì luôn luôn đúng.

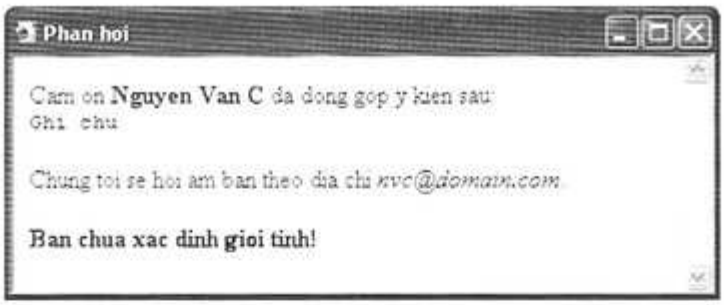
3. Lưu tập tin, tải lên máy chủ Web và chạy thử nó trong trình duyệt (xem hình 2-10, 2-11 và 2-12). Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 2.4.



Hình 2-10: *Điều kiện dựa trên **$gender** sẽ in ra các thông báo khác nhau tùy theo những gì được chọn trong biểu mẫu.*



Hình 2-11: *Cùng một mã kịch bản nhưng sẽ tạo ra các kết quả khác nhau (so sánh với hình 2-10) khi các giá trị trên biểu mẫu thay đổi.*



Hình 2-12: *Nếu trường gender không được chọn, một thông báo lỗi sẽ được in ra.*

Đoạn mã 2.4: *Các câu lệnh điều kiện cho phép điều chỉnh hành động dựa trên các chỉ tiêu nhất định.*

```
<!DOCTYPE html PUBLIC
→ "-//W3C//DTD XHTML 1.0 Transitional//EN"
→ "http://www.w3.org/TR/2000/REC-xhtml1-20000126/DTD/xhtml1-
→transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">

<head>

<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />

<title>Phan hoi</title>

</head>

<body>

<?php # Doan ma 2.4 - handle_form.php

// Xu ly Magic Quotes...

$name = stripslashes($name);

$comments = stripslashes($comments);

echo "Cam on <b>$name</b> da dong gop y kien sau:<br />
→<tt>$comments</tt><p>Chung toi se hoi am ban theo dia chi
→ <i>$email</i>.</p>";

// Conditional replies:

if ($gender == 'M') {

   echo '<b><p>Chuc ong mot ngay tot lanh!</p></b>';
```

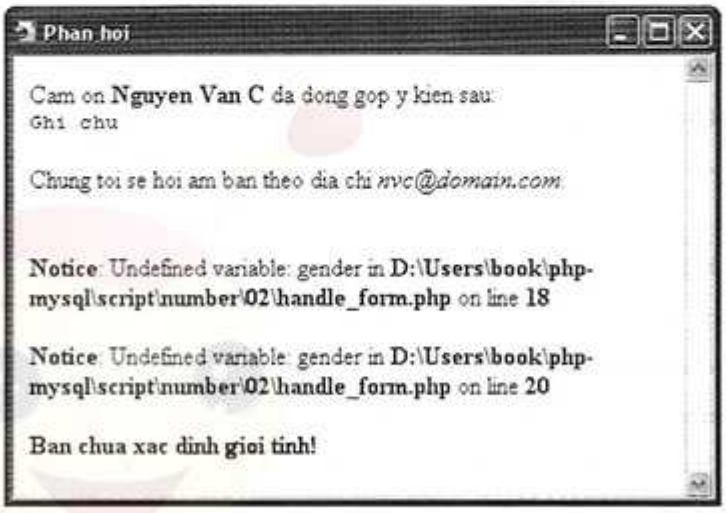
```
} elseif ($gender == 'F') {
   echo '<b><p>Chuc ba mot ngay tot lanh!</p></b>';
} else {
   echo '<p><b>Ban chua xac dinh gioi tinh!</b></p>';
}
?>
</body>
</html>
```

Có thể lồng các câu lệnh điều kiện vào nhau.

Các dấu ngoặc móc ({}) được sử dụng để báo hiệu phần bắt đầu và kết thúc của một câu lệnh điều kiện, điều này không bắt buộc nếu bạn chỉ muốn thực hiện một câu lệnh. Tuy nhiên, nên sử dụng cặp dấu ngoặc móc trong mọi trường hợp để đảm bảo tính rõ ràng của mã lệnh.

Từ ***elseif*** *có thể được viết thành* ***else if****.*

Bạn sẽ tìm hiểu cách ngăn ngừa các thông báo lỗi như hình 2-13 trong phần kế tiếp.



Hình 2-13: *Nếu trường gender không được chọn và máy chủ Web được thiết lập để thông báo lỗi với mức cao nhất, bạn sẽ thấy các lỗi phụ của PHP.*

Vì rất dễ nhập nhầm mã lệnh là ***$gender = 'M'*** *thay vì* ***$gender == 'M'****, nên nhiều người thường viết lại biểu thức này ở dạng* ***'M' == $gender*** *để tạo ra thông báo lỗi nếu nhập sai là* ***'M' = $gender****.*

Switch

PHP còn có một dạng câu lệnh điều kiện nữa là **switch**. Câu lệnh này rất phù hợp cho việc thay thế nếu câu lệnh **if-elseif-else** quá dài. Cú pháp của câu lệnh **switch** như sau:

```
switch($variable) {
        case 'giá trị 1':
                // thực hiện lệnh
                break;
        case 'giá trị 2':
                // thực hiện lệnh
                break;
        default:
                // thực hiện lệnh
}
```

Trong mã lệnh trên, câu lệnh **switch** thực hiện việc so sánh giá trị của biến **$variable** với các giá trị khác nhau, ứng với các mệnh đề **case**. Khi tìm được một giá trị phù hợp, mã lệnh đi ngay sau trường hợp này sẽ được thực hiện cho đến khi bắt gặp câu lệnh **break**. Nếu không có giá trị nào phù hợp, mệnh đề **default** sẽ được thực hiện (nếu có).

Như vậy, điều kiện trong đoạn mã 2.4 có thể được viết lại như sau:

```
switch($gender) {
   case 'M':
      echo '<b><p>Chuc ong mot ngay tot lanh!</p></b>';
      break;
   case 'F':
      echo '<b><p>Chuc ba mot ngay tot lanh!</p></b>';
      break;
   default:
      echo '<p><b>Ban chua xac dinh gioi tinh!</b></p>'; }
```

Câu lệnh **switch** bị giới hạn ở chỗ nó chỉ kiểm tra giá trị của biến với một tập hợp các giá trị, nên không thể sử dụng nó trong các điều kiện phức tạp hơn.

Kiểm tra tính hợp lệ của dữ liệu biểu mẫu

Một khái niệm quan trọng liên quan đến việc xử lý các biểu mẫu HTML là kiểm tra tính hợp lệ của dữ liệu. Về mặt quản lý lỗi và an toàn dữ liệu, chúng ta không nên tin tưởng vào dữ liệu được nhập trong biểu mẫu. Cho dù người dùng vô tình hoặc cố ý khi nhập dữ liệu sai, các giá trị này đều phải được kiểm tra lại.

Việc kiểm tra tính hợp lệ của dữ liệu biểu mẫu đòi hỏi sử dụng các câu lệnh điều kiện, một số hàm, toán tử và biểu thức. Một hàm thường được sử dụng là **isset()**. Hàm này kiểm tra một biến đã có giá trị hay chưa (kể cả giá trị **0**, nhưng không phải là **NULL** hoặc **FALSE**).

```
if(isset($var)){
        // $var có giá trị
}  else {
        // $var chưa có giá trị
}
```

Một vấn đề phát sinh là khi dùng hàm **isset()** kiểm tra chuỗi rỗng sẽ cho giá trị là **TRUE**. Chính vì điều đó, chúng ta sẽ sử dụng hàm **strlen()** để kiểm tra chuỗi. Hàm này đếm số ký tự có trong chuỗi.

```
if(strlen($var) > 0){
        // $var có giá trị
}  else {
        // $var chưa có giá trị
}
```

Hai mục đích chính của việc kiểm tra tính hợp lệ của dữ liệu là đảm bảo rằng mã kịch bản có được thông tin phù hợp và đúng dạng (số, văn bản...). Chương 8 "An toàn", sẽ đề cập đến việc sử dụng các biểu thức quy tắc để kiểm tra dữ liệu với yêu cầu cụ thể. Trong phần này, chúng ta sẽ viết một mã kịch bản **handle_form.php** mới để đảm bảo rằng các biến đã tồn tại trước khi chúng được tham chiếu (để loại bỏ các thông báo lỗi khó chịu như trong hình 2-13).

Thực hành kiểm tra tính hợp lệ của biểu mẫu theo các bước sau:

1. Tạo một mã kịch bản PHP mới trong trình soạn thảo:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
→ Transitional//EN" "http://www.w3.org/TR/2000/REC-xhtml1-
→20000126/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">
```

```
<head>
<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />
<title>Phan hoi</title>
</head>
<body>
<?php # Doan ma 2.5 - handle_form.php
```

2. Kiểm tra trường **name** có được nhập hay không:

```
if (strlen($name) > 0) {
   $name = stripslashes($name);
} else {
   $name = NULL;
   echo '<p><b>Ban quen nhap ten cua minh!</b></p>';
}
```

Cách tốt nhất để kiểm tra xem một hộp văn bản trên biểu mẫu có được nhập hay không là tính chiều dài của chuỗi kết quả. Nếu kết quả là số dương (**strlen($name) > 0**) sẽ có ít nhất một ký tự được nhập. Nếu đúng, chúng ta sẽ loại bỏ các dấu chéo ngược như đã thực hiện trước đây. Nếu sai, thiết lập biến **$name** thành **NULL** và in ra thông báo lỗi.

3. Lặp lại bước này cho các biến **$email** và **$comments**:

```
if (strlen($comments) > 0) {
   $comments = stripslashes($comments);
} else {
   $comments = NULL;
   echo '<p><b>Ban quen nhap ghi chu!</b></p>';
}
if ( !(strlen($email) > 0) ) {
   $email = NULL;
   echo '<p><b>Ban quen nhap email!</b></p>';
}
```

Biến **$comments** được xử lý tương tự như biến **$name**, nhưng biến **$email** thì hơi khác một chút vì hàm **stripslashes()** không áp dụng cho nó. Như vậy, bạn sẽ không cần làm gì cả nếu biến **$email** có giá trị. Thay vì kiểm tra xem chiều dài

của nó có "*lớn hơn* **0**" hay không? Bạn kiểm tra xem nó có "*không lớn hơn* **0**" hay không. Điều kiện **!strlen($email) > 0** cũng có thể được viết lại thành **strlen($email) <= 0**.

4. Kiểm tra trạng thái của biến **$gender**:

```
if (isset($gender)) {
   if ($gender == 'M') {
      $message = '<b><p>Chuc ong mot ngay tot
      → lanh!</p></b>';
   } elseif ($gender == 'F') {
      $message = '<b><p>Chuc ba mot ngay tot lanh!</p></b>';
   }
} else {
   $gender = NULL;
   echo '<p><b>Ban quen xac dinh gioi tinh!</b></p>';
}
```

Vì trường **$gender** (giới tính) là trường chọn lựa, giá trị của nó chỉ là *M* (nam) hoặc *F* (nữ). Đầu tiên, trường này sẽ được kiểm tra bằng hàm **isset()** xem nó có giá trị hay không? Sau đó, tạo ra các thông báo dựa trên giá trị của trường.

Trong mã kịch bản trước (tham khảo lại mã kịch bản 2.4 và hình 2-11), chúng ta đã in ra thông báo theo **$gender** ở cuối của trang Web. Trong mã kịch bản này, bạn sẽ tạo ra biến **$message** để in nó ra trong phần sau.

5. In ra thông báo nếu tất cả các kiểm tra đều đạt.

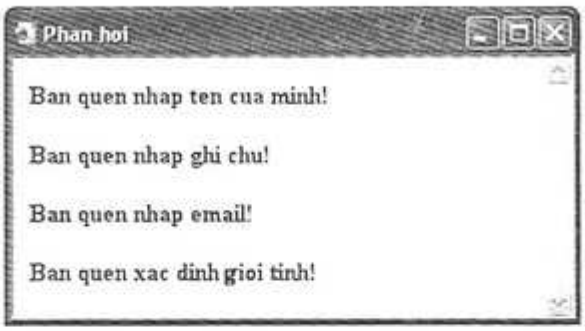
```
if ($name && $comments && $email && $gender) {
   echo "Cam on <b>$name</b> da dong gop y kien sau:<br />
   → <tt>$comments</tt><p>Chung toi se hoi am ban theo dia
   → chi <i>$email</i>.</p>";
   echo $message;
}
```

Mỗi biến sẽ có một giá trị nếu nó vượt qua được bước kiểm tra và sẽ có giá trị **NULL** nếu không hợp lệ. Nếu mọi biến đều có giá trị, biểu mẫu được xem là hoàn chỉnh thì điều kiện sẽ đúng và kết quả là thông báo sẽ được in ra. Nếu một trong số các biến có giá trị **NULL**, thông báo sẽ không được in ra.

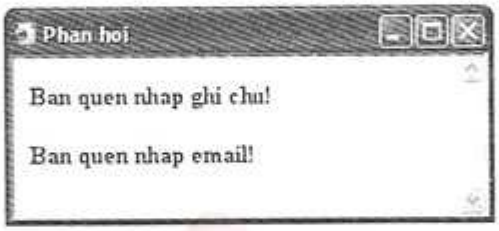
6. Đóng phần mã lệnh PHP và hoàn tất mã lệnh HTML:

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

7. Lưu tập tin theo tên **handle_form.php** (mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 2.5), tải lên máy chủ và thử nó trong trình duyệt Web (xem hình 2-14, 2-15 và 2-16). Điền các giá trị khác nhau vào biểu mẫu để thử nghiệm.



Hình 2-14: *Mã kịch bản sẽ kiểm tra từng phần tử trong biểu mẫu xem có được nhập giá trị không và in ra các thông báo nhắc nhở nếu chúng chưa được nhập.*



Hình 2-15: *Nếu một hoặc hai trường bị bỏ qua, thông báo sẽ không được in ra.*

Hình 2-16: *Nếu tất cả các trường được nhập một cách chính xác, thông báo kết quả sẽ được in ra như trước đây.*

Đoạn mã 2.5: *Kiểm tra tính hợp lệ của dữ liệu trước khi sử dụng chúng là một việc quan trọng để đảm bảo tính an toàn và mang lại kết quả chuyên nghiệp.*

```
<!DOCTYPE html PUBLIC
→ "-//W3C//DTD XHTML 1.0 Transitional//EN"
→ "http://www.w3.org/TR/2000/REC-xhtml1-20000126/DTD/xhtml1-
→transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">

<head>

<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />

<title>Phan hoi</title>

</head>

<body>

<?php # Doan ma 2.5 – handle_form.php

// Kiem tra $name va loai bo cac dau cheo.

If (strlen($name) > 0) {

   $name = stripslashes($name);

} else { // Neu ten khong duoc nhap...

   $name = NULL;

   echo '<p><b>Ban quen nhap ten cua minh!</b></p>';

}

// Kiem tra $comments va loai bo cac dau cheo.

If (strlen($comments) > 0) {

   $comments = stripslashes($comments);

} else { // Neu khong co phan ghi chu

   $comments = NULL;

   echo '<p><b>Ban quen nhap ghi chu!</b></p>';

}

// Kiem tra $email:

if ( !(strlen($email) > 0) ) {

   $email = NULL;
```

```
    echo '<p><b>Ban quen nhap email!</b></p>';
}
// Gioi tinh
if (isset($gender)) {
    if ($gender == 'M') {
        $message = '<b><p>Chuc ong mot ngay tot lanh!</p></b>';
    } elseif ($gender == 'F') {
        $message = '<b><p>Chuc ba mot ngay tot lanh!</p></b>';
    }
} else { // Neu gioi tinh khong duoc chon
    $gender = NULL;
    echo '<p><b>Ban quen xac dinh gioi tinh!</b></p>';
}
// Neu moi thong tin deu duoc dien, in ra thong bao.
If ($name && $comments && $email && $gender) {
    echo "Cam on <b>$name</b> da dong gop y kien sau:<br />
    →<tt>$comments</tt><p>Chung toi se hoi am ban theo dia chi
    →<I>$email</I>.</p>";
    echo $message;
}
?>
</body>
</html>
```

Sử dụng hàm ***is_number()*** *để kiểm tra giá trị nhập có phải là số không.*

Trong Chương 8 "An toàn", JavaScript sẽ được sử dụng để kiểm tra tính hợp lệ của dữ liệu biểu mẫu trên máy khách hàng và các biểu thức quy tắc trên máy chủ.

Chú ý, cần báo cho người dùng biết những trường nào là bắt buộc khi họ điền các giá trị vào biểu mẫu, bạn nên bổ sung các thông tin này vào trang ***form.html****.*

Về mặt ngữ nghĩa và lôgic, có thể thiết lập các biến bằng **FALSE** *hoặc* **0** *thay vì là* **NULL** *mà không làm ảnh hưởng đến kết quả cuối cùng.*

Gửi các giá trị đến mã kịch bản bằng cách thủ công

Trong các ví dụ từ đầu cho tới bây giờ, tất cả dữ liệu nhận được trong mã kịch bản **handle_form.php** đều được người dùng nhập vào trên biểu mẫu. Tuy nhiên, còn có hai cách khác để truyền các biến và giá trị cho một mã kịch PHP, bạn cũng nên biết cả hai cách này.

Phương pháp đầu tiên là sử dụng kiểu trường ẩn:

```
<input type="hidden" name="name" value="value" />
```

Miễn là mã lệnh này được đặt trong phạm vi của thẻ **form**, biến **$name** sẽ có giá trị là *value* trong mã kịch bản PHP xử lý biểu mẫu.

Phương pháp thứ hai là nối một giá trị vào URL trỏ đến mã kịch bản xử lý:

```
http://localhost/page.php?name=value
```

Kỹ thuật này giả lập phương pháp **get** của biểu mẫu HTML và có thể thay thế hoặc sử dụng trong biểu mẫu HTML.

Thực hành gửi dữ liệu tới mã kịch bản bằng cách thủ công theo các bước sau:

1. Tạo một hồ sơ HTML mới trong trình soạn thảo văn bản:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
→ Transitional//EN" "http://www.w3.org/TR/2000/REC-xhtml1-
→20000126/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">

<head>

<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />

<title>May tinh tien</title>

</head>

<body>

<!-- Doan ma 2.6 - calculator.html -->
```

2. Tạo một biểu mẫu HTML:

```
<form action="handle_calculator.php?source=calculator.html"
→ method="post">
```

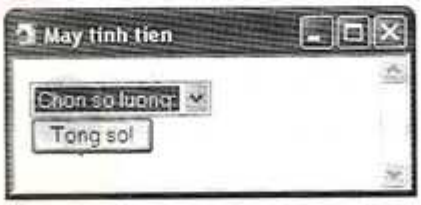
```
<select name="quantity">
<option value="">Select a quantity:</option>
<option value="1">1</option>
<option value="2">2</option>
<option value="3">3</option>
<option value="4">4</option>
<option value="5">5</option>
<option value="6">6</option>
</select>
<div align="left"><input type="submit" name="submit"
→ value="Tong so!" /></div>
<input type="hidden" name="price" value="19.95" />
<input type="hidden" name="taxrate" value=".05" />
</form>
```

Biểu mẫu này có hai đặc điểm mới. Đặc điểm đầu tiên là nó chứa giá và thuế suất trong các trường ẩn. Đặc điểm thứ hai là thuộc tính **attribute** của biểu mẫu có phần **?source=calculator.html** nối vào cuối mã **handle_form.php** thông thường. Bằng cách này, trang xử lý biểu mẫu (**handle_form.php**) sẽ nhận được biến **$source**, cho biết biểu mẫu từ đâu đến.

3. Hoàn tất trang HTML:

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

4. Lưu tập tin theo tên **calculator.html** và xem nó trong trình duyệt Web (xem hình 2-17). Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 2.6.



Hình 2-17: *Biểu mẫu HTML.*

Đoạn mã 2.6: *Biểu mẫu đơn giản này minh họa cách sử dụng các trường ẩn.*

```
<!DOCTYPE html PUBLIC
→ "-//W3C//DTD XHTML 1.0 Transitional//EN"
→ "http://www.w3.org/TR/2000/REC-xhtml1-20000126/DTD/xhtml1-
```

```
→transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">
<head>
<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />
<title>May tinh tien</title>
</head>
<body>
<!-- Doan ma 2.6 - calculator.html -->
<form action="handle_calculator.php?source=calculator.html"
→ method="post">
<select name="quantity">
<option value="">Chon so luong:</option>
<option value="1">1</option>
<option value="2">2</option>
<option value="3">3</option>
<option value="4">4</option>
<option value="5">5</option>
<option value="6">6</option>
</select>
<div align="left"><input type="submit" name="submit"
→ value="Tong so!" /></div>
<input type="hidden" name="price" value="19.95" />
<input type="hidden" name="taxrate" value=".05" />
</form>
<!-Ket thuc bieu mau -->
</body>
</html>
```

Bây giờ, chúng ta sẽ viết mã kịch bản PHP mới để xử lý biểu mẫu trên.

Thực hành xử lý biểu mẫu HTML theo các bước sau:

1. Tạo một hồ sơ PHP mới trong trình soạn thảo văn bản:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
→ Transitional//EN" "http://www.w3.org/TR/2000/REC-xhtml1-
→20000126/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">

<head>

<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />

<title>May tinh tien</title>

</head>

<body>

<?php # Doan ma 2.7 - handle_calculator.php
```

2. Kiểm tra sự chính xác của biến $source:

```
if (isset($source)) {

   if ($source == 'calculator.html') {
```

Điều kiện đầu tiên kiểm tra xem biến $source có hoặc không có giá trị. Điều kiện thứ hai sẽ kiểm tra giá trị của nó có đúng là giá trị mà bạn quan tâm. Nếu bỏ qua điều kiện thứ nhất và biến $source chưa được thiết lập, sẽ có thông báo lỗi khi chạy mã kịch bản này.

3. Kiểm tra tính hợp lệ của số lượng và thực hiện các tính toán:

```
if (is_numeric($quantity)) {

   $total = ($quantity * $price) * ($taxrate + 1);

   $total = number_format ($total, 2);

   echo "Ban mua <b>$quantity</b> mat hang voi gia
   → <b>\$$price</b>. Cong voi thue, tong so tien la
   → <b>\$$total</b>.\n";
```

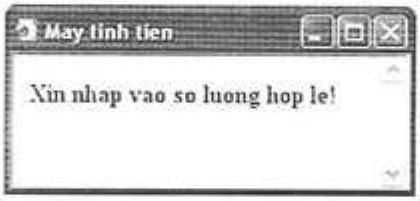
Để kiểm tra tính hợp lệ của số lượng, chúng tôi sử dụng hàm is_numeric(). Có thể sử dụng hàm isset() trước, nhưng biết rằng các giá trị này đã được thiết lập cố định trong biểu mẫu HTML nên bạn có thể bỏ qua phép kiểm tra này. Sau đó, thực hiện các tính toán thích hợp và in kết quả như đã làm trong Chương 1 "Giới thiệu về PHP" (tham khảo lại mã kịch bản 1.1).

4. Hoàn tất điều kiện:

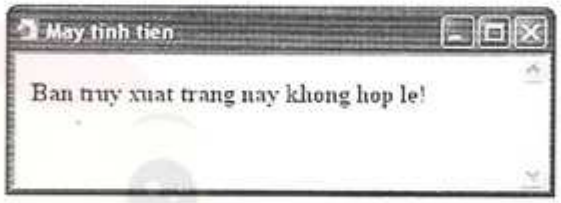
```
   } else {

      echo '<p><b>Xin nhap vao so luong hop le!</b></p>';
```

```
        }
    } else {
        echo '<p><b>Ban truy xuat trang nay khong hop
        → le!</b></p>';
    }
} else {
    echo '<p><b>Ban truy xuat trang nay khong hop
    → le!</b></p>';
}
```

Mệnh đề **else** đầu tiên ứng với trường hợp khi giá trị số lượng không phải là số (sẽ tạo ra thông báo như trong hình 2-18), hai mệnh đề **else** còn lại tham chiếu đến biến **$source** (biến này không được thiết lập chính xác và sẽ dẫn đến thông báo như hình 2-19). Trong ví dụ này, biến **$source** được sử dụng để đảm bảo rằng các tính toán chỉ được thực hiện nếu người dùng gửi dữ liệu đến từ biểu mẫu đúng.



Hình 2-18: *Nếu số lượng không được thiết lập, mã kịch bản sẽ không thực hiện việc tính toán.*

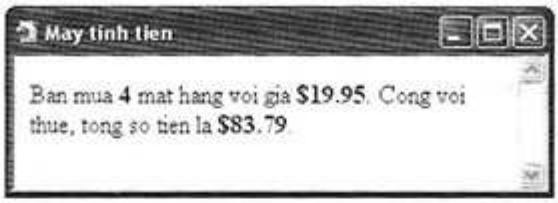


Hình 2-19: *Nếu mã kịch bản không nhận được biến* **$source**, *nó sẽ in ra thông báo lỗi.*

5. Hoàn tất mã kịch bản PHP và trang HTML:

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

6. Lưu tập tin theo tên **handle_calculator.php**, tải nó lên máy chủ cùng với tập tin **calculator.html** và thử cả hai tập tin này trong trình duyệt Web (xem hình 2-20). Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 2.7.



Hình 2-20: *Biểu mẫu cho kết quả giống như những gì đã thực hiện ở Chương 1.*

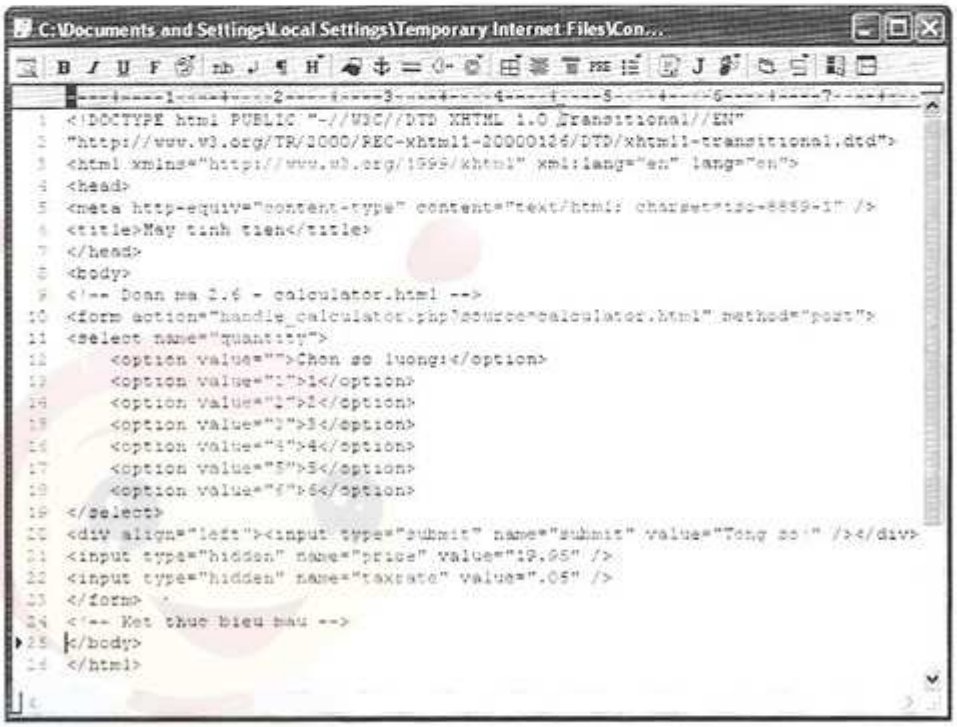
Đoạn mã 2.7: *Mã kịch bản này sử dụng ba biến được thiết lập một cách thủ công.*

```
<!DOCTYPE html PUBLIC
→ "-//W3C//DTD XHTML 1.0 Transitional//EN"
→ "http://www.w3.org/TR/2000/REC-xhtml1-20000126/DTD/xhtml1-
→transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">
<head>
<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />
<title>May tinh tien</title>
</head>
<body>
<?php # Doan ma 2.7 - handle_calculator.php
if (isset($source)) {
   if ($source == 'calculator.html') {
      if (is_numeric($quantity)) {
         $total = ($quantity * $price) * ($taxrate + 1);
         $total = number_format ($total, 2);
         echo "Ban mua <b>$quantity</b> mat hang voi gia
         → <b>\$$price</b>. Cong voi thue, tong so tien la
         → <b>\$$total</b>.\n";
      } else {
         echo '<p><b>Xin nhap vao so luong hop le!</b></p>';
      }
```

```
   } else { // Trang xuat phat khong dung.
      echo '<p><b>Ban truy xuat trang nay khong hop
      → le!</b></p>';
   }
} else {
   echo '<p><b>Ban truy xuat trang nay khong hop le!</b></p>';
}
?>
</body>
</html>
```

*Hàm **parse_url()** của PHP có thể được dùng để phân tách một URL thành các thành phần con (tham khảo tài liệu PHP để biết thêm chi tiết).*

Các trường biểu mẫu ẩn không thể hiện trong trình duyệt Web nhưng vẫn hiện diện trong mã nguồn của biểu mẫu. Vì lý do này, bạn không nên chứa các thông tin cần bảo mật trong các trường này (xem hình 2-21).



Hình 2-21: *Mã nguồn HTML với hai trường ẩn.*

Mảng là gì?

Kiểu biến cuối cùng được đề cập đến trong cuốn sách này là mảng. Không giống với chuỗi và số (là các kiểu biến đơn, chỉ có một giá trị đơn), một mảng có thể chứa nhiều thông tin riêng biệt. Mảng giống như một danh sách các giá trị, mỗi giá trị có thể là một chuỗi, một số hoặc thậm chí là một mảng khác.

Các mảng được cấu tạo dưới dạng một chuỗi các cặp *khóa-giá trị* (*key-value*). Mỗi mục trong danh sách đều có một khóa (hoặc chỉ số *index*) gắn liền với chúng. Cấu trúc kết quả của nó không giống như bảng tính hoặc bảng dữ liệu (xem bảng 2.2).

Bảng 2.2: *Mảng* ***$artists*** *sử dụng các con số làm khóa.*

Khóa	Giá trị
0	Low
1	Aimee Mann
2	Ani Difranco
3	Air

PHP hỗ trợ hai dạng mảng: *sử dụng chỉ số* (*indexed* – sử dụng các con số làm khóa, xem bảng 2.2) và *liên kết* (*associative* – sử dụng chuỗi làm khóa, xem bảng 2.3). Với các mảng *sử dụng chỉ số* sẽ luôn bắt đầu bằng chỉ số `0`, trừ khi khóa được xác định cụ thể.

Bảng 2.3: *Mảng* ***$nations*** *sử dụng chuỗi làm khóa.*

Khóa	Giá trị
VN	Việt Nam
US	Mỹ
RU	Nga
CH	Trung Quốc

Các mảng siêu toàn cục (superglobal arrays)

Mảng rất mạnh và hữu dụng đối với các nhà lập trình, nên PHP còn sử dụng một số dạng mảng mặc định như các mảng siêu toàn cục: ***$_GET, $_POST, $_SESSION, $_COOKIE*** *...*

Biến ***$_GET*** *là nơi PHP chứa tất cả các biến và giá trị được gửi theo phương pháp* ***get*** *(nhưng không nhất thiết phải sử dụng biểu mẫu). Biến* ***$_POST*** *cũng được áp dụng tương tự. Như vậy, nếu tham khảo lại mã kịch bản 2.7,* ***$quantity*** *và* ***$_POST['quantity']*** *sẽ có cùng giá trị.*

So với biến toàn cục, mảng siêu toàn cục có hai lợi điểm. Thứ nhất, chúng an toàn hơn vì chính xác hơn (chúng cho biết các biến đến từ đâu). Thứ hai, chúng có tính toàn cục (như tên gọi của chúng) và bạn sẽ thấy chúng có ý nghĩa hơn khi đọc xong Chương 3 "Tạo Website động". Vì những thuận lợi này, các mảng siêu toàn cục sẽ tiếp tục được sử dụng

cho phần còn lại của cuốn sách. Nếu muốn làm đơn giản cho mã kịch bản (trong khi ***register_globals*** *được kích hoạt trên máy chủ), bạn có thể viết* ***$name*** *tại những chỗ trong cuốn sách này ghi là* ***$_POST['name']*** *hoặc* ***$_GET['name']****.*

Như bạn đã thấy, mảng cũng tuân theo nguyên tắc đặt tên như các biến khác (không giống với ngôn ngữ **PERL**, trong đó mảng sử dụng một tiền tố khác để phân biệt giữa các loại biến). Như vậy, chúng ta không thể nói rằng biến **$var** là một mảng, một chuỗi hay một số. Sự khác biệt về cú pháp sẽ liên quan đến việc truy xuất từng phần tử trong mảng.

Để nhận một giá trị cụ thể từ mảng, cần tham chiếu tên mảng trước và tiếp theo là khóa đặt phía trong dấu ngoặc vuông:

```
echo $artists[2];
echo $nations['VN'];
```

Vì mảng sử dụng cú pháp không giống với các biến khác, nên việc in chúng ra cũng bị giới hạn hơn. Ví dụ, mã lệnh sau sẽ gây ra một lỗi biên dịch:

```
echo "CH là từ viết tắt của $nations['CH']";
```

Để giải quyết vấn đề này, hãy đặt tên mảng và khóa trong cặp dấu ngoặc móc ({}):

```
echo "CH là từ viết tắt của {$nations['CH']}";
```

Để có ví dụ cụ thể hơn, chúng ta sẽ sửa mã kịch bản **handle_calculator.php** (tham khảo đoạn mã 2.7) để sử dụng các mảng siêu toàn cục.

Thực hành sử dụng mảng theo các bước sau:

1. Mở tập tin **handle_calculator.php** trong trình soạn thảo văn bản (tham khảo đoạn mã 2.7).
2. Thay thế hai tham chiếu đến biến **$source** bằng các mảng siêu toàn cục.

```
if (isset($_GET['source'])) {
   if ($_GET['source'] == 'calculator.html') {
```

Vì biến **$source** nằm trong URL, bạn sử dụng mảng **$_GET** để truy xuất nó.

3. Viết lại phần tính toán bằng cách sử dụng các mảng siêu toàn cục:

```
$total = ($_POST['quantity'] * $_POST['price']) *
→ ($_POST['taxrate'] + 1);
$total = number_format ($total, 2);
```

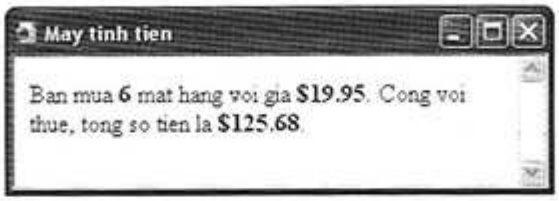
Thay vì tham chiếu đến **$quantity**, **$price** và **$taxrate**, chúng ta sử dụng **$_POST['quantity']**, **$_POST['price']** và **$_POST['taxrate']**. Biến **$_POST** được dùng thay cho **$_GET** vì những biến này được nhận từ biểu mẫu HTML theo phương pháp **POST**. Biến **$total** không nằm trong các mảng siêu toàn cục nên được giữ nguyên.

4. Thay đổi câu lệnh **echo()**:

```
echo "Ban mua <b>{$_POST['quantity']}</b> mat hang voi gia
→ <b>\${$_POST['price']}</b>. Cong voi thue, tong so tien
→ la <b>\$$total</b>.\n";
```

Tuy đã có khái niệm về mảng, bạn vẫn cần biết cú pháp khi muốn in giá trị của nó. Để in giá trị của một mảng, phải sử dụng dấu ngoặc móc (ví dụ: `>{$_POST['quantity']}` và `>{$_POST['price']}`) để tránh các lỗi biên dịch.

5. Lưu lại tập tin, chép nó lên máy chủ Web và thử trong trình duyệt (xem hình 2-22). Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 2.8.



Hình 2-22: *Các biến siêu toàn cục không làm ảnh hưởng đến hoạt động của mã kịch bản và an toàn hơn.*

Đoạn mã 2.8: *Các biến siêu toàn cục chỉ là một dạng mảng trong PHP.*

```
<!DOCTYPE html PUBLIC
→ "-//W3C//DTD XHTML 1.0 Transitional//EN"
→ "http://www.w3.org/TR/2000/REC-xhtml1-20000126/DTD/xhtml1-
→transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">
<head>
<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />
<title>May tinh tien</title>
</head>
<body>
<?php # Doan ma 2.8 - handle_calculator.php
if (isset($_GET['source'])) {
  if ($_GET['source'] == 'calculator.html') {
    if (is_numeric($_POST['quantity'])) {
      $total = ($_POST['quantity'] * $_POST['price']) *
      → ($_POST['taxrate'] + 1);
      $total = number_format ($total, 2);
      echo "Ban mua <b>{$_POST['quantity']}</b> mat hang voi
      →gia <b>\${$_POST['price']}</b>. Cong voi thue, tong
```

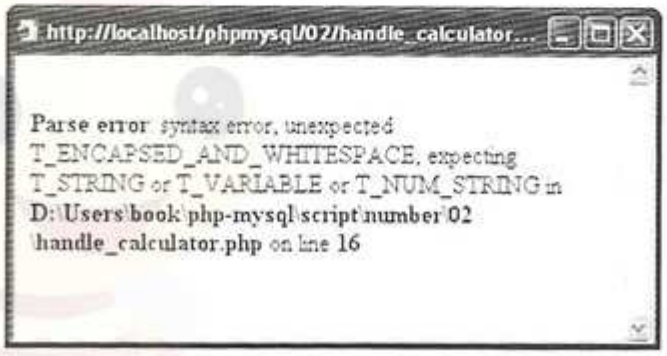
```
        →so tien la <b>\$$total</b>.\n";
      } else {
         echo '<p><b>Xin xac dinh so luong dat mua!</b></p>';
      }
   } else {
      echo '<p><b>Ban truy xuat trang nay khong dung!</b></p>';
   }
} else {
   echo '<p><b>Ban truy xuat trang nay khong dung!</b></p>';
}
?>
</body>
</html>
```

PHP rất "thoải mái" với cấu trúc biến, thậm chí có thể sử dụng số và chuỗi làm khóa của mảng. Một quy tắc quan trọng cần nhớ: Khóa của mảng phải là duy nhất.

Tuy không bắt buộc, nhưng bạn nên đặt khóa trong dấu nháy khi tham chiếu đến một phần tử trong mảng (ví dụ ***$array[key]****).*

Biến siêu toàn cục mới có kể từ phiên bản PHP 4.1. Nếu sử dụng các phiên bản trước, có thể dùng ***$HTTP_POST_VARS*** *thay cho* ***$_POST*** *và* ***$HTTP_GET_VARS*** *thay cho* ***$_GET****.*

Trong những cú pháp phức tạp, rất dễ gặp phải lỗi biên dịch khi sử dụng mảng với các khóa là chuỗi (xem hình 2-23). Bạn nên tuân theo quy tắc quan trọng: Hãy bao tên mảng trong dấu ngoặc nhọn khi sử dụng chúng trong các dấu nháy kép.



Hình 2-23: *Cú pháp mảng quá phức tạp có thể sẽ gây ra lỗi biên dịch.*

Tạo mảng

Trong các ví dụ trước, chúng ta đã sử dụng mảng được tạo bởi PHP, nhưng việc cần tạo ra các mảng của riêng mình khi viết mã cũng là một việc rất thường xuyên. Có hai cách cơ bản để định nghĩa một mảng. Đầu tiên, ta có thể bổ sung từng phần tử một vào mảng.

Trong ví dụ trước, chúng ta đã sử dụng mảng được tạo bởi PHP. Tuy nhiên, bạn sẽ thường xuyên cần tạo các mảng cho riêng mình trong khi viết mã. Có hai cách cơ bản để định nghĩa một mảng.

Cách thứ nhất, bổ sung từng phần tử vào mảng:

```
$array[] = 'giá trị 1';
$array[] = 'giá trị 2';
$array['khóa'] = 'giá trị 3';
```

Một điều quan trọng cần lưu ý là nếu xác định một khóa và một giá trị đã có tại chỉ số ứng với khóa đó, giá trị mới sẽ thay thế cho giá trị cũ. Ví dụ:

```
$array['name'] = 'Larry';
$array['name'] = 'Henrik';
$array[2] = 'giá trị 1';
$array[2] = 'giá trị 2';
```

Cách thứ hai, sử dụng hàm **array()** để xây dựng toàn bộ mảng chỉ trong một bước:

```
$array = array('khóa' => 'giá trị', 'khóa 2' => 'giá trị 2');
```

Hàm này có thể được dùng để xác định cụ thể khóa hoặc không xác định khóa:

```
$array = array('giá trị', 'giá trị 2');
```

Nếu thiết lập giá trị đầu là một giá trị khóa dạng số, các giá trị thêm vào sẽ tạo khóa tăng dần:

```
$days = array(1 => 'Chủ nhật', 'Thứ hai', 'Thứ ba');
echo $days[3]; // Thứ ba
```

Cuối cùng, nếu muốn tạo một mảng số liên tục, bạn có thể sử dụng hàm **range()**:

```
$ten = range(1, 10);
```

Truy xuất mảng

Bạn đã biết cách truy xuất từng phần tử mảng theo khóa (ví dụ: `$_POST['price']`). Tuy nhiên, cách này chỉ thích hợp khi biết chính xác khóa của phần tử muốn truy xuất hoặc khi muốn tham chiếu đến một phần tử duy nhất trong mảng. Để truy xuất mọi phần tử trong mảng, chúng ta sẽ sử dụng vòng lặp **foreach**:

```
foreach ($array as $value){
        // thực hiện lệnh
```

```
}
```

Vòng lặp **foreach** sẽ duyệt qua từng phần tử trong mảng **$array** (từng phần tử trong danh sách mảng), gán giá trị của từng phần tử này cho biến **$value**. Để truy xuất cả khóa và giá trị, sử dụng mã lệnh sau:

```
foreach ($array as $key => $value){
        echo "Giá trị mảng ứng với khóa $key là $value";
}
```

Bằng việc sử dụng mảng, bạn có thể dễ dàng tạo ra một menu thả xuống dùng để chọn một ngày.

Thực hành việc tạo và truy xuất mảng theo các bước sau:

1. Tạo một hồ sơ PHP mới trong trình soạn thảo văn bản:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
→ Transitional//EN" "http://www.w3.org/TR/2000/REC-xhtml1-
→ 20000126/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">
<head>
<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />
<title>Lich</title>
</head>
<body>
<?php # Doan ma 2.9 - calendar.php
```

2. Tạo một mảng chứa tên các tháng trong năm:

```
$months = array (1 => 'January', 'February', 'March',
→ 'April', 'May', 'June', 'July', 'August', 'September',
→ 'October', 'November', 'December');
```

Mảng này sử dụng các con số làm khóa, từ 1 đến 12. Vì giá trị của khóa thứ nhất đã được xác định nên các giá trị sau sẽ được tăng một cách tuần tự.

3. Tạo các mảng chứa các ngày trong tháng và năm:

```
$days = range (1, 31);
$years = range (2003, 2010);
```

Bằng việc sử dụng hàm **range()**, bạn sẽ dễ dàng tạo ra các mảng số.

4. Bắt đầu biểu mẫu HTML:

```
echo '<form action="calendar.php" method="post">';
```

Hiện giờ, biểu mẫu chưa được sử dụng cụ thể nên chúng ta thiết lập thuộc tính **action** để tham chiếu lại chính mã kịch bản này. Các chương sau sẽ sử dụng chúng trong các ứng dụng Web thực tế.

5. Phát sinh menu thả xuống cho tháng:

```
echo '<select name="month">';
foreach ($months as $key => $value) {
   echo "<option value=\"$key\">$value</option>\n";
}
echo '</select>';
```

Vòng lặp **foreach** được dùng để dễ dàng tạo ra toàn bộ mã lệnh cho menu tháng. Mỗi lần thực hiện, vòng lặp sẽ tạo ra một dòng có dạng **<option value="1">January</option>**.

6. Tạo ra các menu ngày và năm:

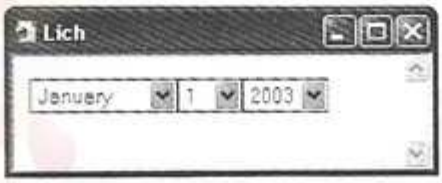
```
echo '<select name="day">';
foreach ($days as $value) {
   echo "<option value=\"$value\">$value</option>\n";
}
echo '</select>';
echo '<select name="year">';
foreach ($years as $value) {
   echo "<option value=\"$value\">$value</option>\n";
}
echo '</select>
</form>';
```

Không giống như menu tháng, cả menu ngày và năm đều sử dụng một giá trị cho cả phần giá trị và phần nhãn của mỗi mục menu. Đó là lý do tại sao không cần xác định khóa khi tạo ra các mảng này.

7. Đóng thẻ PHP và hoàn tất hồ sơ HTML:

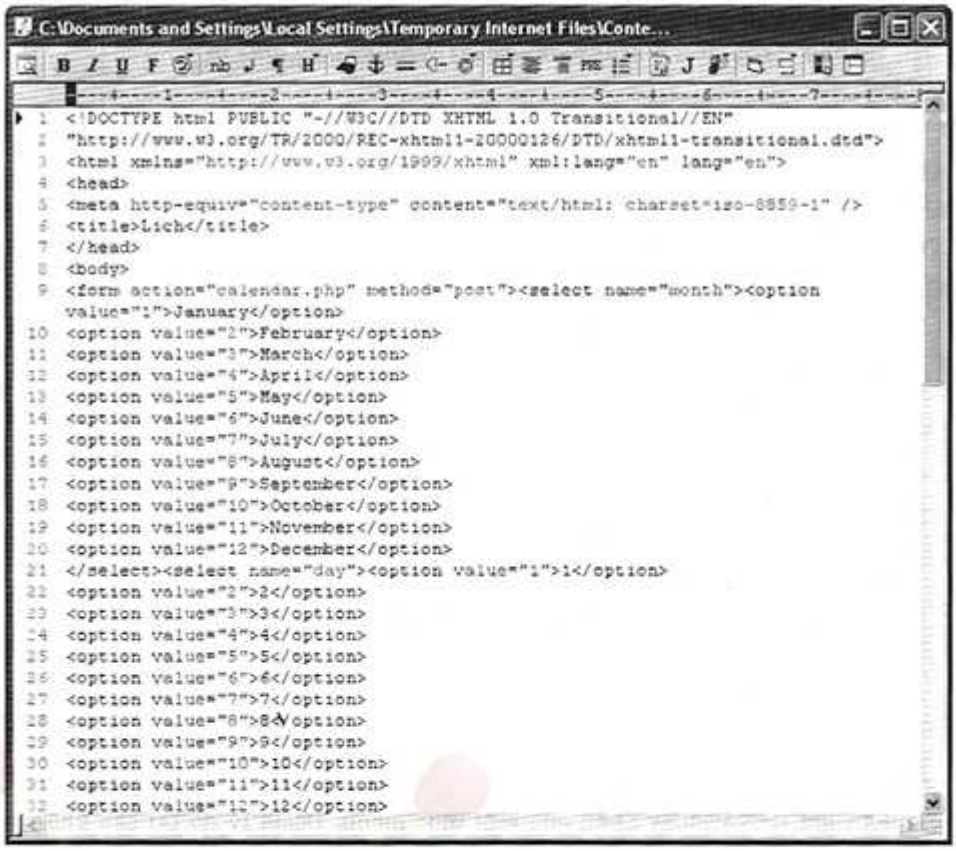
```
?>
</body>
</html>
```

8. Lưu tập tin theo tên **calendar.php**, tải nó lên máy chủ Web và thử trong trình duyệt (xem hình 2-24). Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 2.9.



Hình 2-24: *Các menu thả xuống được tạo bằng các mảng và vòng lặp* ***foreach***

9. Nếu cần, xem mã nguồn trong trình duyệt bằng cách chọn View > Source hoặc View > Page Source (xem hình 2-25).



Hình 2-25: *Phần lớn mã nguồn được phát sinh chỉ với một số dòng lệnh PHP.*

Đoạn mã 2.9: *Mảng được sử dụng để tạo "động" ba menu thả xuống (xem lại hình 2-24).*

```
<!DOCTYPE html PUBLIC
→ "-//W3C//DTD XHTML 1.0 Transitional//EN"
→ "http://www.w3.org/TR/2000/REC-xhtml1-20000126/DTD/xhtml1-
→ transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">

<head>

<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />
```

```
<title>Lich</title>
</head>
<body>
<?php # Doan ma 2.9 - calendar.php
$months = array (1 =>'January', 'February', 'March', 'April',
→ 'May', 'June', 'July', 'August', 'September', 'October',
→ 'November', 'December');
$days = range (1, 31);
$years = range (2003, 2010);
echo '<form action="calendar.php" method="post">';
echo '<select name="month">';
foreach ($months as $key => $value) {
   echo "<option value=\"$key\">$value</option>\n";
}
echo '</select>';
echo '<select name="day">';
foreach ($days as $value) {
   echo "<option value=\"$value\">$value</option>\n";
}
echo '</select>';
echo '<select name="year">';
foreach ($years as $value) {
   echo "<option value=\"$value\">$value</option>\n";
}
echo '</select>
</form>';
?>
</body>
</html>
```

Để xác định số phần tử trong một mảng, hãy sử dụng hàm **`count()`** *hoặc* **`sizeof()`** *(hai hàm này tương đương nhau). Ví dụ:*

```
$num = count($array);
```

Hàm **range()** *cũng có thể tạo ra một mảng các chữ cái liên tục (áp dụng từ PHP 4.1):*

```
$alphabet = range(a,z);
```

Kết cấu **foreach** *sẽ không ảnh hưởng đến bản thân mảng, cho dù bạn làm gì với từng giá trị riêng biệt, phía trong vòng lặp.*

Vì **foreach** *sử dụng bản sao của mảng nên sẽ đòi hỏi nhiều bộ nhớ khi làm việc với các mảng lớn.*

Các khóa của một mảng có thể là một chuỗi gồm nhiều từ, chẳng hạn như **first name** *hoặc* **phone number**. *Trong trường hợp này, phải đặt các khóa trong các dấu nháy khi tham chiếu đến chúng.*

Hàm **is_array()** *kiểm tra xem một biến có phải là một mảng hay không.*

Không nhất thiết phải sử dụng các tên **$key** *và* **$value** *trong vòng lặp* **foreach**, *nhưng đây là các lựa chọn hợp lý (một số lập trình viên lại tham chiếu chúng với tên viết tắt* **$k** *và* **$v**).

Mảng nhiều chiều

Như đã giới thiệu trước đây, các giá trị của một mảng có thể là một tổ hợp bất kỳ của số, chuỗi và thậm chí là các mảng khác. Việc một mảng chứa các mảng khác sẽ tạo ra một mảng nhiều chiều.

Các mảng nhiều chiều khá phổ biến (đặc biệt khi sử dụng các biến siêu toàn cục) và rất dễ dùng. Ví dụ, bảng 2.3 thể hiện một mảng có tên là **$nations** được tạo bởi:

```
$nation = array('VN' => 'Vietnam', 'US' => 'United States',
→ 'UK' => 'United Kingdom');
```

Nếu bạn có một mảng khác tương tự về các tỉnh thành ở Việt Nam, ví dụ:

```
$vietnam = array('HN'=>'Hanoi','HM'=>'HoChiMinh city');
```

Khi đó, hai mảng này có thể kết hợp với nhau thành một mảng nhiều chiều như sau:

```
$abbr = array('VN' => $vietnam, 'US' => $unitedstates, 'UK' =>
$unitedkingdom);
```

Giờ đây, mảng **$nation** là một mảng nhiều chiều. Để truy xuất mảng **$vietnam**, sử dụng cú pháp **$nation['VN']**. Để truy xuất Hà Nội, sử dụng cú pháp **$nation['VN']['HN']**. Bạn chỉ việc sử dụng tên của mảng nhiều chiều, tiếp theo là khóa của mảng thứ nhất trong dấu ngoặc vuông, cuối cùng là khóa của mảng

thứ hai cũng trong dấu ngoặc vuông. Để in ra một trong các giá trị, hãy bao bọc toàn bộ cấu trúc này trong dấu ngoặc móc:

```
echo "Tỉnh/thành phố của Việt Nam có tên viết tắt HN là
{$nation['VN']['HN']}";
```

Dĩ nhiên, bạn vẫn có thể truy xuất các mảng nhiều chiều bằng cách sử dụng vòng lặp **foreach** và lồng chúng vào nhau nếu cần.

Thực hành sử dụng mảng nhiều chiều theo các bước sau:

1. Tạo một hồ sơ HTML mới trong trình soạn thảo văn bản:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
→ Transitional//EN" "http://www.w3.org/TR/2000/REC-xhtml1-
→ 20000126/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">

<head>

<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />

<title>Noi cho chung toi biet ve ban</title>

</head>

<body>

<!-- Doan ma 2.10 - about.html -->
```

2. Bắt đầu biểu mẫu HTML:

```
<form action="handle_about.php" method="post">

<fieldset><legend>Nhap vao thong tin ve ban:</legend>

<p><b>Ten:</b> <input type="text" name="name" size="20"
→ maxlength="40" /></p>
```

3. Tạo một loạt hộp kiểm để người dùng có thể lựa chọn sở thích của mình:

```
<p><b>So thich:</b>

<input type="checkbox" name="interests[]" value="Am nhac"/>
→ Am nhac

<input type="checkbox" name="interests[]" value="Xem phim"/>
→ Xem phim

<input type="checkbox" name="interests[]" value="Doc sach"/>
→ Doc sach
```

```
<input type="checkbox" name="interests[]" value="Truot
→ tuyet"/> Truot tuyet

<input type="checkbox" name="interests[]" value="Da banh"/>
→ Da banh

</p>
```

Phần Interests sẽ cho phép người dùng chọn nhiều tùy chọn. Trong trường hợp này, có hai cách đặt tên cho các trường nhập: đặt tên theo giá trị (ví dụ: *Music*, *Movie*...) hoặc sử dụng mảng. Ở đây, chúng ta sử dụng *interest[]* làm tên gọi và trong mã kịch bản PHP xử lý biểu mẫu sẽ tạo ra mảng có tên **interests** để chứa tất cả những tùy chọn của người dùng.

4. Hoàn tất trang:

```
</fieldset>

<div align="center"><input type="submit" name="submit"
→ value="Gui thong tin" /></div>

</form>

</body>

</html>
```

5. Lưu tập tin theo tên **about.html**. Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 2.10.

Đoạn mã 2.10: *Biểu mẫu này có một loạt hộp kiểm mà người dùng có thể chọn.*

```
<!DOCTYPE html PUBLIC
→ "-//W3C//DTD XHTML 1.0 Transitional//EN"
→ "http://www.w3.org/TR/2000/REC-xhtml1-20000126/DTD/xhtml1-
→transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">
<head>
<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />
<title>Noi voi chung toi ve ban</title>
</head>
<body>
<!-Doan ma 2.10 – about.html -->
<form action="handle_about.php" method="post">
<fieldset><legend>Nhap vao thong tin ve ban:</legend>
<p><b>Ten:</b> <input type="text" name="name" size="20"
```

```
→ maxlength="40" /></p>
<p><b>So thich:</b>
<input type="checkbox" name="interests[]" value="Am nhac"/>
→ Am nhac
<input type="checkbox" name="interests[]" value="Xem phim"/>
→ Xem phim
<input type="checkbox" name="interests[]" value="Doc sach"/>
→ Doc sach
<input type="checkbox" name="interests[]" value="Truot
→ tuyet"/> Truot tuyet
<input type="checkbox" name="interests[]" value="Da banh"/>
→ Da banh
</p>
</fieldset>
<div align="center"><input type="submit" name="submit"
→ value="Gui thong tin" /></div>
</form>
<!-Ket thuc bieu mau -->
</body>
</html>
```

Bây giờ, chúng ta sẽ điều chỉnh tập tin **handle_form.php** để sử dụng mảng **$interests**.

Thực hành xử lý biểu mẫu theo các bước sau:

1. Tạo một hồ sơ PHP mới trong trình soạn thảo văn bản:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
→ Transitional//EN" "http://www.w3.org/TR/2000/REC-xhtml1-
→ 20000126/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">
<head>
<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />
<title>Phan hoi</title>
</head>
<body>
<?php # Doan ma 2.11 - handle_about.php
```

2. Kiểm tra xem tên có được nhập vào hay không:

```
if (strlen($_POST['name']) > 0) {
   $name = stripslashes($_POST['name']);
} else {
   $name = NULL;
   echo '<p><b>Ban quen nhap ten!</b></p>';
}
```

Mã lệnh ở đây giống như mã lệnh trong các mã kịch bản trước đây (tham khảo đoạn mã 2.5), nhưng lần này, chúng ta sử dụng biến siêu toàn cục **$_POST**.

3. Bổ sung các dòng lệnh để xử lý phần sở thích của người dùng:

```
if (isset($_POST['interests'])) {
   $ints = NULL;
   foreach ($_POST['interests'] as $key => $value) {
   $ints .= "$value, ";
   }
   $ints = substr($ints, 0, -2);
   $interests = TRUE;
} else {
   $interests = NULL;
   echo '<p><b>Ban quen nhap so thich!</b></p>';
}
```

Phần đầu tiên của mã sẽ kiểm tra xem biến có giá trị hay không. Nếu có, một vòng lặp sẽ được sử dụng để tạo thông báo (**$ints**) chứa tất cả các sở thích của người dùng.

Để bắt đầu xây dựng thông báo này, chúng ta sẽ khởi tạo biến **$ints** là **NULL**. Sau đó, trong phạm vi vòng lặp, các giá trị sẽ kế tiếp nhau nối vào thông báo, được phân cách bằng một dấu phẩy và khoảng trắng. Lần đầu tiên chạy qua vòng lặp, sở thích đầu tiên sẽ được đưa vào biến **$ints** (Music,), lần thứ hai, sở thích kế tiếp sẽ được thêm vào (Music, Movies,)... Ở cuối chuỗi kết quả sẽ có một khoảng trắng và dấu phẩy dư thừa, hàm **substr()** sẽ được dùng để loại bỏ hai ký tự này.

Do bản chất của hộp kiểm, chỉ có những hộp nào được chọn mới có một giá trị trong mảng **$interests**.

4. Thêm điều kiện cuối cùng:

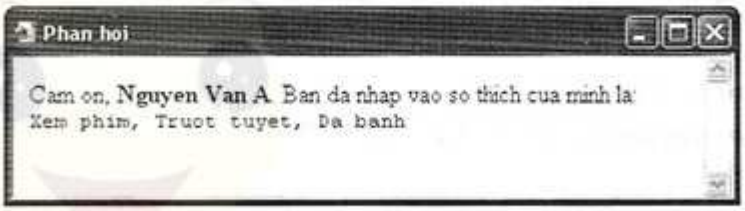
```
if ($name && $interests) {
    echo "Cam on, <b>$name</b>. Ban da nhap vao so thich cua
    → minh la:<br /><tt>$ints</tt></p>";
}
```

5. Hoàn tất mã lệnh PHP và trang HTML:

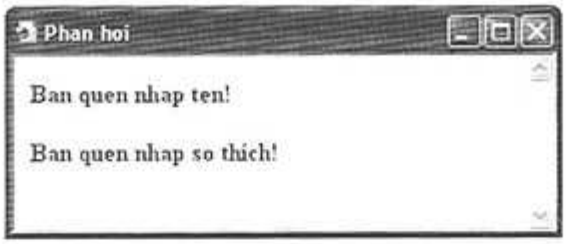
```
?>
</body>
</html>
```

6. Lưu tập tin theo tên **handle_about.php**, tải nó lên máy chủ Web cùng với tập tin **about.html** và thử cả hai tập tin này trong trình duyệt Web (xem hình 2-26, 2-27 và 2-28). Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 2.11.

Hình 2-26: *Biểu mẫu HTML này có sử dụng các hộp kiểm.*



Hình 2-27: *Các sở thích đã chọn được thể hiện trở lại cho người dùng ở dạng danh sách, cách nhau bằng dấu phẩy.*



Hình 2-28: *Nếu tên không được nhập hoặc không có sở thích nào được chọn, các thông báo lỗi sẽ được tạo.*

Đoạn mã 2.11: *Mã kịch bản này sử dụng một mảng nhiều chiều,* ***$interests****.*

```
<!DOCTYPE html PUBLIC
→ "-//W3C//DTD XHTML 1.0 Transitional//EN"
→ "http://www.w3.org/TR/2000/REC-xhtml1-20000126/DTD/xhtml1-
→transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">
<head>
<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />
<title>Phan hoi</title>
</head>
<body>
<?php # Doan ma 2.11 - handle_about.php
// Kiem tra $_POST['name'] va loai bo cac dau cheo.
if (strlen($_POST['name']) > 0) {
   $name = stripslashes($_POST['name']);
} else { // Neu ten khong duoc nhap...
   $name = NULL;
   echo '<p><b>Ban quen nhap ten!</b></p>';
}
// Kiem tra $interests[]
if (isset($_POST['interests'])) {
   $ints = NULL; // Thong bao moi se duoc dung.
   foreach ($_POST['interests'] as $key => $value) {
      $ints .= "$value, ";
```

```
    }
    $ints = substr($ints, 0, -2);
    $interests = TRUE;
} else {
    $interests = NULL;
    echo '<p><b>Ban quen nhap so thich!</b></p>';
}
// Neu tat ca thong tin duoc nhap, in ra thong bao.
if ($name && $interests) {
    echo "Cam on, <b>$name</b>. Ban da nhap vao so thich cua
    → minh la:<br /><tt>$ints</tt></p>";
}
?>
</body>
</html>
```

Nếu không muốn sử dụng biến siêu toàn cục và tùy chọn ***register_globals*** *được kích hoạt trên máy chủ, bạn có thể sử dụng biến* ***$interests*** *và* ***$name*** *thay cho* ***$_POST['interests']*** *và* ***$_POST['name']***.

Ví dụ này xử lý một mảng nhiều chiều ***$_POST*** *với mã lệnh không quá phức tạp. Việc kết hợp tính linh động của chúng và tính dễ dùng của PHP làm cho mảng trở thành một công cụ hấp dẫn đối với các nhà lập trình.*

Mảng và chuỗi

Vì chuỗi và mảng được sử dụng khá phổ biến nên PHP có hai hàm để hoán đổi chúng:

```
$array = explode(separator, $string);
$string = implode(glue, $array);
```

Chìa khóa giúp bạn hiểu và sử dụng hai hàm này chính là mối quan hệ giữa *separator* và *glue*. Khi *glue* được thiết lập để chuyển một mảng thành một chuỗi, các ký tự hoặc mã sẽ được chèn giữa các giá trị của mảng khi tạo ra chuỗi. Ngược lại, khi chuyển một chuỗi thành mảng, bạn phải xác định dấu phân cách (*separator*). Dấu phân cách là ký tự hoặc mã nằm giữa các phần tử khác nhau trong chuỗi.

```
$string = 'Mon - Tue - Wed - Thu - Fri';
```

```
$array = explode(' – ', $string);
```

Biến **$array** đã có năm phần tử, với **Mon** ở chỉ số **0**, **Tue** ở chỉ số **1**...

```
$string = implode (', ', $array);
```

Bây giờ, biến **$string** đã là một danh sách ngày trong tuần, cách nhau bằng dấu phẩy: *Mon, Tue, Wed...*

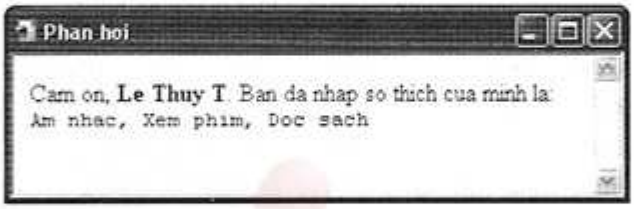
Thực hành việc chuyển đổi một mảng thành chuỗi theo các bước sau:

1. Mở tập tin **handle_form.php** (tham khảo đoạn mã 2.11) trong trình soạn thảo văn bản.
2. Thay thế nội dung của điều kiện sở thích bằng mã lệnh sau:

```
$ints = implode (', ', $_POST['interests']);
```

Hàm này đã thay cho bốn dòng lệnh trong mã kịch bản trước. Vòng lặp không còn cần thiết nữa và không cần khởi tạo biến **$ints**.

3. Lưu tập tin theo tên **handle_about.php**, tải nó lên máy chủ Web và chạy thử trong trình duyệt (xem hình 2-29). Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 2.12.



Hình 2-29: *Mặc dù mã kịch bản này làm việc giống như mã kịch bản trước đây (xem hình 2-27), nhưng mã lệnh của nó ngắn và dễ hiểu hơn (tham khảo mã kịch bản 2.12).*

Đoạn mã 2.12: *Hàm* ***implode()*** *thực hiện việc chuyển đổi giữa mảng và chuỗi.*

```
<!DOCTYPE html PUBLIC
→ "-//W3C//DTD XHTML 1.0 Transitional//EN"
→ "http://www.w3.org/TR/2000/REC-xhtml1-20000126/DTD/xhtml1-
→ transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">
<head>
<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />
<title>Phan hoi</title>
```

```
</head>
<body>
<?php # Doan ma 2.12 - handle_about.php
if (strlen($_POST['name']) > 0) {
   $name = stripslashes($_POST['name']);
} else {
   $name = NULL;
   echo '<p><b>Ban quen nhap ten!</b></p>';
}
if (isset($_POST['interests'])) {
   $ints = implode (', ', $_POST['interests']);
   $interests = TRUE;
} else {
   $interests = NULL;
   echo '<p><b>Ban quen nhap so thich!</b></p>';
}
if ($name && $interests) {
   echo "Cam on, <b>{$_POST['name']}</b>. Ban da nhap so thich
   → cua minh la:<br /><tt>$ints</tt></p>";
}
?>
</body>
</html>
```

Hàm ***join()*** *là một tên gọi khác của hàm* ***implode()****.*

Hàm ***explode()*** *nhận vào một tham số thứ ba để giới hạn số phần tử tạo ra trong mảng.*

Sắp xếp mảng

So với các dạng biến khác, một trong các ưu điểm nổi trội của mảng là khả năng sắp xếp. Trong PHP, có một số hàm được dùng để sắp xếp mảng với cú pháp rất đơn giản:

```
$array = sort($array);
```

Các hàm này thực hiện ba dạng sắp xếp. Dạng thứ nhất, sử dụng hàm **sort()** sắp xếp mảng theo giá trị, bất chấp khóa của chúng. Cần đặc biệt lưu ý là các khóa của

mảng sẽ được thiết lập lại sau quá trình sắp xếp. Bạn không nên sử dụng hàm này nếu mối quan hệ *khóa - giá trị* là quan trọng.

Dạng thứ hai, sử dụng hàm **asort()** sắp xếp mảng theo giá trị nhưng vẫn duy trì khóa. Dạng thứ ba, sử dụng hàm **ksort()** sắp xếp mảng theo khóa. Các dạng sắp xếp này đều có thể đảo ngược chiều sắp xếp đổi chúng thành các hàm **rsort()**, **arsort()** và **ksort()**.

Để minh họa việc sắp xếp mảng, chúng ta sẽ tạo một mảng với các tiêu đề phim và kết quả đánh giá (với tỉ lệ từ 1 đến 10), rồi thể hiện danh sách này theo các cách khác nhau.

Thực hành sắp xếp một mảng theo các bước sau:

1. Tạo một hồ sơ PHP mới trong trình soạn thảo văn bản:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
→ Transitional//EN" "http://www.w3.org/TR/2000/REC-xhtml1-
→ 20000126/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">
<head>
<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />
<title>Sap xep mang</title>
</head>
<body>
<?php # Doan ma 2.13 - sorting.php
```

2. Tạo một mảng mới:

```
$movies = array (
'10' => 'Casablanca',
'9.65' => 'To Kill a Mockingbird',
'2.35' => 'The English Patient',
'8' => 'Traffic'
);
```

Mảng **$movies** sử dụng tiêu đề phim làm giá trị và kết quả đánh giá làm khóa. Cấu trúc này sẽ tạo ra một số khả năng truy xuất đến toàn bộ danh sách. Các con số đánh giá phải được đặt trong dấu nháy hoặc PHP sẽ bỏ qua phần số lẻ (các khóa dạng số sẽ luôn luôn là các số nguyên).

3. In ra giá trị gốc của mảng:

```
echo 'Xep theo thu tu tu nhien:<br /><pre>';
```

```
foreach ($movies as $key => $value) {
   echo "$key\t$value\n";
}
echo '</pre>';
```

Mảng tuân theo trật tự như khi nó được tạo. Đoạn mã trên đã in ra các giá trị của mảng để kiểm tra điều này.

4. Sắp xếp mảng theo thứ tự chữ cái dựa trên tiêu đề phim và in lại mảng:

```
echo 'Xep theo tieu de:<br /><pre>';
asort($movies);
foreach ($movies as $key => $value) {
   echo "$key\t$value\n";
}
echo '</pre>';
```

Hàm **asort()** sắp xếp mảng theo giá trị, trong khi vẫn giữ mối quan hệ *khóa - giá trị*.

5. Sắp xếp mảng theo chiều giảm dần của kết quả đánh giá và in lại mảng:

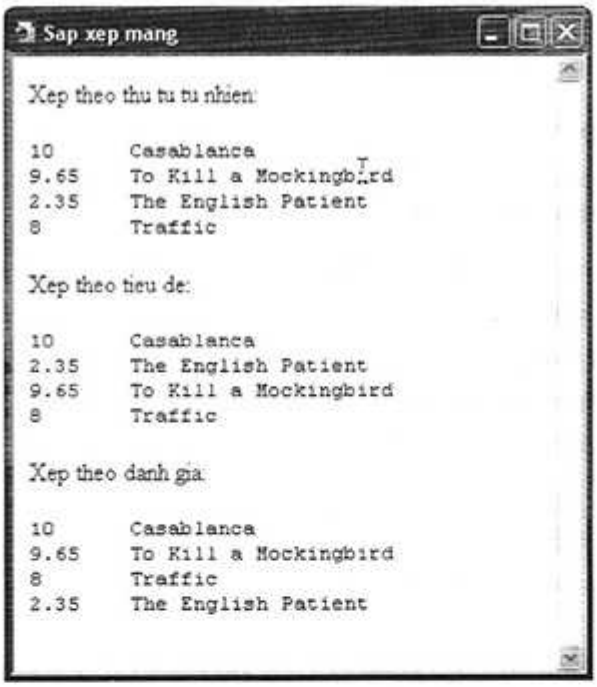
```
echo 'Xep theo danh gia:<br /><pre>';
krsort($movies);
foreach ($movies as $key => $value) {
   echo "$key\t$value\n";
}
echo '</pre>';
```

Hàm **ksort()** sẽ sắp xếp mảng theo khóa, nhưng theo thứ tự tăng dần. Sử dụng hàm **krsort()** nếu muốn đảo ngược thứ tự sắp xếp (phim có kết quả đánh giá cao sẽ xuất hiện trước). Hàm này và hàm **asort()** vẫn duy trì mối quan hệ *khóa - giá trị*.

6. Hoàn tất mã lệnh PHP và trang HTML:

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

7. Lưu tập tin theo tên **sorting.php**, tải nó lên máy chủ Web và thử trong trình duyệt Web (xem hình 2-30). Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 2.13.



Hình 2-30: *Trang này minh họa những cách sắp xếp mảng.*

Đoạn mã 2.13: *Các mảng có thể được sắp xếp theo nhiều cách khác nhau.*

```
<!DOCTYPE html PUBLIC
→ "-//W3C//DTD XHTML 1.0 Transitional//EN"
→ "http://www.w3.org/TR/2000/REC-xhtml1-20000126/DTD/xhtml1-
→ transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">
<head>
<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />
<title>Sap xep mang</title>
</head>
<body>
<?php # Doan ma 2.13 - sorting.php
// Tao mang
$movies = array (
'10' => 'Casablanca',
'9.65' => 'To Kill a Mockingbird',
```

```
'2.35' => 'The English Patient',
'8' => 'Traffic'
);
// Hien thi phim theo thu tu tu nhien:
echo 'Xep theo thu tu tu nhien:<br /><pre>';
foreach ($movies as $key => $value) {
   echo "$key\t$value\n";
}
echo '</pre>';
// Xep theo tieu de:
echo 'Xep theo tieu de:<br /><pre>';
asort($movies);
foreach ($movies as $key => $value) {
   echo "$key\t$value\n";
}
echo '</pre>';
// Xep theo danh gia:
echo 'Xep theo danh gia:<br /><pre>';
krsort($movies);
foreach ($movies as $key => $value) {
   echo "$key\t$value\n";
}
echo '</pre>';
?>
</body>
</html>
```

Để tạo ra một trật tự ngẫu nhiên cho mảng, hãy sử dụng hàm ***shuffle()****.*

Hàm ***natsort()*** *của PHP có thể được dùng để sắp xếp mảng theo một trật tự tự nhiên hơn (chủ yếu để xử lý các con số ở dạng chuỗi được tốt hơn).*

PHP cũng có thể sắp xếp các mảng bằng cách sử dụng hàm sắp xếp do người dùng tự định nghĩa (tham khảo tài liệu về PHP để biết thêm chi tiết).

Vòng lặp for và while

Kết cấu ngôn ngữ sau cùng được đề cập là các vòng lặp. Bạn đã sử dụng vòng lặp ở các ví dụ trên, như vòng lặp **foreach** khi làm việc với mảng. Hai dạng vòng lặp kế tiếp sẽ được sử dụng là vòng lặp **for** và **while**.

Vòng lặp **while** có dạng như sau:

```
while(điều kiện){
        // khối lệnh
}
```

Nếu điều kiện của vòng lặp **while** còn đúng, phần lệnh của nó sẽ được thực hiện. Khi điều kiện sai, vòng lặp sẽ dừng lại. Nếu điều kiện không khi nào đúng, vòng lặp sẽ không bao giờ được thực hiện. Vòng lặp **while** thường được sử dụng khi lấy dữ liệu từ cơ sở dữ liệu (bạn sẽ thấy trong Chương 6 "Sử dụng PHP và MySQL").

Vòng lặp **for** có cú pháp phức tạp hơn:

```
for(biểu thức khởi tạo; điều kiện; biểu thức đóng){
        // khối lệnh
}
```

Lần đầu tiên thực hiện vòng lặp, *biểu thức khởi tạo* sẽ được thực hiện. Kế tiếp, điều kiện sẽ được kiểm tra và nếu đúng, khối lệnh trong vòng **for** sẽ được thực hiện. Sau đó, *biểu thức đóng* được thực hiện và điều kiện sẽ được kiểm tra trở lại. Quá trình này tiếp tục cho đến khi điều kiện trở thành sai. Ví dụ:

```
for($i = 1; $i <=10; $i++){
   echo $i;
}
```

Lần đầu tiên khi vòng lặp được thực hiện, giá trị của biến **$i** sẽ được thiết lập là 1. Kế tiếp, điều kiện (**$i <= 10**) sẽ được kiểm tra. Vì điều này đúng, số 1 sẽ được in ra (**echo $i**). Sau đó, biến **$i** sẽ được tăng lên một đơn vị (2), điều kiện lại được kiểm tra, câu lệnh trong vòng lặp (**echo $i**) lại được thực hiện và quá trình sẽ tiếp diễn như vậy. Kết quả của mã lệnh này là các số từ 1 đến 10 sẽ được in ra.

Thực hành sử dụng vòng lặp theo các bước sau:

1. Mở tập tin **calendar.php** (tham khảo đoạn mã 2.9) trong trình soạn thảo văn bản.
2. Xóa phần tạo các mảng **$days** và **$years**.

Bằng cách sử dụng các vòng lặp, bạn cũng tạo được hai menu thả xuống mà không phải sử dụng các mã lệnh phụ và tiêu tốn nhiều bộ nhớ như khi sử dụng mảng.

3. Viết lại vòng lặp **foreach** cho biến **$day** với vòng lặp **for**:

```
for ($day = 1; $day <= 31; $day++) {
   echo "<option value=\"$day\">$day</option>\n";
}
```

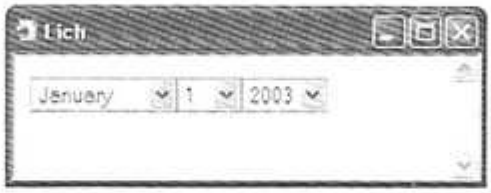
Vòng lặp **for** bắt đầu bằng việc khởi tạo biến **$day** với giá trị 1. Nó sẽ tiếp tục lặp cho đến khi biến **$day** lớn hơn 31 và **$day** sẽ tăng lên 1 sau mỗi lần lặp. Nội dung vòng lặp (được thực hiện 31 lần) là một câu lệnh **echo()**.

4. Thay vòng lặp **foreach** (ứng với biến **$year**) bằng vòng lặp **while**:

```
$year = 2003;
while ($year <= 2010) {
   echo "<option value=\"$year\">$year</option>\n";
   $year++;
}
```

Về bản chất, cấu trúc lặp này cũng giống như vòng lặp **for** ở phần trên, nhưng được viết lại với vòng lặp **while**. Biến **$year** sẽ được khởi tạo và vòng lặp sẽ được thực hiện khi giá trị của biến này vẫn còn nhỏ hơn hoặc bằng 2010. Trong vòng lặp, câu lệnh **echo()** sẽ được thực hiện và biến **$year** được tăng lên 1.

5. Lưu lại tập tin, tải nó lên máy chủ và chạy thử trong trình duyệt Web (xem hình 2-31). Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 2.14.



Hình 2-31: *Biểu mẫu vẫn thể hiện như trước đây (xem hình 2-24) nhưng được tạo ra với số lượng mảng ít hơn so với mã kịch bản trước (tham khảo mã kịch bản 2.14).*

Đoạn mã 2.14: *Các vòng lặp thường được sử dụng cùng với các biến mảng.*

```
<!DOCTYPE html PUBLIC
→ "-//W3C//DTD XHTML 1.0 Transitional//EN"
→ "http://www.w3.org/TR/2000/REC-xhtml1-20000126/DTD/xhtml1-
→ transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">
<head>
<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />
<title>Lich</title>
</head>
<body>
<?php # Doan ma 2.14 - calendar.php
$months = array (1 => 'January', 'February', 'March', 'April',
→ 'May', 'June', 'July', 'August', 'September', 'October',
→ 'November', 'December');
echo '<form action="calendar.php" method="post">';
```

```
echo '<select name="month">';
foreach ($months as $key => $value) {
   echo "<option value=\"$key\">$value</option>\n";
}
echo '</select>';
echo '<select name="day">';
for ($day = 1; $day <= 31; $day++) {
   echo "<option value=\"$day\">$day</option>\n";
}
echo '</select>';
echo '<select name="year">';
$year = 2003;
while ($year <= 2010) {
   echo "<option value=\"$year\">$year</option>\n";
   $year++;
}
echo '</select>
</form>';
?>
</body>
</html>
```

*PHP cũng có vòng lặp **do...while** với một cú pháp hơi khác (xem tài liệu về PHP để biết thêm chi tiết). Vòng lặp này luôn thực hiện ít nhất một lần.*

*Cú pháp và tính năng tương tự của hai vòng lặp **for** và **while** cho phép chúng sử dụng thay cho nhau như trong ví dụ trên. Tuy nhiên, theo kinh nghiệm của nhiều lập trình viên, vòng lặp **for** thường được sử dụng khi muốn thực hiện một điều gì đó một số lần và vòng lặp **while** thường được sử dụng khi muốn thực hiện một điều gì đó trong khi điều kiện hãy còn đúng.*

Khi sử dụng các vòng lặp, hãy cẩn thận với các tham số và điều kiện để tránh rơi vào vòng lặp vô tận (xảy ra khi điều kiện lặp không bao giờ bị sai).

CHƯƠNG 3:

TẠO WEBSITE ĐỘNG

Với những kiến thức cơ bản về PHP, bạn sẽ bắt tay vào xây dựng Website động thực sự. Khác với Website tĩnh (cơ sở ban đầu của Web), Website động rất dễ bảo trì, có tính đáp ứng cao đối với người dùng và có thể nhanh chóng thay đổi dạng thể hiện dựa trên các tình huống khác nhau.

Chương này, sẽ đề cập đến nhiều ý tưởng dùng cho việc tạo các ứng dụng Web động. Các ý tưởng này bao gồm việc kết hợp các tập tin ngoài, viết và sử dụng các hàm của riêng bạn, gửi thư điện tử, chuyển hướng trình duyệt và sử dụng hàm **date()** của PHP. Một vài ví dụ trong chương này sẽ dựa trên những gì đã thực hiện trong các chương trước và phần lớn mã lệnh của nó sẽ được sử dụng trong các chương sau.

Sử dụng các tập tin ngoài

Tại thời điểm này, các mã kịch bản trong cuốn sách chỉ bao gồm một tập tin chứa tất cả các mã lệnh. Nhưng khi phát triển các Website phức tạp hơn, bạn sẽ thấy cách thức này có nhiều hạn chế. PHP có thể dễ dàng sử dụng các tập tin ngoài, cho phép chia mã kịch bản thành các phần riêng biệt. Thông thường, các tập tin ngoài được dùng để tách mã HTML ra khỏi mã lệnh PHP hoặc để phân chia các quá trình thường được sử dụng.

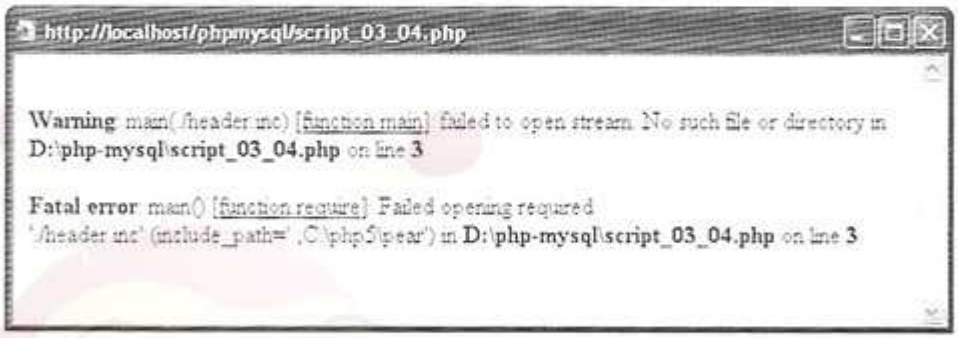
PHP có bốn hàm để sử dụng tập tin ngoài: **include()**, **include_once()**, **require()** và **require_once()**. Để sử dụng chúng, mã kịch bản cần thêm vào như sau:

```
include_once("filename.php");
require('/path/to/filename.php');
```

Các hàm này đưa toàn bộ nội dung của tập tin kèm theo vào trong mã kịch bản hiện tại (mã kịch bản có chứa lời gọi hàm) tại vị trí gọi hàm. Một thuộc tính quan trọng của các tập tin kèm theo là PHP sẽ xem các mã lệnh của tập tin này như mã lệnh HTML (nghĩa là nó sẽ được gửi trực tiếp đến trình duyệt Web), trừ khi nó chứa mã lệnh trong các thẻ PHP.

Các phiên bản trước đây của PHP có sự phân biệt giữa việc sử dụng hàm **include()** và **require()**. Bây giờ, các hàm này hoàn toàn giống nhau khi làm việc chính xác nhưng sẽ làm việc khác nhau khi chúng thực hiện không thành công. Nếu hàm **include()** không thực hiện được (không thể truy xuất tập tin kèm theo vì một lý do nào đó), thông báo lỗi sẽ được in ra trang Web (xem hình 3-1) và mã kịch bản sẽ tiếp tục chạy. Nếu hàm **require()** không thực hiện được, một thông báo lỗi cũng được in ra và mã kịch bản sẽ dừng lại (xem hình 3-2).

Hình 3-1: *Hai lời gọi hàm* `include()` *không thực hiện được.*



Hình 3-2: *Ngay khi không thực hiện được lời gọi hàm* `require()`*, một thông báo lỗi sẽ được in ra và ngưng việc thực hiện mã kịch bản.*

Cả hai hàm đều có phần mở rộng là `_once()`. Hàm có phần mở rộng này sẽ đảm bảo rằng tập tin kèm theo chỉ được đưa vào một lần trong trường hợp chúng được gọi nhiều lần.

```
require_once('filename.php');
```

Ví dụ đầu tiên sẽ sử dụng các tập tin kèm theo để phân tách mã HTML ra khỏi mã lệnh PHP. Các ví dụ còn lại trong chương này sẽ sử dụng bố cục HTML giống nhau, bạn không phải viết lại mã lệnh mỗi khi cần đến nó.

Cấu trúc site

Khi bắt đầu sử dụng nhiều tập tin trong các ứng dụng Web, cấu trúc tổng thể của site sẽ là một điều rất quan trọng. Khi thực hiện bố cục site, bạn cần quan tâm đến các vấn đề sau:

- Dễ bảo trì.
- An toàn.
- Dễ truy cập.

Việc sử dụng các tập tin ngoài để lưu giữ các thủ tục chuẩn và các thiết kế HTML sẽ cải thiện đáng kể tính dễ bảo trì của site, vì mã lệnh thường phải chỉnh sửa sẽ được đặt tại một vị trí trung tâm. Có thể tạo thư mục *includes* hoặc *templates* để chứa các tập tin này và để tách chúng khỏi mã kịch bản chính.

Nên sử dụng phần mở rộng tập tin là `.inc` cho các hồ sơ không cần bảo mật (ví dụ như các mẫu HTML) và `.php` cho các hồ sơ có tính "nhạy cảm" (như chứa các thông tin truy xuất cơ sở dữ liệu). Chương 8 "An toàn", sẽ đề cập đến các rủi ro có thể phát sinh với các tập tin ngoài và cách an toàn để sử dụng chúng.

Cuối cùng, cần cố gắng kết cấu Website sao cho người dùng dễ định hướng theo cả hai cách là nhấp vào liên kết hoặc nhập trực tiếp URL. Tránh tạo ra quá nhiều thư mục lồng vào nhau, tránh sử dụng các thư mục và tên tập tin khó nhập, chữ hoa và chữ thường lẫn lộn và không nên sử dùng các dạng dấu ngắt.

Thực hành sử dụng tập tin ngoài theo các bước sau:

1. Thiết kế một trang HTML trong trình soạn thảo văn bản hoặc trình soạn thảo WYSIWYG (xem đoạn mã 3.1 và hình 3-3).

Hình 3-3: *Trang HTML được thể hiện trong trình duyệt.*

Đầu tiên, chúng ta sẽ tạo bố cục HTML cho ứng dụng của mình, độc lập với mã lệnh PHP. Phần bố cục có sự thay đổi từ trang này sang trang khác sẽ được đánh

dấu bằng hai thẻ ghi chú <!--BẮT ĐẦU NỘI DUNG TRANG--> và <!-- KẾT THÚC NỘI DUNG TRANG-->.

2. Sao chép từ dòng đầu tiên của mã nguồn trong bố cục đến trước nội dung cụ thể của trang và dán nó vào hồ sơ mới:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
→ Transitional//EN" "http://www.w3.org/TR/2000/REC-xhtml1-
→ 20000126/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">
<head>
<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />
<title>TIEU DE TRANG</title>
</head>
<body bgcolor="#FFFFFF">
<table width="100%" border="0" cellspacing="0"
→ cellpadding="4">
<tr>
<td width="100%" bgcolor="#666666"><font
→ color="#CCCCCC"><big><b>Chao mung ban den voi website!
→ </b></big></font></td>
</tr>
<tr>
<td bgcolor="#CCCCCC"> <table width="100%" border="0"
→ cellspacing="1" cellpadding="2">
<tr>
<td align="center"><a href="index.php">Trang chu</a></td>
<td align="center"><a href="dateform.php">Lich</a></td>
<td align="center"><a
→ href="calculator.php">May tinh</a></td>
<td align="center"><a href="register.php">Dang ky</a></td>
</tr>
</table></td>
</tr>
</table>
<br />
<!-- Doan ma 3.2 header.inc -->
```

Tập tin đầu tiên sẽ chứa các thẻ HTML ban đầu (từ DOCTYPE cho đến phần bắt đầu của thân trang).

3. Thay đổi tiêu đề trang từ TIEU DE TRANG thành:

```
<title><?php echo $page_title; ?></title>
```

Tiêu đề trang (phần xuất hiện trên thanh tiêu đề của cửa sổ trình duyệt – xem hình 3-3) cần được thay đổi theo từng trang. Để thực hiện điều này, một biến sẽ được thiết lập và in ra bằng mã lệnh PHP.

4. Lưu tập tin theo tên **header.inc**. Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 3.2.

Tập tin kèm theo này có thể sử dụng phần mở rộng bất kỳ. Một số lập trình viên thích sử dụng phần mở rộng **.inc** để cho biết đây là tập tin kèm theo (xem lại phần "Cấu trúc site" và xem trong Chương 8 "An toàn" để biết thêm chi tiết).

5. Sao chép mọi thứ trong khuôn mẫu gốc, từ phần kết thúc nội dung cụ thể cho đến cuối trang và dán nó vào hồ sơ mới:

```
<!-- Doan ma 3.3 footer.inc -->
<br />
<table width="100%" border="0" cellspacing="0"
→ cellpadding="2" bgcolor="#CCCCCC">
<tr>
<td> <div align="center">&copy; Cong ty ABC.</div></td>
</tr>
</table>
</body>
</html>
```

Tập tin **footer.inc** chứa phần định dạng còn lại của thân trang Web và phần kết thúc hồ sơ HTML.

6. Lưu tập tin theo tên **footer.inc**. Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 3.3.
7. Tạo một hồ sơ PHP mới trong trình soạn thảo văn bản:

```
<?php # Doan ma 3.4 - index.php
```

Vì mã kịch bản này sẽ sử dụng các tập tin kèm theo cho hầu hết các định dạng HTML của nó, nên bạn có thể bắt đầu và kết thúc bằng thẻ PHP thay vì HTML.

8. Thiết lập biến **$page_title** kèm theo tập tin **header.inc**:

```
$page_title = 'Welcome!';
include ('./header.inc');
```

Biến **$page_title** cho phép thiết lập một tiêu đề mới cho mỗi trang Web có sử dụng hệ thống khuôn mẫu này. Vì biến **$page_title** được thiết lập trước khi đưa vào tập tin **header.inc** nên mã lệnh trong tập tin này sẽ truy xuất được biến đó.

9. Đóng thẻ PHP và sao chép phần nội dung của trang từ khuôn mẫu:

```
?>
<table width="90%" border="0" cellspacing="2"
→ cellpadding="4" align="center">
<tr bgcolor="#333333">
<td>
<table width="100%" border="0" cellspacing="0"
→ cellpadding="4">
<tr>
<td bgcolor="#FFFFFF"> ! </td>
```

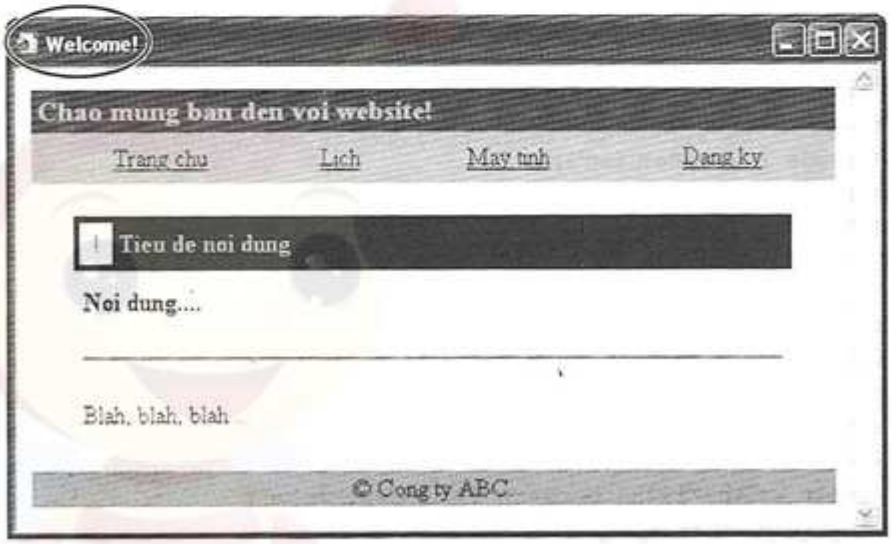
```
<td width="100%"> <font color="#CCCCCC"> <b>Tieu de noi dung
→ </b></font></td>
</tr>
</table></td>
</tr>
</table>
<table width="90%" border="0" cellspacing="4"
→ cellpadding="4" align="center">
<tr>
<td width="70%" valign="top"> <p><b>Noi dung...</b></p>
<hr />
<p>Blah, blah, blah ...</p>
</td>
</tr>
</table>
```

Thông tin này có thể được truyền đến trình duyệt Web bằng lệnh **echo()**, nhưng vì không có nội dung động nào trong phần này nên việc đặt chúng ngoài phạm vi của thẻ PHP sẽ dễ dàng và thuận tiện hơn.

10. Tạo phần PHP cuối cùng và kèm theo tập tin **footer.inc**:

```
<?php
include ('./footer.inc');
?>
```

11. Lưu tập tin theo tên **index.php**, tải nó lên máy chủ Web cùng với hai tập tin **header.inc** và **footer.inc**, chạy thử trong trình duyệt Web (xem hình 3-4). Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 3.4.



Hình 3-4: *Giờ đây, bố cục như trong hình 3-3 được tạo bằng cách sử dụng các tập tin ngoài.*

12. Xem mã nguồn của trang Web, nếu cần (xem hình 3-5).

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/2000/REC-xhtml1-20000126/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<head>
<meta http-equiv="content-type" content="text/html; charset=iso-8859-1" />
<title>Welcome!</title>
</head>
<body bgcolor="#FFFFFF">
<table width="100%" border="0" cellspacing="0" cellpadding="4">
<tr>
<td width="100%" bgcolor="#666666"><font color="#CCCCCC"><big><b>Chao mung ban den voi
website!</b></big></font></td>
</tr>
<tr>
<td bgcolor="#CCCCCC"> <table width="100%" border="0" cellspacing="1" cellpadding="2">
<tr>
<td align="center"><a href="index.php">Trang chu</a></td>
<td align="center"><a href="dateform.php">Lich</a></td>
<td align="center"><a href="calculator.php">May tinh</a></td>
<td align="center"><a href="register.php">Dang Ky</a></td>
</tr>
</table></td>
</tr>
</table>
<hr />
<!-- Doan ma 3.2 header.inc -->
<!-- BAT DAU NOI DUNG TRANG. --><table width="90%" border="0" cellspacing="2" cellpadding="4"
align="center">
<tr bgcolor="#333333">
<td>
<table width="100%" border="0" cellspacing="0" cellpadding="4">
<tr>
<td bgcolor="#FFFFFF">  </td>
<td width="100%"> <font color="#CCCCCC"> <b>Tieu de noi dung</b></font></td>
</tr>
</table></td>
</tr>
</table>
<table width="90%" border="0" cellspacing="4" cellpadding="4" align="center">
<tr>
<td width="70%" valign="top"> <p><b>Noi dung....</b></p>
<hr />
<p>Blah, blah, blah ...</p>
</td>
</tr>
</table>
<!-- Doan ma 3.3 footer.inc -->
<hr />
<table width="100%" border="0" cellspacing="0" cellpadding="2" bgcolor="#CCCCCC">
<tr>
<td> <div align="center">&copy; Cong ty ABC.</div></td>
</tr>
</table>
</body>
</html>
```

Hình 3-5: *Mã nguồn HTML (của trang Web được tạo) giống như mã lệnh trong khuôn mẫu gốc (tham khảo đoạn mã 3.1).*

Đoạn mã 3.1: *Khuôn mẫu HTML dùng cho các trang Web trong chương này.*

```
<!DOCTYPE html PUBLIC
→ "-//W3C//DTD XHTML 1.0 Transitional//EN"
→ "http://www.w3.org/TR/2000/REC-xhtml1-20000126/DTD/xhtml1-
→transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
```

```
→ lang="en">
<head>
<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />
<title>TIEU DE TRANG</title>
</head>
<body bgcolor="#FFFFFF">
<table width="100%" border="0" cellspacing="0"
→ cellpadding="4">
<tr>
<td width="100%" bgcolor="#666666"><font color="#CCCCCC">
→<big><b>Chao mung ban den voi website!</b></big></font></td>
</tr>
<tr>
<td bgcolor="#CCCCCC"> <table width="100%" border="0"
→ cellspacing="1" cellpadding="2">
<tr>
<td align="center"><a href="index.php">Trang chu</a></td>
<td align="center"><a href="dateform.php">Lich</a></td>
<td align="center"><a href="calculator.php">May tinh</a></td>
<td align="center"><a href="register.php">Dang ky</a></td>
</tr>
</table></td>
</tr>
</table>
<br />
<!-BAT DAU NOI DUNG TRANG. -->
<table width="90%" border="0" cellspacing="2" cellpadding="4"
→ align="center">
<tr bgcolor="#333333">
<td>
<table width="100%" border="0" cellspacing="0"
→ cellpadding="4">
<tr>
<td bgcolor="#FFFFFF"> ! </td>
<td width="100%"> <font color="#CCCCCC">
→ <b>Tieu de noi dung</b></font></td>
```

```
</tr>
</table></td>
</tr>
</table>
<table width="90%" border="0" cellspacing="4" cellpadding="4"
→ align="center">
<tr>
<td width="70%" valign="top"> <p><b>Noi dung....</b></p>
<hr />
<p>Blah, blah, blah ...</p>
</td>
</tr>
</table>
<!-- KET THUC NOI DUNG TRANG. -->
<br />
<table width="100%" border="0" cellspacing="0" cellpadding="2"
→ bgcolor="#CCCCCC">
<tr>
<td> <div align="center">&copy; Cong ty ABC.</div></td>
</tr>
</table>
</body>
</html>
```

Đoạn mã 3.2: *Phần bắt đầu của mỗi trang Web sẽ được đặt trong tập tin* ***header.inc***

```
<!DOCTYPE html PUBLIC
→ "-//W3C//DTD XHTML 1.0 Transitional//EN"
→ "http://www.w3.org/TR/2000/REC-xhtml1-20000126/DTD/xhtml1-
→transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">
<head>
<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />
<title><?php echo $page_title; ?></title>
</head>
```

```
<body bgcolor="#FFFFFF">
<table width="100%" border="0" cellspacing="0"
→ cellpadding="4">
<tr>
<td width="100%" bgcolor="#666666"><font
→ color="#CCCCCC"><big><b>Chao mung ban den voi website!
→</b></big></font></td>
</tr>
<tr>
<td bgcolor="#CCCCCC"> <table width="100%" border="0"
→ cellspacing="1" cellpadding="2">
<tr>
<td align="center"><a href="index.php">Trang chu</a></td>
<td align="center"><a href="dateform.php">Lich</a></td>
<td align="center"><a href="calculator.php">May tinh</a></td>
<td align="center"><a href="register.php">Dang ky</a></td>
</tr>
</table></td>
</tr>
</table>
<br />
<!-- Doan ma 3.2 header.inc -->
<!-- BAT DAU NOI DUNG TRANG. -->
```

Doạn mã 3.3: *Phần kết thúc của trang sẽ được lưu trong tập tin* ***footer.inc***

```
<!-- Doan ma 3.3 footer.inc -->
<br />
<table width="100%" border="0" cellspacing="0" cellpadding="2"
→ bgcolor="#CCCCCC">
<tr>
<td> <div align="center">&copy; Cong ty ABC.</div></td>
</tr>
</table>
</body>
</html>
```

Đoạn mã 3.4: *Mã kịch bản này tạo trang Web bằng cách sử dụng khuôn mẫu chứa trong các tập tin ngoài.*

```
<?php # Doan ma 3.4 - index.php
$page_title = 'Chao mung!';
include ('./header.inc');
?>
<table width="90%" border="0" cellspacing="2" cellpadding="4"
→ align="center">
<tr bgcolor="#333333">
<td>
<table width="100%" border="0" cellspacing="0" cellpadding="4">
<tr>
<td bgcolor="#FFFFFF"> ! </td>
<td width="100%"> <font color="#CCCCCC"> <b>Tieu de noi dung
→</b></font></td>
</tr>
</table></td>
</tr>
</table>
<table width="90%" border="0" cellspacing="4" cellpadding="4"
→ align="center">
<tr>
<td width="70%" valign="top"> <p><b>Noi dung....</b></p>
<hr />
<p>Blah, blah, blah ...</p>
</td>
</tr>
</table>
<?php
include ('./footer.inc');
?>
```

Trong tập tin **php.ini**, *cần điều chỉnh thông số* **include_path** *để báo cho PHP biết thư mục chứa các tập tin kèm theo.*

Như sẽ giới thiệu trong Chương 6 "Sử dụng PHP và MySQL", các tập tin kèm theo có chứa các thông tin "nhạy cảm" (như việc truy xuất cơ sở dữ liệu) cần được đặt ngoài thư mục dành cho Web trên máy chủ.

*Nên sử dụng cú pháp **./filename** khi tham chiếu đến tập tin trong cùng thư mục với tập tin cha (tập tin chứa mã lệnh gọi tập tin kèm theo). Tập tin nằm trong thư mục cấp cao hơn tập tin cha sẽ sử dụng đường dẫn **../filename** và tập tin trong thư mục bên dưới tập tin cha sẽ được sử dụng đường dẫn **directory/filename**.*

Có thể kèm theo các tập tin với đường dẫn tương đối hoặc đường dẫn tuyệt đối. Ví dụ:

```
include(/path/to/filename);
```

*Vì hàm **require()** có tác động mạnh hơn khi nó không thực hiện được, bạn nên sử dụng hàm này để đưa vào các tập tin quan trọng (chẳng hạn như các tập tin chứa thông tin truy xuất cơ sở dữ liệu) và sử dụng **include()** cho các tập tin ít quan trọng hơn.*

*Có thể tắt các thông báo lỗi được tạo ra khi hàm **include()** không thực hiện được bằng cách bổ sung tiền tố **@** vào trước lời gọi hàm:*

```
@include('filename');
```

Tạo và gọi các hàm của riêng bạn

Như chúng ta đã biết, PHP có rất nhiều hàm cài sẵn để đáp ứng phần lớn các yêu cầu. Tuy nhiên, nó vẫn cho phép bạn định nghĩa và sử dụng các hàm của riêng mình. Cú pháp để tạo ra một hàm là:

```
function function_name(){
        // mã lệnh của hàm
}
```

Tên của hàm (**function_name**) có thể là một tổ hợp bất kỳ gồm chữ cái, con số và dấu gạch dưới, nhưng phải bắt đầu bằng chữ cái hoặc dấu gạch dưới. Trong PHP, như đã giới thiệu trong chương đầu tiên, tên hàm là không phân biệt dạng chữ (khác với tên biến), nên hàm có thể được gọi bằng cách dùng **function_name()**, **FUNCTION_NAME()**, **Function_Name()**... đều được.

Phía trong hàm có thể thực hiện được mọi mã lệnh, từ việc phát sinh HTML cho đến thực hiện các tính toán. Chương này, sẽ đề cập đến cả hai điều này.

Thực hành tạo hàm của riêng bạn theo các bước sau:

1. Tạo một hồ sơ PHP mới trong trình soạn thảo:

```
<?php # Doan ma 3.5 - dateform.php

$page_title = 'Lich';

include ('./header.inc');
```

Trang này sẽ sử dụng khuôn mẫu HTML giống như ví dụ trước (**index.php**, tham khảo đoạn mã 3.4).

2. Bắt đầu bằng việc định nghĩa một hàm mới:

```
function make_calendar_pulldown() {
```

Hàm được viết ở đây sẽ tạo ra các menu thả xuống trên biểu mẫu, dùng cho tháng, ngày và năm như trong mã kịch bản **calendar.php** (tham khảo lại đoạn mã 2.14 ở Chương 2). Tên hàm sử dụng trong mã kịch bản này cũng tương đối rõ nghĩa.

3. Tạo các menu thả xuống:

```
$months = array (1 => 'January', 'February', 'March',
→ 'April', 'May', 'June', 'July', 'August', 'September',
→ 'October', 'November', 'December');

echo '<select name="month">';

foreach ($months as $key => $value) {

   echo "<option value=\"$key\">$value</option>\n";

}

echo '</select>

<select name="day">';

for ($day = 1; $day <= 31; $day++) {

   echo "<option value=\"$day\">$day</option>\n";

}

echo '</select>

<select name="year">';

$year = 2003;

while ($year <= 2010) {

   echo "<option value=\"$year\">$year</option>\n";

   $year++;

}

echo '</select>';
```

Mã lệnh trên cũng giống như mã lệnh gốc, nhưng chỉ khác là nó được đặt trong phạm vi của một hàm.

4. Đóng phần định nghĩa hàm:

```
} // kết thúc hàm make_calendar_pulldown
```

Trong các mã lệnh phức tạp, nên đặt một ghi chú ở cuối hàm để biết nó bắt đầu và kết thúc ở đâu.

5. Tạo một biểu mẫu và gọi hàm:

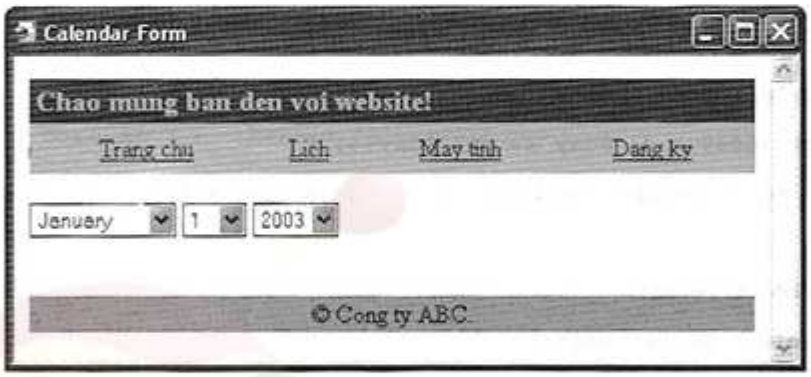
```
echo '<form action="dateform.php" method="post">';
make_calendar_pulldown();
echo '</form>';
```

Mã lệnh này sẽ tạo ra các thẻ cho biểu mẫu và gọi hàm **make_calendar_pulldown()** với kết quả tạo ra là mã lệnh cho ba menu thả xuống.

6. Hoàn tất mã kịch bản PHP bằng cách kèm theo tập tin **footer.inc**:

```
include ('./footer.inc');
?>
```

7. Lưu tập tin theo tên **dateform.php**, tải nó lên máy chủ (trong cùng thư mục với tập tin **header.inc**, **footer.inc** và **index.php**) và thử trong trình duyệt (xem hình 3-6). Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 3.5.



Hình 3-6: *Các menu thả xuống được tạo bởi một hàm trong khuôn mẫu HTML.*

Đoạn mã 3.5: *Hàm này rất hữu dụng cho việc tạo một loạt menu thả xuống.*

```
<?php # Doan ma 3.5 - dateform.php
$page_title = 'Lich';
include ('./header.inc');
function make_calendar_pulldown() {
   $months = array (1 => 'January', 'February', 'March',
   → 'April', 'May', 'June', 'July', 'August', 'September',
   → 'October', 'November', 'December');
```

```
    echo '<select name="month">';
    foreach ($months as $key => $value) {
        echo "<option value=\"$key\">$value</option>\n";
    }
    echo '</select>
    <select name="day">';
    for ($day = 1; $day <= 31; $day++) {
        echo "<option value=\"$day\">$day</option>\n";
    }
    echo '</select>
    <select name="year">';
    $year = 2003;
    while ($year <= 2010) {
        echo "<option value=\"$year\">$year</option>\n";
        $year++;
    }
    echo '</select>';
}
echo '<form action="dateform.php" method="post">';
make_calendar_pulldown();
echo '</form>';
include ('./footer.inc');
?>
```

Trong PHP, bạn có thể định nghĩa một hàm (ví dụ như ***read()****) và gọi nó giống như một biến.*

```
$var = 'read';
$var(); // gọi hàm read()
```

Vì phải tốn bộ nhớ khi mã kịch bản PHP sử dụng các hàm do người dùng định nghĩa, nên bạn cần lưu ý khi nào thì sử dụng chúng và khi nào thì kèm theo chúng trong một mã kịch bản.

Tạo một hàm nhận vào các tham số

Cũng giống như các hàm có sẵn của PHP, các hàm do bạn tạo ra cũng có thể nhận các tham số (còn gọi là đối số). Ví dụ, hàm **print()** nhận vào những gì bạn muốn gửi tới trình duyệt làm đối số và hàm **strlen()** nhận vào một chuỗi để xác định số ký tự trong chuỗi.

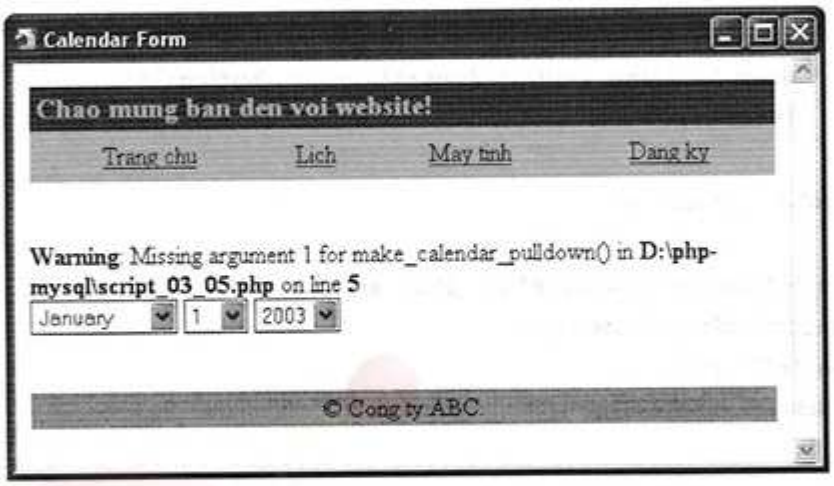
Một hàm có thể có số lượng tham số tùy ý, tuy nhiên thứ tự của chúng rất quan trọng. Để sử dụng các tên biến cho các tham số:

```
function function_name($arg1, $arg2){
        // mã lệnh của hàm
}
```

Để gọi một hàm:

```
function_name('giá trị 1', 'giá trị 2');
function_name('giá trị', $var);
```

Với các hàm trong PHP, việc truyền số tham số sai sẽ phát sinh lỗi (xem hình 3-7). Đồng thời, tên các biến sử dụng cho các tham số sẽ không có liên quan gì đến phần còn lại của mã kịch bản (tìm hiểu thêm ở phần *"Phạm vi của biến"* trong chương này), bạn nên sử dụng tên biến hợp lệ và có nghĩa.



Hình 3-7: *Việc gửi sai số tham số sẽ dẫn đến phát sinh lỗi.*

Để minh họa điều này, ví dụ về máy tính (đã thực hiện trong Chương 2 "Lập trình PHP") sẽ được viết lại dưới dạng một hàm (tham khảo lại đoạn mã 2.6 và 2.8 ở Chương 2).

Thực hành viết các hàm có nhận vào các tham số theo các bước sau:

1. Tạo một hồ sơ PHP mới trong trình soạn thảo văn bản:

```
<?php # Doan ma 3.6 - calculator.php
$page_title = 'May tinh';
include ('./header.inc');
```

2. Thoát ra khỏi mã lệnh PHP và tạo biểu mẫu HTML:

```
?>
<form action="handle_calculator.php" method="post">
<select name="quantity">
<option value="">Chon so luong:</option>
<option value="1">1</option>
<option value="2">2</option>
<option value="3">3</option>
<option value="4">4</option>
<option value="5">5</option>
<option value="6">6</option>
</select>
<div align="left"><input type="submit" name="submit"
→ value="Total!" /></div>
<input type="hidden" name="price" value="19.95" />
<input type="hidden" name="taxrate" value=".05" />
</form>
```

Biểu mẫu HTML đơn giản này cho phép người dùng chọn số lượng để tính tổng giá trị đơn hàng.

3. Hoàn tất trang bằng cách kèm theo tập tin **footer.inc**:

```
<?php
include ('./footer.inc');
?>
```

4. Lưu tập tin theo tên **calculator.php**. Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 3.6.

5. Tạo mã kịch bản PHP khác trong trình soạn thảo văn bản:

```
<?php # Doan ma 3.7 - handle_calculator.php
$page_title = 'May tinh';
include ('./header.inc');
```

6. Định nghĩa hàm **calculate_total()**:

```
function calculate_total ($quantity, $price, $taxrate) {
   $total = ($quantity * $price) * ($taxrate + 1);
   $total = number_format ($total, 2);
   echo "<p>Ban mua <b>$quantity</b> mat hang voi gia
   → <b>\$$price</b>. Cong voi thue, tong so tien la
   → <b>\$$total</b>.</p>\n";
}
```

Hàm này thực hiện các tính toán giống như ví dụ ở Chương 2 và in ra kết quả. Nó nhận vào ba tham số: lượng đặt hàng, giá và thuế suất.

7. Xác lập điều kiện chính của trang:

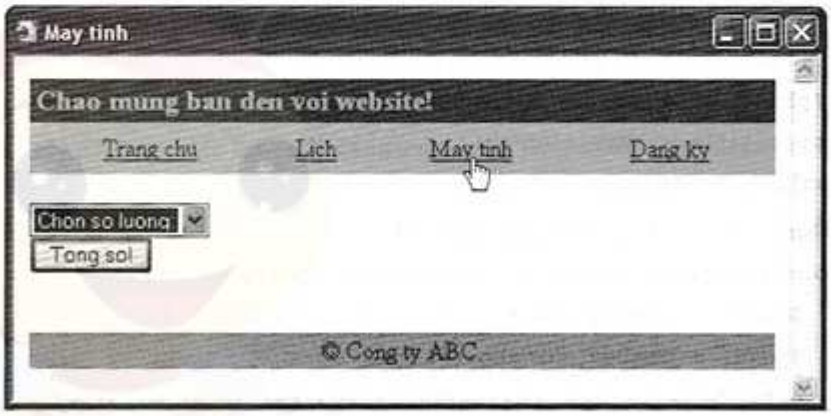
```
if (is_numeric($_POST['quantity'])) {

   calculate_total ($_POST['quantity'], $_POST['price'],
   → $_POST['taxrate']);

} else {

   echo '<p><b>Xin nhap vao so luong can mua!</b></p>';

}
```

Điều kiện này kiểm tra sự hợp lệ của số lượng nhận vào. Nếu đúng, hàm **calculate_total()** sẽ được gọi để xác định và in ra trị giá của đơn hàng. Nếu sai, thông báo lỗi sẽ được gửi đến trình duyệt.

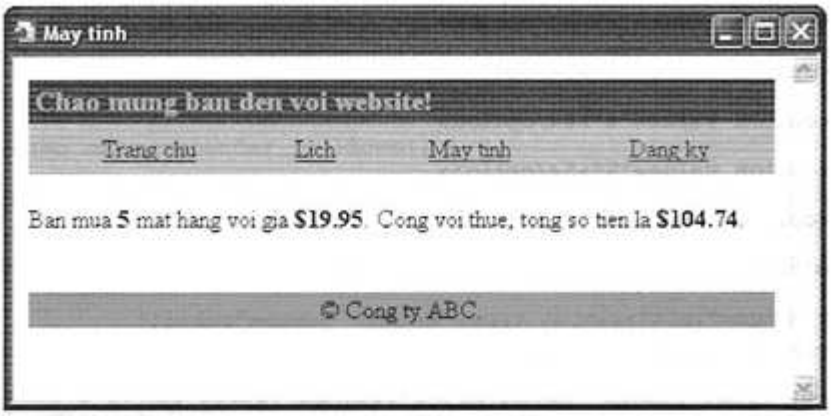
8. Hoàn tất trang PHP:

```
include ('./footer.inc');

?>
```

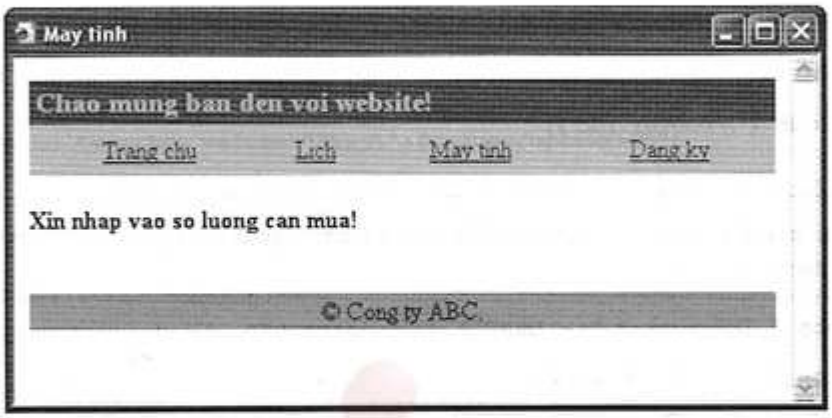
9. Lưu tập tin theo tên **handle_calculator.php**, tải nó lên máy chủ Web cùng với tập tin **calculator.php** (đặt chúng trong cùng thư mục với các tập tin khác mà bạn tạo ra trong chương này) rồi thử trong trình duyệt Web (xem hình 3-8, 3-9 và 3-10). Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 3.7.



Hình 3-8: *Biểu mẫu HTML trong Web mẫu.*



Hình 3-9: *Các tính toán được thực hiện bởi một hàm và in ra kết quả.*



Hình 3-10: *Nếu số lượng đưa vào không hợp lệ, một thông báo lỗi sẽ được in ra.*

Đoạn mã 3.6: *Trang* ***calculator.php*** *được phát triển từ một ví dụ trong Chương 2 "Lập trình PHP".*

```
<?php # Doan ma 3.6 – calculator.php
$page_title = 'May tinh';
include ('./header.inc');
?>
<form action="handle_calculator.php" method="post">
<select name="quantity">
   <option value="">Chon so luong:</option>
   <option value="1">1</option>
```

```
    <option value="2">2</option>
    <option value="3">3</option>
    <option value="4">4</option>
    <option value="5">5</option>
    <option value="6">6</option>
</select>
<div align="left"><input type="submit" name="submit"
→ value="Tong so!" /></div>
<input type="hidden" name="price" value="19.95" />
<input type="hidden" name="taxrate" value=".05" />
</form>
<?php
include ('./footer.inc');
?>
```

Đoạn mã 3.7: *Mã kịch bản* ***handle_calculator.php*** *sử dụng một hàm để thực hiện các tính toán.*

```
<?php # Doan ma 3.7 - handle_calculator.php
$page_title = 'May tinh';
include ('./header.inc');
function calculate_total ($quantity, $price, $taxrate) {
    $total = ($quantity * $price) * ($taxrate + 1);
    $total = number_format ($total, 2);
    echo "<p>Ban mua <b>$quantity</b> mat hang voi gia
    → <b>\$$price</b>. Cong voi thue, tong so tien la
    → <b>\$$total</b>.</p>\n";
}
if (is_numeric($_POST['quantity'])) {
    calculate_total ($_POST['quantity'], $_POST['price'],
    → $_POST['taxrate']);
} else {
    echo '<p><b>Xin nhap vao so luong can mua!</b></p>';
```

```
}
include ('./footer.inc');
?>
```

PHP cho phép định nghĩa một hàm không có tham số nhưng khi gọi vẫn có thể truyền tham số cho nó.

Một ví dụ khác về hàm có nhận tham số là hàm ***make_calendar_pulldown()****, nó được viết để nhận vào một tập hợp tháng, ngày và năm. Các tập hợp này sẽ được chọn lựa trong biểu mẫu ở phần sau của chương này.*

Thiết lập các tham số mặc định

Một dạng khác trong việc định nghĩa hàm là thiết lập trước giá trị của một tham số:

```
function function_name($arg1, $arg2='Bob'){
        // mã lệnh của hàm
}
```

Kết quả cuối cùng của việc thiết lập giá trị tham số mặc định là tham số này sẽ trở thành một tùy chọn khi gọi hàm. Nếu giá trị được truyền vào khi gọi hàm, giá trị đó sẽ được sử dụng. Nếu không có giá trị truyền vào, giá trị mặc định sẽ được sử dụng.

Bạn có thể thiết lập giá trị mặc định cho nhiều tham số theo ý mình, miễn là các tham số này nằm phía sau danh sách tham số trong phần định nghĩa hàm. Nói cách khác, các tham số bắt buộc phải xuất hiện trước.

Với hàm ví dụ được định nghĩa ở trên, các lời gọi hàm sau đây đều hợp lệ:

```
function_name($var1, $var2);
function_name('Robert');
function_name('John', 'Ann');
```

Tuy nhiên, lời gọi hàm `function_name()` sẽ không hoạt động và không có cách nào để truyền tham số `$arg2` mà không truyền vào tham số `$arg1`.

Thực hành thiết lập giá trị mặc định cho các tham số theo các bước sau:

1. Mở tập tin `handle_calculator.php` (tham khảo đoạn mã 3.7) trong trình soạn thảo văn bản.
2. Thay đổi dòng định nghĩa hàm để chỉ có phần số lượng là bắt buộc:

```
function calculate_total ($quantity, $price = 19.95,
→ $taxrate = .05) {
```

Bây giờ, các biến **$price** và **$taxrate** (được gửi dưới dạng các trường ẩn trong biểu mẫu) đã được mã hóa cố định trong hàm.

3. Thay đổi lời gọi hàm thành **calculate_total()** để chỉ có tham số số lượng được thiết lập.

```
calculate_total ($_POST['quantity']);
```

Vì các giá trị về giá và thuế suất được gửi từ biểu mẫu có cùng giá trị mặc định khi định nghĩa hàm, nên không phải truyền chúng vào khi gọi hàm.

4. Gọi lại hàm với giá và thuế suất khác:

```
calculate_total ($_POST['quantity'], 29.99, .0875);
```

Mặc dù giá trị mặc định đã được thiết lập, việc gửi vào các giá trị mới sẽ ghi đè lên chúng (dòng lệnh trên có mục đích minh họa cho điều này).

5. Lưu lại tập tin, tải nó lên máy chủ Web và thử trong trình duyệt (xem hình 3-11). Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 3.8.

Hình 3-11: *Hàm **calculate_total()** được gọi hai lần với số tham số khác nhau.*

Đoạn mã 3.8: *Hàm **calculate_total()** giờ đây xem như tham số giá và thuế suất đã được thiết lập trừ khi chúng được xác định khi gọi hàm.*

```
<?php # Doan ma 3.8 - handle_calculator.php

$page_title = 'Calculator';

include ('./header.inc');

function calculate_total ($quantity, $price = 19.95,
→ $taxrate = .05) {
```

```
    $total = ($quantity * $price) * ($taxrate + 1);
    $total = number_format ($total, 2);
    echo "<p>Ban mua <b>$quantity</b> mat hang voi gia
    → <b>\$$price</b>. Cong voi thue, tong so tien la
    → <b>\$$total</b>.</p>\n";
}
if (is_numeric($_POST['quantity'])) {
    calculate_total ($_POST['quantity']);
    calculate_total ($_POST['quantity'], 29.99, .0875);
} else {
    echo '<p><b>Xin nhap vao so luong can mua!</b></p>';
}
include ('./footer.inc');
?>
```

*Để gọi hàm với một tham số nào đó không có giá trị, bạn sử dụng chuỗi rỗng (''), **NULL** hoặc **FALSE***

Trong tài liệu PHP, các dấu ngoặc vuông ([]) sẽ cho biết các tham số tùy chọn của hàm.

Trả lại các giá trị từ một hàm

Thuộc tính cuối cùng của một hàm có thể sử dụng được là việc trả lại các giá trị. Một số hàm (không phải tất cả) cho phép thực hiện điều này. Ví dụ, hàm **print()** sẽ trả lại giá trị là 1 hoặc 0, cho biết kết quả thực hiện của nó, trong khi hàm **echo()** lại không trả lại giá trị. Hàm **strlen()** trả lại giá trị là số ký tự có trong một chuỗi.

Sử dụng câu lệnh **return** để hàm trả lại một giá trị:

```
function function_name($arg1, $arg2='Bob'){
        // mã lệnh của hàm
    return $somevalue;
}
```

Hàm có thể trả về một giá trị (một chuỗi hoặc một số) hoặc một biến mà giá trị của nó được tạo ra bởi mã lệnh của hàm. Khi gọi các hàm này, bạn nên gán giá trị trả về cho một biến.

```
$var = function_name();
```

Thực hành viết một hàm có trả lại giá trị theo các bước sau:

1. Mở tập tin **handle_calculator.php** (tham khảo đoạn mã 3.8) trong trình soạn thảo văn bản.

2. Thay đổi phần định nghĩa hàm:

```
function calculate_total ($quantity, $price = 19.95,
→ $taxrate = .05) {

   $total = ($quantity * $price) * ($taxrate + 1);

   return number_format ($total, 2);

}
```

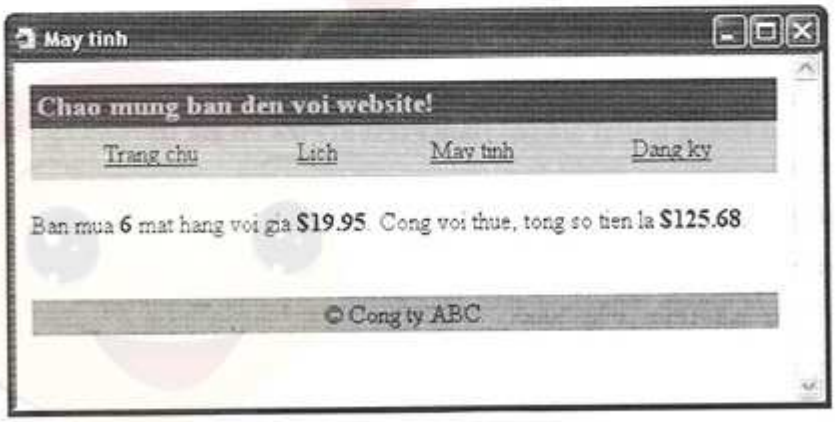
Phiên bản này của hàm chỉ trả lại kết quả tính toán, không có mã lệnh HTML hoặc không gửi bất kỳ cái gì cho trình duyệt Web.

3. Thay đổi các dòng gọi hàm:

```
$total = calculate_total ($_POST['quantity']);

echo "<p>Ban mua <b>{$_POST['quantity']}</b> mat hang voi
→ gia <b>\${$_POST['price']}</b>. Cong voi thue, tong so
→ tien la <b>\$$total</b>.</p>\n";
```

Bây giờ, biến **$total** đã được gán bởi giá trị trả lại của hàm **calculate_total()**, nên câu lệnh **echo()** đã trở lại dạng trước đây (sử dụng các biến siêu toàn cục).

4. Lưu lại tập tin, tải nó lên máy chủ Web và thử trong trình duyệt (xem hình 3-12). Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 3.9.



Hình 3-12: *Mục đích chính của mã kịch bản có thể đạt được theo nhiều cách khác nhau mà không ảnh hưởng đến kết quả cuối cùng.*

Đoạn mã 3.9: *Hàm* ***handle_calculator.php*** *nhận vào ba tham số và trả lại kết quả tính toán.*

```
<?php # Doan ma 3.9 - handle_calculator.php

$page_title = 'Calculator';

include ('./header.inc');

function calculate_total ($quantity, $price = 19.95,
→ $taxrate = .05) {

   $total = ($quantity * $price) * ($taxrate + 1);

   return number_format ($total, 2);

}

if (is_numeric($_POST['quantity'])) {

   $total = calculate_total ($_POST['quantity']);

   echo "<p>Ban mua <b>{$_POST['quantity']}</b> mat hang voi
   → gia <b>\${$_POST['price']}</b>. Cong voi thue, tong so
   → tien la <b>\$$total</b>.</p>\n";

} else {

   echo '<p><b>Xin nhap vao so luong can mua!</b></p>';

}

include ('./footer.inc');

?>
```

Một hàm có thể có nhiều câu lệnh ***return*** *(như trong câu lệnh* ***switch*** *hoặc điều kiện):*

```
function function_name(){
        if(điềukiện){
                return 'giá trị 1';
} else {
                return 'giá trị 2';
}
}
```

Để một hàm trả lại nhiều giá trị, có thể sử dụng mảng:

```
function function_name(){
```

```
    return array('giá trị 1', 'giá trị 2');
}
list($var1, $var2) = function_name();
```

Cấu trúc mã lệnh

Bạn đã biết cách viết và truy xuất các hàm của riêng mình, điều này đưa đến một vấn đề: đó là kết cấu mã lệnh. Trong PHP 4, các hàm tự định nghĩa có thể đặt vị trí bất kỳ trong mã kịch bản và có thể truy xuất được (trong các phiên bản trước, bạn phải định nghĩa một hàm trước khi gọi nó).

Tuy nhiên, bạn cũng nên đặt phần định nghĩa hàm ở phần đầu của mã kịch bản (có sử dụng hàm). Nếu hàm được sử dụng bởi nhiều mã kịch bản trong cùng một ứng dụng Web, hãy đặt hàm trong một tập tin ngoài để có thể kèm theo tập tin này ở phần đầu mã kịch bản (có sử dụng hàm).

Việc sử dụng các hàm tự định nghĩa và đặt chúng tại một nơi dễ truy xuất sẽ làm cho việc chỉnh sửa dễ dàng hơn, đặc biệt khi mã kịch bản và ứng dụng của bạn phát triển về mặt kích thước.

Phạm vi của biến

Phạm vi của biến là một khái niệm rất quan trọng. Mỗi biến trong PHP đều có một phạm vi áp dụng cho chúng. Phạm vi này cho biết khi nào thì một biến có thể truy xuất được. Một quy tắc chung là mọi biến đều có phạm vi là trang nơi nó được định nghĩa. Nói cách khác, khi biến đã được định nghĩa (thiết lập giá trị cho biến), có thể tham chiếu nó ở vị trí bất kỳ trong mã kịch bản. Giả sử cấu hình PHP của bạn đã kích hoạt **register_globals**, có thể được truy xuất các giá trị biểu mẫu ở vị trí bất kỳ trong trang xử lý biểu mẫu.

Vì các tập tin kèm theo làm việc giống như thành phần của mã kịch bản gốc, các biến được định nghĩa trước dòng kèm theo tập tin đều có hiệu lực với tập tin này. Hơn thế nữa, các biến được định nghĩa trong tập tin cũng có hiệu lực với *mã kịch bản cha* ngay sau dòng kèm theo tập tin.

Tất cả điều trên sẽ trở nên phức tạp hơn khi bạn sử dụng các hàm của riêng mình. Các hàm này có phạm vi của riêng chúng, có nghĩa là các biến được sử dụng trong bản thân hàm sẽ không thể truy xuất được từ bên ngoài và các biến được định nghĩa ngoài hàm không có hiệu lực trong phạm vi hàm. Vì lý do này, một biến phía trong một hàm có thể trùng tên với một biến được định nghĩa ngoài hàm và chúng được xem như những biến khác nhau với giá trị khác nhau (ví dụ như biến **$total** trong đoạn mã 3.9).

Thay đổi phạm vi của biến trong một hàm bằng sử dụng từ khóa **global**:

```
function function_name(){
   global $var;
}
$var = 'giá trị';
```

Trong ví dụ này, biến **$var** phía trong hàm cũng chính là biến **$var** phía ngoài hàm. Điều này có nghĩa là biến **$var** của hàm giờ đây cũng có một giá trị (**'giá trị'**) và nếu giá trị đó thay đổi phía trong hàm, giá trị của biến **$var** phía ngoài hàm cũng sẽ thay đổi.

Một tùy chọn khác để đánh lừa phạm vi của biến là sử dụng các biến siêu toàn cục **$_GET**, **$_POST**, **$_SESSION**... Các biến này đều có thể truy xuất một cách tự động trong các hàm (nên chúng được gọi là siêu toàn cục).

Thực hành sử dụng các biến toàn cục theo các bước sau:

1. Mở tập tin **handle_calculator.php** (tham khảo đoạn mã 3.9) trong trình soạn thảo văn bản.
2. Thay đổi định nghĩa hàm:

```
function calculate_total () {
   global $total;
   $total = ($_POST['quantity'] * $_POST['price']) *
   → ($_POST['taxrate'] + 1);
   $total =  number_format ($total, 2);
}
```

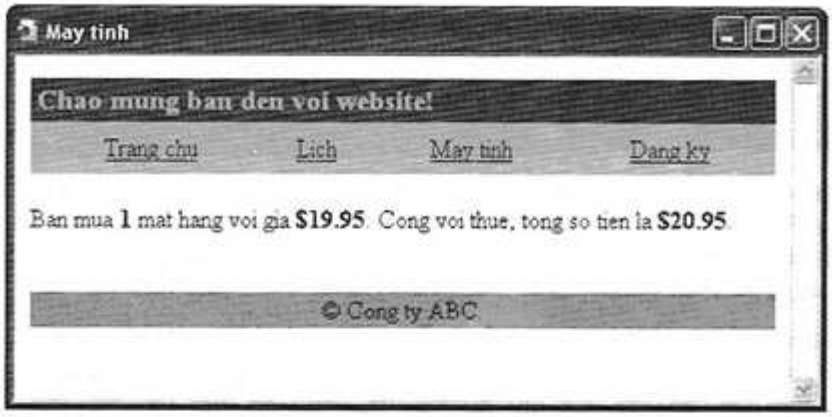
Vì đang sử dụng các biến siêu toàn cục nên bạn có thể viết lại hàm để sử dụng các biến này, thay vì phải truyền giá trị vào cho hàm. Một cách tương tự, thay vì trả lại giá trị của biến **$total**, có thể làm cho biến này trở thành biến có phạm vi toàn cục.

3. Thay đổi lời gọi hàm:

```
$total = 0;
calculate_total ();
```

Mặc dù đã sử dụng biến **$_POST**, nhưng bạn vẫn phải gọi hàm để thực hiện các tính toán và khởi tạo biến **$total** với giá trị **0** để đảm bảo an toàn.

4. Lưu lại tập tin, tải nó lên máy chủ Web và thử trong trình duyệt (xem hình 3-13). Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 3.10.



Hình 3-13: *Kết quả tính toán.*

Đoạn mã 3.10: *Vì biến* ***$_POST*** *là biến siêu toàn cục nên giá trị của nó có thể được truy xuất phía trong một hàm.*

```
<?php # Doan ma 3.10 - handle_calculator.php

$page_title = 'Calculator';

include ('./header.inc');

function calculate_total () {

   global $total;

   $total = ($_POST['quantity'] * $_POST['price']) *
   → ($_POST['taxrate'] + 1);

   $total =  number_format ($total, 2);


}

if (is_numeric($_POST['quantity'])) {

   $total = 0;

   calculate_total ();

   echo "<p>Ban mua <b>{$_POST['quantity']}</b> mat hang voi
   → gia <b>\${$_POST['price']}</b>. Cong voi thue, tong so
   → tien la <b>\$$total</b>.</p>\n";

} else {

   echo '<p><b>Xin nhap vao so luong can mua!</b></p>';
```

```
}
include ('./footer.inc');
?>
```

Một biến **static** *(biến tĩnh) là biến thuộc về một hàm mà giá trị của nó được khởi tạo khi lần đầu tiên hàm được gọi, giá trị này sẽ được nhớ trong những lần gọi sau. Hàm dưới đây sẽ in ra số 1 ở lần đầu tiên nó được gọi, in ra số 2 cho lần gọi thứ nhì...*

```
function counter(){
        static $var = 1;
        echo $var++;
}
```

Một phương pháp khác để đánh lừa phạm vi của biến là truyền theo kiểu tham chiếu thay vì truyền theo kiểu giá trị. Điều này cho phép các thay đổi với biến phía trong hàm sẽ có ảnh hưởng đến các biến được định nghĩa ngoài hàm. Tham khảo tài liệu về PHP để biết thêm chi tiết về truyền giá trị và truyền tham chiếu.

Từ phiên bản 3 của PHP, bạn có thể sử dụng mảng $GLOBALS. Mảng này chứa tất cả các biến toàn cục của mã kịch bản.

Từ phiên bản 4.1 của PHP, có một biến siêu toàn cục là $_REQUEST, chứa toàn bộ các biến và giá trị được truyền bằng phương pháp get, post và cookie.

Mặc dù bạn có thể sử dụng câu lệnh global và các biến siêu toàn cục để viết lại hàm của mình như đã được thực hiện ở đây, tuy nhiên đó là điều không nên.

Xử lý biểu mẫu với PHP Redux

Tất cả các ví dụ trong chương này và các chương trước đều sử dụng hai mã kịch bản riêng biệt để xử lý các biểu mẫu HTML: Một dùng để thể hiện biểu mẫu và một dùng để xử lý nó. Tuy không có gì sai khi sử dụng theo cách này, nhưng việc đặt toàn bộ mã kịch bản trong một tập tin sẽ có những ưu điểm nhất định. Để có một mã kịch bản vừa thể hiện, vừa xử lý biểu mẫu HTML, hãy sử dụng câu lệnh điều kiện:

```
if(/* biểu mẫu đã được gửi */){
        // xử lý biểu mẫu
} else {
```

```
        // hiển thị biểu mẫu
}
```

Để xác định xem một biểu mẫu có được gửi hay không, chúng ta sẽ kiểm tra xem biến **$submit** có được thiết lập chưa (tên của trường submit trong biểu mẫu):

```
if(isset($_POST['submit'])){
        // xử lý biểu mẫu
} else {
        // hiển thị biểu mẫu
   echo '<input type="submit" name="submit" value="Go!">';
}
```

Nếu muốn một trang thực hiện việc xử lý một biểu mẫu và sau đó thể hiện lại nó (ví dụ, bổ sung một mẩu tin vào cơ sở dữ liệu và cung cấp một tùy chọn để bổ sung một mẩu tin khác), sử dụng đoạn mã sau:

```
if(isset($_POST['submit'])){
        // xử lý biểu mẫu
}
// hiển thị biểu mẫu
```

Phiên bản này của mã kịch bản sẽ xử lý biểu mẫu nếu nó được gửi và thể hiện biểu mẫu mỗi khi trang được tải lại.

Để minh họa kỹ thuật quan trọng này, chúng ta sẽ viết một biểu mẫu đăng ký. Biểu mẫu này sẽ được xây dựng thông qua các phần trong cuốn sách.

Thực hành xử lý các biểu mẫu HTML theo các bước sau:

1. Tạo một hồ sơ PHP trong trình soạn thảo văn bản:

```
<?php # Doan ma 3.11 - register.php
$page_title = 'Register!';
include ('./header.inc');
```

2. Viết câu lệnh điều kiện để xử lý biểu mẫu:

```
if (isset($_POST['submit'])) {
```

Như đã để cập trước đây, nếu biến **$submit** (hoặc **$_POST['submit']**) được thiết lập, có nghĩa là biểu mẫu đã được gửi và có thể xử lý nó.

3. Kiểm tra tính hợp lệ của biểu mẫu:

```
if (strlen($_POST['name']) > 0) {
```

```
    $name = TRUE;
} else {
    $name = FALSE;
    echo '<p>Ban quen nhap ten!</p>';
}
if (strlen($_POST['email']) > 0) {
    $email = TRUE;
} else {
    $email = FALSE;
    echo '<p>Ban quen nhap dia chi thu dien tu!</p>';
}
if (strlen($_POST['username']) > 0) {
    $username = TRUE;
} else {
    $username = FALSE;
    echo '<p>Ban quen nhap ten dang nhap!</p>';
}
if (strlen($_POST['password1']) > 0) {
    if ($_POST['password1'] == $_POST['password2']) {
        $password = TRUE;
    } else {
        $password = FALSE;
        echo '<p>Mat khau khong khop voi phan xac nhan!</p>';
    }
} else {
    $password = FALSE;
    echo '<p>Ban quen nhap mat khau!</p>';
}
if ($name && $email && $username && $password) {
    echo '<p>Ban da duoc dang ky.</p>';
} else {
    echo '<p>Xin hay thu lai.</p>';
}
```

Quá trình kiểm tra biểu mẫu HTML cũng giống như những gì đã phát triển trong Chương 2 "Lập trình với PHP". Vì các giá trị nhập là dạng chuỗi, bạn chỉ cần kiểm tra xem chúng có chiều dài là dương hay không. Nếu đúng, thiết lập biến

tương ứng thành **TRUE**. Nếu sai, hãy thiết lập biến tương ứng thành **FALSE** và in ra thông báo lỗi. Trường mật khẩu cũng được dùng để so sánh với trường xác nhận mật khẩu.

Nếu toàn bộ quá trình kiểm tra là hợp lệ, người dùng sẽ được đăng ký. Nếu không, người dùng sẽ được yêu cầu làm lại. Quá trình đăng ký sẽ được phát triển trong Chương 6 "Sử dụng PHP và MySQL".

4. Hoàn tất điều kiện chính và thể hiện biểu mẫu HTML:

```
} else {
?>
<form action="<?php echo $_SERVER['PHP_SELF']; ?>"
→ method="post">
<fieldset><legend>Nhap vao thong tin cua ban:</legend>
<p><b>Ten:</b> <input type="text" name="name" size="20"
→ maxlength="40" /></p>
<p><b>Email:</b> <input type="text" name="email" size="40"
→ maxlength="60" /> </p>
<p><b>Ten dang nhap:</b> <input type="text" name="username"
→ size="20" maxlength="40" /></p>
<p><b>Mat khau:</b> <input type="password" name="password1"
→ size="20" maxlength="40" /></p>
<p><b>Xac nhan:</b> <input type="password" name="password2"
→ size="20" maxlength="40" /></p>
</fieldset>
<div align="center"><input type="submit" name="submit"
→ value="Gui thong tin" /></div>
</form>
```

Biểu mẫu khá rõ ràng và có thể được phát triển sau này. Vì mã của biểu mẫu thường ở dạng HTML, chúng ta tạm thoát khỏi mã lệnh PHP mặc dù đang ở trong phạm vi của câu lệnh điều kiện. Điều này cho phép viết tiếp mã của biểu mẫu ở dạng HTML, thay vì phải dùng PHP để tạo ra chúng.

Ở đây, chúng ta sử dụng biến **$PHP_SELF** và nó được truy xuất thông qua biến siêu toàn cục **$_SERVER**. Biến này có giá trị là tên của mã kịch bản hiện tại. Việc gán giá trị này cho thuộc tính **action** sẽ gửi biểu mẫu tới chính trang chứa nó.

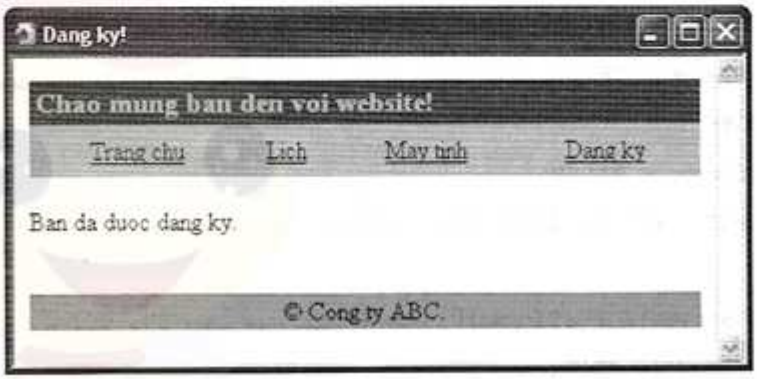
5. Hoàn tất câu lệnh điều kiện và trang PHP:

```
<?php
}
include ('./footer.inc');
?>
```

Khi vào và thoát khỏi mã lệnh PHP, rất dễ bỏ sót dấu ngoặc móc, bạn nên sử dụng các ghi chú để đánh dấu vị trí.

6. Lưu tập tin theo tên **register.php**, tải nó lên máy chủ Web và thử trong trình duyệt (xem hình 3-14, 3-15 và 3-16). Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 3.11.

Hình 3-14: *Biểu mẫu đăng ký.*



Hình 3-15: *Khi đăng ký thành công.*

Hình 3-16: *Nếu biểu mẫu chưa được điền đầy đủ, các thông báo lỗi sẽ được tạo.*

Đoạn mã 3.11: *Trang này sẽ hiển thị và xử lý biểu mẫu đăng ký.*

```
<?php # Doan ma 3.11 - register.php
$page_title = 'Register!';
include ('./header.inc');
if (isset($_POST['submit'])) {
   if (strlen($_POST['name']) > 0) {
      $name = TRUE;
   } else {
      $name = FALSE;
      echo '<p>Ban quen nhap ten!</p>';
   }
   if (strlen($_POST['email']) > 0) {
      $email = TRUE;
   } else {
      $email = FALSE;
```

```
      echo '<p>Ban quen nhap dia chi thu dien tu!</p>';
   }
   if (strlen($_POST['username']) > 0) {
      $username = TRUE;
   } else {
      $username = FALSE;
      echo '<p>Ban quen nhap ten dang nhap!</p>';
   }
   if (strlen($_POST['password1']) > 0) {
      if ($_POST['password1'] == $_POST['password2']) {
         $password = TRUE;
      } else {
         $password = FALSE;
         echo '<p>Mat khau khong khop voi phan xac nhan!</p>';
      }
   } else {
      $password = FALSE;
      echo '<p>Ban quen nhap mat khau!</p>';
   }
   if ($name && $email && $username && $password) {
      echo '<p>Ban da duoc dang ky.</p>';
   } else {
      echo '<p>Xin hay thu lai.</p>';
   }
} else {
?>
<form action="<?php echo $_SERVER['PHP_SELF']; ?>"
→ method="post">
<fieldset><legend>Nhap vao thong tin cua ban:</legend>
<p><b>Ten:</b> <input type="text" name="name" size="20"
```

```
→ maxlength="40" /></p>

<p><b>Email:</b> <input type="text" name="email" size="40"
→ maxlength="60" /> </p>

<p><b>Ten dang nhap:</b> <input type="text" name="username"
→ size="20" maxlength="40" /></p>

<p><b>Mat khau:</b> <input type="password" name="password1"
→ size="20" maxlength="40" /></p>

<p><b>Xac nhan:</b> <input type="password" name="password2"
→ size="20" maxlength="40" /></p>

</fieldset>

<div align="center"><input type="submit" name="submit"
→ value="Gui thong tin" /></div>

</form>

<?php

}

include ('./footer.inc');

?>
```

Có thể kiểm tra biểu mẫu có được gửi hay không bằng cách kiểm tra ***$_SERVER['REQUEST_METHOD']*** *có giá trị là* **post** *hay không (hoặc* **get** *nếu sử dụng phương pháp* **get**).

Hãy nhớ để quá trình này có thể làm việc, tên của trường submit phải có tên chính xác (cả về dạng chữ và đánh vần) như khi nó được thể hiện trong câu lệnh điều kiện.

Để quá trình này có thể làm việc được, nên ghi nhớ là trường submit phải có tên chính xác (cả về dạng chữ) như khi nó thể hiện trong câu lệnh điều kiện.

Mặc dù chưa thực hiện trong biểu mẫu này, nhưng chúng tôi cũng đề nghị bạn nên có dấu hiệu báo cho người dùng biết trường nào là bắt buộc và cho biết thông tin của trường (ví dụ, tên người dùng hoặc số điện thoại...). Nên để người dùng nhập mật khẩu hai lần, để tránh nhầm lẫn trong khi nhập.

Gửi thư điện tử

Một điều rất hấp dẫn của PHP là bạn có thể gửi thư điện tử rất dễ dàng. Trên một máy chủ được cấu hình thích hợp, quá trình này chỉ đơn giản là sử dụng hàm **mail()**:

```
mail($to, $subject, $body);
```

Giá trị tham số `$to` sẽ là địa chỉ thư điện tử hoặc một loạt địa chỉ thư điện tử cách nhau bằng dấu phẩy. Giá trị của tham số `$subject` là dòng chủ đề thư và tham số `$body` là nơi đặt nội dung thư. Khi tạo ra phần thân, bằng cách sử dụng ký tự xuống hàng (`\n`) trong dấu nháy kép sẽ làm cho văn bản thể hiện trên nhiều dòng.

Hàm `mail()` còn nhận tham số tùy chọn cho các tiêu đề phụ. Đây là nơi bạn có thể thiết lập `From`, `Reply-To`, `Cc`, `Bcc` và các thông số khác. Ví dụ:

```
mail('someone@domain.com','Question about PHP', $body,
→ 'From: person@address.com');
```

Để sử dụng nhiều dạng tiêu đề khác nhau trong thư điện tử, các tiêu đề được phân cách bằng `\r\n`:

```
$header = 'From: person@address.com\r\n';
$header .= 'Cc: anotherperson@address.com,
→ otherperson@address.com\r\n';
mail('someone@domain.com','Question about PHP', $body,
→ $header);
```

Thực hành gửi thư điện tử theo các bước sau:

1. Mở tập tin **register.php** (tham khảo lại đoạn mã 3.11) trong trình soạn thảo văn bản.
2. Tại nơi xử lý đăng ký, bổ sung mã lệnh sau:

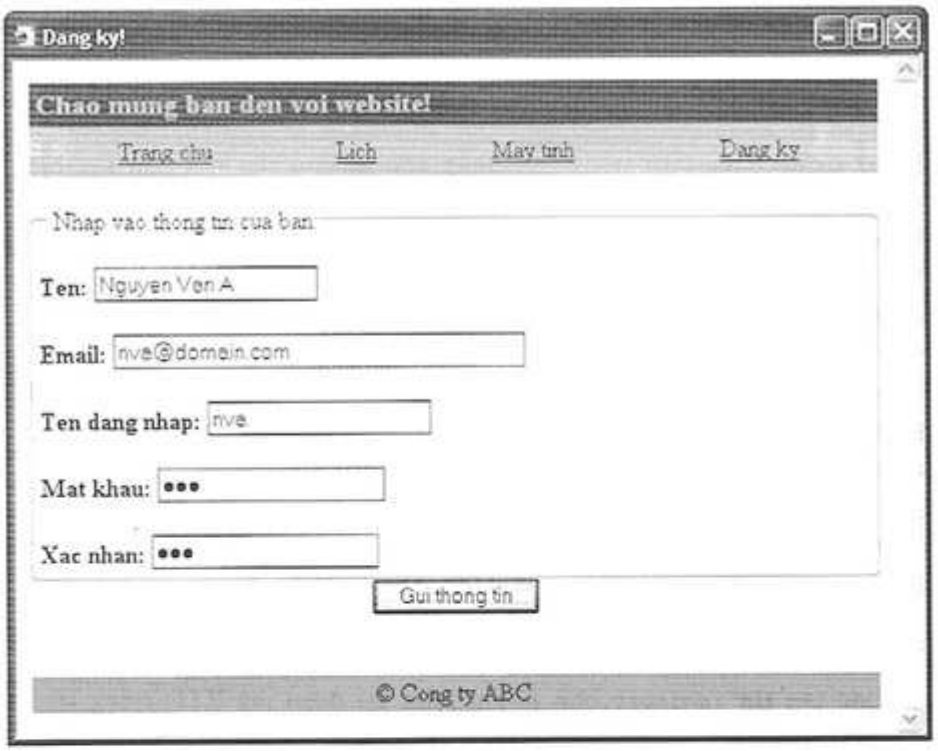
```
$body = "Cam on ban da dang ky!\nTen dang nhap cua ban la
→ '{$_POST['username']}' va mat khau
→ '{$_POST['password1']}'.\n\n";

mail ($_POST['email'], 'Cam on ban da dang ky!', $body,
→ 'From: admin@site.com');
```

Để gửi thư điện tử, đầu tiên sẽ xây dựng phần thân của thư, gán nó cho biến `$body` và sử dụng hàm `mail()`. Trong Chương 8 "An toàn", sẽ giới thiệu cách sử dụng biểu thức quy tắc để kiểm tra tính hợp lệ của địa chỉ thư điện tử.

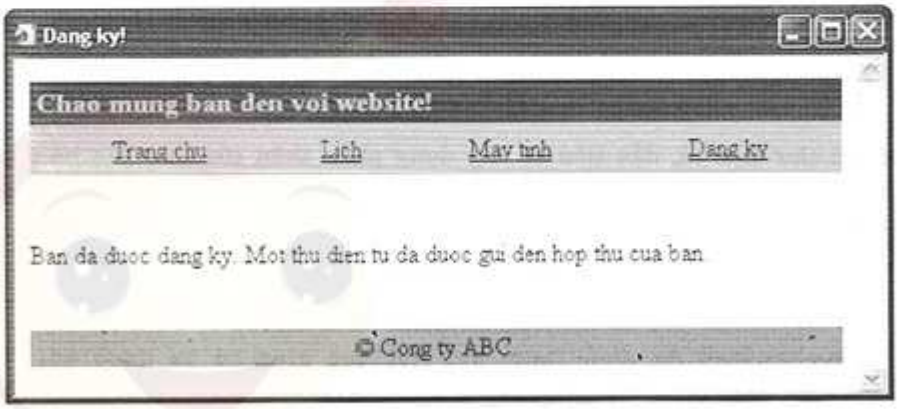
3. Nếu cần, thay đổi dòng lệnh `echo` ở phần đăng ký để phản ánh thay đổi trong mã kịch bản:

```
echo '<p>Ban da duoc dang ky. Mot thu dien tu da duoc gui
→ den hop thu cua ban.</p>';
```

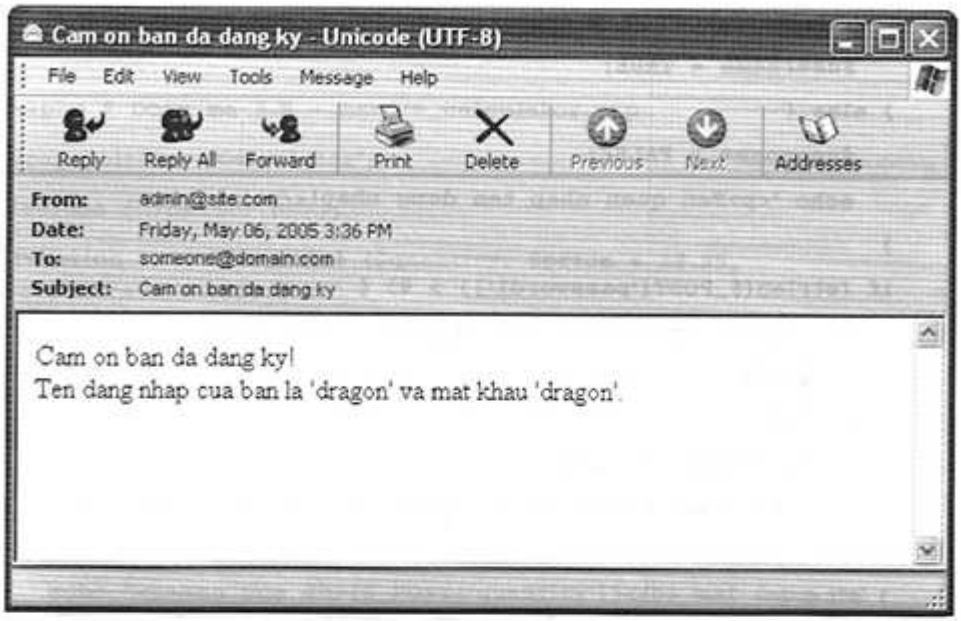
4. Lưu tập tin theo tên **register.php**, tải lên máy chủ Web và thử trong trình duyệt (xem hình 3-17, 3-18 và 3-19). Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 3.12.



Hình 3-17: *Để nhận được thông báo, phải đảm bảo rằng bạn đang sử dụng địa chỉ thư điện tử thật.*



Hình 3-18: *Bây giờ, trang đăng ký sẽ gửi thư điện tử đến người dùng.*



Hình 3-19: *Thư điện tử nhận được từ hàm* ***mail()*** *của PHP.*

Đoạn mã 3.12: *Hàm* ***mail()*** *của PHP rất dễ dùng.*

```
<?php # Doan ma 3.12 - register.php
$page_title = 'Dang ky!';
include ('./header.inc');
if (isset($_POST['submit'])) {
   if (strlen($_POST['name']) > 0) {
      $name = TRUE;
   } else {
      $name = FALSE;
      echo '<p>Ban quen nhap ten!</p>';
   }
   if (strlen($_POST['email']) > 0) {
      $email = TRUE;
   } else {
      $email = FALSE;
      echo '<p>Ban quen nhap dia chi thu dien tu!</p>';
   }
```

```
   if (strlen($_POST['username']) > 0) {
      $username = TRUE;
   } else {
      $username = FALSE;
      echo '<p>Ban quen nhap ten dang nhap!</p>';
   }
   if (strlen($_POST['password1']) > 0) {
      if ($_POST['password1'] == $_POST['password2']) {
         $password = TRUE;
      } else {
         $password = FALSE;
         echo '<p>Mat khau khong khop voi phan xac nhan!</p>';
      }
   } else {
      $password = FALSE;
      echo '<p>Ban quen nhap mat khau!</p>';
   }

   if ($name && $email && $username && $password) {
      // Đang ky nguoi dung.
      // Gui thu dien tu den nguoi dung.
      $body = "Cam on ban da dang ky!\nTen dang nhap cua ban
      → la '{$_POST['username']}' va mat khau
      → '{$_POST['password1']}'.\n\n";
      mail ($_POST['email'], 'Cam on ban da dang ky!', $body,
      → 'From: admin@site.com');
      echo '<p>Ban da duoc dang ky. Mot thu dien tu da duoc gui
      → den hop thu cua ban.</p>';
   } else {
      echo '<p>Hay thu lai.</p>';
   }
} else {
?>
<form action="<?php echo $_SERVER['PHP_SELF']; ?>"
→ method="post">
```

```
<fieldset><legend>Nhap vao thong tin cua ban:</legend>
<p><b>Ten:</b> <input type="text" name="name" size="20"
→ maxlength="40" /></p>
<p><b>Email:</b> <input type="text" name="email" size="40"
→ maxlength="60" /> </p>
<p><b>Ten dang nhap:</b> <input type="text" name="username"
→ size="20" maxlength="40" /></p>
<p><b>Mat khau:</b> <input type="password" name="password1"
→ size="20" maxlength="40" /></p>
<p><b>Xac nhan:</b> <input type="password" name="password2"
→ size="20" maxlength="40" /></p>
</fieldset>
<div align="center"><input type="submit" name="submit"
→ value="Gui thong tin" /></div>
</form>
<?php
}
include ('./footer.inc');
?>
```

Nếu hàm **mail()** *không gửi các thư địa chỉ từ máy chủ, bạn phải điều chỉnh lại các thiết lập SMTP trong tập tin* **php.ini** *(Windows) hoặc kiểm tra ứng dụng thư điện tử (của Unix hoặc Mac OS).*

Hàm **mail()** *trả lại giá trị* **1** *hoặc* **0** *để cho biết hàm có thực hiện thành công hay không, nhưng điều này không có nghĩa là thư điện tử đã được gửi và nhận thành công.*

Khi sử dụng hàm **mail()** *như trong ví dụ trên, bạn không thể gửi thư có đính kèm, mặc dù có thể gửi dưới dạng HTML. Thư đính kèm có thể gửi với hàm* **mail()**, *chủ yếu bằng cách sử dụng một lớp MIME (giống như của PEAR). Xem Chương 10 "Các chủ đề mở rộng" để biết thêm chi tiết về PEAR.*

HTTP Headers

HTTP (HyperText Transfer Protocol) là công nghệ chủ yếu của World Wide Web vì nó định nghĩa cách thức mà máy chủ và khách hàng có thể giao tiếp với nhau. Khi trình duyệt yêu cầu một trang Web, nó sẽ nhận được một loạt HTTP header.

Hàm **header()** cài sẵn của PHP được dùng để tận dụng ưu điểm của giao thức này. Ví dụ phổ biến nhất của hàm sẽ được minh họa trong phần này, khi nó được

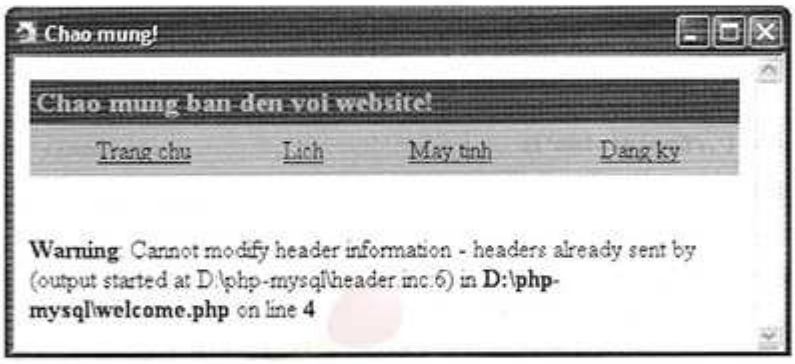
dùng để chuyển hướng trình duyệt từ trang Web này sang trang Web khác. Trong Chương 10 "Các chủ đề mở rộng", bạn sẽ thấy cách sử dụng hàm **header()** để tác động đến việc lưu trang trong bộ đệm. Trong Chương 8 "An toàn", nó sẽ được dùng để thực hiện kiểm tra xác thực HTTP (HTTP Authentication).

Sử dụng hàm **header()** nhằm chuyển hướng trình duyệt bằng dòng lệnh sau:

```
header("Location: http://www.url.com/page.php");
```

Dòng lệnh này phải là dòng lệnh cuối cùng được thực hiện trong trang hiện tại (vì trình duyệt sẽ nhanh chóng chuyển tới trang khác), nên dòng lệnh tiếp theo thường là lời gọi tới hàm **exit()** để làm ngừng việc thực hiện mã kịch bản hiện tại.

Điều quan trọng cần nhớ là hàm **header()** phải được gọi trước khi bất kỳ những gì được gửi xuống trình duyệt (bao gồm cả HTML và các khoảng trắng). Nếu mã lệnh hoặc tập tin kèm theo có lời gọi **echo()** hoặc **print()**, hoặc có các dòng trắng phía ngoài phạm vi thẻ PHP, bạn sẽ thấy thông báo lỗi như hình 3-20.



Hình 3-20: *Thông báo lỗi "headers already sent" rất phổ biến đối với các lập trình viên.*

Để tránh lỗi này, hãy sử dụng hàm **headers_sent()**. Hàm này kiểm tra xem dữ liệu đã được gửi xuống trình duyệt hay chưa.

```
if(!headers_sent()){
   header("Location: http://www.url.com/page.php");
   exit();
} else {
      echo "Không thể chuyển hướng trình duyệt";
}
```

Chúng ta sẽ điều chỉnh mã kịch bản "đăng ký" để chuyển người dùng đến trang "cảm ơn", sau khi họ đăng ký thành công. Để thực hiện điều này, trang phải được kết cấu lại để đảm bảo không có văn bản nào được gửi xuống trình duyệt trước khi gọi hàm **header()**.

Thực hành sử dụng hàm **header()** theo các bước sau:

1. Mở tập tin **register.php** (tham khảo đoạn mã 3.12) trong trình soạn thảo văn bản.
2. Di chuyển toàn bộ điều kiện **isset()** tới đầu trang, phía trên *dòng lệnh đính kèm tập tin*:

```
if (isset($_POST['submit'])) {
```

Nếu tất cả các điều kiện được thỏa mãn, người dùng sẽ được chuyển đến một trang Web khác. Do đó, chúng ta không muốn tự động đính kèm tập tin **header.inc** và mã lệnh HTML của nó. Để thực hiện điều này, bạn sẽ đặt điều kiện tại dòng lệnh đầu tiên của mã kịch bản.

3. Tạo biến **$message** mới:

```
$message = NULL;
```

Biến **$message** sẽ được dùng thay cho các câu lệnh **echo()** khi một điều kiện nào đó không được đáp ứng. Chúng ta phải thực hiện điều này để thể hiện thông báo trong ngữ cảnh của khuôn mẫu HTML (nếu in ngay thông báo xảy ra, chúng sẽ xuất hiện trong phần đầu của khuôn mẫu được tạo bởi tập tin đầu trang (**header.inc**)).

4. Viết lại phần điều kiện và thay thế từng câu lệnh **echo()** bằng **$message .=**:

```
if (strlen($_POST['name']) > 0) {
    $name = TRUE;
} else {
    $name = FALSE;
    $message .= '<p>Ban quen nhap ten!</p>';
}
if (strlen($_POST['email']) > 0) {
    $email = TRUE;
} else {
    $email = FALSE;
    $message .= '<p>Ban quen nhap dia chi thu dien tu!</p>';
}
if (strlen($_POST['username']) > 0) {
```

```
        $username = TRUE;
    } else {
        $username = FALSE;
        $message .= '<p>Ban quen nhap ten dang nhap!</p>';
    }
    if (strlen($_POST['password1']) > 0) {
        if ($_POST['password1'] == $_POST['password2']) {
            $password = TRUE;
        } else {
            $password = FALSE;
            $message .= '<p>Mat khau khong khop voi gia tri xac
            → nhan!</p>';
        }
    } else {
        $password = FALSE;
        $message .= '<p>Ban quen nhap mat khau!</p>';
    }
```

Biến **$message** được tạo dưới dạng một chuỗi dài chứa tất cả các thông báo lỗi phát sinh.

5. Bổ sung hàm **header()** vào quá trình xử lý đăng ký:

```
    if ($name && $email && $username && $password) {
        $body = "Cam on ban da dang ky!\nTen dang nhap cua ban la
        → '{$_POST['username']}' va mat khau la
        →'{$_POST['password1']}'.\n\n";
        mail ($_POST['email'], 'Cam on ban da dang ky!', $body,
        → 'From: admin@site.com');
        header ('Location: thankyou.php');
        exit();
    } else {
        $message .= '<p>Hay thu lai.</p>';
    }
```

Khi mọi chuyện tốt đẹp, hàm **header()** sẽ đưa người dùng đến trang **thankyou.php**. Sau đó, mã kịch bản sẽ thoát ra, bất kỳ những gì còn lại trong mã kịch bản sẽ không được thực hiện.

6. Thay đổi điều kiện gửi.

```
    }
```

Phiên bản này của mã kịch bản sẽ không sử dụng mệnh đề **else** với điều kiện gửi như trong phiên bản trước. Như vậy, chúng ta sẽ thay đổi dòng lệnh ở đây và loại bỏ dấu ngoặc móc đóng trước câu lệnh **include('./footer.inc')**.

7. Kèm theo tập tin tiêu đề đầu trang:

```
$page_title = 'Dang ky!';
include ('./header.inc');
```

Sau điều kiện chính và khi hàm **header()** đã được gọi, bạn có thể kèm theo tập tin một cách an toàn.

8. In ra các thông báo lỗi:

```
if (isset($message)) {
   echo '<font color="red">', $message, '</font>';
}
```

Người dùng sẽ thấy trang này trong hai trường hợp: lần đầu tiên họ truy xuất trang Web này hoặc khi họ đăng ký không thành công. Trong trường hợp thứ hai, biến **$message** sẽ có giá trị (cho biết người dùng thiếu hoặc sai thông tin gì) và nó sẽ được in ra tại đây, phía trên biểu mẫu.

9. Giữ nguyên phần còn lại của trang và lưu các thay đổi. Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 3.13.

Đoạn mã 3.13: *Hàm **header()** có thể chuyển hướng trình duyệt đến một trang khác.*

```
<?php # Doan ma 3.13 - register.php
if (isset($_POST['submit'])) {
   $message = NULL;
   if (strlen($_POST['name']) > 0) {
      $name = TRUE;
   } else {
      $name = FALSE;
      $message .= '<p>Ban quen nhap ten!</p>';
   }
   if (strlen($_POST['email']) > 0) {
      $email = TRUE;
   } else {
      $email = FALSE;
      $message .= '<p>Ban quen nhap dia chi thu dien tu!</p>';
   }
```

```
    if (strlen($_POST['username']) > 0) {
        $username = TRUE;
    } else {
        $username = FALSE;
        $message .= '<p>Ban quen nhap ten dang nhap!</p>';
    }
    if (strlen($_POST['password1']) > 0) {
        if ($_POST['password1'] == $_POST['password2']) {
            $password = TRUE;
        } else {
            $password = FALSE;
            $message .= '<p>Mat khau khong khop voi gia tri xac
            → nhan!</p>';
        }
    } else {
        $password = FALSE;
        $message .= '<p>Ban quen nhap mat khau!</p>';
    }
    if ($name && $email && $username && $password) {
        // Dang ky nguoi dung.
        // Gui mot thu dien tu.
        $body = "Cam on ban da dang ky!\nTen dang nhap cua ban
        → la '{$_POST['username']}' va mat khau la
        → '{$_POST['password1']}'.\n\n";
        mail ($_POST['email'], 'Cam on ban da dang ky!', $body,
        → 'From: admin@site.com');
        header ('Location: thankyou.php');
        exit();
    } else {
        $message .= '<p>Hay thu lai.</p>';
    }
}
$page_title = 'Dang ky!';
include ('./header.inc');
if (isset($message)) {
```

```
    echo '<font color="red">', $message, '</font>';
}
?>
<form action="<?php echo $_SERVER['PHP_SELF']; ?>"
→ method="post">
<fieldset><legend>Nhap vao thong tin cua ban:</legend>
<p><b>Ten:</b> <input type="text" name="name" size="20"
→ maxlength="40" /></p>
<p><b>Email:</b> <input type="text" name="email" size="40"
→ maxlength="60" /> </p>
<p><b>Ten dang nhap:</b> <input type="text" name="username"
→ size="20" maxlength="40" /></p>
<p><b>Mat khau:</b> <input type="password" name="password1"
→ size="20" maxlength="40" /></p>
<p><b>Xac nhan:</b> <input type="password" name="password2"
→ size="20" maxlength="40" /></p>
</fieldset>
<div align="center"><input type="submit" name="submit"
→ value="Gui thong tin" /></div>
</form>
<?php
include ('./footer.inc');
?>
```

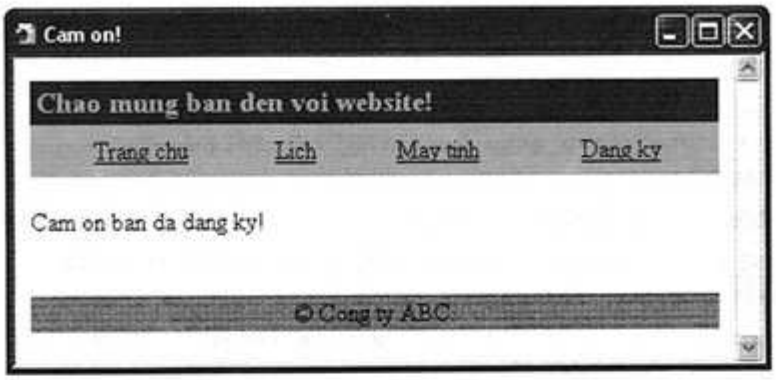
Phần tiếp theo, chúng ta sẽ nhanh chóng hoàn tất trang "cảm ơn".

Thực hành tạo trang "cảm ơn" theo các bước sau:

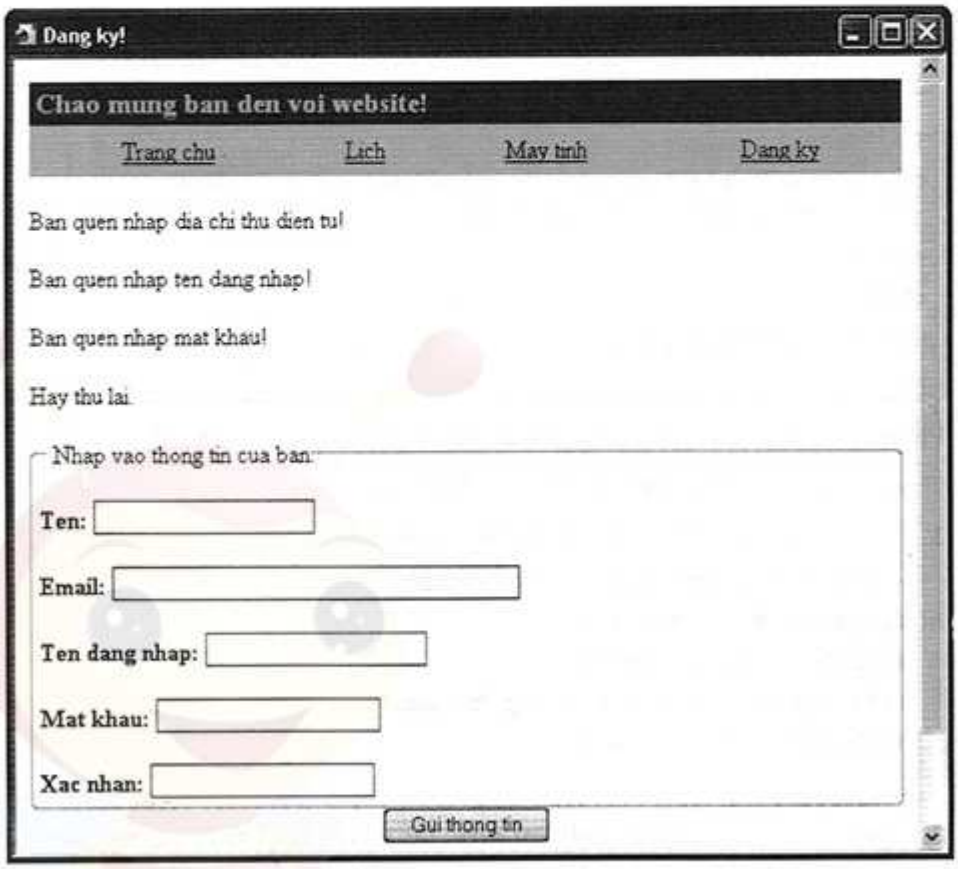
1. Tạo một hồ sơ PHP mới trong trình soạn thảo văn bản:

```
<?php # Doan ma 3.14 - thankyou.php
$page_title = 'Cam on!';
include ('./header.inc');
echo '<p>Cam on ban da dang ky!</p>';
include ('./footer.inc');
?>
```

2. Lưu tập tin theo tên **thankyou.php**, tải nó lên máy chủ cùng với phiên bản sửa đổi của tập tin **register.php** và thử trong trình duyệt (xem hình 3-21 và 3-22). Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 3.14.



Hình 3-21: *Nếu người dùng điền đủ thông tin trong biểu mẫu, họ sẽ được chuyển đến trang này.*



Hình 3-22: *Nếu người dùng điền thiếu thông tin trong biểu mẫu, họ sẽ nhận được thông báo lỗi và biểu mẫu sẽ xuất hiện trở lại.*

Đoạn mã 3.14: *Người dùng sẽ được chuyển đến trang này khi họ đăng ký thành công.*

```
<?php # Doan ma 3.14 – thankyou.php
$page_title = 'Cam on!';
include ('./header.inc');
echo '<p>Cam on ban da dang ky!</p>';
include ('./footer.inc');
?>
```

Hàm ***header()*** *thường được dùng khi mã kịch bản PHP phát sinh ra tập tin PDF, hình ảnh hoặc các media khác.*

Nếu muốn biết thêm chi tiết và thông tin liên quan về HTTP, hãy truy xuất trang Web www.w3.org/protocols.

HTTP là một giao thức không lưu giữ trạng thái (stateless), không ghi nhớ giữa các trang Web. Chương 7 "Cookies và Sessions" sẽ giới thiệu cách giải quyết vấn đề này của PHP.

Tuy chưa thực hiện ở đây, nhưng bạn nên sử dụng địa chỉ URL tuyệt đối với hàm ***header():***

```
header("Location: http://www.example.com/page.html");
```

Tạo biểu mẫu ghi nhớ (sticky form)

Bạn có thể đã gặp các biểu mẫu ghi nhớ, ngay cả khi không biết chúng được gọi là gì. Một biểu mẫu ghi nhớ chỉ là một biểu mẫu HTML chuẩn, ghi lại những gì đã được điền trong một biểu mẫu. Đây là một đặc điểm rất thú vị đối với người dùng cuối, đặc biệt nếu bạn yêu cầu họ điền và gửi lại biểu mẫu.

Để thiết lập những gì người dùng đã nhập vào hộp văn bản, hãy sử dụng thuộc tính **value** của nó.

```
<input type="text" name="name" size="20" maxlength="40"
→ value="Nguyen Van A"/>
```

Để PHP thiết lập giá trị, hãy in ra giá trị thích hợp:

```
<input type="text" name="name" size="20" maxlength="40"
→ value="<?php echo $_POST['name'];?>"/>
```

Với kiến thức này, bạn sẽ viết lại mã kịch bản **register.php** lần cuối cùng để nó ghi nhớ những gì người dùng đã nhập.

Thực hành tạo một biểu mẫu ghi nhớ theo các bước sau:

1. Mở tập tin **register.php** (tham khảo lại đoạn mã 3.13) trong trình soạn thảo văn bản.
2. Thay đổi trường nhập Name:

```
<p><b>Ten:</b> <input type="text" name="name" size="20"
→ maxlength="40" value="<?php if (isset($_POST['name']))
→ echo $_POST['name']; ?>" /></p>
```

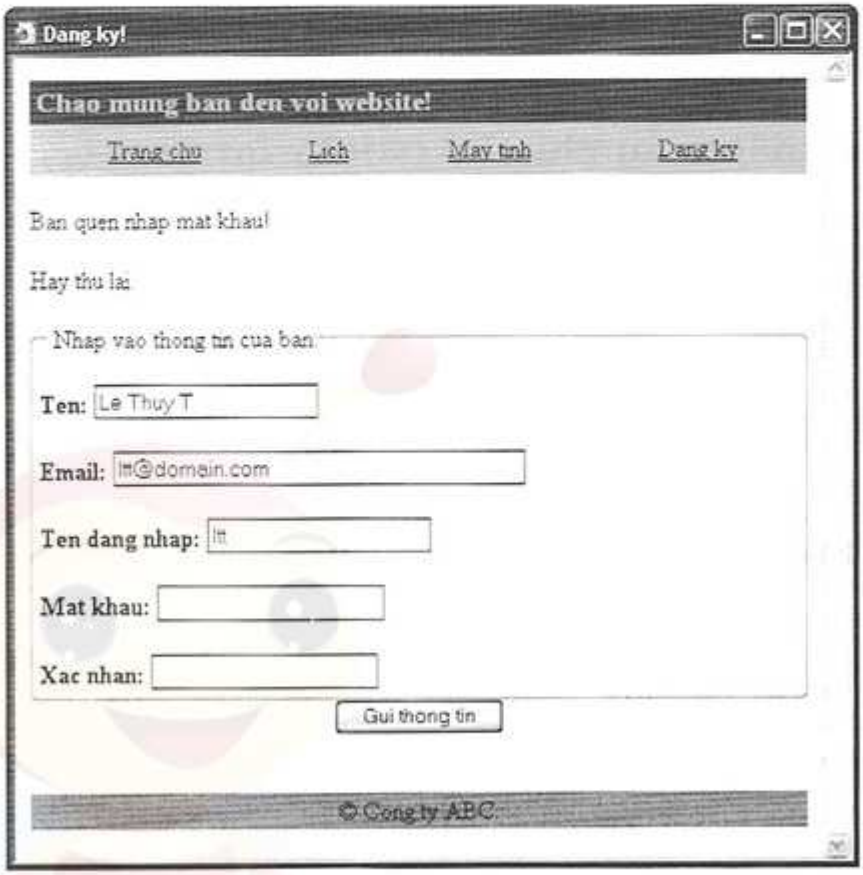
Điều đầu tiên được thực hiện là bổ sung thuộc tính **value** cho thẻ input. Sau đó, in ra giá trị của biến ứng với trường name đã được gửi (**$_POST['name']**). Nhưng trước hết, chúng ta phải kiểm tra xem biến này đã được thiết lập chưa để đảm bảo rằng nó có một giá trị. Kết quả cuối cùng ứng với giá trị của thẻ input là mã lệnh PHP:

```
<?php
if (isset($_POST['name']))
   echo $_POST['name'];
?>
```

3. Lặp lại quá trình cho trường email và username:

```
<p><b>Email:</b> <input type="text" name="email" size="40"
→ maxlength="60" value="<?php if (isset($_POST['email']))
→ echo $_POST['email']; ?>" /> </p>
<p><b>Ten dang nhap:</b> <input type="text" name="username"
→ size="20" maxlength="40" value="<?php if
→ (isset($_POST['username'])) echo $_POST['username']; ?>"
→ /></p>
```

4. Lưu tập tin theo tên **register.php**, tải nó lên máy chủ Web và thử trong trình duyệt (xem hình 3-23). Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 3.15.



Hình 3-23: *Phiên bản chuyên nghiệp này của biểu mẫu đã ghi lại những gì mà người dùng nhập vào.*

Đoạn mã 3.15: *Biểu mẫu đăng ký sẽ ghi nhớ lại các giá trị mà người dùng đã nhập.*

```
<?php # Doan ma 3.15 - register.php
if (isset($_POST['submit'])) {
   $message = NULL;
   if (strlen($_POST['name']) > 0) {
      $name = TRUE;
   } else {
      $name = FALSE;
      $message .= '<p>Ban quen nhap ten!</p>';
   }
   if (strlen($_POST['email']) > 0) {
      $email = TRUE;
   } else {
      $email = FALSE;
      $message .= '<p>Ban quen nhap dia chi thu dien tu!</p>';
   }
   if (strlen($_POST['username']) > 0) {
      $username = TRUE;
   } else {
      $username = FALSE;
      $message .= '<p>Ban quen nhap ten dang nhap!</p>';
   }
   if (strlen($_POST['password1']) > 0) {
      if ($_POST['password1'] == $_POST['password2']) {
         $password = TRUE;
      } else {
         $password = FALSE;
         $message .= '<p>Mat khau khong khop voi gia tri xac
         → nhan!</p>';
      }
   } else {
      $password = FALSE;
      $message .= '<p>Ban quen nhap mat khau!</p>';
   }
   if ($name && $email && $username && $password) {
      // Dang ky nguoi dung.
      // Gui mot thu dien tu.
      $body = "Cam on ban da dang ky!\nTen dang nhap cua ban
```

```
      → la '{$_POST['username']}' va mat khau la
      → '{$_POST['password1']}'.\n\n";
      mail ($_POST['email'], 'Cam on ban da dang ky!', $body,
      → 'From: admin@site.com');
      header ('Location: thankyou.php');
      exit();
   } else {
      $message .= '<p>Hay thu lai.</p>';
   }
}
$page_title = 'Dang ky!';
include ('./header.inc');
if (isset($message)) {
   echo '<font color="red">', $message, '</font>';
}
?>
<form action="<?php echo $_SERVER['PHP_SELF']; ?>"
→ method="post">
<fieldset><legend>Nhap vao thong tin cua ban:</legend>
<p><b>Ten:</b> <input type="text" name="name" size="20"
→ maxlength="40" value="<?php if (isset($_POST['name']))
→ echo $_POST['name']; ?>" /></p>
<p><b>Email:</b> <input type="text" name="email" size="40"
→ maxlength="60" value="<?php if (isset($_POST['email']))
→ echo $_POST['email']; ?>" /> </p>
<p><b>Ten dang nhap:</b> <input type="text" name="username"
→ size="20" maxlength="40" value="<?php if
→ (isset($_POST['username'])) echo $_POST['username']; ?>"
→ /></p>
<p><b>Mat khau:</b> <input type="password" name="password1"
→ size="20" maxlength="40" /></p>
<p><b>Xac nhan:</b> <input type="password" name="password2"
→ size="20" maxlength="40" /></p>
</fieldset>
<div align="center"><input type="submit" name="submit"
→ value="Gui thong tin" /></div>
</form>
<?php
include ('./footer.inc');
?>
```

Do hạn chế của HTML, không thể thiết lập trước giá trị của trường password.

Để thiết lập trạng thái của các nút chọn lựa hoặc hộp kiểm khi được chọn, hãy bổ sung mã **checked="checked"** *vào thẻ input của chúng:*

```
<input type="checkbox" name="interest[]" value="Skiing"
→ checked="checked"/>
```

Để thiết lập giá trị chọn của menu thả xuống, hãy sử dụng thuộc tính **selected="selected"**:

```
<select name="year">
<option value="2003">2003</option>
<option value="2004" selected="selected">2004</option>
</select>
```

Để thiết lập trước giá trị của một vùng văn bản, đặt giá trị này trong phạm vi thẻ mở và đóng của thẻ **textarea**:

```
<textarea name="comments" rows="10"
→ cols="50">Giá trị ban đầu</textarea>
```

Các hàm ngày, giờ

Phần này sẽ giới thiệu về các hàm của PHP có liên quan đến ngày và giờ. Quan trọng nhất là hàm **date()**, hàm này trả lại chuỗi văn bản ứng với ngày, giờ nhất định theo một định dạng xác định.

```
date(format, [timestamp]);
```

Tham số timestamp là một tùy chọn biểu thị cho số giây, tính từ mốc Unix Epoch (nửa đêm ngày 1-1-1970) tới thời điểm quan tâm. Nếu tham số này không được xác định, PHP sẽ sử dụng thời gian hiện tại của hệ thống.

Rất nhiều định dạng có thể áp dụng cho hàm **date()** (xem bảng 3.1) và chúng có thể phối hợp với văn bản thông thường. Ví dụ:

```
echo date('F j, Y');
echo date('H:i');
echo date('D');
```

Bảng 3.1: *Hàm* ***date()*** *có thể nhận vào một tổ hợp bất kỳ của các tham số này để thay đổi kết quả trả lại của nó.*

Ký tự	Ý nghĩa	Ví dụ
Y	Năm với bốn chữ số.	2003

y	Năm với hai chữ số.	03
n	Tháng với một hoặc hai chữ số.	2
m	Tháng với hai chữ số.	02
F	Tháng.	January
M	Tháng với ba chữ cái.	Jan
j	Ngày trong tháng với một hoặc hai chữ số.	5
d	Ngày của tháng với hai chữ số.	15
l	Ngày trong tuần.	Monday
D	Ngày trong tuần với ba chữ cái.	Mon
g	Giờ với định dạng 12 giờ cùng một hoặc hai chữ số.	6
G	Giờ với định dạng 24 giờ cùng một hoặc hai chữ số.	18
h	Giờ với định dạng 12 giờ cùng hai chữ số.	08
H	Giờ với định dạng 24 giờ cùng hai chữ số.	23
i	Phút.	45
a	am hoặc pm.	am
A	AM hoặc PM.	PM
s	Giây.	18

Có thể tìm giá trị của timestamp ứng với một ngày cụ thể bằng cách sử dụng hàm `mktime()`:

```
int mktime(giờ, phút, giây, tháng, ngày, năm);
```

Cuối cùng, hàm `getdate()` có thể dùng để trả lại một mảng giá trị (xem bảng 3.2) ứng với ngày, giờ hiện tại. Ví dụ:

```
$date = getdate();
echo $date['month'];
```

Bảng 3.2: *Hàm `getdate()` trả lại mảng liên kết này.*

Khóa	Giá trị	Ví dụ
year	Năm.	2005
Mon	Tháng.	4
month	Tên tháng.	April
mday	Ngày trong tháng.	25
weekday	Ngày trong tuần.	Tuesday

hours	Giờ.	11
minutes	Phút.	59
seconds	Giây.	30

Thực hành sử dụng các hàm ngày và giờ theo các bước sau:

1. Tạo một hồ sơ PHP mới trong trình soạn thảo văn bản:

```
<?php # Doan ma 3.16 - dateform.php
$page_title = 'Calendar Form';
include ('./header.inc');
```

2. Bắt đầu phần định nghĩa hàm:

```
function make_calendar_pulldown($this_month = NULL,
→ $today = NULL, $year = NULL) {
```

Mã kịch bản này giống như mã kịch bản **dateform.php** trước đây (tham khảo lại đoạn mã 3.5) nhưng các giá trị sẽ được chọn trước trong menu thả xuống dựa theo ngày và giờ hiện tại. Hàm nhận vào ba tham số và các tham số này đều là tùy chọn vì chúng cùng được thiết lập giá trị mặc định là **NULL**.

3. Tạo menu thả xuống ứng với tháng:

```
$months = array (1 => 'January', 'February', 'March',
→ 'April', 'May', 'June', 'July', 'August', 'September',
→ 'October', 'November', 'December');
echo '<select name="month">';
foreach ($months as $key => $value) {
   echo "<option value=\"$key\"";
   if ($value == $this_month) {
      echo ' selected="selected"';
   }
   echo ">$value</option>\n";
}
```

Không có gì thay đổi trong quá trình tạo menu thả xuống, ngoại trừ việc bổ sung điều kiện phía trong vòng lặp **foreach**. Điều kiện này sẽ kiểm tra tháng hiện tại (một con số trong khoảng từ 1 đến 12) so với giá trị được gửi (**$this_month**). Nếu chúng giống nhau, mã lệnh HTML sẽ được thêm vào để tháng đó sẽ được chọn.

4. Tạo menu thả xuống ứng với ngày:

```
echo '</select>
<select name="day">';
for ($day = 1; $day <= 31; $day++) {
   echo "<option value=\"$day\"";
```

```
    if ($day == $today) {
        echo ' selected="selected"';
    }
    echo ">$day</option>\n";
}
```

Các mã lệnh tạo menu thả xuống ứng với ngày cũng tương tự như menu thả xuống ứng với tháng, ngoại trừ điều kiện được thêm vào.

5. Tạo menu thả xuống ứng với năm:

```
<select name="year">';
if (!isset($year)) {
    $year = date('Y');
}
$end = $year + 10;
while ($year <= $end) {
    echo "<option value=\"$year\">$year</option>\n";
    $year++;
}
echo '</select>';
```

Menu thả xuống ứng với năm sẽ hơi khác một chút, nó tạo menu thả xuống ứng với 10 năm kể từ ngày được gửi. Nếu giá trị năm được gửi chưa được thiết lập, năm hiện tại (`date('Y')`) sẽ được sử dụng.

6. Hoàn tất hàm và tạo ra thẻ mở của biểu mẫu:

```
}
echo '<form action="dateform.php" method="post">';
```

7. Nhận thông tin ngày hiện tại và gọi hàm:

```
$dates = getdate();
make_calendar_pulldown ($dates['month'], $dates['mday'],
→ $dates['year']);
```

Để gửi các tham số ứng với tháng, ngày và năm cho hàm, hàm **getdate()** sẽ được sử dụng để lấy ra một mảng các giá trị ứng với ngày hiện tại. Sau đó, gọi hàm và gửi vào các giá trị thích hợp.

8. Hoàn tất biểu mẫu và in ra ngày giờ hiện tại:

```
echo '</form>';
echo '<p>Hom nay la ', date ('d:m:Y'), '. Gio hien tai la ',
→ date ('g:i a'), '.</p>';
```

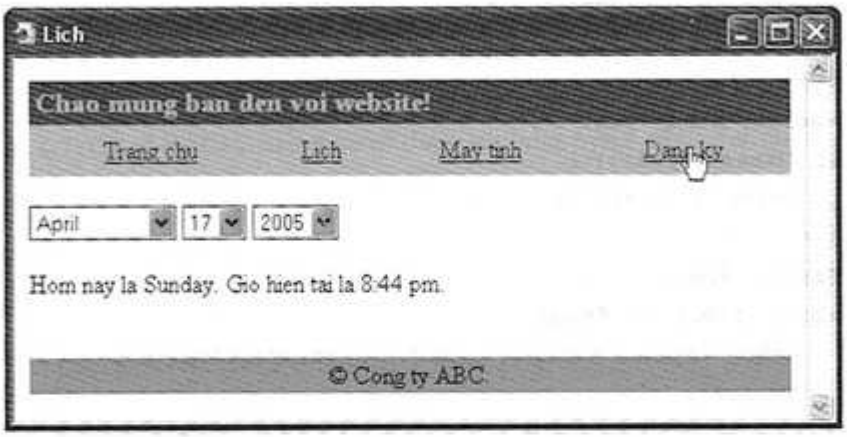
Chuỗi kết quả sẽ có dạng như sau:

```
<p>Hom nay la 10-06-2005. Gio hien tai la 11:30 am.</p>
```

9. Hoàn tất trang:

```
include ('./footer.inc');
?>
```

10. Lưu tập tin theo tên **dateform.php**, tải nó lên máy chủ Web và thử trong trình duyệt (xem hình 3-24). Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 3.16.



Hình 3-24: *Các hàm* ***date()*** *và* ***getdate()*** *bổ sung một số tính "động" cho trang Web.*

Đoạn mã 3.16: *Mã kịch bản sử dụng hàm* ***date()*** *và* ***getdate()***.

```
<?php # Doan ma 3.16 – dateform.php
$page_title = 'Lich';
include ('./header.inc');
function make_calendar_pulldown($this_month = NULL,
→ $today = NULL, $year = NULL) {
   $months = array (1 => 'January', 'February', 'March',
   → 'April', 'May', 'June', 'July', 'August', 'September',
   → 'October', 'November', 'December');
   echo '<select name="month">';
   foreach ($months as $key => $value) {
      echo "<option value=\"$key\"";
      if ($value == $this_month) {
         echo ' selected="selected"';
      }
      echo ">$value</option>\n";
   }
   echo '</select>
```

```
   <select name="day">';
   for ($day = 1; $day <= 31; $day++) {
      echo "<option value=\"$day\"";
      if ($day == $today) {
         echo ' selected="selected"';
      }
      echo ">$day</option>\n";
   }
   echo '</select>
   <select name="year">';
   if (!isset($year)) {
      $year = date('Y');
   }
   $end = $year + 10;
   while ($year <= $end) {
      echo "<option value=\"$year\">$year</option>\n";
      $year++;
   }
   echo '</select>';
}
echo '<form action="dateform.php" method="post">';
$dates = getdate();
make_calendar_pulldown ($dates['month'], $dates['mday'],
→ $dates['year']);
echo '</form>';
echo '<p>Hom nay la ', date ('d:m:Y'), '. Gio hien tai la ',
→ date ('g:i a'), '.</p>';
include ('./footer.inc');
?>
```

MySQL cũng có các hàm về ngày giờ của riêng nó và sẽ được giới thiệu trong Chương 5 "SQL và MySQL nâng cao".

Cần sử dụng JavaScript nếu muốn xác định ngày và giờ trên máy khách hàng.

*Hàm **checkdate()** nhận vào ba tham số (tháng, ngày và năm) và kiểm tra xem đó có phải là ngày hợp lệ không.*

Chương 4:
GIỚI THIỆU VỀ SQL VÀ MYSQL

Chương này sẽ tạm thời để PHP lại phía sau và đi vào nghiên cứu SQL và MySQL. Vì cuốn sách này đề cập đến việc tích hợp hai công nghệ PHP và MySQL, nên việc tìm hiểu từng công nghệ trước khi bắt tay vào viết mã kịch bản PHP sử dụng SQL để tương tác với MySQL là một công việc cần thiết và quan trọng.

SQL (Structured Query Language - Ngôn ngữ Hỏi đáp có Cấu trúc), là một nhóm các từ đặc biệt được sử dụng để tương tác với cơ sở dữ liệu. Phần lớn các cơ sở dữ liệu đều hỗ trợ SQL và MySQL cũng không phải là ngoại lệ. SQL được tạo ra không lâu sau khi Tiến sỹ E.F. Codd đưa ra lý thuyết về cơ sở dữ liệu quan hệ vào đầu những năm 70. Vào năm 1989, tổ chức American National Standards Institute (một tổ chức có trách nhiệm duy trì và phát triển ngôn ngữ này) công bố chuẩn SQL đầu tiên mà ngày nay được gọi là SQL89. SQL2 được công bố vào năm 1992 và vẫn được sử dụng cho đến bây giờ (còn được gọi là SQL92 hoặc đơn giản là SQL).

MySQL là ứng dụng cơ sở dữ liệu mã nguồn mở phổ biến nhất hiện nay (theo như thông tin từ Website của MySQL: *www.mysql.com*) và được sử dụng phối hợp với PHP. Phần mềm MySQL được sử dụng cùng với ứng dụng máy chủ cơ sở dữ liệu, phần mềm máy khách và một số tiện ích khác. Chương này và chương tiếp theo sẽ minh họa cách sử dụng SQL và MySQL để lưu giữ và sử dụng lại dữ liệu. Chương 6 "Sử dụng PHP và MySQL", sẽ áp dụng các thông tin này cho các mã kịch bản PHP.

Định nghĩa bảng

Trước khi làm việc với SQL và MySQL, cần xác định các nhu cầu cho ứng dụng. Điều này có ý nghĩa quyết định đến thiết kế cơ sở dữ liệu. Trong các ví dụ của chương này, chúng ta sẽ tạo ra một cơ sở dữ liệu (với tên gọi phổ thông là *sitename*) để chứa các thông tin đăng ký của người dùng (sẽ được tích hợp trong Chương 6 "Sử dụng PHP và MySQL"). Cơ sở dữ liệu bao gồm chỉ có một bảng, *users* chứa tất cả các cột để lưu trữ định danh người dùng, tên, địa chỉ thư điện tử, mật khẩu và ngày đăng ký. Bảng 4.1 thể hiện bố cục của bảng này theo quy tắc đặt tên của MySQL cho các tiêu đề cột (các tên gồm chữ cái, chữ số, dấu gạch dưới và không có khoảng trắng).

Bảng 4.1: *Bảng dữ liệu được sử dụng trong toàn bộ chương này.*

Tên cột	Ví dụ
`user_id`	834
`first_name`	Lynn

last_name	Mauck
email	phpmysql@domain.com
password	Leonard
registration_date	2005-03-15 17:39:57

Sau khi xác định tất cả các bảng và cột cần phải có của cơ sở dữ liệu, chúng ta cần xác định kiểu dữ liệu của từng trường. Khi tạo cơ sở dữ liệu, MySQL đòi hỏi định nghĩa nhiều loại thông tin cho mỗi trường. Có ba nhóm chính và được áp dụng cho hầu hết các phần mềm cơ sở dữ liệu:

- Văn bản.
- Số.
- Ngày và giờ.

Trong mỗi nhóm còn có nhiều dạng con và một số là dạng riêng của MySQL. Việc chọn kiểu trường không chỉ cho biết trường đó chứa dữ liệu gì, mà còn ảnh hưởng đến sự vận hành của cả cơ sở dữ liệu. Bảng 4.2 liệt kê những kiểu dữ liệu phổ biến của MySQL, kích thước cần sử dụng cho chúng và lời mô tả ngắn gọn.

Bảng 4.2: *Các kiểu dữ liệu phổ biến được dùng để định nghĩa cột.*

Kiểu	Kích thước	Mô tả
CHAR [length]	length byte	Trường kích thước cố định từ 0 đến 255 ký tự.
VARCHAR[length]	Chiều dài chuỗi + 1 byte	Trường kích thước thay đổi từ 0 đến 255 ký tự.
TINYTEXT	Chiều dài chuỗi + 1 byte	Chuỗi với chiều dài tối đa 255 ký tự.
TEXT	Chiều dài chuỗi + 2 byte	Chuỗi với chiều dài tối đa 65.535 ký tự.
MEDIUMTEXT	Chiều dài chuỗi + 3 byte	Chuỗi với chiều dài tối đa 16.777.215 ký tự.
LONGTEXT	Chiều dài chuỗi + 4 byte	Chuỗi với chiều dài tối đa 4.294.967.215 ký tự.
TINYINT[length]	1 byte	Thay đổi từ -128 đến 127 hoặc 0 đến 255 với kiểu không dấu.
SMALLINT[length]	2 byte	Thay đổi từ -32.768 đến 32.767 hoặc từ 0 đến 65.535 với kiểu không dấu.

MEDIUMINT[length]	3 byte	Thay đổi từ -8.388.608 đến 8.388.607 hoặc từ 0 đến 16.777.215 với kiểu không dấu.
INT[length]	4 byte	Thay đổi từ -2.147.483.648 đến 2.147.483.647 hoặc từ 0 đến 4.294.967.295 với kiểu không dấu.
BIGINT[length]	8 byte	Thay đổi từ -9.223.372.036.854.775.808 đến 9.223.372.036.854.775.807 hoặc từ 0 đến 18.448.744.073.709.551.615 với kiểu không dấu.
FLOAT	4 byte	Số thập phân nhỏ.
DOUBLE[length, decimal]	8 byte	Số thập phân lớn.
DECIMAL[length, decimal]	length + 1 hoặc length + 2 byte	Số DOUBLE được chứa ở dạng chuỗi.
DATE	3 byte	Ở dạng YYYY-MM-DD.
DATETIME	8 byte	Ở dạng YYYY-MM-DD HH:MM:SS.
TIMESTAMP	4 byte	Ở dạng YYYYMMDDHHMMSS.
TIME	3 byte	Ở dạng HH:MM:SS.
ENUM	1 hoặc 2 byte	Trường có một giá trị trong một tập hợp được định trước.
SET	1, 2, 4 hoặc 8 byte	Cũng giống ENUM, ngoại trừ trường có thể có nhiều giá trị từ một tập hợp định trước.

NULL và giá trị mặc định

UNSIGNED và ZEROFILL là hai trong số các thuộc tính có thể gán khi định nghĩa kiểu dữ liệu. Ngoài ra, còn có hai tùy chọn: Một tùy chọn dùng để xác định trường có thể nhận giá trị NULL hay không và một tùy chọn dùng để xác định giá trị mặc định cho trường.

Trong lập trình cơ sở dữ liệu, giá trị NULL tương đương với phát biểu "trường không có giá trị". Về mặt lý thuyết, mọi trường trong mẩu tin đều có một giá trị, nhưng điều kiện này trong thực tế rất ít khi được đáp ứng. Để buộc một trường phải có một giá trị, cần bổ sung mô tả NOT NULL cho kiểu cột của trường. Ví dụ:

```
cost DECIMAL(5, 2) UNSIGNED NOT NULL
```

Khi tạo bảng, cũng có thể xác định một giá trị mặc định. Trong trường hợp mà đa số mẩu tin có cùng một giá trị ứng với một cột, việc thiết lập trước một giá trị mặc định sẽ giúp bạn khỏi phải xác định một giá trị khi chèn vào các mẩu tin mới (trừ khi giá trị của mẩu tin cho cột đó khác với giá trị mặc định).

```
gender ENUM('m','f') default 'm'

phone_number_type ENUM('Work', 'Home', 'Mobile', 'Fax')
→ default 'home'
```

Nếu không xác định giá trị cụ thể của trường khi bổ sung mẩu tin, sẽ xảy ra hai trường hợp sau:

- Nếu trường đã được định nghĩa với giá trị mặc định, giá trị này sẽ được sử dụng.
- Nếu giá trị mặc định không được xác định và trường được định nghĩa là **NOT NULL**, sẽ xuất hiện lỗi.

Nhiều kiểu dữ liệu còn có một thuộc tính tùy biến là **Length**. Thuộc tính này giới hạn kích thước của trường. Ngoài ra, các kiểu dữ liệu số còn có thể định nghĩa là **UNSIGNED** (chỉ cho phép giá trị trong trường là số dương hoặc bằng **0**) hoặc **ZEROFILL** (khoảng trống còn lại sẽ được thay bằng các số **0**, **ZEROFILL** cũng bao hàm **UNSIGNED**).

Các kiểu dữ liệu ngày giờ cũng có cách ứng xử khác nhau, có thể tham khảo điều này tại địa chỉ *www.mysql.com/doc/D/A/DATETIME.html*. Các trường **DATE** và **TIME** thường không phải điều chỉnh khi sử dụng nên bạn không cần quan tâm nhiều tới những rắc rối của chúng.

Có hai kiểu đặc biệt là **ENUM** và **SET**, chúng cho phép định nghĩa một loạt giá trị có thể chấp nhận được, ứng với một trường. Một cột **ENUM** chỉ có thể có một giá trị trong số những giá trị được thiết lập (số giá trị thiết lập có thể lên đến vài ngàn), trong khi **SET** cho phép thiết lập tới 64 giá trị khác nhau.

Thực hành thiết kế bảng theo các bước sau:

1. Xác định xem dữ liệu ở mỗi cột trong bảng là dạng văn bản, số hoặc một kiểu dữ liệu khác (xem bảng 4.3).

Bảng 4.3: *Bảng users với các kiểu dữ liệu tổng quát.*

Tên cột	Kiểu
user_id	Số
first_name	Văn bản
last_name	Văn bản
Email	Văn bản
password	Văn bản
registration_date	Văn bản

Đây là bước tương đối đơn giản và rõ ràng. Một vài giá trị số (như mã vùng, số tiền...) có thể lưu dưới dạng chuỗi khi bạn muốn kèm theo ký tự không phải dạng số (như dấu $, dấu phẩy...). Tuy nhiên, bạn nên lưu chúng ở dạng số và định dạng lại trong mã kịch bản khi cần thể hiện.

2. Chọn một kiểu dữ liệu thích hợp cho mỗi trường (xem bảng 4.4).

Bảng 4.4: *Bảng users với các kiểu dữ liệu.*

Tên cột	Kiểu
user_id	MEDIUMINT
first_name	VARCHAR
last_name	VARCHAR
email	VARCHAR
password	CHAR
registration_date	DATETIME

Trong ví dụ này, chúng ta thiết lập kiểu là **MEDIUMINT** cho trường user_id, cho phép nó có khoảng 17 triệu giá trị khác nhau (dưới dạng một số không dấu). Trường registration_date sẽ có kiểu là **DATETIME**, chứa cả ngày và giờ tại thời điểm người dùng đăng ký. Khi quyết định kiểu dữ liệu, bạn cần cân nhắc xem có cần đến ngày hoặc giờ hoặc cả hai.

Các trường khác thường sử dụng kiểu dữ liệu **VARCHAR**, vì chiều dài của chúng sẽ thay đổi khi chuyển từ mẩu tin này sang mẩu tin khác. Ngoại trừ trường mật khẩu, do có chiều dài cố định nên sẽ có kiểu **CHAR** (chúng ta sẽ tìm hiểu nguyên do khi đề cập đến việc chèn các mẩu tin ở phần sau của chương này). Để biết thêm chi tiết, bạn tham khảo phần nói về kiểu dữ liệu **CHAR** và **VARCHAR**.

3. Thiết lập chiều dài lớn nhất và các thuộc tính khác (như **UNSIGNED** và **NULL**) cho các cột văn bản và cột số (xem bảng 4.5).

Bảng 4.5: *Định nghĩa cuối cùng của bảng users.*

Tên cột	Kiểu
user_id	MEDIUMINT UNSIGNED NOT NULL
first_name	VARCHAR(15) NOT NULL
last_name	VARCHAR(30) NOT NULL
email	VARCHAR(40)
password	CHAR(16) NOT NULL
registration_date	DATETIME NOT NULL

Nên giới hạn kích thước trường ở mức nhỏ nhất, dựa theo giá trị lớn nhất của nó. Ví dụ, nếu con số lớn nhất của trường user_id là hàng trăm, hãy thiết lập kiểu dữ liệu của trường là một số gồm ba chữ số, không dấu và có kiểu **SMALLINT** (cho phép tới 999 giá trị).

Trong ví dụ này, mỗi trường đều phải có giá trị (không thể là **NULL**), ngoại trừ trường email. Trường user_id với kiểu **UNSIGNED** phải có giá trị là một số dương. Không có cột nào được gán giá trị mặc định.

Chương 5 "SQL và MySQL nâng cao" sẽ giới thiệu việc thiết kế chi tiết và phát triển một cơ sở dữ liệu phức tạp.

*Nhiều kiểu dữ liệu có ý nghĩa giống nhau: **INT** tương tự như **INTEGER**, **DEC** tương tự như **DECIMAL**...*

*Kiểu dữ liệu **TIMESTAMP** được tự động thiết lập khi thực hiện câu lệnh **INSERT** hoặc **UPDATE**, ngay cả khi không có giá trị được xác định cho trường đó. Nếu một bảng có nhiều cột **TIMESTAMP**, chỉ có cột đầu tiên được cập nhật khi thực hiện câu lệnh **INSERT** hoặc **UPDATE**.*

*Kiểu dữ liệu là **BLOB** (một biến thể của **TEXT**) cho phép chứa các tập tin nhị phân trong bảng.*

*Kiểu dữ liệu **ENUM** và **SET** là phần mở rộng của MySQL và các kiểu dữ liệu này không tồn tại trong các ứng dụng cơ sở dữ liệu khác. Việc sử dụng chúng sẽ giới hạn tính tương thích cơ sở dữ liệu.*

*Chú ý, nếu một chuỗi năm ký tự được chèn vào trường **CHAR(2)**, ba ký tự phía sau sẽ bị cắt bỏ. Điều này đúng với trường bất kỳ có thiết lập chiều dài (**CHAR**, **VARCHAR**, **INT**...).*

So sánh giữa CHAR và VARCHAR

*Cả hai kiểu dữ liệu này đều chứa chuỗi và có thể thiết lập chiều dài cố định. Khác biệt chủ yếu giữa hai dạng này là: những gì chứa trong trường **CHAR** sẽ luôn ở dưới dạng một chuỗi và có chiều dài bằng với kích thước trường (bằng cách thêm vào các khoảng trắng). Ngược lại, các chuỗi **VARCHAR** chỉ chứa đúng bằng số ký tự có trong chuỗi giá trị.*

Điều này nghĩa là:

- *Các trường **VARCHAR** chiếm ít không gian đĩa hơn.*
- *Trừ khi sử dụng kiểu bảng là InnoDB, các trường kiểu **CHAR** được truy xuất nhanh hơn **VARCHAR**.*

Hiện tại, trong phần lớn trường hợp, khác biệt về vận tốc và không gian đĩa giữa hai kiểu dữ liệu này là không đáng kể và khi kiểu bảng InnoDB trở nên phổ biến, khác biệt về tốc độ sẽ không còn nữa.

*Còn một khác biệt thứ ba giữa hai dạng này là: MySQL sẽ cắt các khoảng trắng thừa khỏi các cột **CHAR** (khi dữ liệu được lấy ra) và cột **VARCHAR** (khi dữ liệu được chèn vào).*

Kết luận, kiểu **CHAR** *được sử dụng khi một trường dạng chuỗi luôn luôn cần có chiều dài cố định, các trường hợp còn lại sẽ sử dụng kiểu* **VARCHAR**.

Sử dụng trình quản lý MySQL

Ngoài việc sử dụng mã kịch bản PHP, cách phổ biến nhất để làm việc với máy chủ MySQL là sử dụng *trình quản lý mysql* (*mysql monitor*). Ứng dụng này được dùng để kết nối *mysqld* (ứng dụng thực tế quản lý cơ sở dữ liệu) trên máy tính hiện hành hoặc trên một máy khác. Tất cả các ví dụ trong chương này sẽ được minh họa thông qua *mysql* (từ *mysql* viết thường được dùng khi đề cập đến trình quản lý mysql (*mysql monitor*), còn từ MySQL nói đến bản thân phần mềm). Trong Chương 9 "Phát triển ứng dụng Web", chúng ta sẽ sử dụng trình quản lý mysql dưới dạng một công cụ dò lỗi.

Trình mysql được truy xuất từ giao diện dòng lệnh (ứng dụng Terminal trong Linux hoặc Mac và dấu nhắc DOS trong Windows.) Nó có thể nhận vào một số tham số, bao gồm tên truy cập, mật khẩu và tên máy. Thiết lập các tham số này như sau:

```
mysql -u <têntruycập> -p -h <tênmáy>
```

Tùy chọn **-p** sẽ buộc mysql nhắc bạn cung cấp mật khẩu. Có thể xác định mật khẩu trên dòng lệnh này bằng cách nhập nó ngay sau tham số **-p**, nhưng như vậy sẽ không an toàn vì mọi người đều thấy được mật khẩu. Tham số **-h** là tùy chọn và thường được bỏ qua khi bạn truy cập mysqld trên cùng máy với mysql.

Trong chế độ làm việc của mysql, mỗi câu lệnh SQL cần phải kết thúc bằng dấu chấm phẩy (;). Điều này có nghĩa là bạn có thể tiếp tục cùng câu lệnh trên nhiều dòng khác nhau để thuận tiện cho việc nhập từ bàn phím.

Để minh họa cho việc truy xuất và sử dụng trình theo dõi mysql, chúng tôi sẽ giới thiệu cách khởi động mysql, chọn một cơ sở dữ liệu để sử dụng và thoát khỏi chương trình. Bạn phải chạy chương trình *mysqld* (chương trình quản lý trực tiếp cơ sở dữ liệu) để có thể truy cập được từ chương trình mysql.

Các chương trình thay thế cho mysql

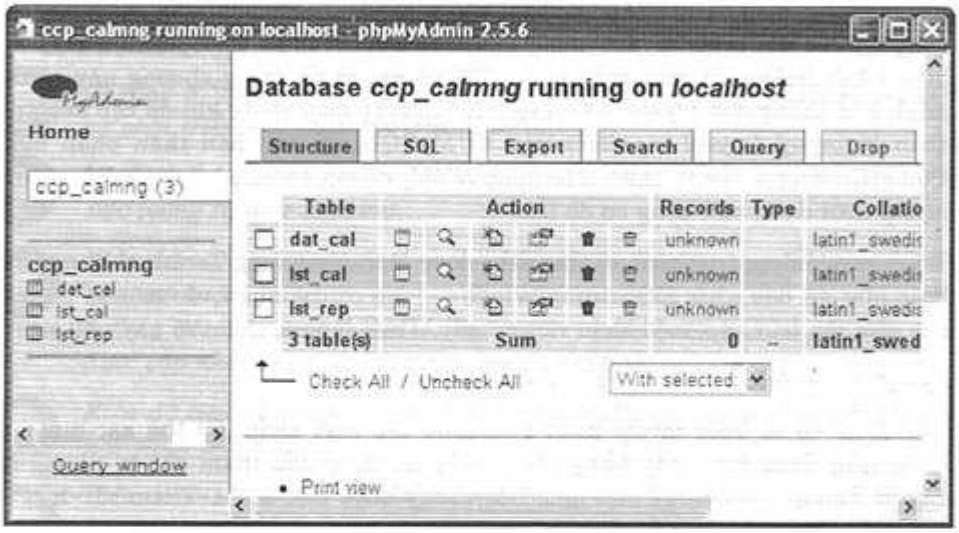
Vì chương trình mysql là một công cụ chạy ở chế độ dòng lệnh, bạn có thể không sử dụng được nó nếu đang làm việc với máy chủ của nhà cung cấp dịch vụ hoặc một máy chủ ký gởi Web. Khi đó, bạn có thể thử hai công cụ sau:

- *Kết nối với máy chủ bằng telnet và sử dụng mysql.*
- *Cài đặt phần mềm MySQL trên máy của bạn và sử dụng trình quản lý mysql để kết nối từ xa với máy chủ bằng cách xác định tên máy chủ (*`mysql -u <tênđăngnhập> -p -h <tênmáychủ>`*).*

Nếu cả hai cách này đều không thực hiện được, bạn có thể sử dụng phpMyAdmin (một công cụ mã nguồn mở phổ biến, cung cấp giao diện Web để làm việc với MySQL - có thể tải nó ở địa chỉ www.phpmyadmin.net). Phần mềm này phổ biến đến nỗi nhiều công ty cung cấp dịch vụ đã cung

cấp nó dưới dạng một dịch vụ mặc định, để khách hàng của họ có thể làm việc với cơ sở dữ liệu của mình.

Nếu không thể truy xuất MySQL thông qua trình quản lý mysql, bạn vẫn có thể thực hành mọi thứ với phpMyAdmin. Thẻ SQL trong phiên bản mới nhất (xem hình 4-1) cho phép nhập trực tiếp các câu lệnh SQL.



Hình 4-1: *Công cụ miễn phí phpMyAdmin là một lựa chọn tuyệt vời để quản trị MySQL.*

Thực hành sử dụng trình quản lý mysql theo các bước sau:

1. Truy xuất hệ thống từ giao diện dòng lệnh.

Trên các hệ thống Unix và Mac, đây chỉ là việc chuyển sang chế độ Terminal hoặc sang một ứng dụng tương tự. Trên Windows, chọn mục Run từ menu Start, nhập **cmd** trong ô nhập và nhấn phím Enter.

2. Chuyển đến thư mục mysql/bin hoặc mysql (tùy thuộc vào hệ điều hành và nơi cài đặt phần mềm MySQL). Xem hình 4-2.

Hình 4-2: *Trên nhiều hệ điều hành, trình quản lý mysql phải được truy xuất từ thư mục mysql/bin hoặc mysql.*

Với Unix:

```
cd /usr/local/mysl/bin
```

hoặc với Mac:

```
cd /usr/local/mysql
```

hoặc với Windows:

```
cd c:\mysql\bin
```

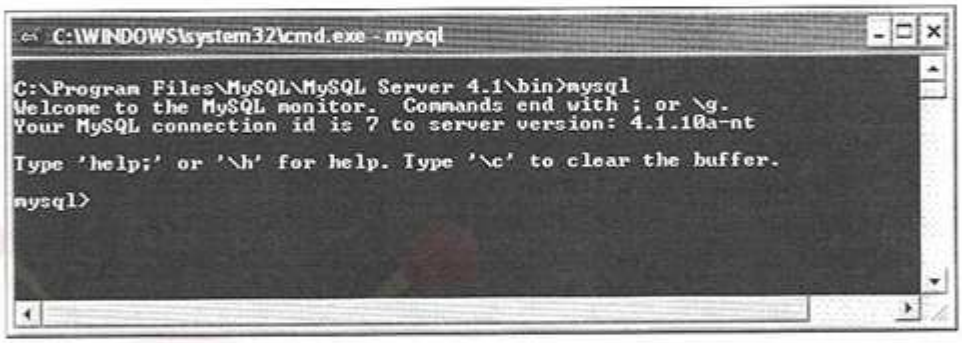
Cú pháp của bước vừa thực hiện và các bước tiếp theo chủ yếu được xác định dựa trên hệ điều hành và nơi cài đặt MySQL. Để truy xuất thành công, bạn cần thử nghiệm các dạng lệnh này.

3. Nhập vào văn bản sau (xem hình 4-3):

```
bin/mysql –u <têndăngnhập> - p
```

hoặc:

```
mysql –u <têndăngnhập> - p
```



Hình 4-3: *Nếu đăng nhập thành công, bạn sẽ thấy một thông điệp chào mừng như trong hình.*

Tên đăng nhập và mật khẩu còn tùy thuộc vào thiết lập cụ thể của cơ sở dữ liệu MySQL. Thông thường, các nhà cung cấp dịch vụ Web sẽ cung cấp cho bạn thông tin này.

Một lần nữa, bạn phải điều chỉnh bước này và bước trước đó để có thể truy xuất trình quản lý mysql. Trên nhiều hệ thống, mysql không thể khởi động trong thư mục mysql và trên một số hệ thống, bạn phải sử dụng `./bin/mysql –u <têndăngnhập> -p`.

4. Chọn cơ sở dữ liệu muốn làm việc (xem hình 4-4):

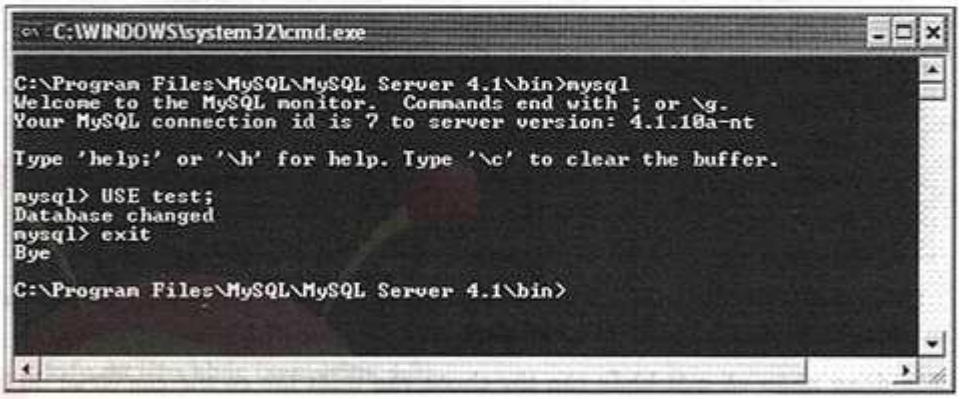
```
USE test;
```

Hình 4-4: *Lệnh USE dùng để xác định cơ sở dữ liệu muốn làm việc.*

Lệnh **USE** báo cho MySQL biết cơ sở dữ liệu muốn làm việc, kể từ thời điểm hiện tại (tránh phải nhập tên cơ sở dữ liệu nhiều lần). Cơ sở dữ liệu *test* là một trong hai cơ sở dữ liệu mà mã kịch bản **mysql_install_db** (được thực hiện trong quá trình cài đặt) tạo ra. Giả sử cơ sở dữ liệu này có mặt trên máy chủ, khi đó, tất cả mọi người đều có thể truy cập được nó.

5. Thoát khỏi mysql (xem hình 4-5):

```
exit
```



Hình 4-5: *Cả hai lệnh **exit** và **quit** sẽ kết thúc phiên làm việc với mysql.*

Cũng có thể sử dụng lệnh **quit** để thoát khỏi trình theo dõi mysql. Chú ý, không giống với các bước khác, bước này không cần dấu chấm phẩy ở cuối câu lệnh.

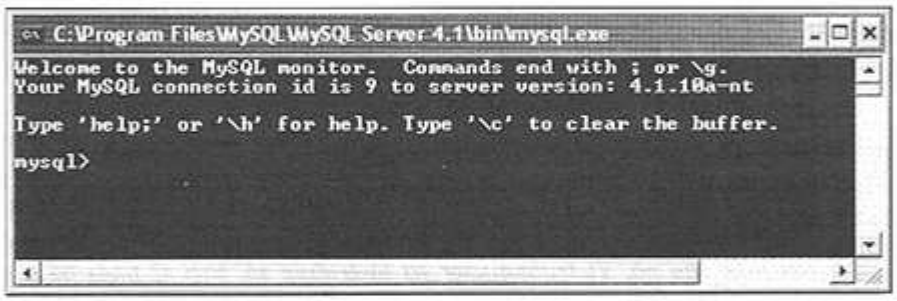
Nếu biết trước cơ sở dữ liệu sẽ làm việc, bạn chỉ việc xác định nó khi khởi động mysql:

```
mysql -u <tênđăngnhập> -p <tên cơ sở dữ liệu>
```

- *Để xem có thể làm những gì với mysql, hãy nhập vào lệnh sau:*

```
mysql --help
```

- *Trình mysql trên Unix cho phép sử dụng các phím mũi tên lên, xuống để lật qua các lệnh đã nhập vào trước đây. Điều này sẽ giúp bạn đỡ mất thời gian nhập lại từng câu lệnh khi làm việc với cơ sở dữ liệu.*
- *Nếu đang nhập một câu lệnh dài và phạm phải sai lầm, hãy hủy bỏ thao tác hiện tại bằng cách nhập* `\c` *và nhấn phím Enter hoặc Return. Nếu mysql thông báo thiếu dấu nháy (đơn hoặc kép), bạn cần bổ sung dấu nháy thích hợp.*
- *Mặc dù chúng ta kết thúc câu lệnh trong mysql với dấu chấm phẩy, nhưng các câu truy vấn (được gửi từ mã kịch bản PHP đến máy chủ) lại không sử dụng dấu chấm phẩy, đây cũng là một lỗi thường gặp.*
- *Phụ thuộc vào cách MySQL được cài đặt trên hệ thống, một số người dùng Windows có thể chạy trình theo dõi mysql (xem hình 4-6) bằng cách nhấp đúp vào tập tin thực thi có trong thư mục mysql/bin.*



Hình 4-6: *Trình mysql chạy trên Windows.*

- *Để đảm bảo an toàn khi sử dụng mysql, ứng dụng cần được bắt đầu bằng tham số* `--i-am-a-dummy`*. Tham số này sẽ giới hạn những gì có thể hoặc không thể thực hiện.*

Tạo cơ sở dữ liệu và bảng

Ứng dụng đầu tiên với SQL và MySQL là tạo một cơ sở dữ liệu. Cú pháp để tạo một cơ sở dữ liệu là:

```
CREATE DATABASE <tên cơ sở dữ liệu>;
```

Thuật ngữ `CREATE` cũng được dùng để tạo bảng dữ liệu:

```
CREATE TABLE <tên bảng> (<tên cột> <mô tả>, <tên cột>
→ <mô tả>,...);
```

Trong cú pháp trên, sau khi đặt tên bảng, từng cột sẽ được định nghĩa trong dấu ngoặc. Mỗi cột và phần mô tả phải được cách nhau bằng dấu phẩy. Nếu muốn tạo chỉ mục tại thời điểm này (xem thêm phần *"Chỉ mục và khóa"*), bạn có thể bổ sung chúng ở cuối của câu lệnh tạo bảng, tuy nhiên cũng có thể tạo chỉ mục sau này.

Chỉ mục và khóa

Hai khái niệm khá thân thuộc với thiết kế cơ sở dữ liệu là chỉ mục và khóa.

Chỉ mục (index) thông báo cho chương trình quản lý cơ sở dữ liệu chú ý đến giá trị của một cột hoặc một tổ hợp các cột cụ thể. Điều này sẽ cải thiện khả năng vận hành khi lấy các mẩu tin, tuy có trở ngại đôi chút khi chèn hoặc cập nhật chúng.

Khóa (key) trong bảng dữ liệu sẽ giúp đơn giản hóa quá trình thiết kế các cơ sở dữ liệu phức tạp. Có hai dạng khóa: khóa chính (primary key) và khóa ngoại (foreign key). Mỗi bảng nên có một khóa chính (bạn sẽ tìm hiểu điều này trong chương kế tiếp) và khóa chính trong bảng này thường được liên kết như một khóa ngoại trong một bảng khác.

Khóa chính của bảng là một cách tham chiếu đến mẩu tin theo các quy tắc sau:

- *Phải có giá trị.*
- *Giá trị không bao giờ thay đổi.*
- *Giá trị đó phải là duy nhất ứng với mỗi mẩu tin trong bảng.*

Trong bảng users, trường user_id được thiết kế là ***PRIMARY KEY****. Điều này sẽ cho biết cả phần mô tả của trường và báo cho MySQL biết để tạo chỉ mục cho nó. Vì trường user_id là trường số, bạn sẽ phải bổ sung mô tả* ***AUTO_INCREMENT*** *cho cột. Mô tả này báo cho MySQL biết để sử dụng con số kế tiếp làm giá trị user_id cho mỗi mẩu tin được thêm vào. Bạn sẽ thấy ý nghĩa của nó trong phần thực hành khi bắt đầu chèn các mẩu tin.*

Chương 5 "SQL và MySQL nâng cao" sẽ đề cập chi tiết về chỉ mục, khóa và thiết kế cơ sở dữ liệu.

Thực hành tạo cơ sở dữ liệu và bảng theo các bước sau:

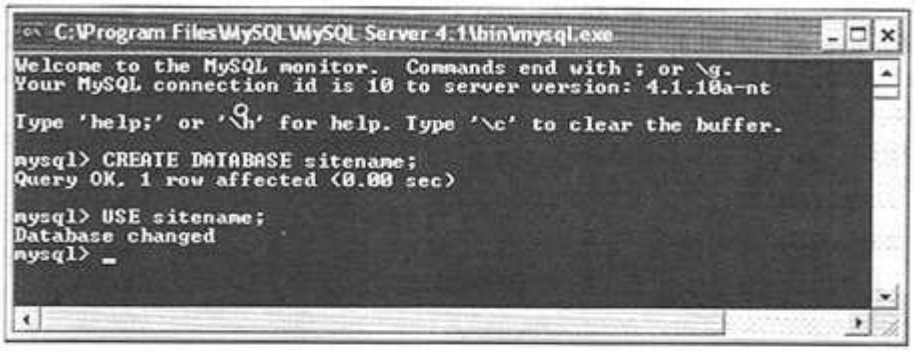
1. Truy xuất trình quản lý mysql:

```
mysql -u <tên đăng nhập> -p
```

Trong phần còn lại của cuốn sách, tất cả các câu lệnh SQL sẽ được đưa vào bằng mysql. Sử dụng các bước như ở phần trước của chương này, mở mysql, nhập vào thông tin thích hợp (tên đăng nhập, mật khẩu), tương ứng với cấu hình và hệ thống hiện hành. Bạn cũng có thể sử dụng phpMyAdmin hoặc một công cụ giao tiếp khác.

2. Tạo và chọn cơ sở dữ liệu mới (xem hình 4-7):

```
CREATE DATABASE sitename;

USE sitename;
```

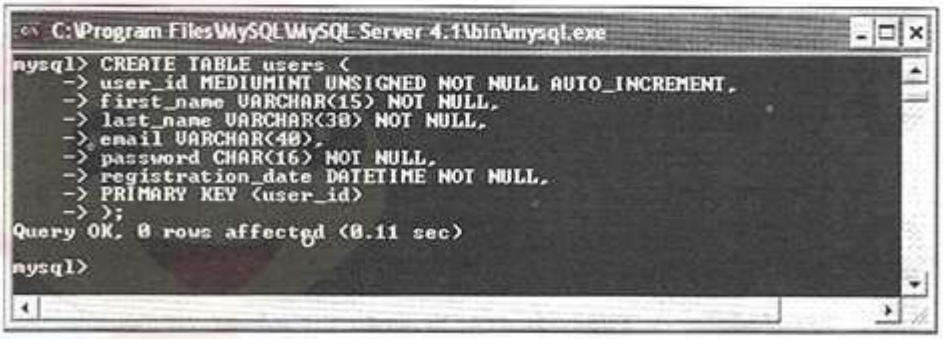


Hình 4-7: *Cơ sở dữ liệu sitename sẽ được sử dụng trong phần còn lại của cuốn sách.*

Dòng lệnh đầu tiên sẽ tạo cơ sở dữ liệu (giả sử bạn đã đăng nhập vào mysql với quyền được phép tạo cơ sở dữ liệu). Dòng lệnh thứ hai báo cho MySQL biết cơ sở dữ liệu sẽ sử dụng từ thời điểm này. Chú ý, trong chế độ làm việc của mysql, bạn cần kết thúc câu lệnh bằng dấu chấm phẩy (;).

3. Tạo bảng users (xem hình 4-8):

```
CREATE TABLE users (
user_id MEDIUMINT UNSIGNED NOT NULL AUTO_INCREMENT,
first_name VARCHAR(15) NOT NULL,
last_name VARCHAR(30) NOT NULL,
email VARCHAR(40),
password CHAR(16) NOT NULL,
registration_date DATETIME NOT NULL,
PRIMARY KEY (user_id)
);
```



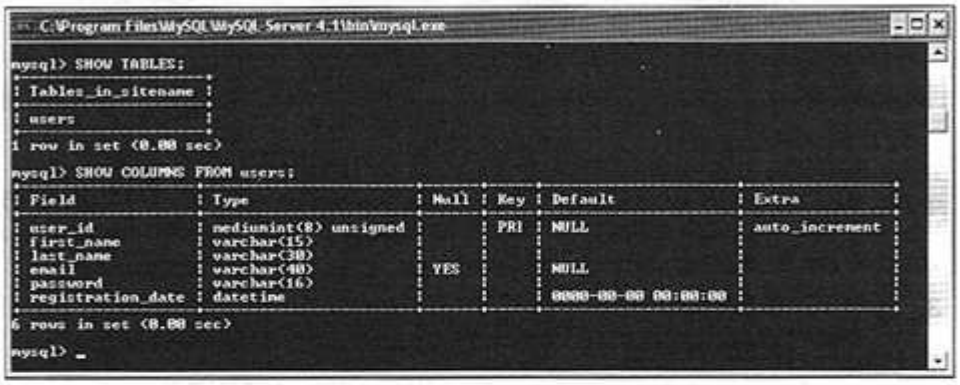
Hình 4-8: *Dùng câu lệnh* ***SQL CREATE*** *để tạo bảng users.*

Bước này định nghĩa cấu trúc cho bảng users và tích hợp nó trong cú pháp của lệnh tạo bảng. Thứ tự nhập các cột sẽ cho biết thứ tự các trường sẽ xuất hiện trong bảng.

Vì trình quản lý mysql sẽ không chạy câu lệnh nếu nó chưa bắt gặp dấu chấm phẩy, bạn có thể nhập các câu lệnh trên nhiều dòng như trong hình 4-8 (bằng cách nhấn phím Enter hoặc Return ở cuối mỗi dòng). Điều này sẽ làm câu lệnh dễ đọc và dễ dò lỗi.

4. Kiểm tra sự tồn tại của bảng (xem hình 4-9):

```
SHOW TABLES;
SHOW COLUMNS FROM users;
```



Hình 4-9: *Có thể kiểm tra sự tồn tại và bố cục của bảng bằng lệnh* ***SHOW***.

Câu lệnh **SHOW** sẽ cho biết các bảng trong một cơ sở dữ liệu hoặc các cột và kiểu của chúng trong một bảng.

Trong phần còn lại của chương này, chúng tôi giả sử bạn sử dụng công cụ mysql (hoặc một công cụ tương đương), đã đăng nhập và chọn cơ sở dữ liệu sitename với lệnh ***USE***.

Mặc dù SQL không phân biệt chữ hoa và chữ thường, nhưng bạn nên tạo thói quen viết hoa các từ khóa SQL. Điều này giúp cho việc phân biệt chúng với cơ sở dữ liệu, bảng và các cột được dễ dàng.

Khi tạo một bảng, bạn có tùy chọn xác định một kiểu, với MyISAM, BDB, InnoDB, Temporary và HEAP là những kiểu phổ biến nhất. Nếu không xác định kiểu cho bảng, MySQL sẽ tự động tạo bảng theo kiểu mặc định, thường là MyISAM (một phiên bản nâng cấp của ISAM).

Cũng có thể tạo cơ sở dữ liệu (nhưng không tạo được bảng) bằng ứng dụng mysqladmin (đi kèm MySQL) với cú pháp sau:

```
mysqladmin -u root -p create <tên cơ sở dữ liệu>
```

DESCRIBE ***<tên bảng>****, là một tên gọi khác của câu lệnh* ***SHOW COLUMNS FROM <tên bảng>;***

Chèn mẩu tin

Sau khi tạo cơ sở dữ liệu và các bảng, chúng ta bắt đầu đưa dữ liệu vào bằng lệnh **INSERT**. Có hai định dạng để chèn dữ liệu, với dạng đầu, chúng ta sẽ xác định các cột được sử dụng:

```
INSERT INTO <bảng> (<cột 1>, <cột 2>, ...)
→ VALUES ('giá trị 1', 'giá trị 2');
INSERT INTO <bảng> (<cột 4>, <cột 8>, ...)
→ VALUES ('giá trị X', 'giá trị Y');
```

Cấu trúc trên sẽ giúp bạn thêm các dòng của các mẩu tin và dữ liệu chỉ được đưa vào các cột được đề cập. Kết quả là những cột không được đề cập sẽ có giá trị **NULL** (hoặc một giá trị mặc định nếu giá trị này được định nghĩa trước). Chú ý, nếu một cột không thể có giá trị **NULL** (nó được thiết lập là **NOT NULL**), việc không xác định giá trị cho cột đó sẽ gây ra lỗi.

Định dạng thứ hai để chèn các mẩu tin sẽ không xác định cột nhưng lại xác định tất cả các giá trị cho từng mẩu tin.

```
INSERT INTO <bảng> VALUES ('giá trị 1', 'giá trị 2', ...);
```

Nếu sử dụng định dạng thứ hai, bạn phải xác định một giá trị (ngay cả khi nó là **NULL**) cho từng cột trong bảng. Nếu có sáu cột trong bảng, bạn phải liệt kê sáu giá trị. Khi số giá trị không khớp với số cột sẽ gây ra lỗi. Chính vì điều này, định dạng thứ nhất thường được sử dụng nhiều hơn.

MySQL cũng cho phép chèn đồng thời nhiều mẩu tin bằng cách dùng dấu phẩy để phân cách các mẩu tin:

```
INSERT INTO <bảng> (<cột 1>, < cột 2>) VALUES ('giá trị 1',
→ 'giá trị 2'), VALUES ('giá trị A', 'giá trị B'),
→ VALUES ('giá trị X', 'giá trị Y');
```

Điều này có thể thực hiện với MySQL, nhưng không được chuẩn ANSI SQL2 chấp nhận.

Hai hàm của MySQL

Chúng ta sẽ làm quen với hai hàm của MySQL: **PASSWORD()** và **NOW()** (các hàm này sẽ được đề cập chi tiết trong Chương 5 "SQL và MySQL nâng cao").

Hàm **PASSWORD()** được dùng để mã hóa dữ liệu (như mật khẩu) khi đưa chúng vào MySQL. Hàm này tạo chuỗi mã hóa có chiều dài cố định là 16 ký tự (đó là lý do tại sao chúng ta thiết lập cột password của bảng users là **CHAR(16)**). Trong các phiên bản sau này, kết quả của hàm **PASSWORD** lên đến 41 ký tự. **PASSWORD()** là kỹ thuật mã hóa một chiều và không thể đảo ngược. Nó rất hữu dụng trong việc lưu giữ các thông tin "nhạy cảm", không cần xem hoặc giải mã trở lại. Tuy nhiên, nó lại không thích hợp để lưu giữ các thông tin mà bạn muốn xem lại sau này (chẳng hạn như số thẻ tín dụng).

Hàm NOW() rất tiện lợi cho các cột ngày, giờ và timestamp vì nó sẽ chèn vào ngày, giờ hiện tại cho cột.

Khi sử dụng hàm bất kỳ trong câu lệnh SQL, đừng đặt chúng trong dấu nháy. Cũng không được đặt khoảng trống giữa hàm và các dấu ngoặc đi sau.

Thực hành chèn dữ liệu vào bảng theo các bước sau:

1. Chèn một mẩu tin mới vào trong bảng users (xem hình 4-10).

Có thể sử dụng một trong hai cú pháp sau:

```
INSERT INTO users (first_name, last_name, email, password,
→ registration_date) VALUES ('John', 'Ullman',
→ 'john@domain.com',PASSWORD('password'), NOW());
```

hoặc:

```
INSERT INTO users VALUES (NULL, 'John', 'Ullman',
→ 'john@domain.com',PASSWORD('password'), NOW());
```

Hình 4-10: *Câu lệnh này sẽ chèn một mẩu tin vào trong bảng users.*

Xin nhắc lại, cú pháp thứ nhất (xác định cụ thể các cột) rõ ràng hơn nhưng không phải lúc nào cũng là tiện lợi. Trong cả hai trường hợp, chúng ta sử dụng hàm NOW() để thiết lập giá trị trường registration_date tại thời điểm chèn dữ liệu (chú ý, hàm không được đặt trong dấu nháy). Giá trị trường user_id là NULL và điều này sẽ buộc MySQL sử dụng giá trị số kế tiếp cho nó, vì trong phần tạo bảng, trường này được mô tả là AUTO_INCREMENT.

2. Chèn một số mẩu tin vào trong bảng users (xem hình 4-11):

```
INSERT INTO users (first_name, last_name, email, password,
→ registration_date) VALUES ('John', 'Lennon',
→ 'john@beatles.com', PASSWORD('Happin3ss'), NOW()),
→ ('Paul', 'McCartney', 'paul@beatles.com',
→ PASSWORD('letITbe'), NOW()),
→ ('George', 'Harrison', 'george@beatles.com ',
→ PASSWORD('something'), NOW()),
→ ('Ringo', 'Starr', 'ringo@beatles.com',
→ PASSWORD('thisboy'), NOW());
```

Hình 4-11: *Câu lệnh này (được hỗ trợ trong MySQL nhưng không được hỗ trợ trong các cơ sở dữ liệu SQL khác) sẽ chèn đồng thời một số mẩu tin vào bảng.*

Vì MySQL cho phép chèn đồng thời nhiều mẩu tin vào bảng, bạn có thể tận dụng ưu điểm này để đưa các mẩu tin vào bảng.

3. Tiếp tục bước 1 và 2 cho đến khi các dữ liệu được đưa hết vào bảng users.

Trong phần còn lại của chương này, các câu truy vấn sẽ được thực hiện dựa trên các mẩu tin đã đưa vào trước đây trong cơ sở dữ liệu. Nếu cơ sở dữ liệu của bạn không giống cơ sở dữ liệu của chúng tôi, hãy thay đổi các chi tiết trong câu lệnh cho phù hợp. Phần cơ bản của các câu truy vấn sẽ được áp dụng mà không phụ thuộc vào dữ liệu, vì cơ sở dữ liệu *sitename* đã thiết lập cấu trúc bảng và cột.

Nếu muốn chèn một giá trị có chứa một dấu nháy đơn, hãy mã hóa escape nó với một dấu chéo ngược, ví dụ:

```
INSERT INTO users (last_name, first_name)
→ VALUES ('O\'Malley','Juan');
```

Các khoảng trắng ở cuối giá trị khi được chèn vào các cột **VARCHAR** *sẽ được tự động cắt bỏ bởi MySQL. Đây cũng là một khác biệt giữa MySQL và chuẩn ANSI SQL2.*

Thuật ngữ **INTO** *trong câu lệnh* **INSERT** *là tùy chọn trong các phiên bản hiện tại của MySQL.*

Một quy tắc cần nhớ là các chuỗi phải đặt trong cặp dấu nháy đơn, còn các giá trị số được nhập vào sẽ không cần đặt trong dấu nháy (tương tự cho **NULL** *và lời gọi hàm). Điều này áp dụng cho mọi câu lệnh SQL, không chỉ riêng cho câu lệnh* **INSERT***.*

Truy xuất dữ liệu

Bây giờ, cơ sở dữ liệu đã có một số mẩu tin, chúng ta có thể lấy thông tin từ cơ sở dữ liệu bằng lệnh `SELECT` (một câu lệnh SQL thường dùng nhất). Dạng câu truy vấn này trả lại một tập các mẩu tin dựa trên những điều kiện nhất định, bằng cú pháp sau:

```
SELECT <danh sách cột> FROM <tên bảng>
```

Câu truy vấn `SELECT` đơn giản nhất là:

```
SELECT * FROM <tên bảng>;
```

Dấu hoa thị có nghĩa là lấy ra tất cả các cột có trong bảng. Một lựa chọn khác là xác định cụ thể các cột cần lấy ra với một danh sách cột cách nhau bằng dấu phẩy.

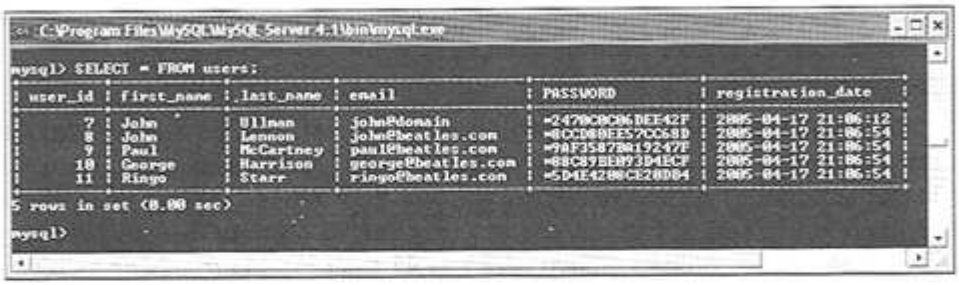
```
SELECT user_id, first_name, last_name FROM users;
```

Có một số ưu điểm trong việc xác định cụ thể các cột cần lấy ra. Đầu tiên là khả năng vận hành: Không có lý do gì lại phải lấy ra những thứ không cần. Thứ hai là thứ tự: Có thể lấy ra các trường theo một trật tự khác với trật tự của chúng trong bố cục bảng. Thứ ba, cho phép thao tác các giá trị trong các cột bằng cách sử dụng các hàm (điều này sẽ được giới thiệu trong Chương 5 "SQL và MySQL nâng cao").

Thực hành truy xuất dữ liệu từ một bảng theo các bước sau:

1. Lấy ra tất cả dữ liệu từ bảng users (xem hình 4-12):

```
SELECT * FROM users;
```

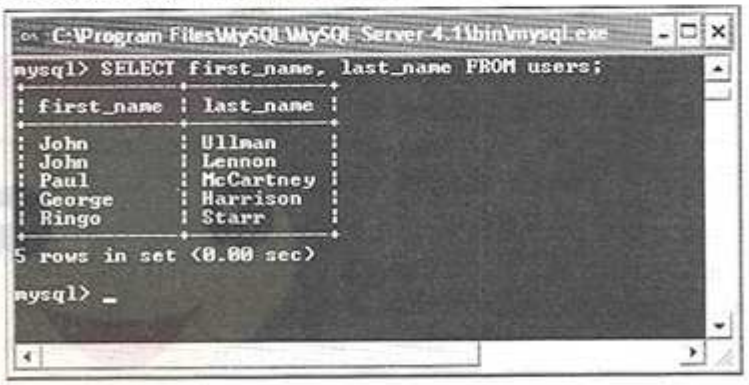


Hình 4-12: *Câu truy vấn* ***SELECT * FROM*** *users sẽ trả lại tất cả các mẩu tin có trong bảng users.*

Câu lệnh SQL rất cơ bản này sẽ lấy ra tất cả các cột của tất cả các mẩu tin có trong bảng dữ liệu.

2. Lấy ra trường first_name và last_name của bảng users (xem hình 4-13).

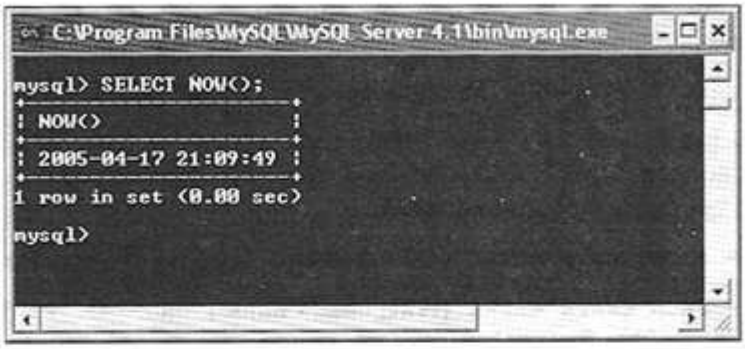
```
SELECT first_name, last_name FROM users;
```



Hình 4-13: *Câu hỏi truy vấn này sẽ lấy ra hai cột của tất cả các mẩu tin.*

Thay vì thể hiện tất cả dữ liệu của toàn bộ các trường trong bảng users, bạn có thể sử dụng câu lệnh **SELECT** để giới hạn các thông tin mà mình cần.

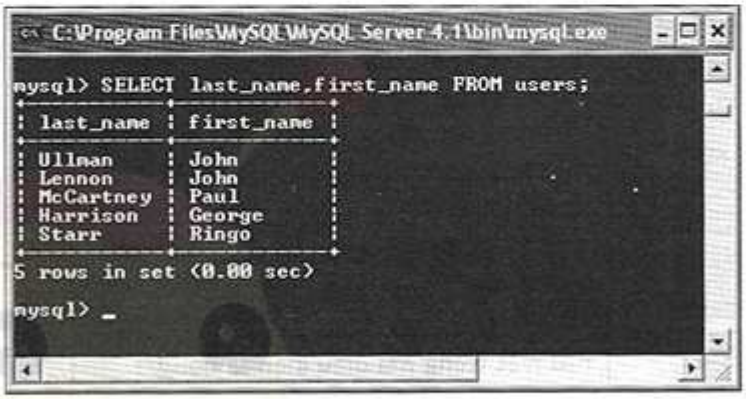
*Cũng có thể sử dụng câu lệnh **SELECT** mà không cần xác định cột hoặc bảng dữ liệu. Ví dụ: **SELECT NOW()**. Xem hình 4-14.*



Hình 4-14: *Nhiều câu truy vấn có thể được thực hiện mà không cần xác định cơ sở dữ liệu hoặc bảng. Câu truy vấn này trả lại ngày và giờ hiện tại.*

*Thứ tự các cột được liệt kê trong câu lệnh **SELECT** (giả sử bạn không lấy ra tất cả) cho biết thứ tự trả về của các giá trị (so sánh hình 4-13 và hình 4-15).*

*Với lệnh **SELECT**, có thể lấy ra nhiều lần từ một cột, điều này cho phép thao tác cột theo nhiều cách khác nhau.*



Hình 4-15: *Các cột thể hiện theo thứ tự được nêu trong câu truy vấn.*

Sử dụng điều kiện

Cho tới lúc này, câu lệnh **SELECT** được sử dụng để tự động lấy ra tất cả các mẩu tin. Sẽ không có vấn đề gì nếu chỉ làm việc với vài mẩu tin, nhưng sẽ gây ra nhiều trở ngại về khả năng vận hành của cơ sở dữ liệu khi số mẩu tin lớn. Để cải thiện tính hiệu quả của câu lệnh **SELECT**, bạn có thể sử dụng một số điều kiện

với khả năng kết hợp gần như là vô tận. Các điều kiện này sử dụng thuật ngữ của SQL là **WHERE** và được viết giống như một điều kiện trong PHP.

```
SELECT * FROM <tên bảng> WHERE <cột>='giá trị';
```

```
SELECT email FROM users WHERE last_name='Lennon';
```

```
SELECT name FROM people WHERE birth_date='2005-04-16';
```

Bảng 4.6 liệt kê các toán tử phổ biến nhất, có thể sử dụng trong điều kiện **WHERE**. Các toán tử này có thể được sử dụng phối hợp với nhau, với các dấu ngoặc, để tạo ra các biểu thức phức tạp.

```
SELECT * FROM users WHERE (user_id>=10) AND (user_id<=20);
```

```
SELECT * FROM users WHERE (last_name='Bank') OR
→ (last_name='Banks');
```

Bảng 4.6: *Các toán tử MySQL này thường được dùng với các biểu thức trong mệnh đề* ***WHERE***

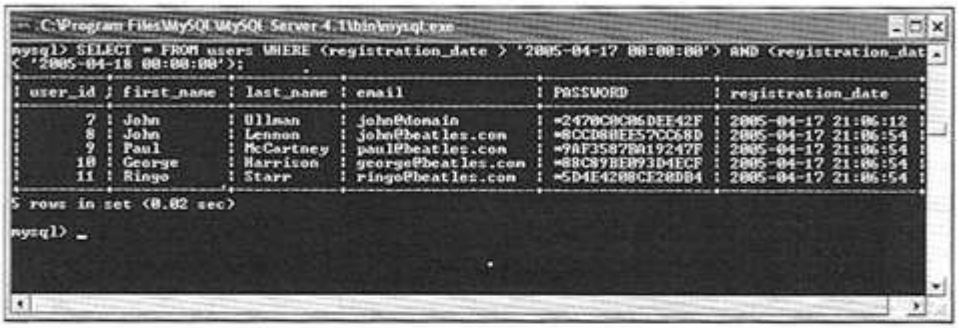
Toán tử	Ý nghĩa
=	Bằng.
<	Nhỏ hơn.
>	Lớn hơn.
<=	Nhỏ hơn hoặc bằng.
>=	Lớn hơn hoặc bằng.
!=	Không bằng.
IS NOT NULL	Có giá trị.
IS NULL	Không có giá trị.
BETWEEN	Trong phạm vi.
NOT BETWEEN	Không nằm trong phạm vi.
OR (hoặc ||)	Khi một trong hai điều kiện là đúng.
AND (hoặc &&)	Khi cả hai điều kiện cùng đúng.
NOT (hoặc !)	Khi điều kiện không đúng.

Để minh họa cách sử dụng các điều kiện, chúng ta sẽ lấy ra các dữ liệu cụ thể từ cơ sở dữ liệu *sitename*. Các ví dụ dưới đây chỉ minh họa một số trường hợp, bạn sẽ thấy các dạng khác của điều kiện với câu lệnh **SELECT** trong chương này và phần còn lại của cuốn sách.

Thực hành sử dụng điều kiện theo các bước sau:

1. Lấy ra các mẩu tin ứng với những ai đăng ký vào một ngày cụ thể (xem hình 4-16):

```
SELECT * FROM users WHERE (registration_date >
→ '2005-04-17 00:00:00') AND (registration_date <
→ '2005-04-18 00:00:00');
```



Hình 4-16: *Sử dụng mệnh đề* **WHERE** *với hai biểu thức và toán tử* **AND** *để ta lấy ra các mẩu tin theo yêu cầu.*

Để lấy thông tin người dùng đăng ký vào ngày 17-04-2005, chúng ta lấy các mẩu tin có thời gian đăng ký từ nửa đêm ngày 17-04-2005 cho đến nửa đêm ngày 18-04-2005 (có thể sử dụng toán tử **BETWEEN** trong trường hợp này). Nếu trường registration_date có dạng **DATE** (dạng YYYY-MM-DD), chúng ta có thể sử dụng toán tử bằng: **registration_date = '2005-04-17';**

2. Lấy tất cả trường first_name của những ai có trường last_name là **Lennon** (xem hình 4-17):

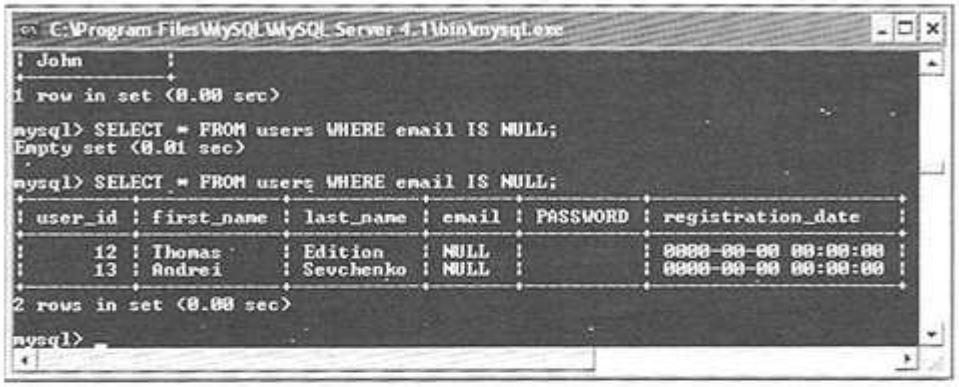
```
SELECT first_name FROM users WHERE last_name = 'Lennon';
```

Hình 4-17: *Lấy ra tất cả những ai có last_name là* **Lennon**

Ở đây, ta chỉ lấy ra một trường ứng với mỗi mẩu tin (first_name). Bản thân các mẩu tin trả lại được xác định bởi giá trị của một trường khác (last_name).

3. Trong bảng users, lấy tất cả các trường từ các mẩu tin không có địa chỉ thư điện tử (xem hình 4-18):

```
SELECT * FROM users WHERE email IS NULL;
```



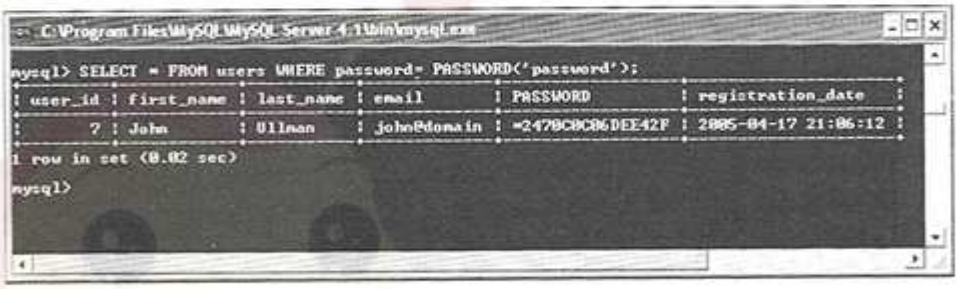
Hình 4-18: *Tất cả các mẩu tin không có địa chỉ thư điện tử.*

Điều kiện `IS NULL` có ý nghĩa là "không có giá trị". Chú ý, một chuỗi rỗng vẫn có giá trị (hiểu theo thuật ngữ `NULL`), vì thế nó sẽ không đáp ứng điều kiện này. Tuy nhiên, chúng sẽ đáp ứng điều kiện trong câu lệnh sau:

```
SELECT * FROM users WHERE email='';
```

4. Lấy ra các mẩu tin có trường password giống như mật khẩu được cung cấp (xem hình 4-19):

```
SELECT * FROM users WHERE password= PASSWORD('password');
```



Hình 4-19: *Sử dụng hàm `PASSWORD()` để kiểm tra các giá trị đã được mã hóa trong cơ sở dữ liệu.*

Vì các mật khẩu đã được mã hóa với hàm `PASSWORD()`, chúng ta sẽ lấy ra các mẩu tin đáp ứng bằng cách so sánh giá trị đã được mã hóa với kết quả của hàm `PASSWORD('mật khẩu')`. Hàm `PASSWORD()` có phân biệt chữ hoa và chữ thường nên câu truy vấn này chỉ lấy ra các mẩu tin có mật khẩu giống như giá trị được cung cấp.

Không nhất thiết phải lấy ra cột sử dụng trong mệnh đề `WHERE`:

```
SELECT user_id FROM users WHERE first_name='John';
```

Các cột được liệt kê sau từ SELECT cho biết giá trị nào sẽ được trả lại. Các cột được liệt kê trong mệnh đề WHERE cho biết giá trị nào sẽ dùng làm cơ sở để lấy dữ liệu.

Có thể sử dụng toán tử `IN` *và* `NOT IN` *để xác định giá trị của cột được hoặc không được liệt kê trong một tập giá trị:*

```
SELECT * FROM people WHERE birth_date
→ IN ('2003-01-05', '2003-04-30', 2004-01-05'');
```

Có thể thực hiện các phép tính toán học trong câu truy vấn bằng các phép cộng (+), trừ (-), nhân (*) và chia (/).

Mặc dù các minh họa trên đây chỉ sử dụng điều kiện với câu lệnh `SELECT`*, nhưng chúng cũng sẽ được dùng trong nhiều dạng câu truy vấn khác (bạn sẽ thấy trong phần sau của chương này).*

Sử dụng LIKE và NOT LIKE

Sẽ tương đối đơn giản nếu sử dụng số, ngày và `NULL` trong các điều kiện, nhưng với chuỗi thì phức tạp hơn. Bạn có thể kiểm tra một chuỗi có bằng với một giá trị xác định hay không:

```
SELECT * FROM users WHERE user_name= 'trout';
```

Tuy nhiên, việc so sánh chuỗi theo những cách thức khác sẽ đòi hỏi sử dụng một số ký tự và toán tử nữa. Ví dụ, nếu muốn tìm kiếm những ai có last_name là Smith, Smiths hoặc Smithson, bạn sẽ cần một điều kiện mềm dẻo và linh động hơn. Đây chính là lúc mà `LIKE` và `NOT LIKE` thể hiện vai trò của mình. Các toán tử này (thường dùng với chuỗi) được sử dụng với hai ký tự đại diện (wildcard): ký tự gạch dưới (_) ứng với một ký tự bất kỳ và ký tự phần trăm (`%`) ứng với `0` hoặc nhiều ký tự. Trong ví dụ trên, có thể sử dụng câu truy vấn sau:

```
SELECT * FROM users WHERE last_name LIKE 'Smith%';
```

Câu truy vấn này thực hiện việc tìm kiếm trên tất cả các cột mà giá trị của trường last_name bắt đầu bằng Smith. Việc tìm kiếm theo mặc định sẽ không phân biệt chữ hoa và chữ thường nên câu truy vấn lấy ra các mẩu tin bắt đầu với smith, SMITH... trong trường last_name.

Thực hành sử dụng `LIKE` theo các bước sau:

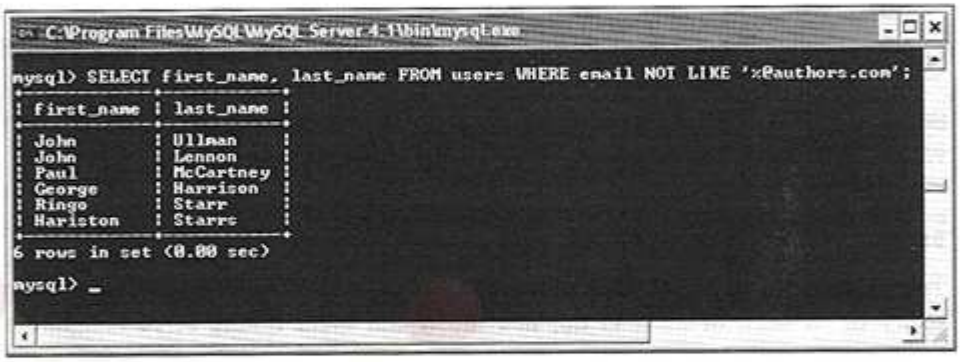
1. Lấy ra tất cả các mẩu tin mà trong đó trường last_name bắt đầu bằng Star (xem hình 4-20):

```
SELECT * FROM users WHERE last_name LIKE 'Star%';
```

Hình 4-20: *Thuật ngữ LIKE của SQL cho phép tạo các điều kiện có tính linh động cao.*

2. Lấy ra phần tên của tất cả các mẩu tin mà địa chỉ thư điện tử không có dạng *something@authors.com* (xem hình 4-21).

```
SELECT first_name, last_name FROM users WHERE email
→ NOT LIKE '%@authors.com';
```



Hình 4-21: *Điều kiện **NOT LIKE** trả lại các mẩu tin dựa trên các giá trị không giống với giá trị được đưa vào.*

Nếu muốn loại trừ một số khả năng, chúng ta sẽ sử dụng **NOT LIKE** với các ký tự đại diện.

*Các câu truy vấn với điều kiện **LIKE** thường chậm hơn, vì chúng không tận dụng được lợi thế của các chỉ mục, bạn nên thận trọng khi sử dụng chúng.*

Các ký tự đại diện có thể sử dụng ở trước và (hoặc) sau một chuỗi trong các câu truy vấn.

```
SELECT * FROM users WHERE user_name= '_Smith%';
```

*Mặc dù **LIKE** và **NOT LIKE** thường được sử dụng với chuỗi, nhưng cũng có thể áp dụng chúng với các cột số.*

*Để sử dụng ký tự gạch dưới và ký tự phần trăm trong câu truy vấn có sử dụng **LIKE** hoặc **NOT LIKE**, chúng phải được mã hóa escape (bằng cách thêm vào một dấu chéo ngược trước ký tự) để không bị nhầm với các ký tự đại diện.*

Ký tự gạch dưới có thể được sử dụng ở dạng tổ hợp với chính nó, ví dụ `LIKE '__'` *sẽ tìm một tổ hợp gồm hai ký tự bất kỳ.*

Sắp xếp kết quả truy vấn

Trong khi điều kiện `WHERE` giới hạn dữ liệu nào sẽ được trả về, mệnh đề `ORDER BY` sẽ ảnh hưởng đến cách dữ liệu trả về. Cũng giống như việc liệt kê các cột trong bảng sẽ ảnh hưởng đến thứ tự các cột trong kết quả trả về (so sánh hình 4-13 và 4-15), `ORDER BY` tác động đến toàn bộ danh sách. Khi không xác định cụ thể trật tự sắp xếp, dữ liệu trả về sẽ thể hiện theo cách thức không thể đoán trước (thường là sắp xếp theo thứ tự của khóa chính với chiều tăng dần).

```
SELECT * FROM <tên bảng> ORDER BY <cột>;
```

```
SELECT email FROM users ORDER BY registration_date;
```

Thứ tự mặc định khi sử dụng `ORDER BY` là tăng dần (viết tắt là `ASC`), các con số sẽ tăng từ nhỏ tới lớn và ngày sẽ đi từ quá khứ đến hiện tại. Đảo ngược thứ tự này bằng cách xác định `DESC`.

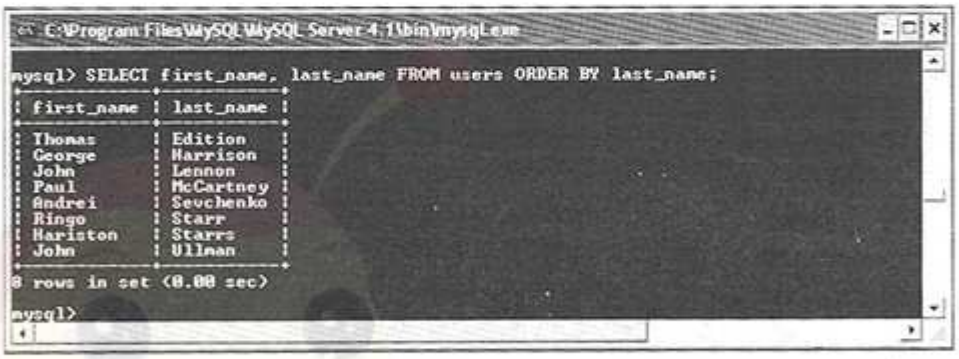
```
SELECT email FROM users ORDER BY registration_date DESC;
```

Thậm chí, bạn có thể sắp xếp thứ tự các giá trị trả về theo nhiều cột như ở ví dụ sau.

Thực hành sắp xếp dữ liệu theo các bước dưới đây:

1. Lấy ra tất cả các mẩu tin, sắp xếp theo thứ tự chữ cái trong trường last_name (xem hình 4-22).

   ```
   SELECT first_name, last_name FROM users ORDER BY last_name;
   ```

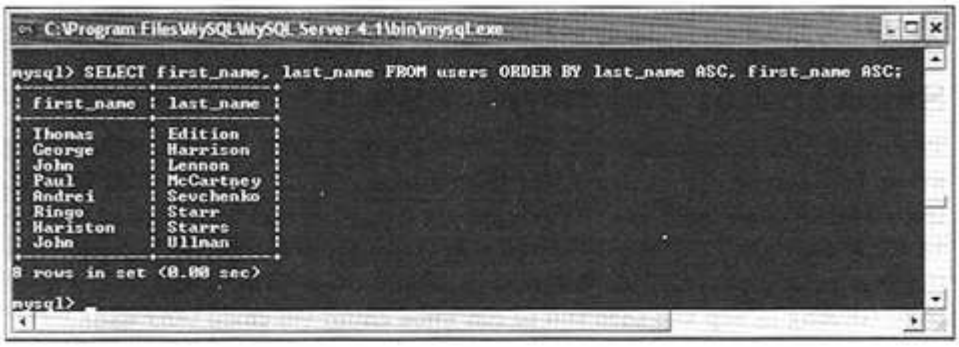


Hình 4-22: *Các mẩu tin được sắp xếp theo thứ tự chữ cái trong trường last_name.*

Nếu so sánh kết quả này với các kết quả trong hình 4-13, bạn sẽ thấy ưu điểm của việc sử dụng `ORDER BY`.

2. Hiển thị tất cả người dùng theo thứ tự chữ cái trong trường last_name và first_name (xem hình 4-23):

   ```
   SELECT first_name, last_name FROM users ORDER BY last_name
   → ASC, first_name ASC;
   ```



Hình 4-23: *Các mẩu tin được sắp xếp theo thứ tự chữ cái, đầu tiên là theo last_name, sau đó là theo first_name.*

Trong câu truy vấn này, kết quả trả về sẽ được sắp xếp trước hết theo last_name và sau đó là theo first_name (trong các mẩu tin có cùng last_name). Bạn sẽ thấy kết quả rõ ràng trong các mẩu tin có cùng last_name.

3. Thể hiện tất cả người dùng theo ngày đăng ký (xem hình 4-24):

```
SELECT * FROM users ORDER BY registration_date DESC;
```

Hình 4-24: *Tất cả người dùng được thể hiện theo ngày đăng ký (người đăng ký gần nhất được thể hiện trước).*

Cũng có thể sử dụng **ORDER BY** trên cột bất kỳ, bao gồm cả số và ngày.

*Do bản chất của cơ sở dữ liệu, thứ tự các mẩu tin được chứa trong một bảng không tuần tự. Để việc sắp xếp kết quả trả về có ý nghĩa, bạn nên sử dụng **ORDER BY** trên một cột hợp lý (ví dụ như user_id hoặc registration_date).*

*Vì MySQL có thể làm việc với nhiều ngôn ngữ, nên **ORDER BY** sẽ được dựa trên ngôn ngữ hiện đang được sử dụng bởi cơ sở dữ liệu (mặc định là tiếng Anh).*

*Nếu cột chọn để sắp xếp có chứa giá trị **NULL**, những mẩu tin có giá trị này sẽ được xếp trước, cả trong hai dạng sắp xếp: tăng dần hoặc giảm dần.*

Nếu sử dụng `ORDER BY` *với* `WHERE` *hoặc các mệnh đề khác, bạn nên đặt* `ORDER BY` *sau các điều kiện.*

```
SELECT * FROM users WHERE registration_date
→ >= '2002-03-01' ORDER BY last_name ASC;
```

Giới hạn kết quả trả về

Một thuật ngữ SQL có thể sử dụng cho câu lệnh truy vấn là **LIMIT**. Không giống với **WHERE** (ảnh hưởng đến mẩu tin nào sẽ được trả lại) hoặc **ORDER BY** (xác định cách sắp xếp các kết quả trả về), **LIMIT** xác định số mẩu tin tối đa được trả về. Ví dụ:

```
SELECT * FROM <tên bảng> LIMIT 0,10;
SELECT * FROM <tên bảng> LIMIT 10,20;
```

Trong ví dụ đầu tiên, chỉ có mười mẩu tin đầu tiên được trả về. Trong câu lệnh thứ hai, 20 mẩu tin sẽ được trả về, bắt đầu từ mẩu tin thứ 11. Cũng giống như mảng trong PHP, các chỉ số trong cơ sở dữ liệu bắt đầu từ 0, vì thế chỉ số 10 trong mệnh đề **LIMIT** sẽ ứng với mẩu tin thứ 11.

Có thể sử dụng **LIMIT** với **WHERE** và (hoặc) **ORDER BY** bằng cách đặt nó vào cuối câu truy vấn:

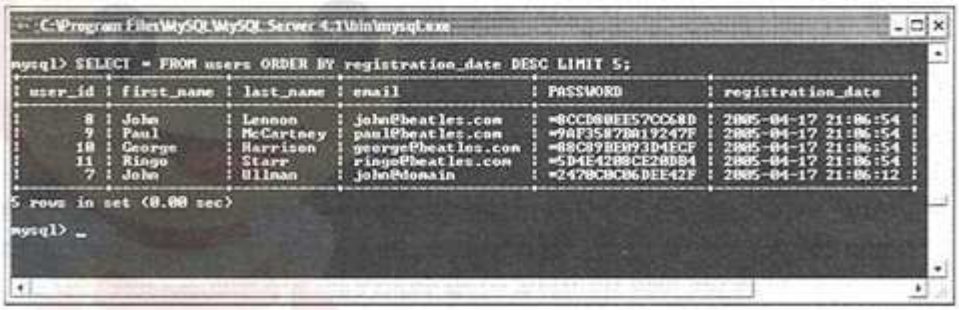
```
SELECT * FROM users WHERE last_name='Simpson' ORDER BY
→ registration_date DESC LIMIT 10;
```

Mặc dù **LIMIT** không làm giảm việc truy vấn cơ sở dữ liệu (vì nó vẫn phải lấy ra các mẩu tin, sau đó mới cắt bỏ), nhưng sẽ làm giảm thiểu lượng dữ liệu cần xử lý khi làm việc với mysql hoặc mã kịch bản PHP. Khi viết các câu truy vấn, không có lý do gì lại phải lấy ra các cột hoặc hàng không được sử dụng.

Thực hành giới hạn lượng dữ liệu trả về theo các bước sau:

1. Lấy ra năm người dùng đăng ký gần đây nhất (xem hình 4-25).

```
SELECT * FROM users ORDER BY registration_date DESC LIMIT 5;
```

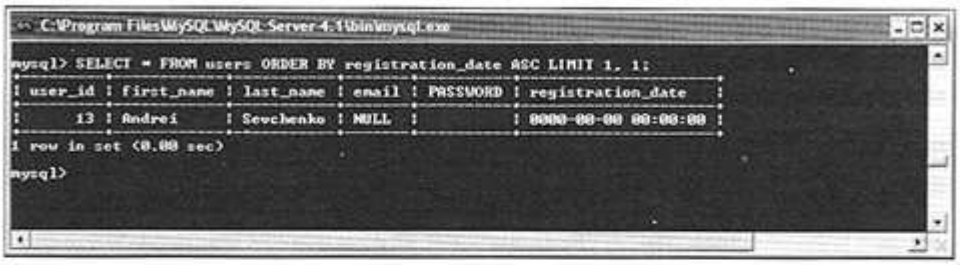


Hình 4-25: *Bằng cách sử dụng mệnh đề* `LIMIT`*, chúng ta có thể lấy ra một số lượng xác định các mẩu tin.*

Để trả lại các mẩu tin mới nhất, bạn phải sắp xếp dữ liệu theo ngày đăng ký và theo thứ tự giảm dần. Sau đó, để lấy ra 5 mẩu tin mới nhất, chúng ta sử dụng mệnh đề `LIMIT 5`.

2. Lấy ra thông tin của người đăng ký thứ hai (xem hình 4-26):

```
SELECT * FROM users ORDER BY registration_date ASC
→ LIMIT 1, 1;
```



Hình 4-26: *Với SQL, bạn có thể lấy ra các mẩu tin nằm ở khoảng giữa của một nhóm, bằng cách sử dụng định dạng* ***LIMIT x, y.***

Điều này có vẻ hơi lạ, nhưng đây chỉ là một ứng dụng hợp lý về thông tin mà bạn đang tìm hiểu. Đầu tiên, chúng ta sắp xếp các mẩu tin theo trường registration_date với chiều tăng dần, người đăng ký đầu tiên sẽ nằm ở đầu danh sách. Sau đó, giới hạn nhóm này để bắt đầu từ mẩu tin thứ hai (chỉ số 1) và chỉ lấy ra một mẩu tin.

Mệnh đề ***LIMIT x, y*** *thường được sử dụng để hiển thị nhiều trang của các câu truy vấn trong khi bạn chỉ muốn thể hiện mỗi trang với số lượng mẩu tin nhất định.*

Trong chương kế tiếp "SQL và MySQL nâng cao", bạn sẽ tìm hiểu mệnh đề cuối cùng là ***GROUP BY*** *và sử dụng nó với các câu lệnh* ***SELECT*** *của mình.*

Thuật ngữ ***LIMIT*** *không phải là thành phần của chuẩn SQL, nên nó không có hiệu lực với các hệ quản trị cơ sở dữ liệu SQL khác (thật tiếc).*

Mệnh đề ***LIMIT*** *có thể được sử dụng với hầu hết dạng câu truy vấn, không chỉ với* ***SELECT.***

Cập nhật dữ liệu

Khi bảng đã có dữ liệu, bạn sẽ có các tùy chọn dùng để thay đổi các mẩu tin. Lý do chính để thay đổi các mẩu tin là thông tin nhập vào không chính xác, hoặc khi người dùng thay đổi thông tin (như địa chỉ thư điện tử) và các thay đổi này cần được cập nhật trong cơ sở dữ liệu.

Cú pháp cập nhật các mẩu tin trong cơ sở dữ liệu là:

```
UPDATE <tên bảng> SET <cột>='giá trị';
```

Có thể thay đổi đồng thời nhiều cột của một mẩu tin bằng cách liệt kê một danh sách với các cột cách nhau bằng dấu phẩy.

```
UPDATE <tên bảng> SET <cột 1>='giá trị 1', <cột 2>='giá trị 2';
```

Thông thường, mệnh đề WHERE được sử dụng để xác định mẩu tin nào sẽ bị ảnh hưởng, nếu không, tất cả các mẩu tin sẽ được cập nhật.

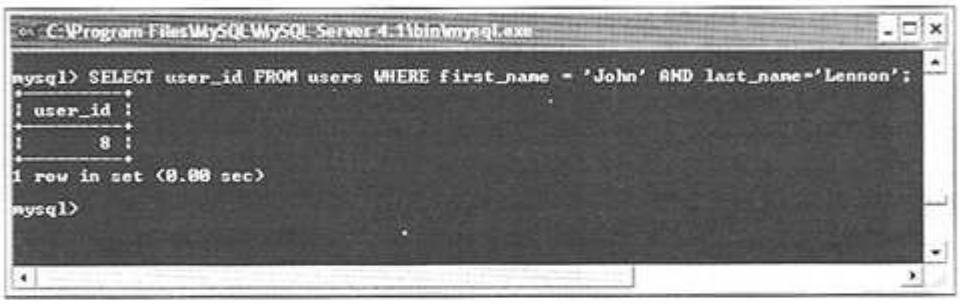
```
UPDATE <tên bảng> SET <cột 1>='giá trị 1'
→ WHERE <cột 2>='giá trị 2';
```

Cập nhật và xóa, đây là một trong những lý do quan trọng nhất cần tới khóa chính. Con số này (vốn không bao giờ thay đổi) đóng vai trò như điểm tham chiếu trong mệnh đề WHERE, ngay cả khi các trường khác bị thay đổi.

Thực hành cập nhật các mẩu tin theo các bước sau:

1. Xác định mẩu tin nào sẽ được cập nhật (xem hình 4-27):

```
SELECT user_id FROM users WHERE first_name = 'John' AND
→ last_name='Lennon';
```



Hình 4-27: *Trước khi cập nhật một mẩu tin, cần xác định khóa chính nào sẽ được sử dụng.*

Trong ví dụ, chúng ta sẽ thay đổi địa chỉ thư điện tử của người dùng. Đầu tiên, bạn phải tìm khóa chính của mẩu tin và đây là nhiệm vụ của câu lệnh truy vấn này.

2. Cập nhật mẩu tin (xem hình 4-28):

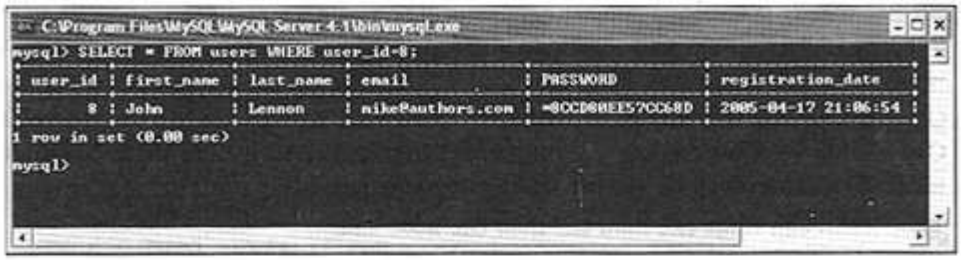
```
UPDATE users SET email='mike@authors.com' WHERE
→ user_id = 8;
```

Hình 4-28: *Câu truy vấn này làm thay đổi giá trị của một cột trong một mẩu tin.*

Để thay đổi địa chỉ thư điện tử, bạn sử dụng câu truy vấn UPDATE và xác định mẩu tin muốn thay đổi bằng khóa chính (user_id). MySQL sẽ thông báo kết quả của câu truy vấn và số lượng mẩu tin được cập nhật.

3. Kiểm tra lại kết quả cập nhật (xem hình 4-29).

```
SELECT * FROM users WHERE user_id=8;
```



Hình 4-29: *Bước cuối cùng, kiểm tra lại kết quả cập nhật bằng cách lấy lại mẩu tin.*

Mặc dù MySQL đã thông báo việc cập nhật thành công (xem lại hình 4-28), nhưng việc lấy lại mẩu tin để kiểm tra xem nó có thực sự được cập nhật hay không sẽ không phải là vô ích.

Phải thật cẩn thận với mệnh đề ***WHERE*** *mỗi khi sử dụng* ***UPDATE****, trừ khi bạn muốn thay đổi tất cả mẩu tin.*

Để tránh vô tình thay đổi quá nhiều mẩu tin, hãy sử dụng mệnh đề ***LIMIT*** *cho câu lệnh* ***UPDATE****.*

Không áp dụng lệnh ***UPDATE*** *cho cột khóa chính vì giá trị này sẽ không bao giờ thay đổi. Việc thay đổi khóa chính trong một bảng sẽ phá vỡ mối quan hệ với các bảng khác.*

Xóa dữ liệu

Bạn có thể dễ dàng xóa các dữ liệu đã có ra khỏi cơ sở dữ liệu. Điều này sẽ được thực hiện bằng lệnh **DELETE**.

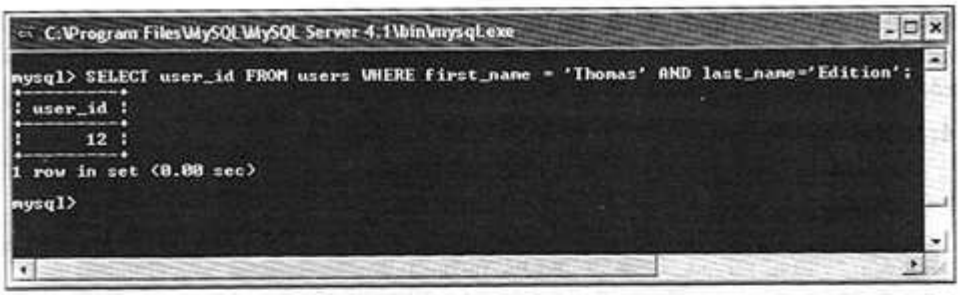
```
DELETE FROM <tên bảng> WHERE <cột>='giá trị';
```

Chú ý, khi đã xóa một mẩu tin, sẽ không có cách nào để lấy lại nó. Vì vậy, bạn nên sao lưu cơ sở dữ liệu trước khi thực hiện hành động xóa bất kỳ. Cũng nên có thói quen sử dụng mệnh đề **WHERE** khi xóa dữ liệu, nếu không sẽ vô tình xóa toàn bộ dữ liệu trong bảng. Câu truy vấn **DELETE FROM <tên bảng>** sẽ xóa trống một bảng và chỉ giữ lại cấu trúc của nó. Tương tự như **DELETE FROM**, câu lệnh **TRUNCATE TABLE <tên bảng>** xóa toàn bộ bảng (cả dữ liệu và cấu trúc của nó) và sau đó tạo lại bảng. Tuy kết quả cuối cùng như nhau, nhưng cách này (xuất phát từ Oracle) có vẻ nhanh và an toàn hơn.

Thực hành xóa dữ liệu theo các bước sau:

1. Xác định mẩu tin cần xóa (xem hình 4-30):

```
SELECT user_id FROM users WHERE first_name = 'Thomas' AND
→ last_name='Edition';
```

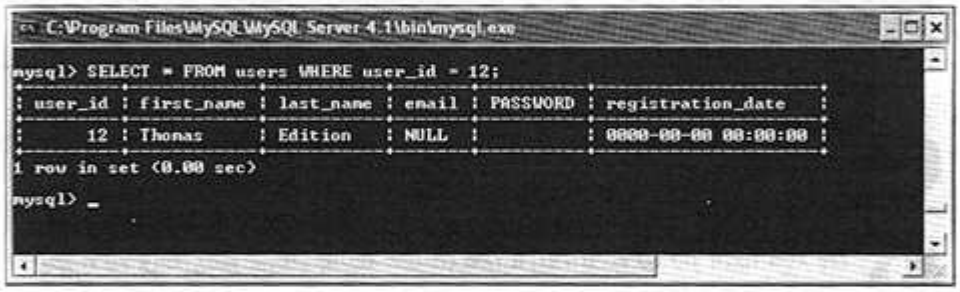


Hình 4-30: *Trường user_id sẽ được sử dụng để tham chiếu đến mẩu tin sẽ bị xóa.*

Cũng như trong ví dụ với câu lệnh **UPDATE**, đầu tiên, phải xác định khóa chính của mẩu tin cần xóa.

2. Kiểm tra lại mẩu tin nào sẽ bị xóa (xem hình 4-31):

```
SELECT * FROM users WHERE user_id = 12;
```

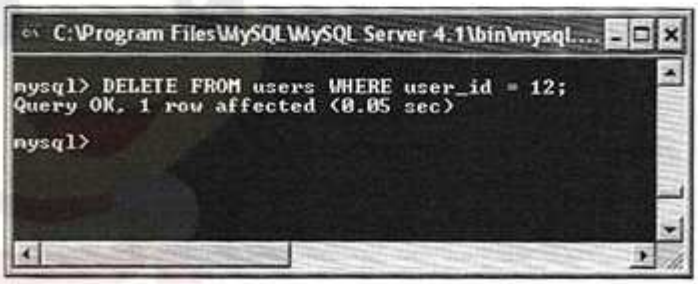


Hình 4-31: *Để xem trước ảnh hưởng của câu truy vấn* ***DELETE****, bạn thực hiện câu truy vấn* ***SELECT*** *tương tự.*

Một thủ thuật để đảm bảo an toàn khi xóa dữ liệu là thực hiện câu lệnh **SELECT** * thay cho **DELETE**. Kết quả là các mẩu tin bị ảnh hưởng bởi câu lệnh **DELETE** sẽ được thể hiện.

3. Xóa mẩu tin (xem hình 4-32):

```
DELETE FROM users WHERE user_id = 12;
```

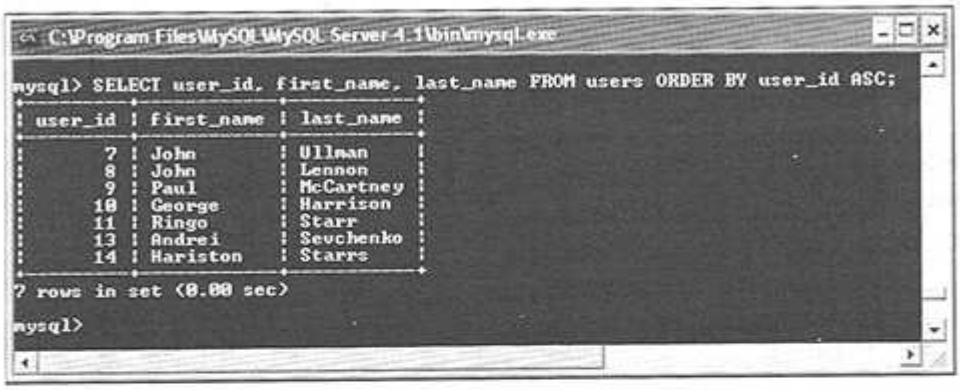


Hình 4-32: *Xóa mẩu tin ra khỏi bảng.*

Cũng như với câu lệnh **UPDATE**, MySQL sẽ thông báo kết quả thực hiện và số mẩu tin bị ảnh hưởng. Tại thời điểm này, không có cách nào lấy lại mẩu tin đã xóa, trừ khi bạn đã sao lưu cơ sở dữ liệu trước đó.

4. Kiểm tra các thay đổi đã thực hiện (xem hình 4-33):

```
SELECT user_id, first_name, last_name FROM users ORDER BY
→ user_id ASC;
```



Hình 4-33: *Mẩu tin với user_id bằng 12 đã không còn trong bảng này.*

Để xóa tất cả dữ liệu trong bảng và bản thân bảng, hãy sử dụng lệnh ***DROP TABLE:***

DROP TABLE <tên bảng>;

Để xóa toàn bộ cơ sở dữ liệu (kể cả các bảng trong đó), hãy sử dụng lệnh ***DROP DATABASE:***

DROP DATABASE <tên cơ sở dữ liệu>;

Lưu ý, nếu đăng nhập vào mysql với tùy chọn ***--i-am-a-dummy****, mysql sẽ không cho phép bạn chạy lệnh* ***UPDATE*** *và* ***DELETE*** *mà không có điều kiện* ***WHERE***

Từ phiên bản MySQL 4.0, bạn có thể thực hiện câu lệnh ***DELETE*** *trên nhiều bảng cùng lúc.*

Để đảm bảo an toàn khi xóa dữ liệu, hãy bổ sung mệnh đề ***LIMIT*** *(nếu chỉ muốn xóa một số mẩu tin):*

DELETE FROM <tên bảng> WHERE id=4 LIMIT 1;

Chương 5:

SQL VÀ MYSQL NÂNG CAO

Chương này sẽ tiếp tục với chương trước, chủ yếu để cập đến các chủ đề SQL và MySQL nâng cao. Bạn có thể hài lòng với những kiến thức cơ bản về cả hai công nghệ có được cho đến giờ. Tuy nhiên, với những đặc điểm nâng cao sẽ được đề cập trong chương này, bạn sẽ đưa các ứng dụng Web của mình lên một tầm cao mới.

Chúng ta sẽ bắt đầu với việc thiết kế cơ sở dữ liệu chi tiết hơn bằng hệ thống quản lý nội dung làm ví dụ. Điều này cũng dẫn đến chủ đề SQL về liên kết, tích hợp các phần khác nhau của một cơ sở dữ liệu quan hệ bất kỳ. Phần lớn nội dung sau đó sẽ giới thiệu một số hàm có sẵn của MySQL. Cuối cùng, bạn sẽ được học về chỉ mục và cách thay đổi cấu trúc cơ bản của các bảng đã có.

Thiết kế cơ sở dữ liệu

Bước đầu tiên trong việc sử dụng một cơ sở dữ liệu là thiết lập cấu trúc của nó. Thiết kế cơ sở dữ liệu, còn được gọi là lập mô hình dữ liệu, rất cần thiết cho việc quản lý hiệu quả thông tin của bạn trong thời gian dài. Bằng việc sử dụng một quá trình được gọi là tiêu chuẩn hóa, bạn sẽ loại trừ các phần dư thừa và các vấn đề khác một cách cẩn thận để xác định tính toàn vẹn cho dữ liệu của mình.

Chương này sẽ giới thiệu các kỹ thuật giúp bạn đảm bảo tính bền vững, tính vận hành và tính hiện thực cho cơ sở dữ liệu của mình. Ví dụ được sử dụng trong chương này là một hệ thống quản lý nội dung để chứa các liên kết Web. Nó sẽ được cụ thể hóa trong Chương 11 "Ví dụ - Quản lý nội dung", nhưng các nguyên lý của việc chuẩn hóa sẽ được áp dụng cho bất kỳ ứng dụng cơ sở dữ liệu nào mà bạn tạo ra.

Chuẩn hóa

Chuẩn hóa được phát triển bởi nhà nghiên cứu của IBM có tên là E.F. Codd trong những năm đầu 1970 (ông cũng là người phát minh ra cơ sở dữ liệu quan hệ). Một cơ sở dữ liệu quan hệ là một tập hợp dữ liệu, được tổ chức theo một cách thức đặc biệt và Codd tạo ra một loạt quy tắc được gọi là các dạng chuẩn hóa giúp định nghĩa tổ chức đó. Chương này sẽ đề cập đến ba dạng chuẩn hóa đầu tiên vì chúng vừa đủ cho phần lớn các thiết kế cơ sở dữ liệu.

Trước khi bắt tay vào việc chuẩn hóa cơ sở dữ liệu, chúng ta phải định nghĩa mục đích, yêu cầu cho ứng dụng sẽ phát triển. Cho dù bạn bàn bạc chi tiết về chủ đề này với khách hàng hoặc tự mình nghĩ ra thì việc hiểu được cách truy xuất thông tin sẽ nói lên mô hình dữ liệu. Như vậy, quá trình này chủ yếu cần giấy và bút

hơn là bản thân phần mềm MySQL (thiết kế cơ sở dữ liệu được áp dụng cho mọi cơ sở dữ liệu quan hệ, không chỉ MySQL).

Ví dụ này sẽ nói về quản lý các liên kết Web (URL). Mẩu tin ví dụ được nêu trong bảng 5.1.

Bảng 5.1: *Một số mẩu tin mẫu về loại thông tin sẽ chứa trong cơ sở dữ liệu.*

Mục	Ví dụ
URL	www.php.net
Date submitted	2-3-2005
Approved	Yes
URL Title	PHP: Hypertext Preprocessor
URL Description	The home page of...
URL Types	General PHP, Programming, Web Development

Một trong những cách tốt nhất để xác định thông tin nào sẽ được chứa trong cơ sở dữ liệu là làm rõ các câu hỏi được đặt ra và các dữ liệu nào sẽ kèm theo trong mỗi câu trả lời.

Ví dụ trong chương này minh họa quá trình thiết kế cơ sở dữ liệu thủ công, tuy nhiên đã có những ứng dụng làm sẵn phục vụ cho mục đích này.

Các ví dụ trong Chương 4 "Giới thiệu về SQL và MySQL", chỉ sử dụng một bảng và không đòi hỏi việc chuẩn hóa nâng cao.

Khóa

Như đã được đề cập sơ bộ trong chương trước, khóa là các mẩu dữ liệu giúp xác định một dòng thông tin trong một bảng. Có hai dạng khóa: khóa chính (primary key) và khóa ngoại (foreign key). Khóa chính là một định danh (identifier) duy nhất phải được tồn tại theo các quy tắc nhất định. Chúng phải:

- Luôn luôn có một giá trị (không thể là **NULL**).
- Có một giá trị không đổi.
- Có một giá trị duy nhất trong toàn bộ các mẩu tin có trong bảng.

Ví dụ thực tế tốt nhất về khóa chính là số chứng minh nhân dân: mỗi cá nhân sẽ có một số chứng minh nhân dân duy nhất và con số này không bao giờ thay đổi.

Dạng khóa thứ hai là khóa ngoại. Khóa ngoại là đại diện của khóa chính của bảng A trong bảng B. Nếu bạn có một cơ sở dữ liệu *movies* với một bảng *movie* và một bảng *director*, khóa chính trong bảng director sẽ được liên kết như một khóa ngoại trong bảng movie. Bạn sẽ hiểu rõ hơn về điều này khi tiếp tục với quá trình chuẩn hóa.

Cơ sở dữ liệu *content* hiện chỉ có một bảng độc lập, tuy nhiên trước khi bắt đầu quá trình chuẩn hóa, chúng ta cần đảm bảo ít nhất nó có một khóa chính (khóa ngoại sẽ được đề cập trong những bước sau).

Để thực hành gán một khóa chính, hãy thực hiện theo các bước dưới đây:

- Tìm trường thỏa mãn các điều kiện cho một khóa chính được nêu ở trên.

Trong ví dụ này, chỉ có một cột đáp ứng được các quy tắc trên: trường URL. Tuy nhiên, dùng nó làm khóa sẽ bất tiện vì một URL có thể được gửi đến nhiều lần và có thể thay đổi. Điều này sẽ phá vỡ quy tắc về khóa chính.

- Nếu không có khóa chính nào tồn tại, chúng ta sẽ tạo ra một khóa chính (xem bảng 5.2).

Bảng 5.2: *Bổ sung một khóa chính cho bảng để có thể tham chiếu đến các mẩu tin một cách dễ dàng.*

Mục	Ví dụ
URL ID	1
URL	www.php.net
Date submitted	2-3-2005
Approved	Yes
URL Title	PHP: Hypertext Preprocessor
URL Description	The home page of …
URL Types	General PHP, Programming, Web Development

Thông thường, chúng ta cần tạo một trường làm khóa chính vì không có giải pháp nào tốt hơn. Ngay cả số chứng minh nhân dân hoặc mã số sách ISBN (mà thoạt nhìn có thể đáp ứng các tiêu chuẩn) cũng không thích hợp cho việc làm khóa chính so với việc tạo một trường mới. Trong ví dụ này, chúng ta tạo ra trường mới có tên URL ID làm khóa chính.

Hiện tại, MySQL chỉ cài đặt khóa ngoại khi bạn sử dụng cơ sở dữ liệu kiểu InnoDB và bỏ qua các khóa này khi dùng các loại cơ sở dữ liệu khác. Do đó, các khóa ngoại trong MySQL chỉ tồn tại ở dạng lý thuyết hơn là dạng liên kết, mặc dù điều này có thể thay đổi trong tương lai.

Nên đặt tên cho khóa chính trùng với tên bảng cộng với hậu tố id (ví dụ user_id).

Một khi MySQL buộc phải cài đặt khóa ngoại (trong phiên bản 4.1 hoặc sau này), việc chuẩn hóa cơ sở dữ liệu sẽ cần thiết hơn so với hiện tại.

MySQL chỉ cho phép một khóa chính trong một bảng, mặc dù bạn có thể tạo ra một khóa chính dựa trên nhiều trường (nghĩa là tổ hợp của các trường này phải có giá trị duy nhất).

Khóa chính nên là một số nguyên vì điều này sẽ đưa đến việc MySQL vận hành tốt hơn. Đây cũng là lý do tại sao số chứng minh nhân dân và số ISBN (có chứa gạch ngang) lại ít được sử dụng làm khóa chính.

Quan hệ

Nói đến quan hệ (relationship) cơ sở dữ liệu là nói đến việc dữ liệu trong bảng này có quan hệ thế nào với dữ liệu trong một bảng khác. Một quan hệ giữa hai bảng có thể là quan hệ một-một, một-nhiều và nhiều-nhiều.

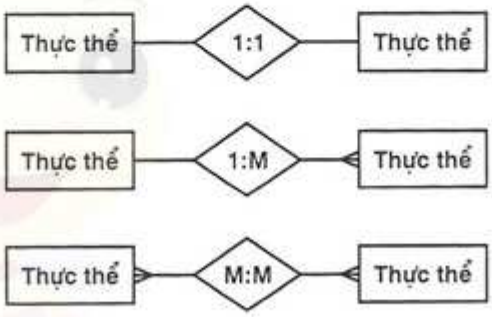
Quan hệ được gọi là một-một nếu một và chỉ một mục trong bảng A có quan hệ với duy nhất một mục trong bảng B (ví dụ: mỗi công dân Việt Nam có một số chứng minh nhân dân và mỗi số chứng minh nhân dân chỉ được sử dụng cho một công dân Việt Nam. Không ai có hai số chứng minh nhân dân và không có số chứng minh nhân dân nào tham chiếu đến hai người).

Quan hệ được gọi là một-nhiều nếu một mục trong bảng A có thể áp dụng cho nhiều mục trong bảng B. Ví dụ giới tính (nam và nữ) có thể được áp dụng cho nhiều người, nhưng mỗi người chỉ có một giới tính nhất định. Quan hệ một-nhiều là dạng quan hệ phổ biến giữa các bảng trong cơ sở dữ liệu.

Quan hệ được gọi là nhiều-nhiều nếu nhiều mục trong bảng A có thể áp dụng cho nhiều mục trong bảng B. Ví dụ, một album nhạc có thể chứa các bài hát của nhiều nhạc sỹ và các nhạc sỹ có thể tạo ra nhiều album. Bạn nên tránh các quan hệ nhiều-nhiều trong thiết kế của mình vì chúng dẫn đến tính dư thừa và các vấn đề về tính tích hợp dữ liệu.

Quan hệ và khóa cùng làm việc với nhau, trong đó khóa trong bảng này thường có quan hệ với một trường trong bảng khác như chúng ta đã đề cập đến trước đây.

Khi tạo mô hình dữ liệu, người ta thường dùng các quy ước nhất định để biểu thị cho cấu trúc của cơ sở dữ liệu và chúng ta sẽ sử dụng chúng trong chương này. Các biểu tượng biểu thị cho ba loại quan hệ được thể hiện trong hình 5-1.



Hình 5-1: *Các biểu tượng thường được dùng để mô tả các quan hệ trong lược đồ mô hình cơ sở dữ liệu.*

Quá trình thiết kế cơ sở dữ liệu dẫn đến ERD (Entity Relationship Diagram - lược đồ quan hệ thực thể), sử dụng các hình hộp và các biểu tượng để biểu thị cho các bảng và quan hệ giữa chúng.

Thuật ngữ "quan hệ" trong RDBMS vốn xuất phát từ bảng mà về mặt kỹ thuật còn được gọi là quan hệ.

Dạng chuẩn hóa thứ nhất

Đối với một cơ sở dữ liệu ở dạng chuẩn hóa thứ nhất (1NF - First Normal Form), mỗi cột chỉ chứa một giá trị. Một bảng có một trường chứa nhiều giá trị (ví dụ, trường điện thoại có thể chứa điện thoại nhà, di động, fax...) sẽ không đáp ứng dạng chuẩn hóa này.

Chúng ta sẽ bắt đầu quá trình chuẩn hóa bằng cách kiểm tra xem cấu trúc bảng có đáp ứng dạng chuẩn hóa 1NF không.

Thực hành tạo một cơ sở dữ liệu tuân theo dạng chuẩn hóa 1NF theo các bước sau:

1. Xác định các trường có thể có nhiều giá trị.

Quan sát lại bảng 5.2, trường URL Types không đáp ứng dạng chuẩn hóa 1NF. Ví dụ mẫu cho thấy trường này có ba giá trị, mặc dù với các mẩu tin khác, nó có thể có nhiều hoặc ít giá trị hơn. Trường Date Submitted cũng chứa nhiều giá trị con (ngày, tháng, năm). Tuy nhiên, việc phân chia nó thành nhiều giá trị như vậy sẽ không có nhiều ý nghĩa thực tế.

2. Tách các trường tìm thấy trong bước 1 thành các trường riêng biệt (xem bảng 5.3).

Bảng 5.3: *Bảng tương thích với 1NF.*

Mục	Ví dụ
URL ID	1
URL	www.php.net
Date submitted	2-3-2005
Approved	Yes
URL Title	PHP: Hypertext Preprocessor
URL Description	The home page of...
URL Type	General PHP

Trong ví dụ này, chúng ta tạo trường URL Type để nó chỉ chứa một giá trị. Ta vẫn có thể liệt kê URL với nhiều giá trị khác nhau bằng cách đưa nhiều mẩu tin vào trong bảng. Bạn sẽ hiểu rõ hơn về điều này khi tiếp tục với quá trình chuẩn hóa.

3. Kiểm tra lại để đảm bảo rằng các trường được tạo ra trong bước 2 đáp ứng được yêu cầu của dạng chuẩn hóa 1NF.

Dạng chuẩn hóa 1NF yêu cầu không có các nhóm lặp lại. Ví dụ, việc thay đổi cột URL Types thành URL Type 1, URL Type 2... sẽ phá vỡ quy tắc của 1NF vì những gì chúng ta làm ở đây là đưa vào nhiều giá trị và đặt chúng vào nhiều cột thay vì đặt chúng sang một bảng khác.

Một yêu cầu của dạng chuẩn hóa 1NF là mỗi mẩu tin đều phải có các thuộc tính (trường) giống nhau (nói cách khác, các mẩu tin phải có cấu trúc nhất quán). Vì thế, nếu trong bảng có một mẩu tin có ba kiểu URL trong khi mẩu tin khác lại chỉ có một, thì nó sẽ không đáp ứng dạng chuẩn hóa 1NF.

Cách hiểu về dạng chuẩn hóa 1NF có thể khác nhau. Ví dụ, nếu chúng ta xem địa chỉ là một giá trị gồm nhiều thành phần (số nhà, đường, quận...) thì nó đáp ứng được dạng chuẩn hóa 1NF. Nhưng, nếu ta xem nó là một trường chứa nhiều giá trị, thì nó lại không đáp ứng được dạng chuẩn hóa này.

Dạng chuẩn hóa thứ hai

Để một cơ sở dữ liệu đáp ứng dạng chuẩn hóa thứ hai (2NF), trước hết, nó phải thỏa mãn dạng chuẩn hóa 1NF (bạn cần phải chuẩn hóa theo đúng trình tự). Tất cả các cột có giá trị giống nhau trong nhiều mẩu tin phải được chuyển sang một bảng riêng và quan hệ ngược trở lại với bảng hiện tại. Ví dụ, nếu liệt kê nhà sản xuất với mỗi mẩu tin album nhạc, bạn sẽ thấy giá trị này có thể lặp lại nhiều lần và như vậy nó cần phải được tách thành một bảng riêng.

Quan sát cơ sở dữ liệu content (xem bảng 5.3), chúng ta thấy có một vấn đề rất rõ ràng: giá trị của trường URL Type sẽ giống nhau cho nhiều mẩu tin trong bảng. Đồng thời, các trường URL, URL Title và URL Description sẽ lặp lại nếu một URL được liệt kê nhiều lần (mỗi lần với một giá trị của trường URL Type).

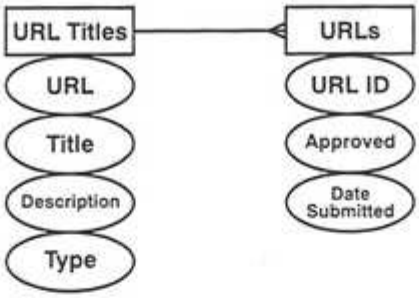
Để chuyển cơ sở dữ liệu này tương thích với dạng chuẩn hóa 2NF, chúng ta cần tách các cột có vấn đề thành một bảng riêng, trong đó các giá trị chỉ được thể hiện một lần. Trong thực tế, việc chuẩn hóa sẽ dẫn đến quá trình tạo các bảng cho đến khi các thông tin dư thừa được loại bỏ hoàn toàn.

Thực hành tạo cơ sở dữ liệu tương thích dạng chuẩn hóa 2NF theo các bước sau:

1. Xác định các trường có các giá trị lặp lại.

Như đã đề cập ở trên, các trường URL, URL Title, URL Description và URL Type lặp lại nhiều lần trên nhiều mẩu tin.

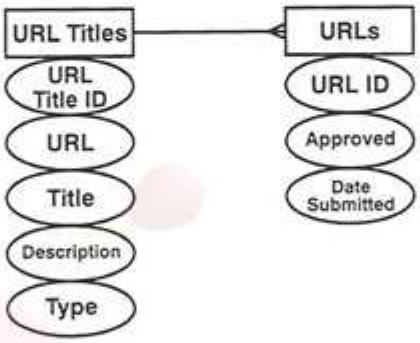
2. Tạo các bảng mới (xem hình 5-2).



Hình 5-2: *Tạo bảng thứ hai để đảm bảo cơ sở dữ liệu tương thích dạng chuẩn hóa 2NF.*

Thay đổi hợp lý nhất đối với cấu trúc hiện có là tạo bảng URLs và bảng URL Titles. Để biểu diễn cơ sở dữ liệu này, chúng ta sẽ tạo ra từng bảng ứng với mỗi hình hộp trong lược đồ để chứa tất cả các cột phía dưới nó. Tên bảng được đặt theo tiêu đề hộp. Hai bảng URLs và URL Titles sẽ có quan hệ một-nhiều (một mẩu tin trong bảng URL Titles có thể xuất hiện nhiều lần trong bảng URLs).

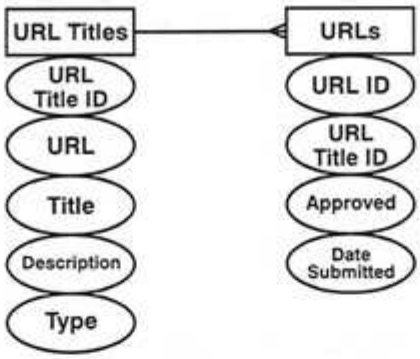
3. Gán hoặc tạo khóa chính (xem hình 5-3).



Hình 5-3: *Mỗi bảng nên có một khóa chính.*

Bạn nên tạo khóa chính cho mỗi bảng bằng các kỹ thuật đã được mô tả trước đây. Trong ví dụ này, chúng ta tạo các trường URL Title ID và URL ID để làm khóa chính cho mỗi bảng.

4. Tạo các khóa ngoại để liên kết hai bảng (xem hình 5-4).



Hình 5-4: *Để liên kết hai bảng, chúng ta bổ sung một khóa ngoại (URL Title ID) cho bảng URLs. Khóa ngoại này sẽ liên kết với khóa chính trong bảng URL Titles.*

Bước cuối cùng để đạt được dạng chuẩn hóa 2NF là kết hợp khóa ngoại và khóa chính để xác định quan hệ của toàn bộ dữ liệu. Chú ý, khóa chính trong một bảng thường là một khóa ngoại trong một bảng khác. Nếu thấy cơ sở dữ liệu của mình không đáp ứng được điều này, thì có thể bạn đã bỏ quên một điều gì đó (tuy rằng không nhất thiết là như vậy).

Một cách khác để kiểm tra dạng chuẩn hóa 2NF là quan sát mối quan hệ giữa các bảng. Mục đích là tạo ra các quan hệ một-nhiều. Các bảng có quan hệ nhiều-nhiều cần phải được thiết kế lại.

Một cơ sở dữ liệu được chuẩn hóa đúng sẽ không bao giờ có các mẩu tin trùng lặp (hai hoặc nhiều mẩu tin có cùng giá trị trong nhiều cột) trong cùng một bảng.

Trong ví dụ đang xét, tất cả URL đều có giá trị yes hoặc no ứng với cột Approved. Bạn có thể giữ nguyên trường này mà không cần phải chuẩn hóa tiếp vì nó chỉ có hai giá trị không đổi.

Chú ý, có nhiều cách mô tả quy tắc của dạng chuẩn hóa 2NF. Cho dù phát biểu ra sao và tiến hành các bước như thế nào thì quá trình chuẩn hóa vẫn không thay đổi.

Dạng chuẩn hóa thứ ba

Một cơ sở dữ liệu được xem là đáp ứng dạng chuẩn hóa thứ ba (3NF) nếu nó đáp ứng dạng chuẩn hóa 2NF và các cột không phải là khóa đều phụ thuộc vào khóa chính. Nếu theo quá trình chuẩn hóa 1NF và 2NF một cách chính xác, bạn sẽ không cần phải thực hiện bất kỳ thay đổi nào trong giai đoạn này. Tuy nhiên, nếu bạn thực hiện một số thay đổi nào đó trong quá trình chuẩn hóa, thì đây sẽ là bước kiểm tra cuối cùng.

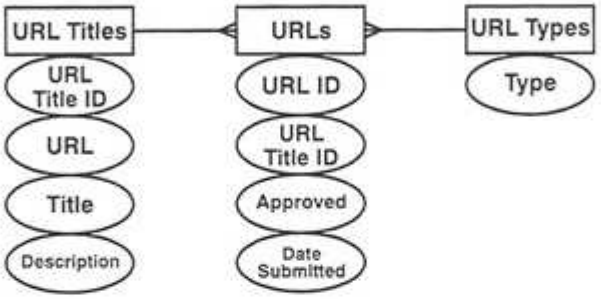
Trong ví dụ đang xét (xem hình 5-4), trường URL Type không phụ thuộc vào trường URL Title ID và như vậy nó cần phải được chuyển thành một bảng riêng.

Thực hành tạo một cơ sở dữ liệu tuân thủ dạng chuẩn hóa 3NF theo các bước sau:

1. Xác định các trường không có quan hệ trực tiếp với khóa chính.

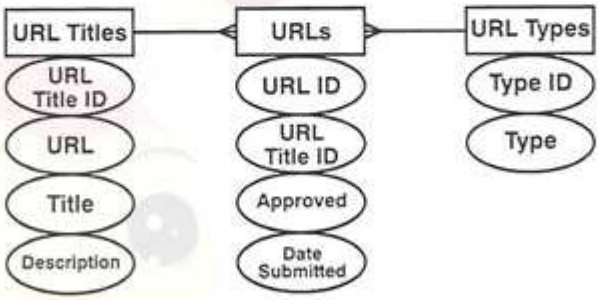
Như đã nói ở trên, trường URL Type không phải là trường duy nhất với mỗi mẩu tin.

2. Tạo các bảng mới (xem hình 5-5).



Hình 5-5: *Tạo bảng URL Types.*

Điều chỉnh hợp lý nhất cho cấu trúc hiện có là tạo bảng URL Types. Theo hướng phát triển của cơ sở dữ liệu này, URL Type sẽ liên hệ ngược trở lại với bảng URL và đóng vai trò như một cầu nối giữa hai bảng URL Types và URL Titles. Hai bảng này có quan hệ một-nhiều (một mẩu tin trong bảng URL Titles có thể xuất hiện nhiều lần trong bảng URL).

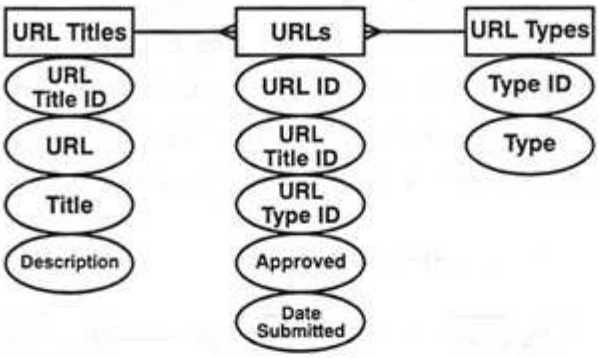


Hình 5-6: *Bảng URL Types giờ đây đã có khóa chính.*

3. Gán hoặc tạo các khóa chính mới (xem hình 5-6).

Bằng các kỹ thuật được mô tả trước đây trong chương này, bạn nên đảm bảo mỗi bảng có một khóa chính. Ở đây, chúng ta bổ sung trường URL Type ID vào bảng URL Types để đóng vai trò như một khóa chính.

4. Tạo các khóa ngoại cần thiết để tạo ra các quan hệ (xem hình 5-7).



Hình 5-7: *Cơ sở dữ liệu đã được chuẩn hóa.*

Cuối cùng, bổ sung một khóa ngoại URL Type ID vào trong bảng URL.

Ví dụ này sẽ không quá phức tạp nếu như nó không có quan hệ nhiều-nhiều giữa trường URL Types và trường URL Titles (vì chúng ta muốn liệt kê mỗi URL với nhiều loại khác nhau). Bảng URLs đóng vai trò trung gian giữa hai bảng này.

Phá bỏ quy tắc chuẩn hóa

Miễn là cơ sở dữ liệu của bạn đáp ứng chuẩn 3NF, nó sẽ đảm bảo tính ổn định và tính lâu dài. Bạn không nhất thiết phải chuẩn hóa hoàn toàn cơ sở dữ liệu. Tuy nhiên, trước khi phá bỏ quy tắc, cần hiểu rằng điều đó có thể dẫn đến kết quả không được tốt.

Hai lý do chính để phá bỏ quy tắc chuẩn hóa là để đảm bảo tính tiện lợi và tính vận hành. Càng ít bảng thì càng dễ quản lý. Hơn nữa, vì bản chất phức tạp của mình, các cơ sở dữ liệu được chuẩn hóa thường chiếm nhiều thời gian khi cập nhật, truy xuất dữ liệu hoặc chỉnh sửa. Chuẩn hóa là một một sự thỏa hiệp giữa tính toàn vẹn dữ liệu và tính mở rộng, tính đơn giản và tốc độ. Có nhiều cách để cải thiện tính vận hành của cơ sở dữ liệu mà không đòi hỏi nhiều đến sự phá vỡ các quy tắc chuẩn hóa.

Việc thực hành và trải nghiệm sẽ giúp tìm ra cách tốt nhất để mô hình hóa cơ sở dữ liệu. Tuy nhiên, bạn nên thực hành theo hướng chuẩn hóa như được nêu trong phần trên.

Tạo cơ sở dữ liệu

Bước cuối cùng trong thiết kế cơ sở dữ liệu là xác định kiểu dữ liệu cho từng cột và đặt tên cho chúng. MySQL rất linh động trong việc đặt tên cơ sở dữ liệu, tên bảng và tên trường, nhưng bạn cũng nên tuân thủ theo các nguyên tắc sau:

- Sử dụng các ký tự chữ cái và số, cùng với dấu gạch dưới (_) để phân tách các từ (MySQL không thể sử dụng khoảng trắng hoặc dấu phân cách trong tên gọi).

- Giới hạn tên gọi trong phạm vi 64 ký tự.
- Sử dụng số nhiều cho tên gọi của bảng (ý nói chứa nhiều mẩu tin) và số ít cho tên cột.
- Kết thúc khóa chính và khóa ngoại với hậu tố id (hoặc ID), hoặc pk và fk.
- Liệt kê khóa chính trong một bảng, sau đó đến các khóa ngoại.
- Tên trường nên có tính gợi tả.
- Tên trường cần phải duy nhất trong các bảng, ngoại trừ khóa.

Đây chỉ là những đề nghị chứ không bắt buộc, ngoại trừ việc giới hạn chiều dài tên trong phạm vi 64 ký tự và không sử dụng khoảng trắng, dấu phân cách khi đặt tên. Một số nhà phát triển thích sử dụng chữ in hoa để phân cách giữa các từ (thay cho dấu gạch dưới). Một số người khác lại thích nêu rõ kiểu dữ liệu hoặc bảng trong tên gọi của trường. Điều quan trọng nhất mà bạn cần nhớ là phải đảm bảo tính nhất quán khi thiết kế.

Bảng 5.4: *Thiết kế cuối cùng cho cơ sở dữ liệu content.*

Tên cột	Bảng	Kiểu
title_id	url_titles	SMALLINT(4) UNSIGNED NOT NULL AUTO_INCREMENT
url	url_titles	VARCHAR(60) NOT NULL
title	url_titles	VARCHAR(60) NOT NULL
description	url_titles	TINYTEXT NOT NULL
url_id	urls	SMALLINT(4) UNSIGNED NOT NULL AUTO_INCREMENT
title_id	urls	SMALLINT(4) UNSIGNED NOT NULL
type_id	urls	TINYINT(3) UNSIGNED NOT NULL
approved	urls	CHAR(1)
date_submitted	urls	TIMESTAMP
type_id	url_types	TINYINT(3) UNSIGNED NOT NULL AUTO_INCREMENT
type	url_types	VARCHAR(20) NOT NULL

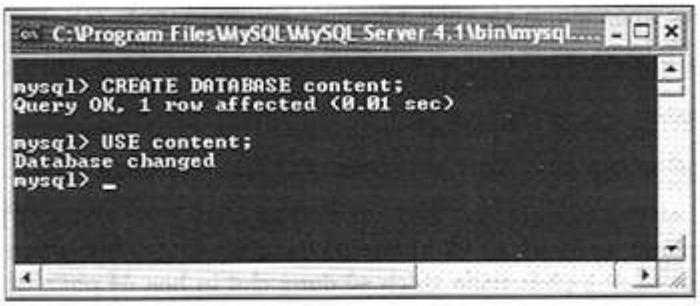
Thực hành tạo cơ sở dữ liệu theo các bước sau:

1. Đăng nhập vào trình quản lý mysql.

Cũng như trong chương trước, các bài thực hành trong chương này đều sử dụng trình quản lý mysql. Bạn có thể sử dụng phpMyAdmin hoặc một công cụ khác tương tự.

2. Tạo cơ sở dữ liệu content (xem hình 5-8):

```
CREATE DATABASE content;
USE content;
```



Hình 5-8: *Bước đầu tiên để tạo và chọn cơ sở dữ liệu.*

Tùy thuộc vào thiết lập, bạn có thể không được phép tạo cơ sở dữ liệu. Nếu đúng vậy, hãy sử dụng cơ sở dữ liệu hiện có và đưa các bảng vào trong cơ sở dữ liệu này (giả sử cơ sở dữ liệu hiện có không có các bảng urls, url_titles và url_types).

3. Tạo bảng urls (xem hình 5-9):

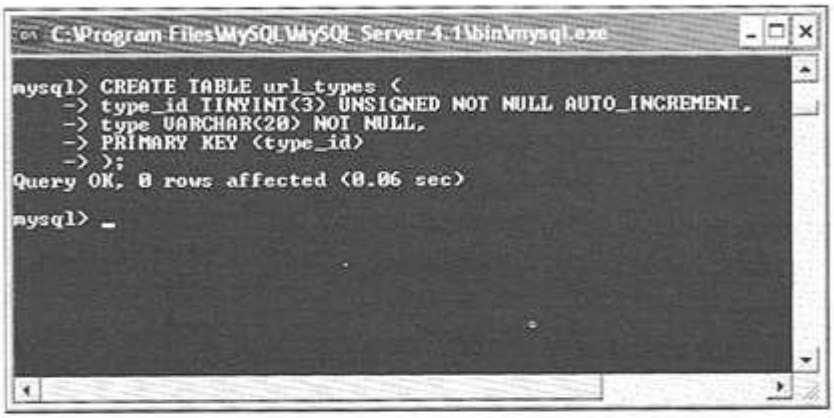
```
CREATE TABLE urls (
url_id SMALLINT(4) UNSIGNED NOT NULL AUTO_INCREMENT,
title_id SMALLINT(4) UNSIGNED NOT NULL,
type_id TINYINT(3) UNSIGNED NOT NULL,
approved CHAR(1),
date_submitted TIMESTAMP,
PRIMARY KEY (url_id)
);
```

Hình 5-9: *Bảng đầu tiên.*

Thứ tự tạo các bảng không quan trọng, tuy nhiên chúng ta sẽ tạo bảng urls trước. Chú ý, bạn có thể nhập câu lệnh SQL trên nhiều dòng.

4. Tạo bảng url_types (xem hình 5-10):

```
CREATE TABLE url_types (
type_id TINYINT(3) UNSIGNED NOT NULL AUTO_INCREMENT,
type VARCHAR(20) NOT NULL,
PRIMARY KEY (type_id)
);
```



Hình 5-10: *Bảng thứ hai.*

5. Tạo bảng url_titles (xem hình 5-11):

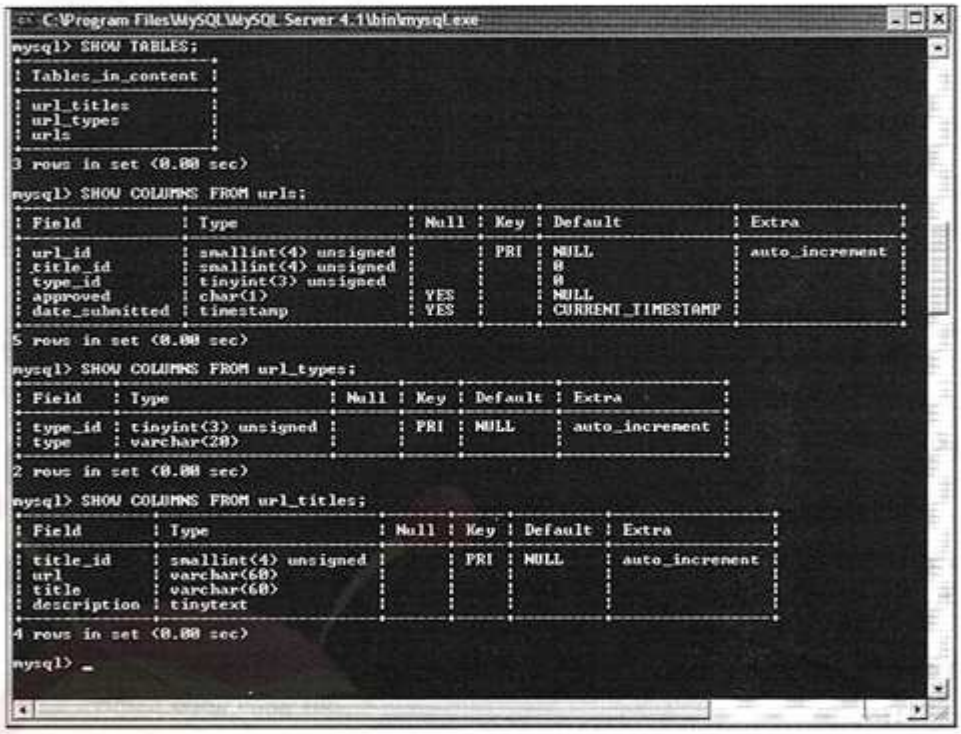
```
CREATE TABLE url_titles (
title_id SMALLINT(4) UNSIGNED NOT NULL AUTO_INCREMENT,
url VARCHAR(60) NOT NULL,
title VARCHAR(60) NOT NULL,
description TINYTEXT NOT NULL,
PRIMARY KEY (title_id)
);
```

Hình 5-11: *Bảng thứ ba và cũng là bảng cuối cùng của cơ sở dữ liệu.*

6. Kiểm tra lại cấu trúc của cơ sở dữ liệu nếu cần (xem hình 5-12):

```
SHOW TABLES;

SHOW COLUMNS FROM urls;

SHOW COLUMNS FROM url_types;

SHOW COLUMNS FROM url_titles;
```

Bước này là tùy chọn vì MySQL thường thông báo kết quả sau mỗi lần thực hiện một câu lệnh.

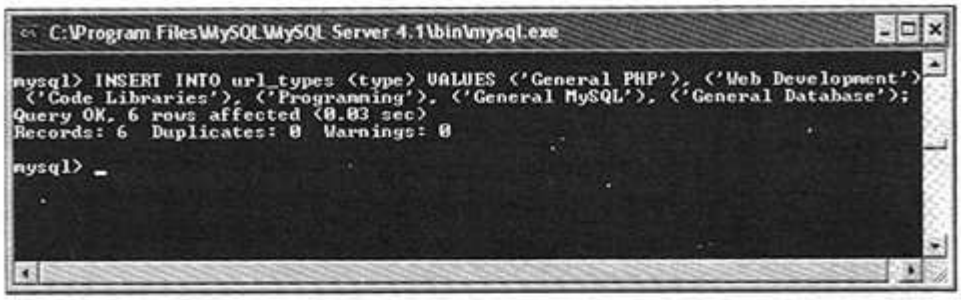


Hình 5-12: *Kiểm tra lại cấu trúc của cơ sở dữ liệu hoặc bảng với lệnh* ***SHOW***

Thực hành đưa dữ liệu vào cơ sở dữ liệu theo các bước sau:

1. Bổ sung một số mẩu tin mới vào bảng url_types (xem hình 5-13):

```
INSERT INTO url_types (type) VALUES ('General PHP'),
→ ('Web Development'), ('Code Libraries'),
→ ('Programming'), ('General MySQL'),
→ ('General Database');
```

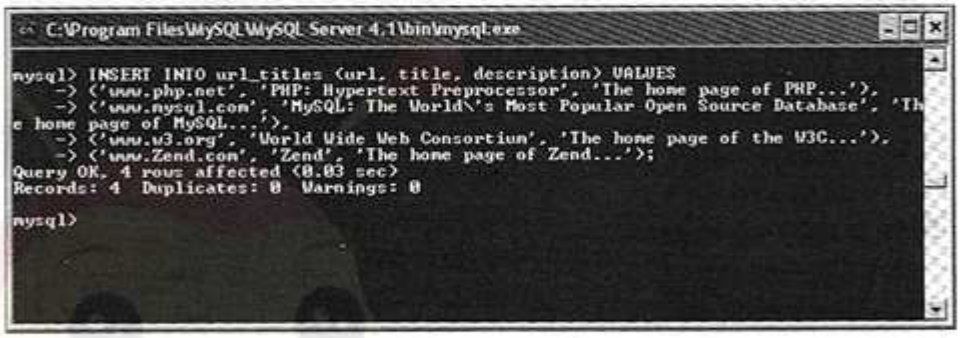


Hình 5-13: *Bổ sung các mẩu tin vào trong bảng url_types.*

Vì bảng urls sử dụng các giá trị được lấy từ bảng url_types và url_titles, chúng ta cần đưa dữ liệu vào các bảng này trước.

2. Đưa các mẩu tin vào trong bảng url_titles (xem hình 5-14):

```
INSERT INTO url_titles (url, title, description) VALUES
→ ('www.php.net', 'PHP: Hypertext Preprocessor',
→ 'The home page of PHP...'),
→ ('www.mysql.com', 'MySQL: The World\'s Most Popular Open
→ Source Database', 'The home page of MySQL...'),
→ ('www.w3.org', 'World Wide Web Consortium', 'The home
→ page of the W3C...'), ('www.Zend.com', 'Zend', 'The home
→ page of Zend...');
```



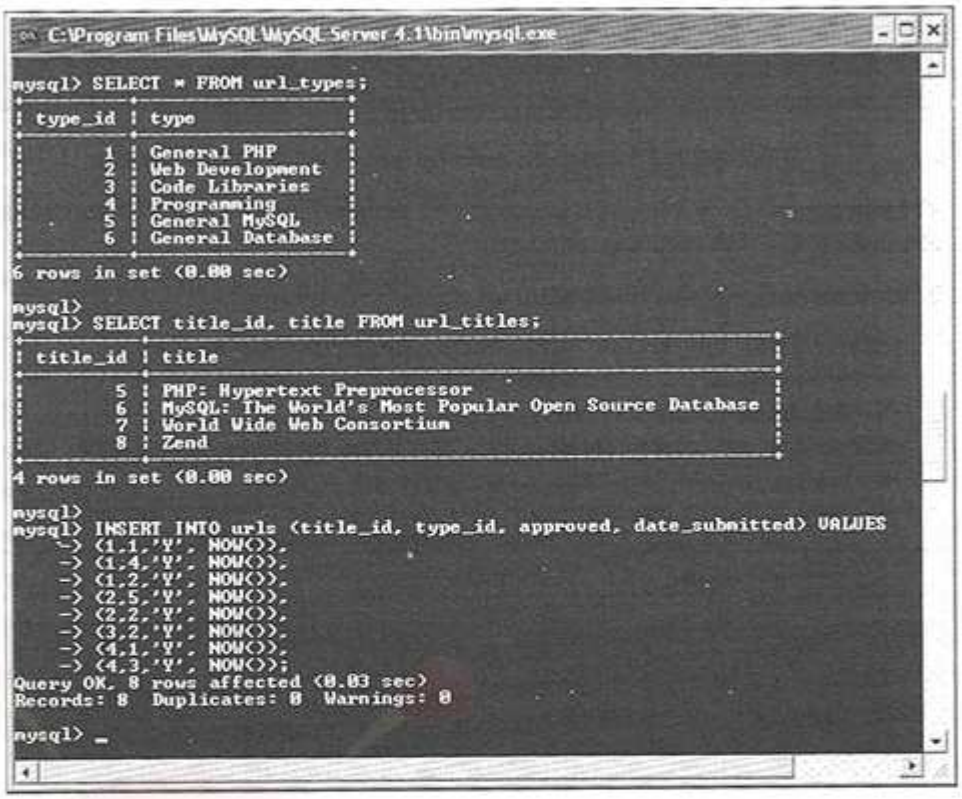
Hình 5-14: *Đưa dữ liệu vào trong bảng url_titles.*

Bạn có thể sử dụng các dữ liệu mẫu này hoặc nhập vào các dữ liệu của riêng mình. Chỉ cần đảm bảo không có các ký tự gây vấn đề (chẳng hạn như các dấu nháy) trong dữ liệu.

3. Đưa dữ liệu vào trong bảng urls (xem hình 5-15):

```
SELECT * FROM url_types;

SELECT title_id, title FROM url_titles;
```

```
INSERT INTO urls (title_id, type_id, approved,
→ date_submitted) VALUES (1,1,'Y', NOW()),
→ (1,4,'Y', NOW()),(1,2,'Y', NOW()),(2,5,'Y', NOW()),
→ (2,2,'Y', NOW()),(3,2,'Y', NOW()),(4,1,'Y', NOW()),
→ (4,3,'Y', NOW());
```



Hình 5-15: *Cơ sở dữ liệu chuẩn hóa thường đòi hỏi phải có các giá trị trong một số bảng trước khi có thể nhập dữ liệu vào trong một bảng khác.*

Vì hai trường trong bảng urls (title_id và type_id) liên quan đến các giá trị trong hai bảng còn lại, chúng ta sẽ lấy ra các giá trị này trước khi chèn vào các mẩu tin. Ví dụ, để tạo một mẩu tin URL mới cho *www.php.net* (title_id có giá trị 1) trong nhóm General PHP (type_id có giá trị 1), chúng ta sử dụng câu lệnh:

```
INSERT INTO urls (title_id, type_id, approved,
→ date_submitted) VALUES (1,1,'Y', NOW());
```

Với mã kịch bản PHP, quá trình này sẽ dễ hơn rất nhiều. Tuy nhiên, điều cần nắm ở đây cách sử dụng các thuật ngữ SQL.

4. Lặp lại bước 1 và 2 để đưa dữ liệu vào trong cơ sở dữ liệu.

Các ví dụ còn lại trong chương này sẽ sử dụng cơ sở dữ liệu đã được bổ sung dữ liệu. Bạn có thể sử dụng các câu lệnh **INSERT** để chèn dữ liệu vào cơ sở dữ liệu hiện có hoặc tạo cơ sở dữ liệu của riêng mình.

Để có một thiết kế tốt, bạn nên tạo các cột trong bảng theo một trật tự nhất định: khóa chính, khóa ngoại và các cột khác.

Trong các bảng trên, chúng ta thiết lập trường approved có kiểu là ***CHAR(1)*** *để nó có thể chứa giá trị Y hoặc N và trường date_submitted có kiểu là* ***TIMESTAMP****. Bạn có thể sử dụng kiểu* ***DATETIME*** *cho trường date_submitted, nhưng nó sẽ chiếm không gian gấp đôi so với kiểu* ***TIMESTAMP****.*

Sau khi phác họa cơ sở dữ liệu trên giấy, chúng ta sẽ tạo ra một loạt các bảng tính (spreadsheet) để phản ánh thiết kế hoặc sử dụng một ứng dụng chuyên biệt để thực hiện điều đó. Kết quả này là tài liệu tham khảo cho các nhóm phát triển Web và là tài liệu cung cấp cho khách hàng khi dự án hoàn tất.

Tên cơ sở dữ liệu và bảng trên các hệ thống Unix có phân biệt chữ hoa và chữ thường, trong khi trên Windows thì không. Riêng tên cột thì không phân biệt chữ hoa và chữ thường.

Nếu tuân thủ đầy đủ các nguyên lý thiết kế như đã nêu, bạn sẽ tránh được các lỗi có thể xảy ra khi lập trình một giao tiếp cơ sở dữ liệu, như bài thực hành trong Chương 6 "Sử dụng PHP và MySQL".

Vì các cơ sở dữ liệu quan hệ thường có kết cấu phức tạp, nên đôi khi cần phải dùng những câu truy vấn phức tạp để lấy ra thông tin cần thiết. Chúng ta sử dụng liên kết (các câu truy vấn SQL được thực hiện thông qua nhiều bảng) để lấy các dữ liệu từ các cơ sở dữ liệu quan hệ.

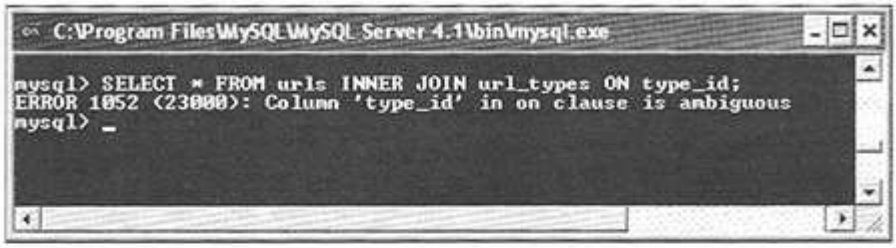
Có một số dạng liên kết theo chuẩn SQL (MySQL có một chút giới hạn trong lĩnh vực này). Chương này sẽ giới thiệu hai dạng liên kết cơ bản, phù hợp với người dùng trình độ trung bình. Dạng liên kết thường được sử dụng nhất là *liên kết trong* (inner join). Ví dụ:

```
SELECT * FROM urls, url_types WHERE
→ urls.type_id=url_types.type_id;
```

Ưu điểm của dạng liên kết này là nó sẽ lấy ra các thông tin từ cả hai bảng urls và url_types khi usrls.type_id bằng với url_types.type_id. Câu truy vấn sẽ thay thế khóa ngoại type_id trong bảng urls với tất cả thông tin ứng với loại đó từ bảng url_types. Một liên kết trong giống như trên sẽ chỉ lấy ra các mẩu tin phù hợp trong cả hai bảng. Vì thế, nếu có một loại URL không được sử dụng bởi một URL hiện có, nó sẽ không được lấy ra.

Khi lấy dữ liệu từ nhiều bảng và các cột trùng tên, chúng ta phải sử dụng cú pháp dấu chấm (*bảng.cột*). Điều này thường xảy ra khi làm việc với các cơ sở dữ

liệu quan hệ, vì khóa chính trong một bảng sẽ cùng tên với khóa ngoại trong bảng khác. Nếu không xác định cụ thể khi tham chiếu các cột trùng tên, MySQL sẽ phát sinh lỗi (xem hình 5-16).



Hình 5-16: *Việc tham chiếu một cột có trong nhiều bảng sẽ gây ra lỗi do sự không rõ ràng.*

Loại liên kết thứ hai được đề cập là *liên kết ngoài* (outer join) hoặc *liên kết trái* (left join). Khác với liên kết trong, liên kết ngoài có thể trả lại các mẩu tin không phù hợp giữa hai bảng. Cú pháp của liên kết ngoài là:

```
SELECT * FROM  url_types LEFT JOIN urls ON
→ urls.type_id=url_type.type_id;
```

Chú ý, dấu phẩy trong mệnh đề FROM của liên kết trong được thay thế bằng từ **LEFT JOIN** và từ **WHERE** được thay thế bằng từ **ON**. Điều cần quan tâm với liên kết ngoài là bảng nào được nêu trước. Trong ví dụ này, tất cả các mẩu tin trong bảng url_types sẽ được lấy cùng với tất cả các thông tin của bảng urls nếu thông tin trong bảng này đáp ứng điều kiện. Bạn sẽ thấy và hiểu hơn về liên kết này trong các ví dụ cụ thể.

Nếu cả hai bảng trong một liên kết trái có các cột cùng tên, bạn có thể đơn giản hóa câu truy vấn với mệnh đề USING:

```
SELECT * FROM  url_types LEFT JOIN urls USING (type_id);
```

Vì cú pháp liên kết khá phức tạp, nên SQL đưa ra khái niệm tên hiệu (alias) để giúp viết các câu truy vấn tiện lợi hơn (xem phần *"Tên hiệu"*).

Tên hiệu

Tên hiệu chỉ đơn giản ký hiệu đại diện cho một bảng hoặc cột. Tên hiệu được xác định với từ khóa AS:

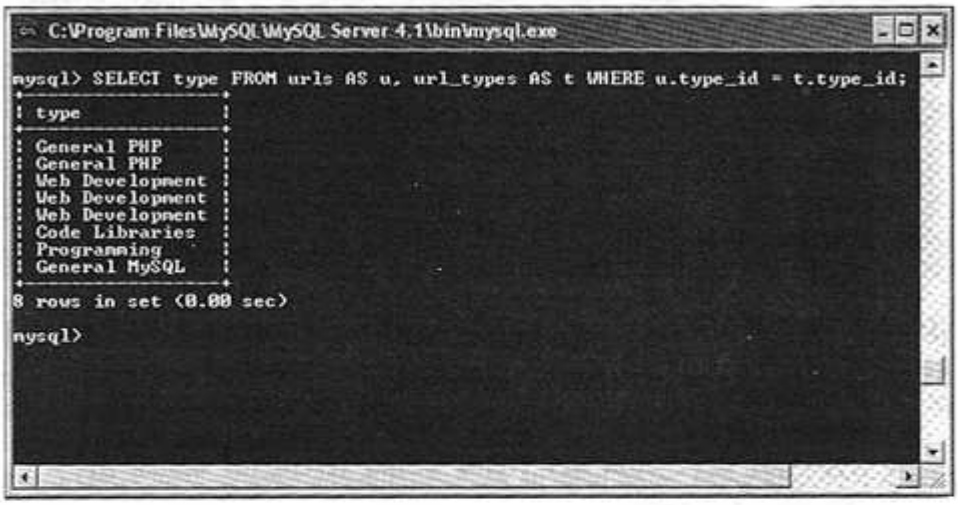
```
SELECT * FROM url_types AS t LEFT JOIN urls AS u
→ ON u.type_id=t.type_id;
```

Tên hiệu là chuỗi được tạo nên từ các con số, chữ cái và dấu gạch dưới, có phân biệt chữ hoa và chữ thường. Nó thường có dạng rất ngắn gọn, cho phép chúng ta viết các câu truy vấn súc tích hơn. Nếu đã định nghĩa một tên hiệu cho một bảng hoặc một cột, bạn nên sử dụng nó thay cho tên gốc trong toàn bộ câu truy vấn.

Thực hành sử dụng liên kết theo các bước sau:

1. Lấy ra loại URL ứng với mỗi mẩu tin trong bảng urls (xem hình 5-17):

```
SELECT type FROM urls AS u, url_types AS t WHERE
→ u.type_id = t.type_id;
```



Hình 5-17: *Một liên kết trong cơ bản.*

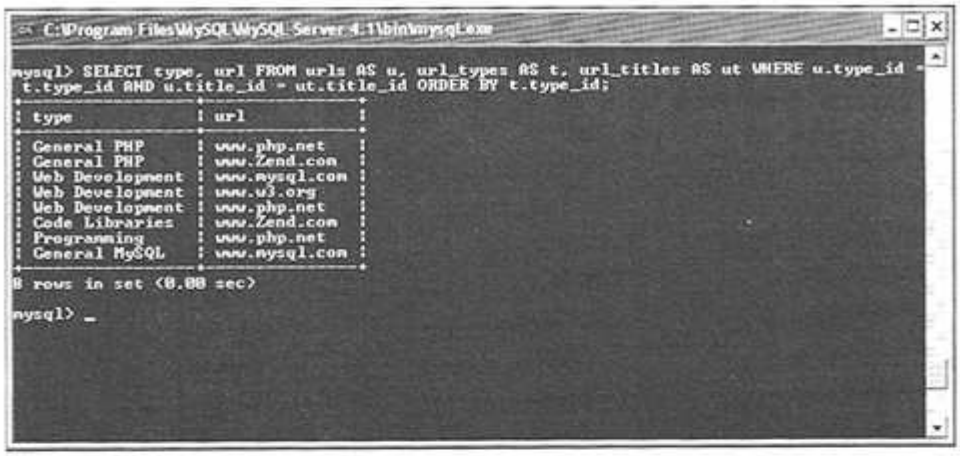
Câu truy vấn này sử dụng một liên kết trong để lấy ra giá trị của trường type trong bảng url_types ứng với mỗi mẩu tin trong bảng urls. Kết quả cuối cùng là nó hiển thị tên loại cho mỗi URL được chứa trong cơ sở dữ liệu.

2. Lấy ra URL và loại URL ứng với mỗi mẩu tin trong bảng urls, sắp xếp kết quả theo loại (xem hình 5-18):

```
SELECT type, url FROM urls AS u, url_types AS t, url_titles
→ AS ut WHERE u.type_id = t.type_id AND u.title_id =
→ ut.title_id ORDER BY t.type_id;
```

Có hai khác biệt giữa câu lệnh này và câu lệnh trước. Trước hết, chúng ta muốn lấy ra URL, nên cần liên kết với bảng url_titles. Thứ hai, chúng ta bổ sung mệnh đề **ORDER BY** để sắp xếp kết quả.

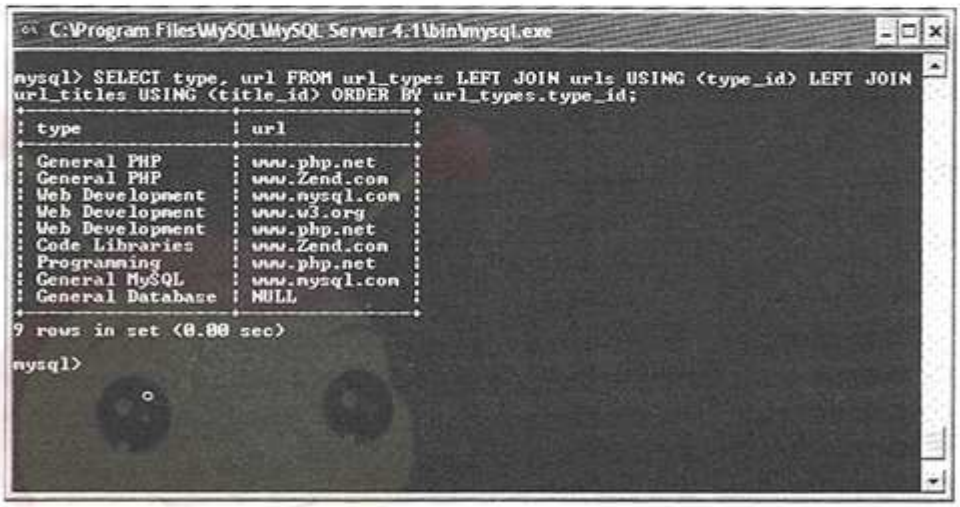
Cần chú ý đến cách viết một liên kết ba bảng (với một điều kiện **AND**) và cách đặt tên hiệu để sử dụng cách viết ngắn gọn khi tham chiếu các bảng và cột. Kết quả của câu truy vấn này sẽ là các mẩu tin trong bảng urls, với tên loại URL dạng văn bản thay thế cho giá trị type_id và tên URL dạng văn bản thay cho title_id.



Hình 5-18: *Một liên kết trong phức tạp hơn đôi chút (so sánh với hình 5-17).*

3. Lấy ra URL và loại của nó (xem hình 5-19):

```
SELECT type, url FROM url_types LEFT JOIN urls USING
→ (type_id) LEFT JOIN url_titles USING (title_id) ORDER BY
→ url_types.type_id;
```



Hình 5-19: *Liên kết trái này trả lại nhiều mẩu tin hơn so với liên kết trong (so sánh với hình 5-18).*

Đây là một liên kết trái được sửa đổi từ phiên bản liên kết trong của bước 2. Trong khi câu truy vấn trong bước 2 chỉ trả lại URL và loại của nó, ứng với các mẩu tin trong bảng urls, câu truy vấn ở bước này lấy ra tất cả URL và loại URL dựa trên bảng types. Kết quả là có nhiều mẩu tin được trả lại hơn. Các loại URL

không phù hợp với liên kết nội (không có URL nào được gán cho nó), sẽ được gán giá trị **NULL**.

Chúng ta có thể tạo liên kết với nhiều bảng cùng lúc, thậm chí liên kết một bảng với chính nó (tự liên kết).

Các liên kết có thể tạo bằng cách sử dụng các điều kiện với sự tham gia của các cột bất kỳ, không nhất thiết phải có sự tham gia của khóa chính và khóa ngoại như trong các ví dụ đã nêu.

Có thể thực hiện liên kết trên nhiều cơ sở dữ liệu với cú pháp ***<tên cơ sở dữ liệu>.<tên bảng>.<tên cột>****, miễn là chúng cùng nằm trên một máy chủ (không thể thực hiện điều này với nhiều cơ sở dữ liệu nằm rải rác trên mạng).*

Các liên kết không có kèm theo mệnh đề ***WHERE*** *(ví dụ,* ***SELECT * FROM*** *urls, url_types) được gọi là liên kết đầy đủ và sẽ lấy ra các mẩu tin xuất hiện trong cả hai bảng. Kết cấu này tạo ra những kết quả khó sử dụng đối với các bảng dữ liệu lớn.*

Giá trị ***NULL*** *trong một cột được tham chiếu trong một liên kết sẽ không bao giờ được trả lại vì* ***NULL*** *sẽ không phù hợp với bất kỳ giá trị nào, kể cả nó.*

Thuật ngữ AS được dùng để tạo một tên hiệu là tùy chọn và thường bị bỏ qua. Có thể viết một câu truy vấn đơn giản hơn với dạng sau: ***SELECT column alias_name FROM table;***

Vì cú pháp của các liên kết phức tạp hơn so với các câu truy vấn ***SELECT*** *chuẩn, bạn cần phải thực hành nhiều và cẩn thận khi viết chúng. Nếu mã kịch bản PHP của bạn làm việc với một câu lệnh SQL có sử dụng liên kết, hãy đảm bảo câu lệnh đó hoạt động trong trình quản lý mysql (hoặc phpMyAdmin) trước.*

Trong phần tiếp theo của chương này, chúng ta sẽ sử dụng tên hiệu khi chỉnh sửa giá trị của các cột để tạo cho các kết quả trả về với tên gọi mới.

Sử dụng hàm

Phần lớn các hàm được sử dụng với các câu truy vấn SQL cho đến giờ là để định dạng và thay đổi giá trị trả về. Để sử dụng một hàm bất kỳ, bạn cần phải sửa đổi câu truy vấn của mình để xác định cột nào sẽ được sử dụng với nó.

```
SELECT FUNCTION(<cột>) FROM <bảng>;
```

Để xác định nhiều cột, bạn có thể viết câu truy vấn theo một trong hai cách sau:

- `SELECT *, FUNCTION(<cột>) FROM <bảng>;`
- `SELECT <cột 1>, FUNCTION(<cột 2>), <cột 3> FROM <bảng>;`

```
SELECT DATE_FORMAT(date_submitted, '%a %b %e %Y') AS Date,
→ type AS Type, url AS URL FROM url_types t, urls u,
→ url_titles ut WHERE t.type_id=u.type_id AND
→ u.title_id=ut.title_id ORDER BY u.date_submitted DESC;
```

Đây là một ví dụ nữa về cách sử dụng các hàm định dạng để thay đổi kết quả xuất ra của một câu truy vấn SQL.

Trong các ứng dụng Web, bạn nên sử dụng SQL và hàm để định dạng ngày bất kỳ có trong cơ sở dữ liệu.

Cách duy nhất để truy xuất ngày hoặc giờ trên máy khách hàng là sử dụng JavaScript. Bạn không thể thực hiện điều đó với PHP và MySQL.

Các hàm gom nhóm

Cũng giống như việc giới hạn và sắp xếp kết quả, việc gom nhóm kết quả trả về trong câu lệnh SQL cũng sử dụng một mệnh đề riêng. Tuy nhiên, mệnh đề này có khác biệt đáng kể so với các mệnh đề kia ở chỗ nó gom nhóm dữ liệu được trả lại thành các khối thông tin tương tự nhau. Ví dụ, để gom nhóm tất cả các URL theo loại, chúng ta dùng:

```
SELECT * FROM urls GROUP BY type_id;
```

Với câu lệnh này, các mẩu tin có cùng giá trị type_id sẽ được gom lại thành một dòng thay vì xuất hiện dưới dạng từng mẩu tin như trước đây. Mệnh đề `GROUP BY` thường được sử dụng với các hàm gom nhóm (hoặc kết hợp). Bảng 5.9 liệt kê các hàm này.

Bảng 5.9: *Các hàm gom nhóm của MySQL.*

Hàm	Sử dụng	Mục đích
`AVG()`	`AVG(cột)`	Trả lại giá trị trung bình của một cột.
`MIN()`	`MIN(cột)`	Trả lại giá trị nhỏ nhất của một cột.
`MAX()`	`MAX(cột)`	Trả lại giá trị lớn nhất của một cột.
`SUM()`	`SUM(cột)`	Trả lại tổng các giá trị của một cột.
`COUNT()`	`COUNT(cột)`	Trả lại số mẩu tin.

Bạn có thể áp dụng một tổ hợp bất kỳ của `WHERE`, `ORDER BY` và `LIMIT` với `GROUP BY` bằng cách viết câu truy vấn như sau:

```
SELECT cột FROM bảng WHERE mệnh đề GROUP BY cột ORDER BY cột
→ LIMIT x;
```

Thực hành gom nhóm dữ liệu theo các bước sau:

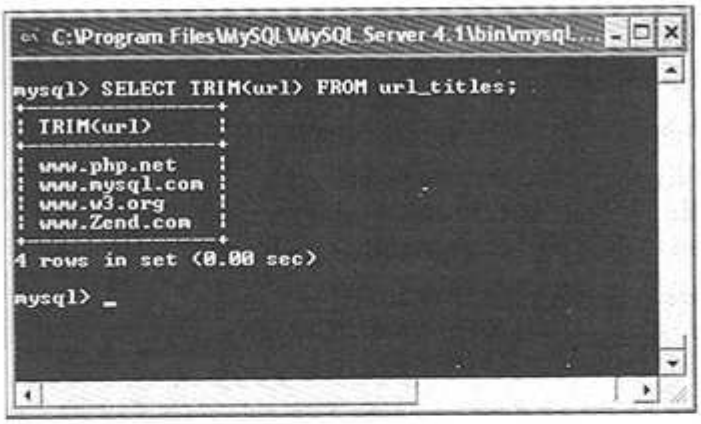
1. Chọn tất cả các tiêu đề đã được gửi (xem hình 5-31):

```
SELECT CONCAT(last_name, ' ', first_name) AS name FROM users;
```

Thực hành sử dụng các hàm văn bản theo các bước sau:

1. Loại bỏ các khoảng trắng dư thừa ra khỏi URL (xem hình 5-20):

```
SELECT TRIM(url) FROM url_titles;
```

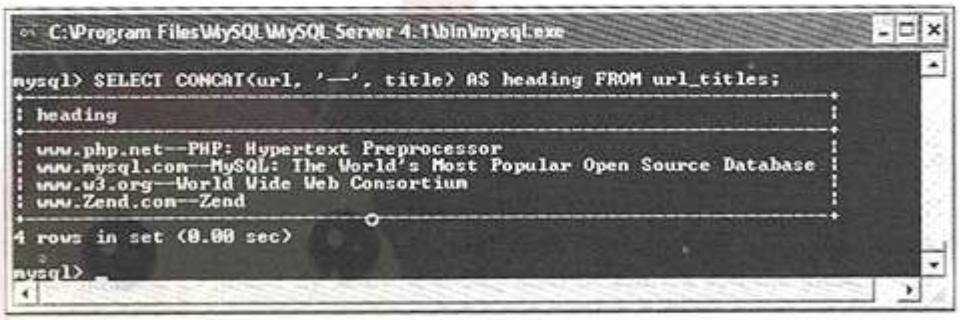


Hình 5-20: *Các khoảng trắng ở đầu và cuối của một cột sẽ được loại bỏ với hàm TRIM().*

Hàm `TRIM()` sẽ tự động loại bỏ các khoảng trắng (bao gồm ký tự khoảng trắng, tab, xuống dòng và về đầu dòng) cả ở đầu và cuối chuỗi.

2. Xem tất cả các URL và tiêu đề của chúng dưới dạng một chuỗi (xem hình 5-21):

```
SELECT CONCAT(url, '--', title) AS heading FROM url_titles;
```

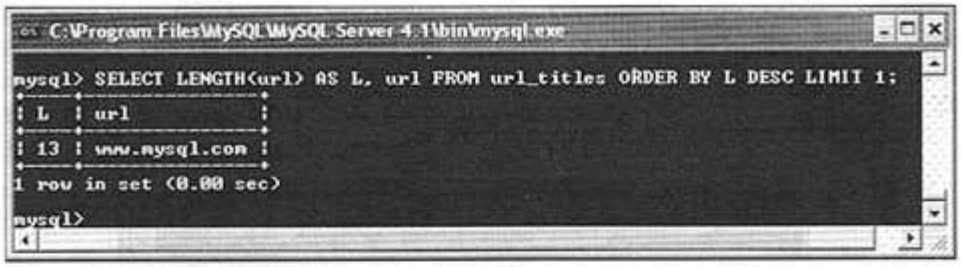


Hình 5-21: *Kết hợp là một cách mà chúng ta có thể dùng để định dạng các kết quả trả về.*

Câu lệnh SQL này sẽ kết hợp hai cột (url và title) và một chuỗi (--) sau đó trả lại kết quả với tên gọi mới (heading).

3. Tìm ra URL dài nhất (xem hình 5-22):

```
SELECT LENGTH(url) AS L, url FROM url_titles ORDER BY L DESC
→ LIMIT 1;
```



Hình 5-22: *Hàm* ***LENGTH()*** *trả lại chiều dài của giá trị chứa trong cột.*

Câu truy vấn này trước tiên sẽ xác định chiều dài của từng URL và gọi nó là L. Sau đó, toàn bộ danh sách có được sẽ được sắp xếp theo chiều giảm dần và mẩu tin đầu tiên sẽ được trả lại với hai giá trị: URL và chiều dài của nó.

Câu truy vấn giống như trong bước 3 (xem hình 5-22) có thể giúp bạn xác định và điều chỉnh chiều rộng của các cột sau khi đã có đầy đủ dữ liệu trong bảng.

Bạn có thể sử dụng các hàm SQL cho nhiều loại câu lệnh khác nhau, ngoài SELECT. Ví dụ, sử dụng hàm TRIM để cắt xén dữ liệu trong câu lệnh ***INSERT****.*

Hàm có thể được áp dụng cho cột hoặc giá trị được nhập vào. Ví dụ, câu lệnh sau là hợp lệ:

```
SELECT UPPER('abcdef');
```

CONCAT() *có một hàm tương tự có tên* ***CONCAT_WS()****, trong đó chữ WS phía sau có nghĩa là with separator (với ký tự phân cách). Cú pháp của hàm này là* ***CONCAT_WS(<chuỗi phân cách>, <cột 1>, <cột 2>,...)****. Chuỗi phân cách sẽ được chèn vào giữa các cột khi thực hiện việc kết hợp.*

Các hàm về số

Ngoài các toán tử toán học chuẩn mà MySQL sử dụng (cộng, trừ, nhân, chia), còn có rất nhiều hàm được dùng để định dạng và thực hiện các tính toán trên các giá trị số. Bảng 5.6 liệt kê một số hàm phổ biến nhất và chúng ta sẽ cùng tìm hiểu một số hàm trong chương này.

Bảng 5.6: *Các hàm xử lý số trong MySQL.*

Hàm	Ví dụ	Mục đích
`ABS()`	`ABS(x)`	Trả lại giá trị tuyệt đối của x.
`CEILING()`	`CEILING(x)`	Trả lại giá trị nguyên cao nhất kế tiếp dựa trên giá trị của x.

FLOOR()	FLOOR(x)	Trả lại giá trị nguyên của x.
FORMAT()	FORMAT(x, y)	Trả lại x được định dạng dưới dạng số với y số lẻ; sử dụng dấu phẩy làm ký tự phân cách sau mỗi ba chữ số.
MOD()	MOD(x,y)	Trả lại phần dư của phép chia x cho y.
RAND()	RAND()	Trả lại một số ngẫu nhiên trong phạm vi từ 0 đến 1.0.
ROUND()	ROUND(x, y)	Trả lại số x được làm tròn đến y số lẻ.
SIGN()	SIGN(x)	Trả lại một giá trị cho biết x là một số âm (-1), zero (0) hoặc một số dương (1).
SQRT()	SQRT(x)	Trả lại căn bậc hai của x.

Hàm **ROUND()** nhận vào một giá trị (thường là từ một cột) và làm tròn nó đến một con số với số số lẻ xác định. Nếu không có phần lẻ nào được xác định, nó sẽ làm tròn đến số nguyên gần nhất. Nếu số số lẻ xác định nhiều hơn số số lẻ thực tế của giá trị, phần còn thiếu sẽ được điền bằng các con số 0 (vào phía phải).

Hàm **RAND()** được sử dụng để trả lại các con số ngẫu nhiên trong khoảng 0 đến 1.0. Ví dụ:

```
SELECT RAND();
```

Một ưu điểm của hàm **RAND()** là nó có thể được sử dụng với các câu truy vấn để sắp xếp các giá trị trả lại theo một trật tự ngẫu nhiên:

```
SELECT * FROM <bảng> ORDER BY RAND();
```

Thực hành sử dụng các hàm về số theo các bước sau:

1. Hiển thị một số, định dạng nó với dấu $ (xem hình 5-23):

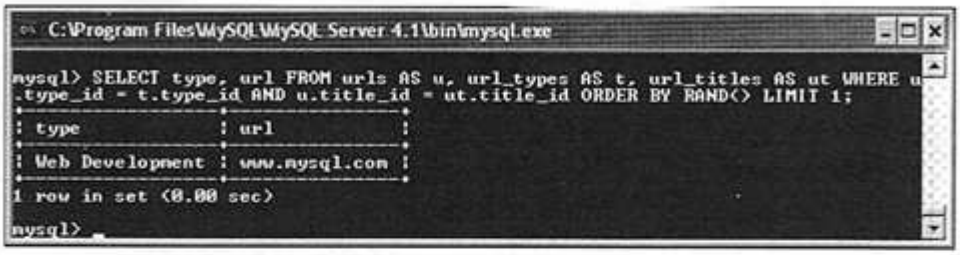
```
SELECT CONCAT('$', FORMAT(5639.6, 2)) AS cost;
```

Hình 5-23: *Câu truy vấn này thể hiện cách sử dụng hàm* ***FORMAT()****.*

Bằng cách dùng hàm **FORMAT()** cùng với hàm **CONCAT()**, bạn có thể chuyển một số bất kỳ thành định dạng tiền tệ khi muốn hiển thị trong một trang Web.

2. Lấy ngẫu nhiên một URL cùng với loại của nó (xem hình 5-24 và 5-25):

```
SELECT type, url FROM urls AS u, url_types AS t, url_titles
→ AS ut WHERE u.type_id = t.type_id AND u.title_id =
→ ut.title_id ORDER BY RAND() LIMIT 1;
```



Hình 5-24: *Hàm **RAND()** được sử dụng để trả lại một mẩu tin ngẫu nhiên từ cơ sở dữ liệu.*

Hình 5-25: *Các lần thực hiện tiếp theo của cùng câu truy vấn (so sánh với hình 5-24) sẽ trả lại các kết quả khác nhau.*

Phần lớn mã lệnh trong câu truy vấn này là liên kết trong giữa ba bảng đã được viết trong phần trước của chương này. Chúng ta chỉ bổ sung thêm mệnh đề **ORDER BY** và **LIMIT**. Tuy hàm **RAND()** không phải là ngẫu nhiên hoàn toàn, nhưng nó cũng đủ đáp ứng cho phần lớn các trường hợp. Chú ý, bạn không phải xác định cột nào khi sử dụng hàm **RAND()**.

Ngoài các hàm toán học được liệt kê ở đây, MySQL còn có một số hàm lượng giác, hàm lũy thừa và các hàm liên quan đến số khác.

*Hàm **MOD()** có cùng ý nghĩa như toán tử %. Ví dụ: **SELECT MOD(9, 2)** và **SELECT 9%2** có kết quả giống nhau.*

Các hàm về ngày, giờ

Các kiểu dữ liệu ngày, giờ trong MySQL đặc biệt linh động. Nhưng vì nhiều người dùng cơ sở dữ liệu không quen với các hàm về ngày và giờ, nên các tùy chọn này ít được sử dụng.

Cho dù bạn thực hiện tính toán dựa trên ngày hoặc chỉ trả lại tên tháng từ một giá trị có sẵn, MySQL đều có các hàm giúp thực hiện điều đó. Bảng 5.7 liệt kê phần lớn các hàm về ngày, giờ.

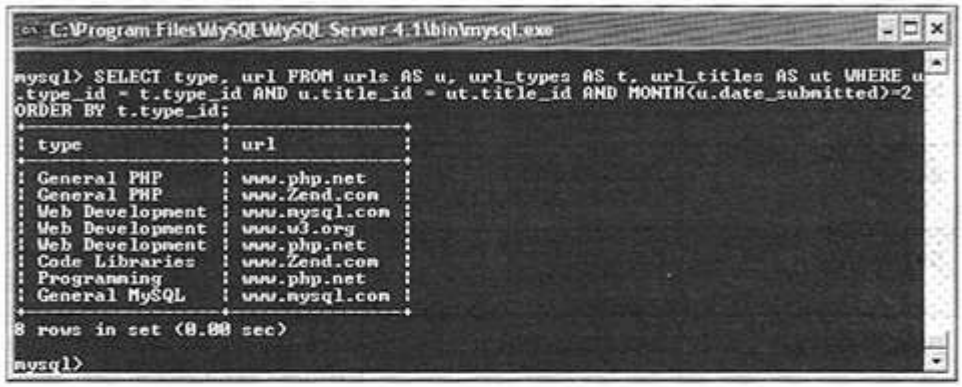
Bảng 5.7: *Các hàm liên quan đến ngày, giờ trong MySQL.*

Hàm	Sử dụng	Mục đích
HOUR()	HOUR(cột)	Trả lại giá trị giờ của giá trị chứa trong cột.
MINUTE()	MINUTE(cột)	Trả lại giá trị phút của giá trị chứa trong cột.
SECOND()	SECOND(cột)	Trả lại giá trị giây của giá trị chứa trong cột.
DAYNAME()	DAYNAME(cột)	Trả lại tên ngày của giá trị chứa trong cột.
DAYOFMONTH()	DAYOFMONTH(cột)	Trả lại con số thể hiện ngày trong tháng của giá trị chứa trong cột.
MONTHNAME()	MONTHNAME(cột)	Trả lại tên tháng của giá trị chứa trong cột.
MONTH()	MONTH(cột)	Trả lại con số thể hiện tháng của giá trị chứa trong cột.
YEAR()	YEAR(cột)	Trả lại giá trị năm của giá trị chứa trong cột.
ADDDATE()	ADDDATE(cột, INTERVAL x loại)	Trả lại kết quả của phép tính x cộng với giá trị cột.
SUBDATE()	SUBDATE(cột, INTERVAL x loại)	Trả lại kết quả của phép tính trừ giá trị cột đi x đơn vị.
CURDATE()	CURDATE()	Trả lại ngày hiện tại.
CURTIME()	CURTIME()	Trả lại giờ hiện tại.
NOW()	NOW()	Trả lại ngày, giờ hiện tại.
UNIX_TIMESTAMP()	UNIX_TIMESTAMP(ngày)	Trả lại số giây tính từ mốc thời gian của ngày xác định.

Thực hành sử dụng các hàm thời gian theo các bước sau:

1. Hiển thị URL và loại cho mỗi mẩu tin đã được thêm vào trong tháng 2 (xem hình 5-26):

```
SELECT type, url FROM urls AS u, url_types AS t, url_titles
→ AS ut WHERE u.type_id = t.type_id AND u.title_id =
→ ut.title_id AND MONTH(u.date_submitted)=2 ORDER BY
→ t.type_id;
```

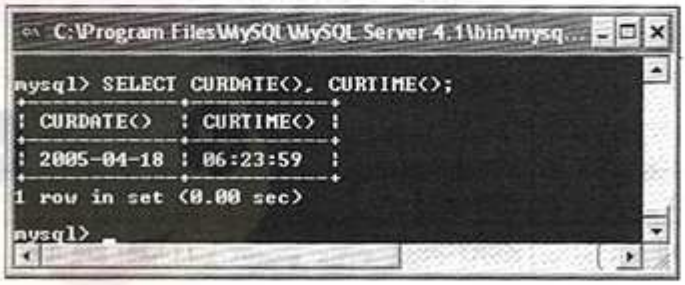


Hình 5-26: *Các hàm thời gian có thể được sử dụng để định dạng các cột hoặc giới hạn các mẩu tin được trả lại.*

Phần lớn mã lệnh của câu truy vấn này cũng giống như liên kết đã được sử dụng trước đây trong chương này. Chúng ta chỉ chỉnh sửa một chút để lấy các mẩu tin được đăng ký vào tháng 2 (`MONTH(u.date_submitted)=2`). Một cách khác để viết câu lệnh này là sử dụng `MONTHNAME(u.date_submitted) = 'February'`. Tuy nhiên, bạn nên sử dụng phương án trước vì nó thực hiện nhanh và hiệu quả hơn.

2. Thể hiện ngày, giờ hiện tại (xem hình 5-27):

```
SELECT CURDATE(), CURTIME();
```



Hình 5-27: *Câu truy vấn này trả lại ngày, giờ hiện tại thay vì trả lại các giá trị từ một bảng.*

Để thể hiện ngày và giờ hiện tại (theo MySQL), chúng ta sử dụng hàm `CURDATE()` và `CURTIME()`. Đây là ví dụ về một câu truy vấn không tham chiếu cụ thể đến một bảng nào.

Ngày và giờ được trả lại bởi các hàm ngày và giờ của MySQL sẽ phụ thuộc vào máy chủ, chứ không phụ thuộc vào máy khách hàng.

Cần cẩn thận khi sử dụng hàm `ADDDATE()` *để đảm bảo có đủ ba chữ D. Rất dễ viết hàm này thành* `ADDATE()` *và điều đó sẽ gây ra lỗi. Nếu thường xuyên phạm lỗi với hàm này, bạn nên chuyển sang sử dụng hàm* `DATE_ADD()`.

Hàm ADDDATE() và SUBDATE()

Các hàm `ADDDATE()` *và* `SUBDATE()` *(là tên gọi khác của các hàm* `DATE_ADD()` *và* `DATE_SUB()`*) thực hiện các tính toán dựa trên các giá trị thời gian. Cú pháp sử dụng các hàm này là:*

```
ADDDATE(ngày, INTERVAL x loại);
```

Trong đó, ngày có thể được nhập vào hoặc là một giá trị được lấy ra từ một cột. Giá trị x thay đổi tùy theo loại mà bạn xác định. Các loại có hiệu lực gồm `SECOND`, `MINUTE`, `HOUR`, `DAY`, `MONTH` *và* `YEAR`. *Thậm chí còn có sự kết hợp của các dạng này:* `MINUTE_SECOND`, `HOUR_MINUTE`, `DAY_HOUR` *và* `YEAR_MONTH`.

Để cộng thêm hai giờ vào một ngày cho trước, chúng ta viết như sau:

```
ADDDATE(ngày, INTERVAL 2 HOUR);
```

Để cộng thêm hai tuần vào ngày 31-12-2002:

```
ADDDATE('2002-12-31', INTERVAL 14 DAY);
```

Để trừ đi 15 tháng từ một ngày:

```
SUBDATE(ngày, INTERVAL '1-3' MONTH);
```

Câu truy vấn này yêu cầu MySQL trừ đi 1 năm và 3 tháng từ ngày được cung cấp.

Định dạng ngày, giờ

Có hai hàm về ngày, giờ được sử dụng nhiều hơn các hàm khác: `DATE_FORMAT()` và `TIME_FORMAT()`. Giữa hai hàm này có một chút trùng lắp.

`DATE_FORMAT()` được sử dụng để định dạng cả ngày và giờ nếu một giá trị có chứa cả hai thành phần này (ví dụ: YYYY-MM-DD HH:MM:SS). Trong khi đó, hàm `TIME_FORMAT()` chỉ dùng để định dạng thời gian và chỉ được sử dụng nếu có thành phần thời gian trong giá trị được định dạng (tức là HH:MM:SS). Cú pháp sử dụng như sau:

```
SELECT DATE_FORMAT(<cột>, 'định dạng') FROM <bảng>;
```

Phần định dạng là một tổ hợp dấu phần trăm (%) và các mã để xác định giá trị muốn lấy ra. Bảng 5.8 liệt kê các tham số định dạng dùng cho hai hàm này. Bạn có thể sử dụng các tham số này với một tổ hợp bất kỳ, cùng với văn bản bổ sung, dấu ngắt... để trả lại ngày và giờ ở dạng thể hiện rõ hơn.

Bảng 5.8: *Các mã định dạng dùng với hàm* ***DATE_FORMAT()*** *và* ***TIME_FORMAT()***.

Mã	Sử dụng	Ví dụ
%e	Ngày của tháng.	1-31
%d	Ngày của tháng ở dạng hai chữ số.	01-31
%D	Ngày với hậu tố.	1st-31st
%W	Tên ngày trong tuần.	Wednesday
%a	Tên viết tắt của ngày trong tuần.	Sun-Sat
%c	Số hiệu tháng.	1-12
%m	Số hiệu tháng ở dạng hai chữ số.	01-12
%M	Tên tháng.	January-December
%b	Tên tháng ở dạng viết tắt.	Jan-Dec
%Y	Năm ở dạng bốn chữ số.	2005
%y	Năm ở dạng hai chữ số.	05
%l	Giờ.	1-12
%h	Giờ ở dạng hai chữ số.	01-12
%k	Giờ (24 giờ).	0-23
%H	Giờ (24 giờ) ở dạng hai chữ số.	00-23
%i	Phút.	00-59
%S	Giây.	00-59
%r	Thời gian.	8:17:02 PM
%T	Thời gian (24 giờ).	20:17:02
%p	AM hoặc PM.	AM hoặc PM

Giả sử, cột *the_date* trong một bảng có giá trị là 2002-04-30 23:07:45, các định dạng phổ biến cho trường này có thể như sau:

- Thời gian (11:07:45 PM):

```
TIME_FORMAT(the_date,'%r');
```

- Thời gian không có phần giây (11:07 PM):

```
TIME_FORMAT(the_date, '%l:%i %p');
```

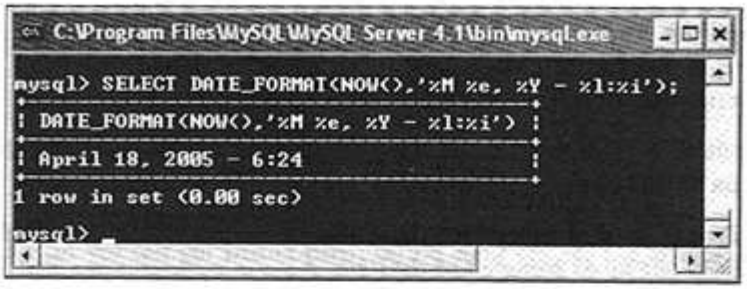
- Ngày (April 30th, 2002):

```
DATE_FORMAT(the_date, '%M %D, %Y');
```

Thực hành định dạng thời gian theo các bước sau:

1. Lấy ngày, giờ hiện tại ở dạng Tháng DD, YYYY - HH:MM:SS (xem hình 5-28):

```
SELECT DATE_FORMAT(NOW(),'%M %e, %Y - %l:%i');
```

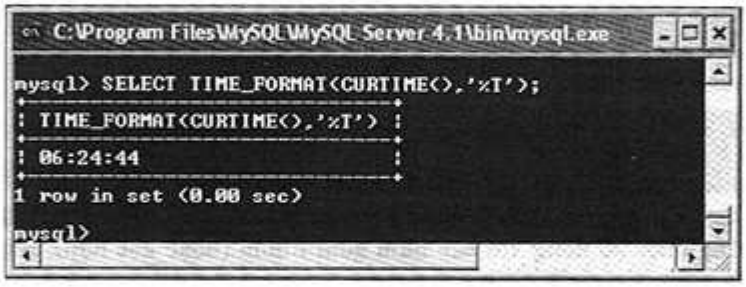


Hình 5-28: *Ngày và giờ hiện tại đã được định dạng.*

Bằng cách sử dụng hàm *NOW()* - hàm trả lại ngày và giờ hiện tại, chúng ta có thể thực hành việc định dạng để xem kết quả được trả lại ra sao.

2. Hiển thị giờ hiện tại bằng cách sử dụng hệ thống giờ 24 (xem hình 5-29):

```
SELECT TIME_FORMAT(CURTIME(),'%T');
```



Hình 5-29: *Giờ hiện tại ở định dạng 24 giờ.*

3. Lấy ra URL và loại, sắp xếp thứ tự theo thời gian nhập và định dạng ngày ở dạng *Thứ (viết tắt) Ngày Tháng (viết tắt) Năm* (xem hình 5-30):

Hình 5-30: *Trong liên kết này, trường date_submitted được định dạng và được sử dụng để sắp xếp các mẩu tin.*

```
SELECT DATE_FORMAT(date_submitted, '%a %b %e %Y') AS Date,
→ type AS Type, url AS URL FROM url_types t, urls u,
→ url_titles ut WHERE t.type_id=u.type_id AND
→ u.title_id=ut.title_id ORDER BY u.date_submitted DESC;
```

Đây là một ví dụ nữa về cách sử dụng các hàm định dạng để thay đổi kết quả xuất ra của một câu truy vấn SQL.

Trong các ứng dụng Web, bạn nên sử dụng SQL và hàm để định dạng ngày bất kỳ có trong cơ sở dữ liệu.

Cách duy nhất để truy xuất ngày hoặc giờ trên máy khách hàng là sử dụng JavaScript. Bạn không thể thực hiện điều đó với PHP và MySQL.

Các hàm gom nhóm

Cũng giống như việc giới hạn và sắp xếp kết quả, việc gom nhóm kết quả trả về trong câu lệnh SQL cũng sử dụng một mệnh đề riêng. Tuy nhiên, mệnh đề này có khác biệt đáng kể so với các mệnh đề kia ở chỗ nó gom nhóm dữ liệu được trả lại thành các khối thông tin tương tự nhau. Ví dụ, để gom nhóm tất cả các URL theo loại, chúng ta dùng:

```
SELECT * FROM urls GROUP BY type_id;
```

Với câu lệnh này, các mẩu tin có cùng giá trị type_id sẽ được gom lại thành một dòng thay vì xuất hiện dưới dạng từng mẩu tin như trước đây. Mệnh đề `GROUP BY` thường được sử dụng với các hàm gom nhóm (hoặc kết hợp). Bảng 5.9 liệt kê các hàm này.

Bảng 5.9: *Các hàm gom nhóm của MySQL.*

Hàm	Sử dụng	Mục đích
`AVG()`	`AVG(cột)`	Trả lại giá trị trung bình của một cột.
`MIN()`	`MIN(cột)`	Trả lại giá trị nhỏ nhất của một cột.
`MAX()`	`MAX(cột)`	Trả lại giá trị lớn nhất của một cột.
`SUM()`	`SUM(cột)`	Trả lại tổng các giá trị của một cột.
`COUNT()`	`COUNT(cột)`	Trả lại số mẩu tin.

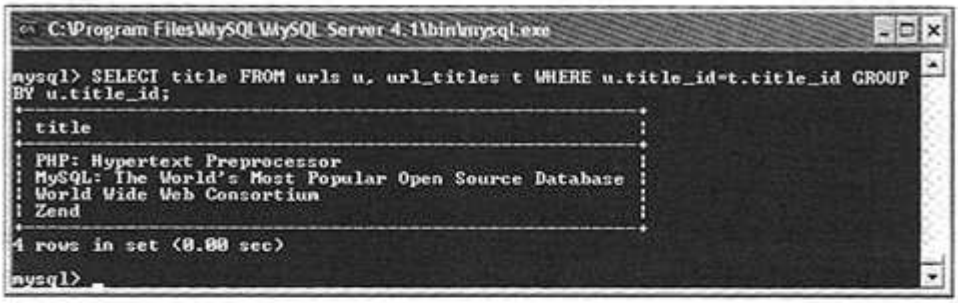
Bạn có thể áp dụng một tổ hợp bất kỳ của `WHERE`, `ORDER BY` và `LIMIT` với `GROUP BY` bằng cách viết câu truy vấn như sau:

```
SELECT cột FROM bảng WHERE mệnh đề GROUP BY cột ORDER BY cột
→ LIMIT x;
```

Thực hành gom nhóm dữ liệu theo các bước sau:

1. Chọn tất cả các tiêu đề đã được gửi (xem hình 5-31):

```
SELECT title FROM urls u, url_titles t WHERE
→ u.title_id=t.title_id GROUP BY u.title_id;
```

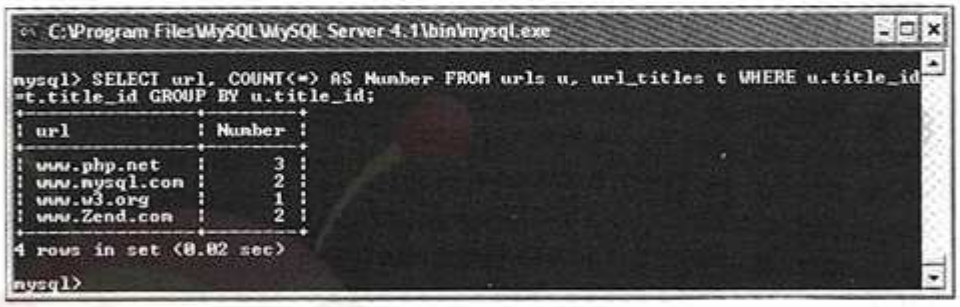


Hình 5-31: *Câu truy vấn* ***GROUP BY*** *này không sử dụng hàm.*

Đây là một trong nhiều cách để lấy các tiêu đề hiện có trong bảng urls. Trong khi có 8 URL được gửi tới dưới dạng 8 mẩu tin khác nhau (tại các thời điểm và loại khác nhau), nhưng chỉ có 4 URL khác nhau được sử dụng.

2. Đếm số lần xuất hiện của các URL (xem hình 5-32):

```
SELECT url, COUNT(*) AS Number FROM urls u, url_titles t
→ WHERE u.title_id=t.title_id GROUP BY u.title_id;
```

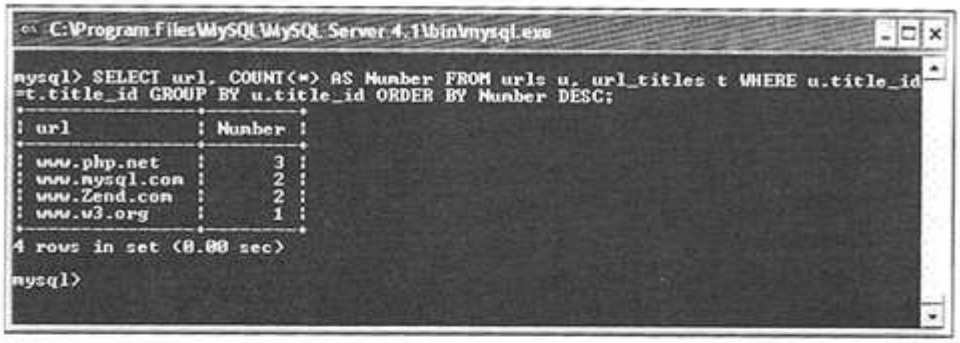


Hình 5-32: *Câu truy vấn* ***GROUP BY*** *này tính số lần xuất hiện của mỗi URL.*

Câu truy vấn này được mở rộng từ câu truy vấn trong bước 1, bằng cách bổ sung thêm hàm `COUNT()` để trả lại số lần xuất hiện của mỗi URL (mỗi lần với một loại khác nhau). Hàm `COUNT()` không nhất thiết phải áp dụng cho mỗi cột.

3. Sắp xếp URL theo số lần xuất hiện (xem hình 5-33):

```
SELECT url, COUNT(*) AS Number FROM urls u, url_titles t
→ WHERE u.title_id=t.title_id GROUP BY u.title_id ORDER BY
→ Number DESC;
```



Hình 5-33: *Cuối cùng, mệnh đề ORDER BY được bổ sung để sắp xếp các URL theo số lần xuất hiện.*

Với việc gom nhóm, bạn có thể sắp xếp các kết quả giống như với các câu truy vấn khác. Việc gán giá trị của `COUNT(*)` với tên hiệu Number sẽ giúp bạn thực hiện ý tưởng này.

`GROUP BY` sẽ gom các giá trị `NULL` lại với nhau vì chúng đều không có giá trị.

Hàm `COUNT()` chỉ tính các mẩu tin không có giá trị `NULL`.

Mệnh đề `GROUP BY` và các hàm được liệt kê trên đây thường tốn nhiều thời gian để thực hiện. Do đó, bạn nên thử nghiệm các câu truy vấn với trình quản lý mysql để đảm bảo chúng chính xác trước khi dùng trong mã kịch bản của mình.

Hàm `DISTINCT()` được dùng để loại bỏ các giá trị trùng lắp và thường được áp dụng cho một cột:

```
SELECT DISTINCT(cột) FROM bảng;
```

Câu truy vấn dạng này thường được sử dụng với `GROUP BY` để tính số mẩu tin duy nhất, dựa trên một nhóm loại nhất định.

Chỉ mục

Chỉ mục là một hệ thống đặc biệt giúp cải thiện khả năng vận hành của các cơ sở dữ liệu. Bằng cách thiết lập các chỉ mục trên các bảng, chúng ta thông báo cho MySQL chú ý đến cột đó. Trong thực tế, MySQL tạo ra các tập tin phụ trợ để chứa và theo dõi các chỉ mục một cách hiệu quả.

MySQL cho phép tới 32 chỉ mục cho mỗi bảng và mỗi chỉ mục có thể có tới 16 cột. Các chỉ mục với nhiều cột sẽ rất hữu hiệu khi thường xuyên thực hiện thao tác tìm kiếm trên nhiều trường (ví dụ, last_name và first_name, city và state...).

Tuy nhiên, bạn cũng không nên quá lạm dụng chỉ mục. Trong khi nó cải thiện tốc độ truy xuất cơ sở dữ liệu, nó lại làm chậm quá trình thay đổi dữ liệu trong một cơ sở dữ liệu (vì các thay đổi này phải được ghi nhận trong bảng chỉ mục). Các chỉ mục được sử dụng tốt nhất trên các cột:

- Thường được sử dụng với mệnh đề **WHERE** trong một câu truy vấn.
- Thường được sử dụng trong mệnh đề **ORDER BY** của câu truy vấn.
- Thường được dùng như là điểm trọng tâm của một liên kết.
- Có nhiều giá trị khác nhau (các cột với các giá trị số lặp lại có thể không cần phải tạo chỉ mục).

MySQL có ba dạng chỉ mục: **INDEX**, **UNIQUE** (đòi hỏi mỗi mẩu tin có một giá trị duy nhất) và **PRIMARY KEY** (là một trường hợp đặc biệt của **UNIQUE**).

Thay đổi bảng

Thuật ngữ SQL ***ALTER*** *được dùng chủ yếu để thay đổi cấu trúc của một bảng trong cơ sở dữ liệu, kể cả việc bổ sung các chỉ mục. Thông thường, lệnh này đề cập đến việc bổ sung, xóa hoặc thay đổi các cột. Nó cũng được dùng để thay đổi tên của bảng, khóa cũng như các chỉ mục. Việc thực hiện các thay đổi về cấu trúc dữ liệu là một việc làm bình thường trong thiết kế và cài đặt cơ sở dữ liệu. Cú pháp cơ bản của* ***ALTER*** *như sau:*

```
ALTER TABLE <bảng> <mệnh đề>;
```

Bảng 5.10 liệt kê các mệnh đề được dùng phổ biến với ALTER.

Bảng 5.10: *Các dạng khác nhau của câu lệnh* ***ALTER***

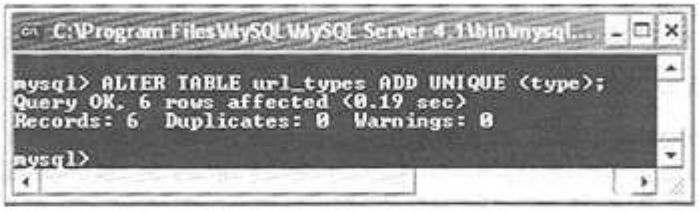
Mệnh đề	Sử dụng	Ý nghĩa
ADD COLUMN	ALTER TABLE <bảng> ADD COLUMN <cột> VARCHAR(10)	Thêm một cột mới vào cuối bảng.
CHANGE COLUMN	ALTER TALBE <bảng> CHANGE COLUMN <cột> <cột> VARCHAR(60)	Cho phép thay đổi kiểu dữ liệu và các thuộc tính của một cột trong bảng.
DROP COLUMN	ALTER TABLE <bảng> DROP COLUMN <cột>	Loại bỏ một cột ra khỏi một bảng, kể cả dữ liệu của nó.
ADD INDEX	ALTER TABLE <bảng> ADD INDEX <chỉ mục> (<tên cột>)	Thêm một chỉ mục mới trên cột <tên cột>.
DROP INDEX	ALTER TABLE <bảng> DROP INDEX <chỉ mục>	Xóa bỏ một chỉ mục hiện có.
RENAME AS	ALTER TABLE <bảng> RENAME AS <tên mới>	Đổi tên của một bảng.

Với những kiến thức có được, chúng ta sẽ điều chỉnh lại bảng bằng cách bổ sung các chỉ mục.

Thực hành bổ sung một chỉ mục cho một bảng đã có theo các bước sau:

1. Bổ sung một chỉ mục trên cột type trong bảng url_types (xem hình 5-34):

```
ALTER TABLE url_types ADD UNIQUE (type);
```

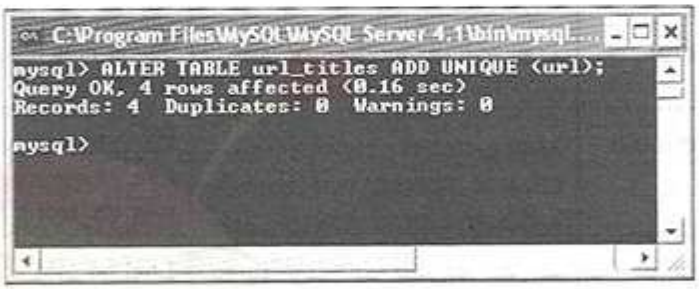


Hình 5-34: *MySQL sẽ thông báo kết quả thực hiện câu truy vấn và số mẩu tin bị ảnh hưởng khi sử dụng các câu lệnh* ***ALTER***

Bảng url_types đã có một chỉ mục khóa chính trên trường type_id. Vì type cũng là một trường thường được tham chiếu và giá trị của nó cũng duy nhất trên toàn bộ các mẩu tin, nên chúng ta bổ sung một chỉ mục **UNIQUE** cho bảng.

2. Bổ sung một chỉ mục trên cột url trong bảng url_titles (xem hình 5-35):

```
ALTER TABLE url_titles ADD UNIQUE (url);
```

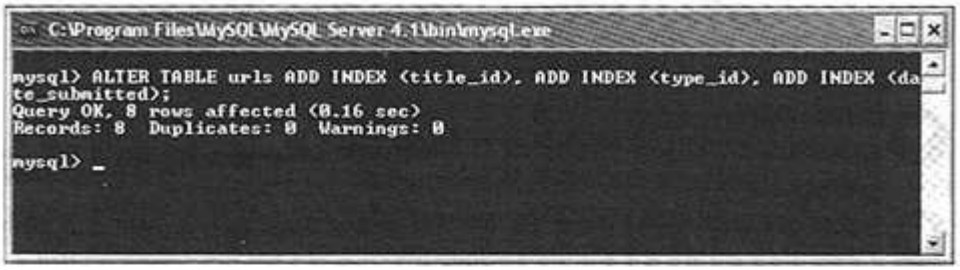


Hình 5-35: *Một chỉ mục mới được thêm vào trong bảng url_titles.*

Cũng tương tự như bảng url_types, bảng url_titles cũng có một chỉ mục khóa chính trên trường title_id. Vì url cũng là một trường thường được tham chiếu và giá trị của nó cũng duy nhất trên nhiều mẩu tin, chúng ta bổ sung một chỉ mục **UNIQUE** cho bảng. Cả hai trường title và description không được sử dụng giống như vậy nên ta sẽ không tạo chỉ mục cho các trường này.

3. Bổ sung các chỉ mục cho bảng urls (xem hình 5-36):

```
ALTER TABLE urls ADD INDEX (title_id), ADD INDEX (type_id), ADD
INDEX (date_submitted);
```

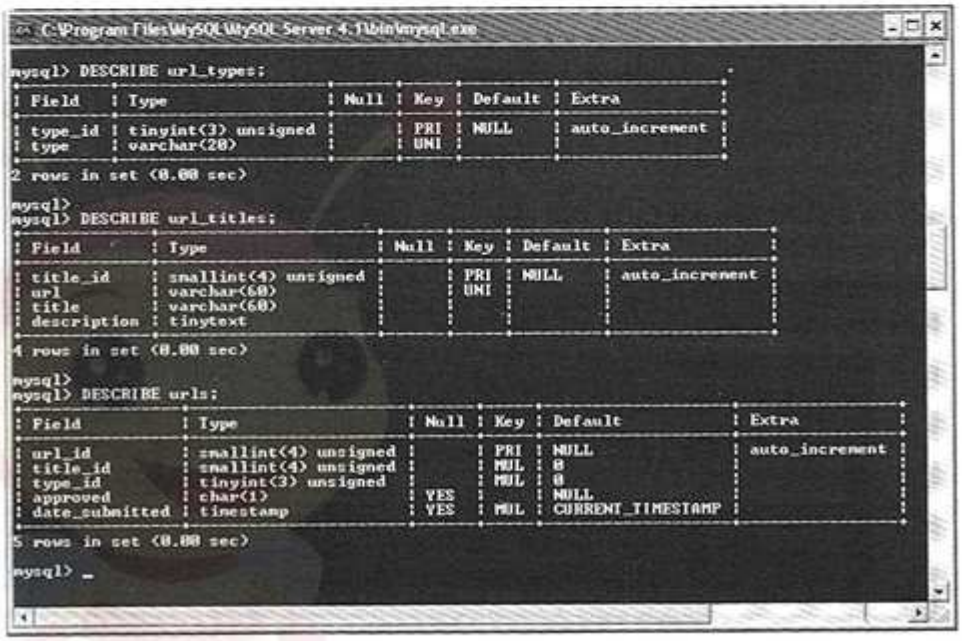


Hình 5-36: *Với SQL, chúng ta có thể bổ sung nhiều chỉ mục cùng lúc.*

Cuối cùng, chúng ta bổ sung ba chỉ mục cho bảng urls. Đầu tiên là các chỉ mục không có tính duy nhất cho các khóa ngoại (title_id và type_id) vì các trường này được sử dụng trong các điều kiện **WHERE** và liên kết. Tiếp đến là chỉ mục cho ngày gửi, vốn cũng được sử dụng trong các điều kiện. Trường url_id của bảng urls được tự động tạo chỉ mục khi thiết lập nó là khóa chính và cột approved không cần tạo chỉ mục vì giá trị của nó chỉ là Y hoặc N.

4. Xem cấu trúc hiện tại của mỗi bảng (xem hình 5-37):

```
DESCRIBE url_types;

DESCRIBE url_titles;

DESCRIBE urls;
```



Hình 5-37: *Sử dụng lệnh DESCRIBE để xem chi tiết cấu trúc của một bảng.*

Thuật ngữ SQL **DESCRIBE** sẽ cho biết thông tin về tên, thứ tự, kiểu của các cột trong bảng cùng các chỉ mục. Nó cũng cho biết một trường có thể là **NULL** hay không, giá trị mặc định được thiết lập là gì...

Các chỉ mục sẽ kém hiệu quả trên các cột có chiều dài thay đổi, vì MySQL nói chung thường xử lý chậm trên các trường có chiều dài thay đổi.

Các chỉ mục có thể được đặt tên khi chúng được tạo ra. Nếu bạn không xác định cụ thể tên gọi, nó sẽ được lấy theo tên cột mà chúng áp dụng.

*Vì lệnh **ALTER** có thể có những tác động nghiêm trọng đến một bảng, bạn nên sao lưu lại bảng trước khi thực hiện lệnh này.*

*Từ **COLUMN** trong phần lớn câu lệnh **ALTER** là tùy chọn.*

*Khi bổ sung một cột mới vào trong một bảng, chúng ta sử dụng mệnh đề **AFTER <tên cột>** để cho biết cột mới sẽ được đặt vào nơi nào trong bảng.*

```
ALTER TABLE <tên bảng> ADD COLUMN last_name VARCHAR(25)
→ AFTER first_name;
```

*Chương 9 "Phát triển ứng dụng Web", sẽ đề cập đến một thuật ngữ SQL khác, **EXPLAIN**. Lệnh này sẽ giúp tinh chỉnh các câu lệnh SQL.*

Chương 6:

SỬ DỤNG PHP VÀ MYSQL

Trong số các ngôn ngữ có thể sử dụng với MySQL, bao gồm PERL, Java, C, C++, Python..., PHP có lẽ là ngôn ngữ phổ biến nhất. Việc tích hợp mạnh mẽ PHP và MySQL cùng với triết lý mã nguồn mở, là lý do khiến nhiều nhà lập trình gắn chặt với nó.

Việc hỗ trợ cơ sở dữ liệu MySQL đã được đưa vào theo mặc định từ khi PHP 4 ra đời. Phiên bản hiện tại của PHP đã kèm theo một số bản sửa chữa, cập nhật và hỗ trợ cho MySQL 4.0.

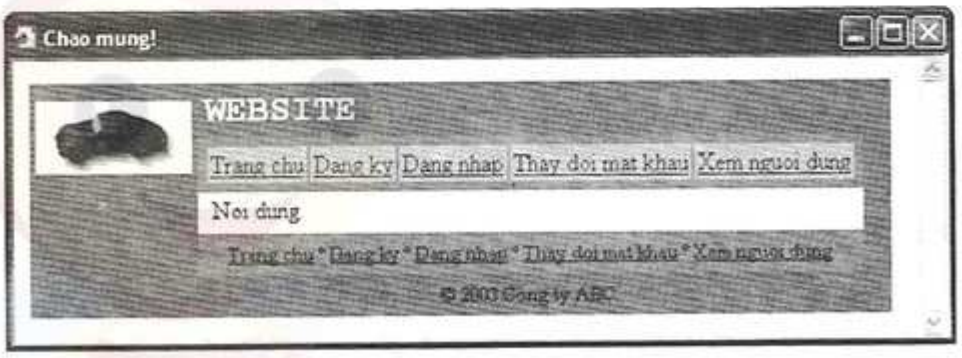
Trong chương này, chúng ta sẽ sử dụng cơ sở dữ liệu *sitename* đã được tạo trong Chương 4 "Giới thiệu về SQL và MySQL" để xây dựng một giao diện PHP, nhằm tương tác với bảng users. Kiến thức bạn có được cho đến bây giờ cùng với các ví dụ được nêu trong chương này sẽ là cơ sở cho tất cả các ứng dụng Web dựa trên PHP và MySQL, vì các nguyên lý được đưa ra ở đây sẽ giống nhau cho mọi trường hợp.

Tạo khuôn mẫu

Vì tất cả trang được tạo trong chương này và chương sau sẽ là thành phần của cùng một ứng dụng Web, nên chúng ta sẽ sử dụng chung một hệ thống khuôn mẫu. Ta sẽ đặt phần đầu và phần cuối trang vào các tập tin ngoài, giống như đã thực hiện trong Chương 3 "Tạo Website động".

Thực hành tạo tập tin đầu trang theo các bước sau:

1. Thiết kế khuôn mẫu trong trình soạn thảo văn bản hoặc một trình soạn thảo WYSIWYG (xem hình 6-1).



Hình 6-1: *Để tạo các tập tin khuôn mẫu, trước hết chúng ta sẽ thiết kế một trang tổng quát làm mô hình.*

Như đã được nêu trong Chương 3, khi sử dụng hệ thống khuôn mẫu (để tách phần HTML ra khỏi PHP), chúng ta tạo thiết kế HTML trước, rồi sau đó ngắt nó ra thành các phần tương ứng.

2. Tạo một hồ sơ mới trong trình soạn thảo văn bản bằng cách sử dụng khuôn mẫu.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
→ Transitional//EN" "http://www.w3.org/TR/2000/REC-xhtml1-
→ 20000126/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">

<head>

<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />
```

3. Tạo tiêu đề trang

```
<title><?php echo $page_title; ?></title>

</head>
```

Tạo tiêu đề trang bằng cách in giá trị của một biến để dễ dàng điều chỉnh trên cơ sở từng trang một.

4. Bắt đầu phần thân và tạo ra dòng đầu tiên của bảng.

```
<body>

<table border="0" cellspacing="0" cellpadding="4">

<tr>

<td rowspan="2" bgcolor="#999966"><img
→ src="images/logo.jpg" alt="LOGO" width="100"
→ height="47" /></td>

<td width="*" bgcolor="#999966"><font color="#FFFFFF"
→ size="+2" face="Courier New, Courier, mono">
→<strong>Website</strong></font></td>

<td width="10" rowspan="2" bgcolor="#999966"> </td>

</tr>
```

Bạn có thể thay đổi mã lệnh HTML nếu sử dụng một thiết kế khác.

5. Tạo ra dòng chứa các liên kết di chuyển.

```
<tr>

<td bgcolor="#CC9933">
```

```
<table width="100%" border="0" cellspacing="2"
→ cellpadding="2">

<tr>

<td width="20%" align="center" bgcolor="#FFCC66">
→ <a href="index.php">Trang chu</a></td>

<td width="20%" align="center" bgcolor="#FFCC66">
→ <a href="register.php">Dang ky</a></td>

<td width="20%" align="center" bgcolor="#FFCC66">
→ <a href="login.php">Dang nhap</a></td>

<td width="20%" align="center" bgcolor="#FFCC66"
→ nowrap="nowrap">
→ <a href="change_password.php">Thay doi mat khau</a></td>

<td width="20%" align="center" bgcolor="#FFCC66"
→ nowrap="nowrap">
→ <a href="view_users.php">Xem nguoi dung</a></td>

</tr>

</table>

</td>

</tr>
```

6. Tạo ra dòng nội dung.

```
<tr>

<td bgcolor="#999966"> </td>

<td bgcolor="#FFFFFF">

<!-- Bat dau noi dung trang -->
```

Bạn nên bổ sung ghi chú HTML vào trong khuôn mẫu nhằm cho biết nơi kết thúc của mã lệnh khuôn mẫu cũng như mã lệnh cụ thể của từng trang xuất hiện.

7. Lưu tập tin với tên gọi **header.inc**. Mã lệnh đầy đủ của tập tin này được thể hiện trong đoạn mã 6.1.

Đoạn mã 6.1: *Tệp tin đầu trang.*

```
<!DOCTYPE html PUBLIC
→ "-//W3C//DTD XHTML 1.0 Transitional//EN"
→ "http://www.w3.org/TR/2000/REC-xhtml1-20000126/DTD/xhtml1-
→transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
```

```
→ lang="en">
<head>
<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />
<title><?php echo $page_title; ?></title>
</head>
<body>
<!-- Doan ma 6.1 - header.inc -->
<table border="0" cellspacing="0" cellpadding="4">
<tr> <!-- Dong tren cung -->
<td rowspan="2" bgcolor="#999966"><img src="images/logo.jpg"
→ alt="LOGO" width="100" height="47" /></td>
<td width="*" bgcolor="#999966"><font color="#FFFFFF"
→ size="+2" face="Courier New, Courier, mono">
→ <strong>WEBSITE</strong></font></td>
<td width="10" rowspan="2" bgcolor="#999966"> </td>
</tr>
<tr> <!-- Dong chua thanh di chuyen -->
<td bgcolor="#CC9933">
<table width="100%" border="0" cellspacing="2"
→ cellpadding="2">
<tr>
<td width="20%" align="center" bgcolor="#FFCC66">
→ <a href="index.php">Trang chu</a></td>
<td width="20%" align="center" bgcolor="#FFCC66">
→ <a href="register.php">Dang ky</a></td>
<td width="20%" align="center" bgcolor="#FFCC66">
→ <a href="login.php">Dang nhap</a></td>
<td width="20%" align="center" bgcolor="#FFCC66"
→ nowrap="nowrap">
→ <a href="change_password.php">Thay doi mat khau</a></td>
<td width="20%" align="center" bgcolor="#FFCC66"
→ nowrap="nowrap">
→ <a href="view_users.php">Xem nguoi dung</a></td>
</tr>
</table>
</td>
</tr>
```

```
<tr> <!-- Dong noi dung -->
<td bgcolor="#999966"> </td>
<td bgcolor="#FFFFFF">
<!-- Bat dau noi dung trang -->
```

Thực hành tạo tập tin cuối trang thực hiện theo các bước sau:

1. Tạo một hồ sơ văn bản mới, kết thúc dòng nội dung trang.

```
<!-- Doan ma 6.2 - footer.inc -->
<!-- Ket thuc noi dung trang --></td>
<td width="10" bgcolor="#999966"> </td>
</tr>
```

2. Thực hiện dòng chứa thanh di chuyển cuối trang.

```
<tr>
<td bgcolor="#999966"> </td>
<td bgcolor="#CC9933"><div align="center"><small><a
→ href="index.php">Trang chu</a> &ordm; <a
→ href="register.php">Dang ky</a> &ordm; <a
→ href="login.php">Dang nhap</a> &ordm; <a
→ href="change_password.php">Thay doi mat khau</a> &ordm;
→ <a href="view_users.php">Xem nguoi dung</a></small>
→</div></td>
<td width="10" bgcolor="#999966"> </td>
</tr>
```

3. Bổ sung dòng chứa thông tin bản quyền và hoàn tất trang.

```
<tr>
<td bgcolor="#999966"> </td>
<td bgcolor="#999966">
<div align="center"><font size="-1">© 2003 Cong ty ABC.
→</font></div></td>
<td bgcolor="#999966"> </td>
</tr>
</table>
</body>
</html>
```

4. Lưu tập tin với tên gọi **footer.inc**. Mã lệnh đầy đủ của tập tin này được thể hiện trong đoạn mã 6.2.

Đoạn mã 6.2: *Tập tin cuối trang.*

```
<!-- Doan ma 6.2 - footer.inc -->
<!-- Ket thuc noi dung trang --></td>
<td width="10" bgcolor="#999966"> </td>
</tr>
<tr><!-- Dong chua thanh di chuyen cuoi trang -->
<td bgcolor="#999966"> </td>
<td bgcolor="#CC9933"><div align="center"><small><a
→ href="index.php">Trang chu</a> &ordm; <a
→ href="register.php">Dang ky</a> &ordm; <a
→ href="login.php">Dang nhap</a> &ordm; <a
→ href="change_password.php">Thay doi mat khau</a> &ordm; <a
→ href="view_users.php">Xem nguoi dung</a></small></div>
→ </td>
<td width="10" bgcolor="#999966"> </td>
</tr>
<tr><!-- Dong chua thong tin ban quyen -->
<td bgcolor="#999966"> </td>
<td bgcolor="#999966"><div align="center"><font size="-1">
→ © 2003 Cong ty ABC.</font></div></td>
<td bgcolor="#999966"> </td>
</tr>
</table>
</body>
</html>
```

Thực hành tạo trang chủ thực hiện theo các bước sau:

1. Tạo một hồ sơ PHP mới.

```
<?php # Doan ma 6.3 - index.php
```

2. Định nghĩa tiêu đề trang và kèm theo tập tin đầu trang.

```
$page_title = 'Chao mung!';
include ('templates/header.inc');
```

3. Với ứng dụng này, chúng ta đặt tập tin đầu và cuối trang vào trong thư mục templates, phía trong thư mục chính. Để tham khảo chúng, ta sử dụng cú pháp *thư mục/tên tập tin* trong câu lệnh `include()`.

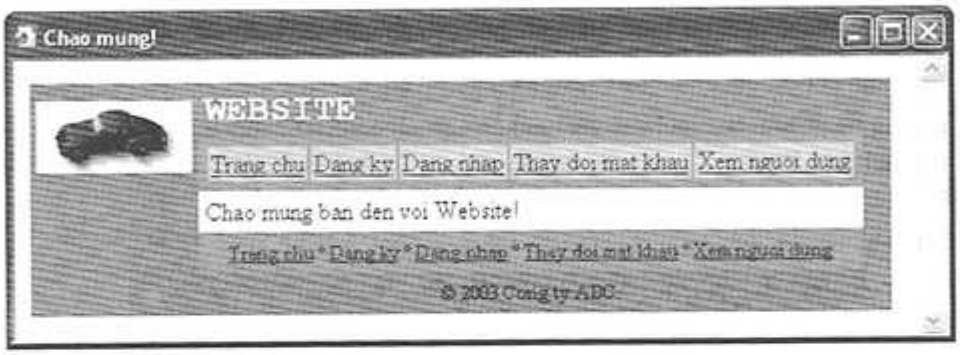
4. Tạo ra nội dung của trang chủ.

```
echo '<p>Chao mung ban den voi Website!</p>';
```

5. Kèm theo tập tin cuối trang và hoàn tất trang.

```
include ('templates/footer.inc');

?>
```

6. Lưu tập tin với tên gọi **index.php**, tải nó cùng với các tập tin **header.inc** và **footer.inc** (hai tập tin này đặt trong thư mục templates) lên máy chủ rồi chạy thử trong trình duyệt (xem hình 6-2). Mã lệnh đầy đủ của tập tin index.php được thể hiện trong đoạn mã 6.3.



Hình 6-2: *Trang chủ được tạo động.*

Đoạn mã 6.3: *Trang chủ của site.*

```
<?php # Doan ma 6.3 - index.php
$page_title = 'Chao mung!';
include ('templates/header.inc');
echo '<p>Chao mung ban den voi Website!</p>';
include ('templates/footer.inc');
?>
```

Tham khảo mục "Tổ chức hồ sơ của bạn" trong phần sau để xem trước cấu trúc của site này.

Bạn có thể sử dụng bất kỳ phần mở rộng nào cho các tập tin khuôn mẫu, kể cả ***.inc*** *và* ***.html****.*

Kết nối với MySQL và truy xuất cơ sở dữ liệu

Bước đầu tiên khi làm việc với MySQL là kết nối với máy chủ. Chúng ta sử dụng hàm `mysql_connect()` để thực hiện điều này.

```
$dbc = mysql_connect($host, $user, $password);
```

Các tham số được gửi cho hàm (máy chủ, tên đăng ký và mật khẩu) được dựa trên cơ sở từng tài khoản và độ ưu tiên được thiết lập trong cơ sở dữ liệu MySQL. Thông thường, máy chủ được xác định là localhost, nhưng không nhất thiết phải như vậy (việc sử dụng một địa chỉ máy chủ cụ thể sẽ cho phép truy xuất đến MySQL chạy trên một máy chủ khác).

Nếu kết nối được thực hiện, biến **$dbc** sẽ trở thành điểm tham chiếu. Phần lớn các hàm PHP làm việc với MySQL đều nhận vào tham số này như một tham số mặc định, nhưng nếu bị bỏ qua, chúng sẽ tự động sử dụng kết nối hiện đang được mở.

Sau khi đã kết nối với MySQL, chúng ta cần chọn cơ sở dữ liệu để làm việc. Điều này tương đương với việc sử dụng lệnh USE **<tên cơ sở dữ liệu>** trong trình quản lý mysql và được thực hiện với hàm **mysql_select_db()**:

```
mysql_select_db($database_name);
```

Chúng ta sẽ bắt đầu phần thiết lập kết nối với MySQL bằng cách tạo một tập tin đặc biệt chỉ dùng cho mục đích này. Các mã kịch bản PHP khác cần đến kết nối cơ sở dữ liệu sẽ sử dụng tập tin này.

Thực hành kết nối với cơ sở dữ liệu theo các bước sau:

1. Tạo một hồ sơ PHP trong trình soạn thảo văn bản.

   ```
   <?php # Doan ma 6.4 - mysql_connect.php
   ```

2. Thiết lập địa chỉ máy chủ, tên đăng nhập, mật khẩu và tên cơ sở dữ liệu.

   ```
   DEFINE ('DB_USER', 'username');
   DEFINE ('DB_PASSWORD', 'password');
   DEFINE ('DB_HOST', 'localhost');
   DEFINE ('DB_NAME', 'sitename');
   ```

Chúng ta thiết lập các giá trị này dưới dạng các hằng vì lý do an toàn (chúng sẽ không thay đổi được), tuy nhiên điều đó không bắt buộc. Nói chung, việc thiết lập các giá trị này theo một loại biến nào đó sẽ giúp ta tách các tham số cấu hình ra khỏi các hàm có sử dụng chúng. Tuy nhiên, một lần nữa xin nhấn mạnh đây là điều không bắt buộc.

Khi viết mã kịch bản của riêng mình, hãy thay đổi các giá trị để phù hợp với thực tế của bạn.

3. Kết nối với MySQL

   ```
   $dbc = mysql_connect (DB_HOST, DB_USER, DB_PASSWORD);
   ```

4. Khi thực hiện kết nối thành công, hàm **mysql_connect()** sẽ trả lại một liên kết ứng với kết nối được mở. Liên kết này sẽ được gán cho biến **$dbc** và

chúng ta sẽ dùng nó mỗi khi sử dụng hàm MySQL có cần đến kết nối cơ sở dữ liệu.

5. Chọn cơ sở dữ liệu sẽ sử dụng và đóng trang PHP.

```
mysql_select_db (DB_NAME);

?>
```

Bước cuối cùng là báo cho MySQL biết cơ sở dữ liệu sẽ sử dụng. Việc không chọn được cơ sở dữ liệu sẽ dẫn đến các vấn đề trong các mã kịch bản tiếp theo. Tuy nhiên, nếu ứng dụng sử dụng nhiều cơ sở dữ liệu, bạn không nhất thiết phải chọn chúng tại bước này.

6. Lưu tập tin với tên gọi **mysql_connect.php**. Mã lệnh đầy đủ của tập tin này được thể hiện trong đoạn mã 6.4. Chương 8 "An toàn", sẽ đề cập đến các khía cạnh an toàn của phần mở rộng tập tin chi tiết hơn. Còn bây giờ, bạn chỉ cần hiểu là do tập tin này chứa các thông tin "nhạy cảm" và cần phải được đặt tại một vị trí an toàn. Chúng ta sử dụng phần mở rộng cho tập tin là **.php** thay vì **.inc**. Xem thêm phần *"Tổ chức hồ sơ của bạn"* để biết chi tiết.

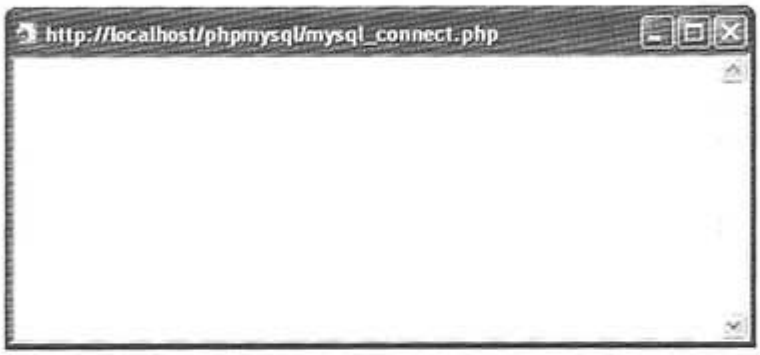
7. Tải tập tin lên máy chủ và đặt nó trong thư mục gốc của Website (xem hình 6-3).

Hình 6-3: *Một dạng thể hiện trực quan của các hồ sơ Web trên máy chủ, trong đó* ***mysql_connect.php*** *không được đặt trong thư mục chính của Web.*

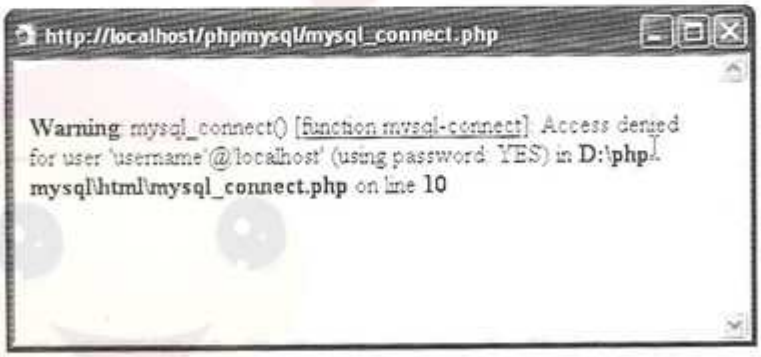
Vì tập tin chứa thông tin "nhạy cảm", nó cần phải được quản lý một cách cẩn thận. Nếu được, bạn nên đặt nó trong một thư mục ngay trước thư mục Web hoặc một thư mục nào đó phía ngoài thư mục này. Như vậy sẽ làm cho tập tin không thể truy cập được từ một trình duyệt Web.

8. Tạm thời, chúng ta đặt một bản sao của mã kịch bản này trong thư mục Web và chạy thử nó trong trình duyệt (xem hình 6-4 và 6-5).

Để thử mã kịch bản, bạn cần phải đặt một bản sao trên máy chủ để nó có thể truy xuất được từ trình duyệt Web (nghĩa là nó phải nằm trong thư mục Web). Nếu mã kịch bản làm việc chính xác, kết quả sẽ là một trang trắng (xem hình 6-4). Nếu thấy xuất hiện dòng Access denied... nghĩa là tên đăng nhập, mật khẩu và máy chủ không cho phép bạn truy xuất cơ sở dữ liệu.



Hình 6-4: *Nếu mã kịch bản làm việc chính xác, kết quả cuối cùng sẽ là một trang trắng (không có mã HTML được phát sinh từ mã kịch bản này).*



Hình 6-5: *Nếu mã kịch bản gặp phải bất kỳ vấn đề gì, bạn sẽ thấy các thông điệp giống như trong hình.*

9. Loại bỏ bản sao tạm thời của tập tin **mysql_connect.php** khỏi thư mục Web.

Đoạn mã 6.4: *Mã kịch bản* ***mysql_connect.php*** *sẽ được dùng trong các mã kịch bản PHP khác trong ứng dụng để thiết lập kết nối với MySQL và chọn cơ sở dữ liệu sẽ sử dụng.*

```
<?php # Doan ma 6.4 - mysql_connect.php

// Thiet lap thong tin truy xuat CSDL duoi dang cac hang.

DEFINE ('DB_USER', 'username');

DEFINE ('DB_PASSWORD', 'password');

DEFINE ('DB_HOST', 'localhost');

DEFINE ('DB_NAME', 'sitename');

// Ket noi va chon CSDL.

$dbc = mysql_connect (DB_HOST, DB_USER, DB_PASSWORD);

mysql_select_db (DB_NAME);

?>
```

Các giá trị mà bạn sử dụng trong Chương 4 "Giới thiệu về SQL và MySQL", Chương 5 "SQL và MySQL nâng cao", để đăng nhập vào trình quản lý mysql cũng có thể được sử dụng để tạo ra kết nối trong chương này.

Nếu nhận được một thông báo lỗi cho biết ***mysql_connect()*** *là một hàm chưa được định nghĩa, điều đó có nghĩa là PHP chưa được biên dịch hoặc thiết lập để hỗ trợ MySQL.*

Nếu thấy thông báo Access denied... khi chạy mã kịch bản (xem hình 6-5), bạn nên sử dụng trình quản lý mysql để thử lại các thông tin kết nối của mình.

PHP có hàm ***mysql_pconnect()*** *để tạo ra một kết nối thường trực với MySQL. Việc sử dụng nó buộc bạn phải thay đổi cách nghĩ về ứng dụng của mình vì kết nối cơ sở dữ liệu sẽ được sử dụng theo một cách thức rất khác.*

Một khi đã viết xong tập tin ***mysql_connect.php****, bạn có thể dễ dàng thực hiện các thay đổi cho các dòng* ***define()*** *để sử dụng mã kịch bản này cho các dự án khác.*

Tổ chức hồ sơ của bạn

Khái niệm cấu trúc site đã được đề cập trong Chương 3 khi phát triển ứng dụng Web đầu tiên. Bây giờ, chúng ta bắt đầu sử dụng một mã kịch bản kết nối cơ sở dữ liệu, chủ đề này sẽ được đề cập lại vì nó rất quan trọng.

Nếu thông tin kết nối cơ sở dữ liệu (tên đăng nhập, mật khẩu và cơ sở dữ liệu) rơi vào tay một kẻ xấu, nó sẽ được sử dụng để ăn cắp thông tin của bạn hoặc phá hủy toàn bộ cơ sở dữ liệu. Do đó, bạn không thể đặt mã kịch bản như ***mysql_connect.php*** *quá lơi lỏng.*

Đề nghị quan trọng đầu tiên để tạo an toàn cho các tập tin như vậy là chứa nó phía ngoài thư mục Web. Ví dụ, thư mục html trong hình 6-3 là thư mục gốc của Web, khi đó bạn không nên chứa tập tin ***mysql_connect.php*** *bất kỳ nơi nào trong thư mục này. Điều này nghĩa là nó không thể truy cập được từ trình duyệt Web.*

Bạn cũng có thể không cần quá cẩn thận đến như vậy, miễn sao mã nguồn của mã kịch bản này không thể xem trong trình duyệt Web là được.

Đề nghị thứ hai, bạn nên sử dụng phần mở rộng cho tập tin là ***.php*** *cho các mã kịch bản kết nối cơ sở dữ liệu. Một cấu hình thích hợp sẽ buộc máy chủ thực hiện mã kịch bản thay vì thể hiện mã lệnh của nó trong các tập tin như vậy. Nếu sử dụng tập tin* ***.inc****, toàn bộ nội dung của nó sẽ được hiển thị nếu người dùng truy xuất trực tiếp tập tin đó.*

Xử lý lỗi

Xử lý lỗi là một việc quan trọng trong bất kỳ mã kịch bản nào, thậm chí còn quan trọng hơn rất nhiều khi làm việc với cơ sở dữ liệu, vì xác suất xảy ra lỗi sẽ tăng lên một cách đáng kinh ngạc. Các lỗi thường gặp là:

- Không kết nối được với cơ sở dữ liệu.
- Không chọn được cơ sở dữ liệu.
- Không thể thực hiện được câu truy vấn.
- Không có kết quả nào được trả lại từ các câu truy vấn.

Kinh nghiệm sẽ cho bạn biết tại sao các lỗi này lại thường xuyên xảy ra, nhưng việc ngay lập tức thấy được vấn đề khi chạy mã kịch bản sẽ giúp tiết kiệm được rất nhiều thời gian dò lỗi. Để mã kịch bản cung cấp các thông tin về lỗi xảy ra, bạn nên sử dụng các hàm `mysql_error()` và `mysql_errno()`. Hàm thứ nhất in ra thông báo lỗi dạng văn bản và hàm thứ hai sẽ trả lại mã lỗi ở dạng số.

Cùng với các hàm này là hai thuật ngữ PHP mà ta có thể sử dụng để xử lý lỗi: `@` và `die()`. Biểu tượng `@` khi được dùng phía trước tên một hàm, sẽ loại bỏ tất cả các thông báo lỗi hoặc cảnh báo nào mà hàm có thể phát sinh. Hàm có sức mạnh hơn là hàm `die()`. Hàm này sẽ kết thúc việc thực hiện một mã kịch bản và gửi thông báo lỗi phía trong dấu ngoặc đến trình duyệt Web. Cú pháp sử dụng thông thường của hàm `die()` như sau:

```
$dbc = mysql_connect(DB_HOST, DB_USER, DB_PASSWORD) or
→ die(mysql_error());
```

Bạn nên sử dụng hàm **die()** với bất kỳ hàm nào mà việc thực hiện thành công của nó là một điều bắt buộc cho phần còn lại của mã kịch bản (ví dụ, kết nối với MySQL và chọn cơ sở dữ liệu). Sử dụng **@** với các hàm có thể có vấn đề nhưng không nhất thiết phải ngừng mã kịch bản (như kèm theo các tập tin HTML hoặc thực hiện một câu truy vấn).

Thực hành xử lý lỗi theo các bước sau:

1. Mở tập tin **mysql_connect.php** (tham khảo mã kịch bản 6.4) trong trình soạn thảo văn bản.
2. Thay đổi mã lệnh kết nối để đưa vào hàm **die()** và **mysql_error()**.

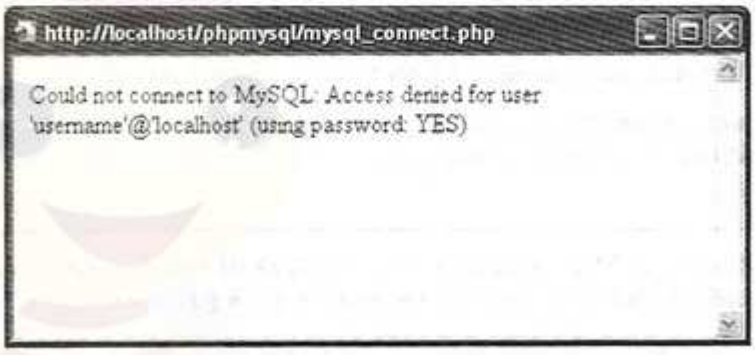
```
$dbc = @mysql_connect (DB_HOST, DB_USER, DB_PASSWORD) OR
→ die ('Could not connect to MySQL: ' . mysql_error() );
```

Vì hàm **die()** có thể nhận vào một chuỗi bất kỳ trong dấu ngoặc của nó, bạn có thể tạo ra các thông báo lỗi của riêng mình. Ở đây, chúng ta sử dụng hàm **mysql_error()** cùng với một chuỗi văn bản bằng cách kết hợp chúng lại với nhau. Biểu tượng **@** được dùng để ngăn các thông báo lỗi mà PHP tự động in ra.

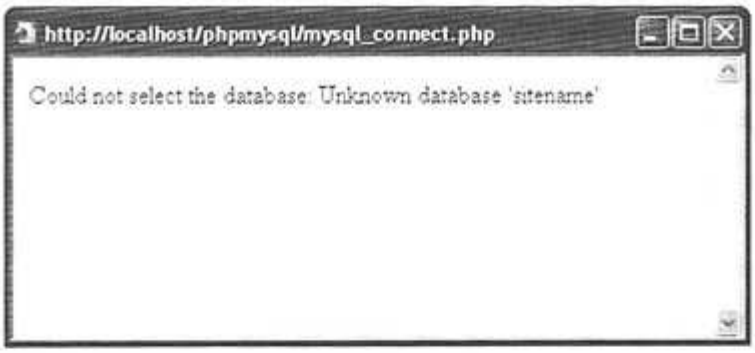
3. Thay đổi mã lệnh chọn cơ sở dữ liệu để sử dụng các ý tưởng mới này.

```
mysql_select_db (DB_NAME) OR die ('Could not select the
→ database: ' . mysql_error() );
```

4. Lưu tập tin và tải lên máy chủ. Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 6.5.
5. Kiểm tra xem các thay đổi này ảnh hưởng thế nào đến việc thực hiện các mã kịch bản bằng cách thay đổi các giá trị **DB_USER**, **DB_PASSWORD**, **DB_HOST**, **DB_NAME** với mục đích tạo ra lỗi (xem hình 6-6 và 6-7).



Hình 6-6: *Nếu có một vấn đề trong việc kết nối với MySQL, một thông báo sẽ được hiển thị và mã kịch bản sẽ ngừng lại (so sánh với hình 6-5).*



Hình 6-7: *Nếu mã kịch bản có thể kết nối với MySQL nhưng không thể chọn được cơ sở dữ liệu, nó sẽ báo cho bạn biết tại sao.*

Tùy vào thiết lập máy chủ, bạn có thể cần phải tạm thời di chuyển một bản sao của mã kịch bản này vào trong thư mục Web để có thể thử nó trong trình duyệt. Hãy nhớ loại bỏ nó khỏi thư mục Web một khi mã kịch bản đã làm việc tốt.

Đoạn mã 6.5: *Với phiên bản trước của* ***mysql_connect.php****, chúng ta bổ sung thêm các kỹ thuật xử lý lỗi để tạo ra các kết quả "chuyên nghiệp" hơn.*

```
<?php # Doan ma 6.5 - mysql_connect.php

// Thiet lap thong tin truy xuat CSDL duoi dang cac hang.

DEFINE ('DB_USER', 'username');

DEFINE ('DB_PASSWORD', 'password');

DEFINE ('DB_HOST', 'localhost');

DEFINE ('DB_NAME', 'sitename');

// Ket noi va chon CSDL.

$dbc = @mysql_connect (DB_HOST, DB_USER, DB_PASSWORD) OR
→ die ('Could not connect to MySQL: ' . mysql_error() );

mysql_select_db (DB_NAME) OR die ('Could not select the
→ database: ' . mysql_error() );

?>
```

Chương 9 "Phát triển ứng dụng Web", sẽ đề cập đến các lỗi MySQL phổ biến chi tiết hơn, cùng với các nguyên nhân gây ra chúng.

Biểu tượng @ được sử dụng để loại bỏ các lỗi, thông báo hoặc cảnh báo có thể xuất phát từ các hàm, chứ không chỉ áp dụng cho các hàm liên quan đến MySQL. Ví dụ:

```
@include("./filename.php");
```

Một cách tương tự, hàm **die()** *không chỉ áp dụng riêng cho MySQL, mà nó còn được áp dụng cho nhiều tình huống khác nữa.*

Hàm **exit()** *là một tên gọi khác của hàm* **die()**. *Nó thường được sử dụng mà không gửi thông báo lỗi xuống cho trình duyệt.*

Bạn có thể đưa một định dạng HTML vào trong thông báo lỗi của mình để chúng nổi bật hơn.

Thực hiện các câu truy vấn đơn giản

Sau khi đã kết nối thành công và chọn được cơ sở dữ liệu làm việc, bạn có thể bắt đầu thực hiện các câu truy vấn. Chúng có thể là những câu lệnh đơn giản (như chèn, cập nhật và xóa) hoặc phức tạp (như các liên kết trả lại nhiều thông tin). Trong bất kỳ trường hợp nào, hàm PHP để thực hiện một câu truy vấn cũng là **mysql_query()**:

```
$result = mysql_query();
```

Đối với những câu truy vấn đơn giản như **INSERT**, **UPDATE**, **DELETE**... (không trả lại các mẩu tin), biến **$result** sẽ chỉ có giá trị **TRUE** hoặc **FALSE** dựa trên việc thực hiện nó có thành công hay không. Đối với các câu truy vấn phức tạp có trả lại các mẩu tin (**SELECT**, **SHOW**, **DESCRIBE** và **EXPLAIN**), biến **$result** sẽ chứa kết quả của câu truy vấn nếu nó thành công và giá trị **FALSE** nếu không thành công.

Bước cuối cùng trong mã kịch bản là đóng lại kết nối MySQL hiện tại khi kết thúc làm việc với nó:

```
mysql_close();
```

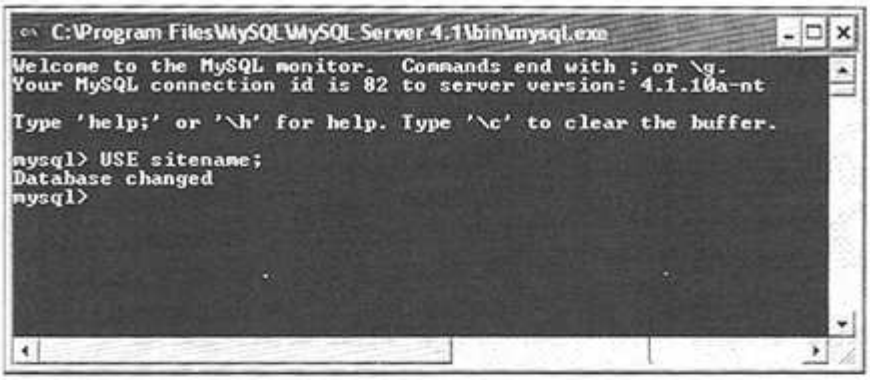
Bước này là tùy chọn vì PHP sẽ tự động đóng lại các kết nối khi mã kịch bản kết thúc. Tuy nhiên, việc tự mình đóng lại các kết nối không còn cần đến nữa là một phong cách lập trình tốt mà bạn nên theo.

Để minh họa toàn bộ quá trình, chúng ta sẽ viết một mã kịch bản đăng ký giống như mã kịch bản trong Chương 3 (tham khảo lại đoạn mã 3.15) để đăng ký người dùng vào bảng users của cơ sở dữ liệu *sitename* (được xây dựng trong Chương 4 "Giới thiệu SQL và MySQL"). Việc này đòi hỏi chúng ta phải thực hiện trước một số công việc với cơ sở dữ liệu.

Thực hành chỉnh sửa bảng users theo các bước sau:

1. Truy xuất MySQL thông qua trình quản lý mysql hoặc công cụ ưa thích của bạn (như phpMyAdmin).
2. Chọn cơ sở dữ liệu *sitename* (xem hình 6-8).

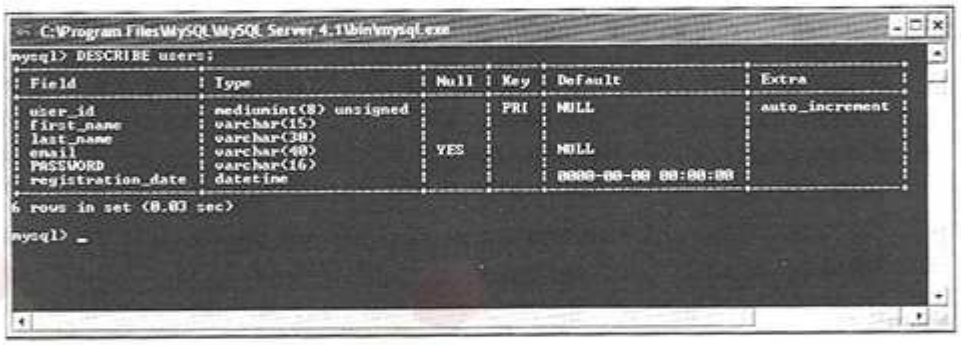
```
USE sitename;
```



Hình 6-8: *Để làm việc với bảng users, cần phải thông báo cho MySQL biết rằng chúng ta muốn làm việc với cơ sở dữ liệu sitename.*

3. Xem cấu trúc hiện tại của bảng (xem hình 6-9).

```
DESCRIBE users;
```

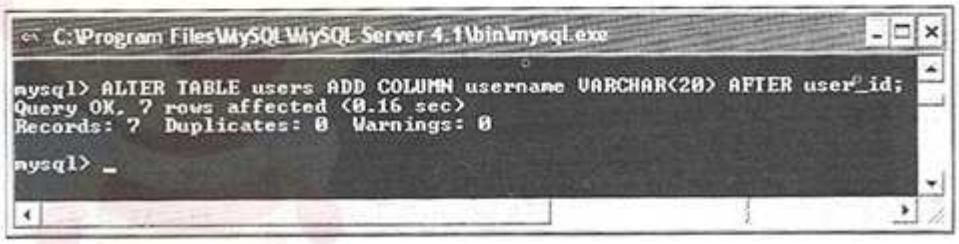


Hình 6-9: *Bảng đã được tạo ra từ trong Chương 4.*

Trước khi bắt tay vào viết mã kịch bản tương tác với cơ sở dữ liệu, chúng ta cần biết chính xác cấu trúc của các bảng.

4. Bổ sung trường username (xem hình 6-10).

```
ALTER TABLE users ADD COLUMN username VARCHAR(20) AFTER
→ user_id;
```

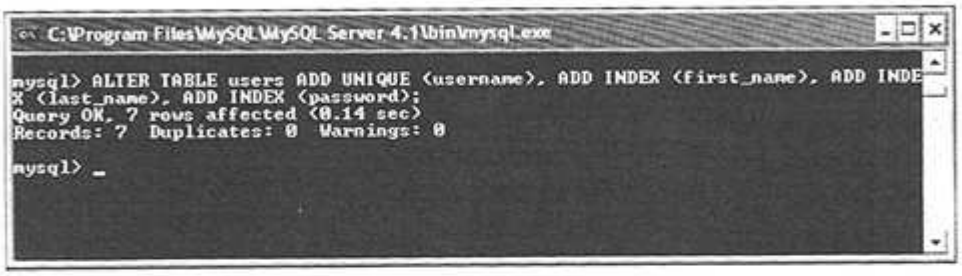


Hình 6-10: *Chúng ta sử dụng câu lệnh* ***ALTER*** *để bổ sung một cột vào bảng.*

Bảng ban đầu được thiết kế thiếu cột chứa tên người dùng (username). Câu truy vấn này bổ sung thêm một trường có kiểu là **VARCHAR** với chiều dài tối đa là 20 ký tự, nằm sau trường user_id.

5. Bổ sung các chỉ mục cần thiết (xem hình 6-11).

```
ALTER TABLE users ADD UNIQUE (username), ADD INDEX
→ (first_name), ADD INDEX (last_name), ADD INDEX
→ (password);
```

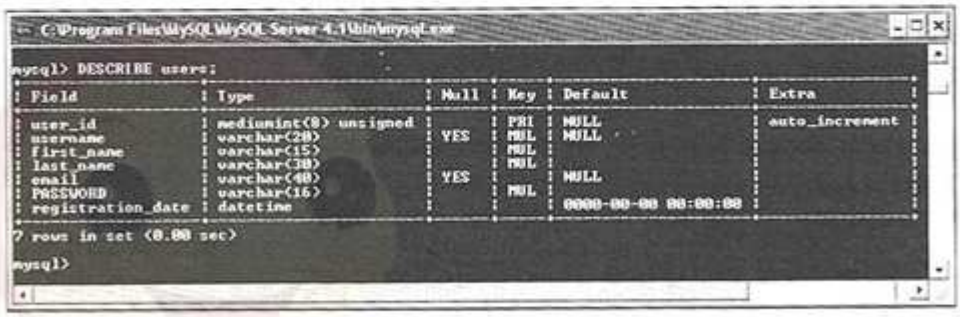


Hình 6-11: *Lệnh **ALTER** (được giới thiệu trong Chương 5) được dùng để tạo chỉ mục cho bảng hiện có.*

Bảng users đã có một chỉ mục cho khóa chính (user_id), nhưng chúng ta cần bổ sung thêm bốn chỉ mục khác để cải thiện khả năng vận hành khi truy xuất các mẩu tin. Bốn cột có trong câu lệnh trên sẽ được sử dụng nhiều trong các câu truy vấn. Chúng ta cũng định nghĩa trường username phải có giá trị duy nhất giữa các mẩu tin. Nếu cần, bạn cũng có thể bổ sung chỉ mục cho trường first_name và last_name, bằng cách dùng bổ sung **ADD INDEX** (**last_name**, **first_name**).

6. Kiểm tra lại cấu trúc bảng sau khi thực hiện các điều chỉnh (xem hình 6-12).

```
DESCRIBE users;
```



Hình 6-12: *Bảng users với các thay đổi vừa được thực hiện (các chỉ mục được liệt kê phía trong cột Key).*

Việc xem lại cấu trúc bảng lần cuối này sẽ giúp xác nhận các thay đổi vừa thực hiện và giúp ghi nhớ cấu trúc bảng khi viết mã kịch bản PHP.

Thực hành các câu truy vấn đơn giản theo các bước sau:

1. Tạo ra một mã kịch bản PHP trong trình soạn thảo văn bản.

```
<?php # Doan ma 6.6 - register.php
$page_title = 'Dang ky';
include ('templates/header.inc');
```

Mặc dù phần lớn mã kịch bản này giống với mã kịch bản **register.php** trong Chương 3, nhưng chúng ta sẽ viết lại nó từ đầu.

2. Tạo một điều kiện và khởi tạo biến **$message**.

```
if (isset($_POST['submit'])) {
   $message = NULL;
```

Mã kịch bản sẽ hiển thị và xử lý biểu mẫu HTML. Điều kiện này kiểm tra xem có xử lý biểu mẫu hay không (chỉ xử lý biểu mẫu khi nó được gửi). Biến **$message** được khởi tạo với giá trị **NULL** để không có cảnh báo nào xuất hiện khi được sử dụng sau này.

3. Kiểm tra tính hợp lệ của first_name.

```
if (empty($_POST['first_name'])) {
   $fn = FALSE;
   $message .= '<p>Ban chua nhap ten ho!</p>';
} else {
   $fn = $_POST['first_name'];
}
```

Trong các ví dụ, chúng ta kiểm tra một giá trị có được nhập hay không bằng cách xem chiều dài của nó có phải là một số dương không và bạn có thể đạt được kết quả tương tự với hàm **empty()**. Hàm này kiểm tra một biến có được thiết lập hay chưa (không có giá trị). Nếu điều kiện là FALSE, biến **$fn** sẽ được gán giá trị **FALSE** và chuỗi thông báo được nối vào biến **$message**. Nếu điều kiện là TRUE, biến **$fn** được gán giá trị gửi. Việc dùng biến mới này sẽ giúp chúng ta gặp nhiều thuận lợi khi viết câu truy vấn sau này.

4. Kiểm tra tính hợp lệ của last_name, email và username.

```
if (empty($_POST['last_name'])) {
   $ln = FALSE;
   $message .= '<p>Ban chua nhap ten!</p>';
} else {
   $ln = $_POST['last_name'];
}
if (empty($_POST['email'])) {
   $e = FALSE;
   $message .= '<p>Ban chua nhap dia chi thu dien tu!</p>';
```

```
} else {
   $e = $_POST['email'];
}
if (empty($_POST['username'])) {
   $u = FALSE;
   $message .= '<p>Ban chua nhap ten dang ky!</p>';
} else {
   $u = $_POST['username'];
}
```

5. Kiểm tra tính hợp lệ của password.

```
if (empty($_POST['password1'])) {
   $p = FALSE;
   $message .= '<p>Ban chua nhap mat khau!</p>';
} else {
   if ($_POST['password1'] == $_POST['password2']) {
      $p = $_POST['password1'];
   } else {
      $p = FALSE;
      $message .= '<p>Mat khau khong khop voi phan xac
      → nhan!</p>';
   }
}
```

Để kiểm tra tính hợp lệ của mật khẩu, chúng ta cần kiểm tra giá trị của trường password1. Sau đó, so sánh giá trị này với giá trị của trường password2 để đảm bảo rằng hai giá trị này giống nhau.

6. Kiểm tra xem có đủ điều kiện để đăng ký người dùng mới không.

```
if ($fn && $ln && $e && $u && $p) {
```

Nếu dữ liệu được gửi lên đáp ứng mọi điều kiện, biểu thức kiểm tra này sẽ có giá trị `TRUE` và người dùng đủ điều kiện đăng ký. Nếu không, một thông báo lỗi tương ứng sẽ được in ra và người dùng được cung cấp một cơ hội khác để đăng ký lại.

7. Đưa thông tin người dùng vào trong cơ sở dữ liệu.

```
require_once ('../mysql_connect.php');
$query = "INSERT INTO users (username, first_name,
→ last_name, email, password, registration_date) VALUES
→ ('$u', '$fn', '$ln', '$e', PASSWORD('$p'), NOW() )";
```

```
$result = @mysql_query ($query);
```

Dòng lệnh đầu tiên sẽ chèn nội dung của mã kịch bản **mysql_connect.php** vào trong mã kịch bản hiện tại, thông qua đó tạo ra kết nối với MySQL và chọn cơ sở dữ liệu. Bạn cần thay đổi tham chiếu đến tập tin theo vị trí của nó trên máy chủ.

Bản thân câu truy vấn này cũng tương tự như các câu lệnh SQL được minh họa trong Chương 4. Hàm **PASSWORD()** được sử dụng để mã hóa mật khẩu và hàm **NOW()** được dùng để thiết lập ngày đăng ký tại thời điểm hiện tại.

Sau khi gán câu truy vấn cho một biến, chúng ta sẽ thực hiện nó bằng hàm **mysql_query()**. Hàm này sẽ gửi câu lệnh SQL tới MySQL.

8. Thông báo kết quả của việc đăng ký.

```
if ($result) {
    echo '<p><b>Ban da duoc dang ky!</b></p>';
    include ('templates/footer.inc');
    exit();
} else {
    $message = '<p>Ban khong the dang ky do mot loi he thong.
    → Chung toi xin loi vi su co nay.</p><p>' .
    → mysql_error() . '</p>';
}
```

Biến **$result**, chứa giá trị trả về của hàm **mysql_query()**, có thể được sử dụng để kiểm tra sự thành công của việc thực hiện câu truy vấn. Trong ví dụ này, chúng ta viết lại điều kiện như sau:

```
if(@mysql_query($query)){
```

Nếu biến **$result** có giá trị là **TRUE**, khi đó thông báo sẽ được hiển thị, phần cuối trang được đưa vào và mã kịch bản ngừng lại (bằng cách sử dụng hàm **exit()**). Nếu không đưa vào phần cuối trang mà thoát khỏi mã kịch bản, khi đó biểu mẫu đăng ký sẽ không được hiển thị lại. Chúng ta cũng có thể gửi một thư điện tử khi đăng ký thành công hoặc chuyển người dùng đến một trang khác như đã thực hiện trong Chương 3.

Nếu biến **$result** có giá trị **FALSE**, biến **$message** sẽ được gán một giá trị, chủ yếu là kết quả từ hàm **mysql_error()**.

9. Đóng kết nối cơ sở dữ liệu và hoàn tất điều kiện đăng ký.

```
    mysql_close();
} else {
    $message .= '<p>Hay thu lai.</p>';
}
```

10. Đóng câu lệnh điều kiện và in ra thông báo lỗi nếu có.

```
}
if (isset($message)) {
   echo '<font color="red">', $message, '</font>';
}
```

11. Đóng phần mã lệnh PHP và bắt đầu biểu mẫu HTML.

```
?>
<form action="<?php echo $_SERVER['PHP_SELF']; ?>"
→ method="post">
<fieldset><legend>Nhap vao thong tin cua ban:</legend>
<p><b>Ten ho:</b> <input type="text" name="first_name"
→ size="15" maxlength="15" value="<?php if
→ (isset($_POST['first_name'])) echo $_POST['first_name'];
→ ?>" /></p>
```

Biểu mẫu này gần giống với biểu mẫu trong Chương 3. Chúng ta dùng `$PHP_SELF` (được truy xuất thông qua biến `$_SERVER['PHP_SELF']`) khi thiết lập thuộc tính action của biểu mẫu để trang tự xử lý.

Bạn nên sử dụng tên các trường trên biểu mẫu trùng với các cột tương ứng trong cơ sở dữ liệu nơi mà giá trị của chúng sẽ được lưu giữ. Ngoài ra, bạn nên thiết lập chiều dài tối đa cho trường nhập trên biểu mẫu bằng với chiều dài tối đa của cột trong cơ sở dữ liệu.

12. Hoàn tất biểu mẫu HTML.

```
<p><b>Ten:</b> <input type="text" name="last_name" size="30"
→ maxlength="30" value="<?php if
→ (isset($_POST['last_name'])) echo $_POST['last_name'];
→ ?>" /></p>

<p><b>Email:</b> <input type="text" name="email" size="40"
→ maxlength="40" value="<?php if (isset($_POST['email']))
→ echo $_POST['email']; ?>" /> </p>

<p><b>Ten dang nhap:</b> <input type="text" name="username"
→ size="10" maxlength="20" value="<?php if
→ (isset($_POST['username'])) echo $_POST['username']; ?>"
→ /></p>

<p><b>Mat khau:</b> <input type="password" name="password1"
→ size="20" maxlength="20" /></p>

<p><b>Xac nhan:</b> <input type="password" name="password2"
→ size="20" maxlength="20" /></p>

</fieldset>
```

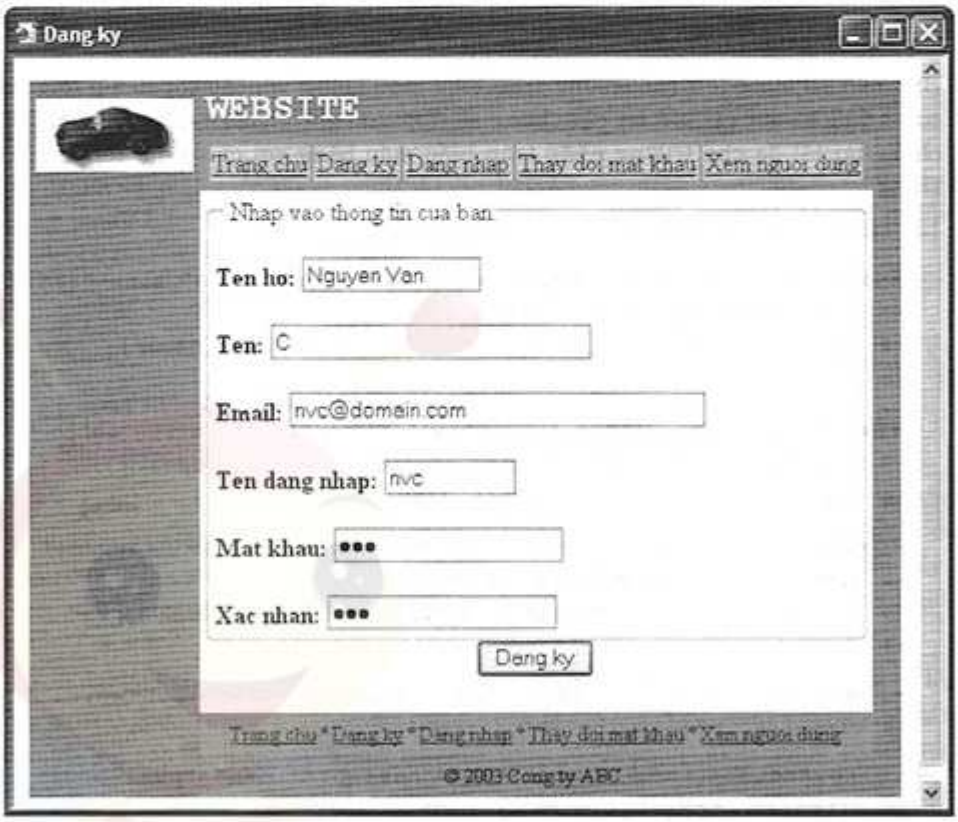
```
<div align="center"><input type="submit" name="submit"
→ value="Dang ky" /></div>

</form>
```

Không cần áp dụng chiều dài tối đa cho trường nhập mật khẩu vì chúng ta sẽ sử dụng hàm `PASSWORD()` để mã hóa nó và hàm này luôn trả lại một chuỗi với chiều dài cố định.

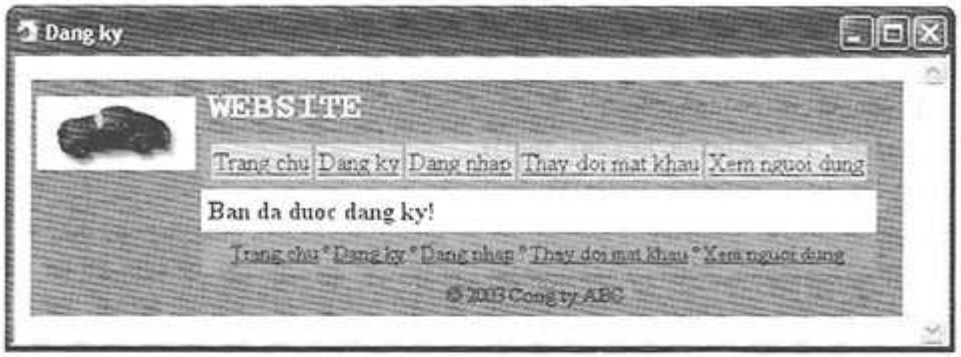
13. Hoàn tất trang bằng cách sử dụng tập tin cuối trang.

```
<?php
include ('templates/footer.inc');
?>
```

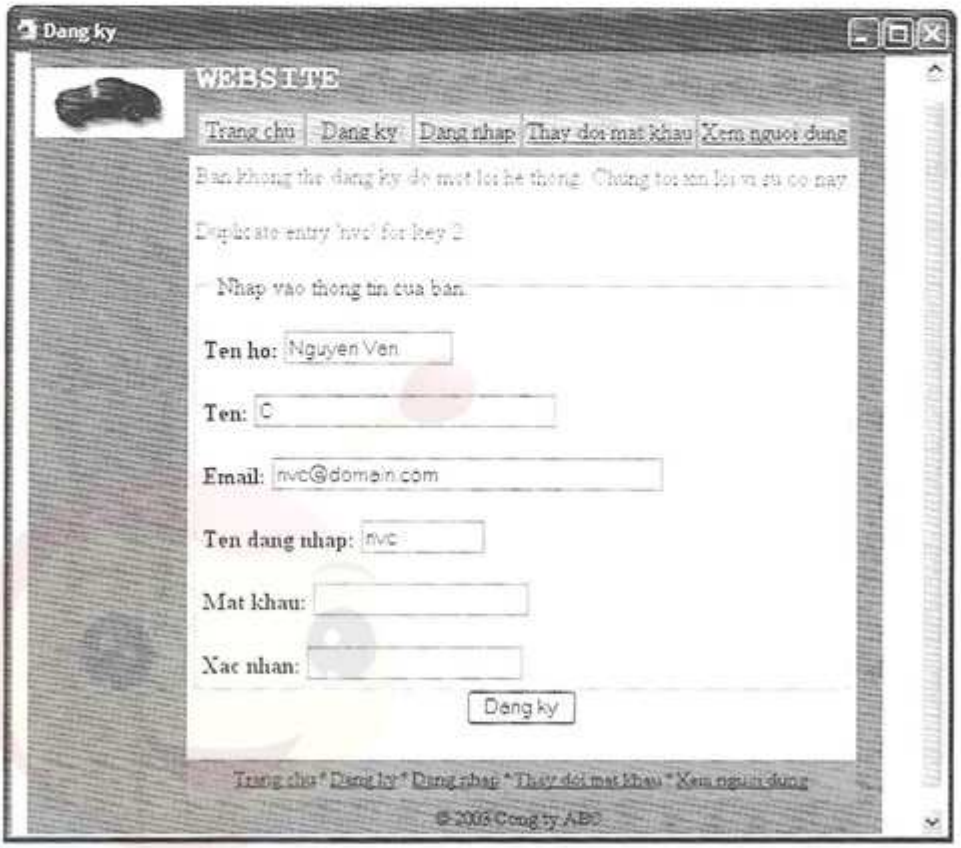
14. Lưu tập tin với tên gọi **register.php**, tải nó lên máy chủ Web trong cùng thư mục với tập tin **index.php** (tham khảo lại mã kịch bản 6.3) và kiểm tra nó bằng cách chạy mã kịch bản trong trình duyệt (xem hình 6-13, 6-14 và 6-15). Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 6.6.



Hình 6-13: *Biểu mẫu đăng ký.*



Hình 6-14: *Nếu người dùng được đăng ký vào trong cơ sở dữ liệu, thông báo này sẽ được hiển thị.*



Hình 6-15: *Nếu có bất kỳ lỗi MySQL nào trong quá trình thực hiện, chúng sẽ được hiển thị.*

Đoạn mã 6.6: *Mã kịch bản để thêm một mẩu tin vào trong cơ sở dữ liệu.*

```
<?php # Doan ma 6.6 - register.php
$page_title = 'Dang ky';
include ('templates/header.inc');
if (isset($_POST['submit'])) {
   $message = NULL;
   if (empty($_POST['first_name'])) {
      $fn = FALSE;
      $message .= '<p>Ban chua nhap ten ho!</p>';
   } else {
      $fn = $_POST['first_name'];
   }
   if (empty($_POST['last_name'])) {
      $ln = FALSE;
      $message .= '<p>Ban chua nhap ten!</p>';
   } else {
      $ln = $_POST['last_name'];
   }
   if (empty($_POST['email'])) {
      $e = FALSE;
      $message .= '<p>Ban chua nhap dia chi thu dien tu!</p>';
   } else {
      $e = $_POST['email'];
   }
   if (empty($_POST['username'])) {
      $u = FALSE;
      $message .= '<p>Ban chua nhap ten dang ky!</p>';
   } else {
      $u = $_POST['username'];
   }
   if (empty($_POST['password1'])) {
      $p = FALSE;
      $message .= '<p>Ban chua nhap mat khau!</p>';
   } else {
      if ($_POST['password1'] == $_POST['password2']) {
```

```
        $p = $_POST['password1'];
      } else {
        $p = FALSE;
        $message .= '<p>Mat khau khong khop voi phan xac
        → nhan!</p>';
      }
   }
   if ($fn && $ln && $e && $u && $p) {
      require_once ('../mysql_connect.php');
      $query = "INSERT INTO users (username, first_name,
      → last_name, email, password, registration_date) VALUES
      → ('$u', '$fn', '$ln', '$e', PASSWORD('$p'), NOW() )";
   $result = @mysql_query ($query);
      if ($result) {
        // Gui thong tin den hop thu nguoi dung de thong bao
        echo '<p><b>Ban da duoc dang ky!</b></p>';
        include ('templates/footer.inc');
        exit();
      } else {
        $message = '<p>Ban khong the dang ky do mot loi he
        → thong. Chung toi xin loi vi su co nay.</p><p>'
        → mysql_error() . '</p>';
      }
      mysql_close();
   } else {
      $message .= '<p>Hay thu lai.</p>';
   }
}
if (isset($message)) {
   echo '<font color="red">', $message, '</font>';
}
?>
<form action="<?php echo $_SERVER['PHP_SELF']; ?>"
method="post">
<fieldset><legend>Nhap vao thong tin cua ban:</legend>
<p><b>Ten ho:</b> <input type="text" name="first_name"
→ size="15" maxlength="15" value="<?php if
```

```
→ (isset($_POST['first_name'])) echo $_POST['first_name'];
→ ?>" /></p>
<p><b>Ten:</b> <input type="text" name="last_name" size="30"
→ maxlength="30" value="<?php if (isset($_POST['last_name']))
→ echo $_POST['last_name']; ?>" /></p>
<p><b>Email:</b> <input type="text" name="email" size="40"
→ maxlength="40" value="<?php if (isset($_POST['email']))
→ echo $_POST['email']; ?>" /> </p>
<p><b>Ten dang nhap:</b> <input type="text" name="username"
→ size="10" maxlength="20" value="<?php if
→ (isset($_POST['username'])) echo $_POST['username']; ?>"
→ /></p>
<p><b>Mat khau:</b> <input type="password" name="password1"
→ size="20" maxlength="20" /></p>
<p><b>Xac nhan:</b> <input type="password" name="password2"
→ size="20" maxlength="20" /></p>
</fieldset>
<div align="center"><input type="submit" name="submit"
→ value="Dang ky" /></div>
</form>
<?php
include ('templates/footer.inc');
?>
```

Sau khi chạy mã kịch bản, bạn có thể kiểm tra kết quả của nó bằng cách sử dụng trình quản lý mysql để xem các giá trị trong cơ sở dữ liệu.

Không nên kết thúc các câu truy vấn với một dấu chấm phẩy khi sử dụng chúng trong PHP. Với MySQL, lỗi này không ảnh hưởng gì đến kết quả, nhưng với các ứng dụng cơ sở dữ liệu khác (như Oracle), lỗi này sẽ làm cho câu truy vấn không thể thực hiện được.

Hàm ***mysql_query()*** *trả lại giá trị* ***TRUE*** *nếu câu truy vấn thực hiện thành công. Điều này không có nghĩa kết quả của câu truy vấn đúng theo những gì bạn mong đợi. Các mã kịch bản sau sẽ minh họa cách kiểm tra kết quả của câu truy vấn một cách chính xác hơn.*

Phương pháp tốt nhất để dò lỗi các mã kịch bản PHP tương tác với MySQL là dùng PHP để in ra câu truy vấn và sau đó chạy nó thông qua trình quản lý mysql hoặc một công cụ tương tự.

Bạn không bắt buộc phải tạo biến ***$query*** *như trong các ví dụ (có thể chèn ngay câu truy vấn vào phía trong dấu ngoặc của hàm* ***mysql_query()****). Tuy nhiên, khi việc tạo câu truy vấn trở nên phức tạp, việc sử dụng một biến như vậy sẽ là một tùy chọn tốt.*

Để làm cho mã kịch bản hiệu quả hơn, bạn nên xóa biến ***$dbc*** *sau khi kết nối đã được đóng lại bằng cách dùng hàm* ***unset($dbc)****. Nếu không, bạn không thể gán các kết quả của* ***mysql_connect()*** *cho một biến.*

Về mặt thực hành, bất kỳ câu truy vấn nào thực hiện được trong trình quản lý MySQL đều có thể thực hiện với hàm ***mysql_query()****.*

Lấy ra các kết quả truy vấn

Phần trước của chương này đã đề cập và minh họa cách thực hiện các câu truy vấn đơn giản trên một cơ sở dữ liệu MySQL. Đó là các câu lệnh SQL được bắt đầu với **INSERT**, **UPDATE**, **DELETE** hoặc **ALTER**. Cả bốn câu truy vấn này đều không trả lại dữ liệu, mà chỉ cho biết thực hiện thành công hay không. Ngược lại, câu truy vấn **SELECT** thường tạo ra thông tin (tức là trả lại các mẩu tin) cần được xử lý bằng các hàm PHP khác.

Công cụ chủ yếu để xử lý câu truy vấn **SELECT** là **mysql_fetch_array()**. Hàm này nhận vào biến kết quả của câu truy vấn (biến **$result** trong ví dụ trên) và trả lại từng mẩu tin ở dạng mảng. Chúng ta thường sử dụng hàm này trong một vòng lặp để truy xuất từng mẩu tin được trả lại. Kết cấu cơ bản của việc đọc ra từng mẩu tin kết quả của một câu truy vấn là:

```
while ($row = mysql_fetch_array($resutl)){
   // thực hiện điều gì đó với $row
}
```

Hàm **mysql_fetch_array()** nhận vào một tham số tùy biến cho biết loại mảng sẽ được trả lại: liên kết, chỉ mục hoặc cả hai. Mảng liên kết cho phép tham chiếu đến các giá trị cột theo tên, trong khi mảng chỉ mục đòi hỏi sử dụng các con số (bắt đầu với **0** cho cột đầu tiên). Mỗi tham số được định nghĩa bởi một hằng được liệt kê trong bảng 6.1. Nhiều người dùng **MYSQL_NUM** vì tùy chọn này nhanh hơn, ít chiếm bộ nhớ hơn so với các tùy chọn khác. Một số nhà lập trình khác sử dụng **MYSQL_ASSOC**, vì khi đó họ có thể truy xuất một trường dưới dạng **$row['cột']** thay vì **$row[3]**. Cách này vẫn làm việc tốt khi cấu trúc dữ liệu bị thay đổi.

Bảng 6.1: *Các hằng được dùng với hàm* ***mysql_fetch_array()*** *để quy định cách trả lại mảng kết quả. Giá trị mặc định là* ***MYSQL_BOTH***

Hằng	Ví dụ
MYSQL_ASSOC	$row['cột']
MYSQL_NUM	$row[0]
MYSQL_BOTH	$row[0] hoặc $row['cột ']

Bạn nên giải phóng kết quả của hàm **mysql_fetch_array()** khi không còn sử dụng đến nó nữa.

```
mysql_free_result($result);
```

Dòng lệnh này loại bỏ phần bộ nhớ được chiếm giữ bởi **$result**. Cũng giống như khi sử dụng hàm **mysql_connect()**, đây là một bước tùy chọn vì PHP sẽ tự động giải phóng tài nguyên hệ thống khi kết thúc mã kịch bản. Tuy nhiên, đây là một phong cách lập trình tốt mà bạn nên theo.

Để minh họa cách xử lý kết quả được trả lại bởi một câu truy vấn, chúng ta sẽ tạo mã kịch bản thể hiện tất cả người dùng đã đăng ký.

Thực hành lấy ra kết quả truy vấn theo các bước sau:

1. Tạo một hồ sơ PHP mới trong trình soạn thảo văn bản.

```
<?php # Doan ma 6.7 - view_users.php
$page_title = 'Xem nguoi dung hien tai';
include ('templates/header.inc');
```

2. Kết nối và truy vấn cơ sở dữ liệu.

```
require_once ('../mysql_connect.php');
$query = "SELECT CONCAT(last_name, ', ', first_name) AS
→ name, DATE_FORMAT(registration_date, '%M %d, %Y') AS dr
→ FROM users ORDER BY registration_date ASC";
$result = @mysql_query ($query);
```

Câu truy vấn này sẽ trả lại hai cột: tên người dùng (định dạng là *họ và tên*) và ngày đăng ký (định dạng Month DD, YYYY).

3. Hiển thị kết quả truy vấn.

```
if ($result) {
   echo '<table align="center" cellspacing="2"
   → cellpadding="2"><tr><td align="left"><b>Ten</b></td>
   →<td align="left"><b>Ngay dang ky</b></td></tr>';
   while ($row = mysql_fetch_array($result, MYSQL_NUM)) {
      echo "<tr><td align=\"left\">$row[0]</td>
      →<td align=\"left\">$row[1]</td></tr>\n";
   }
   echo '</table>';
```

Để hiển thị kết quả, đầu tiên chúng ta tạo bảng và dòng tiêu đề trong HTML. Sau đó, duyệt qua kết quả với hàm **mysql_fetch_array()** và in ra từng dòng kết quả. Cuối cùng, chúng ta đóng bảng lại.

4. Giải phóng tài nguyên của câu truy vấn.

```
mysql_free_result ($result);
```

Một lần nữa xin nhắc lại, đây là một bước tùy chọn, nhưng bạn nên thực hiện theo.

5. Hoàn tất điều kiện.

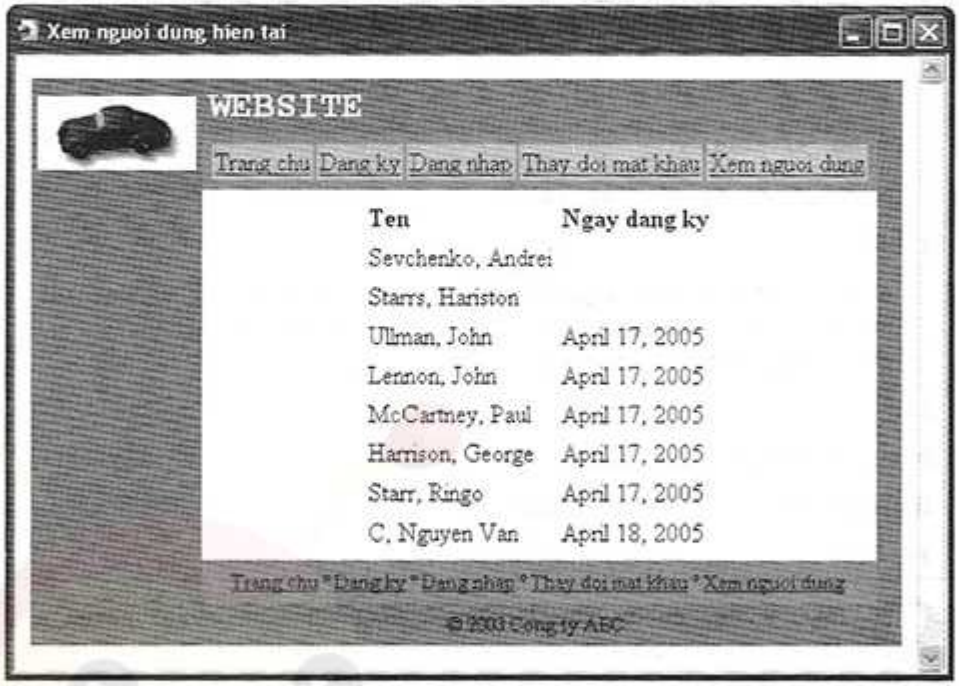
```
} else {
```

```
        echo '<p>Danh sach nguoi dung khong the hien thi vi mot
        → loi he thong. Chung toi xin loi vi su co nay.</p><p>'
        → . mysql_error() . '</p>';
    }
```

6. Đóng kết nối và hoàn tất trang.

```
mysql_close();
include ('templates/footer.inc');
?>
```

7. Lưu tập tin với tên gọi **view_users.php**, tải nó lên máy chủ và thử trong trình duyệt (xem hình 6-16). Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 6.7.



Hình 6-16: *Tất cả các mẩu tin người dùng được lấy từ cơ sở dữ liệu.*

Đoạn mã 6.7: *Mã kịch bản* ***view_users.php*** *thực hiện câu truy vấn với cơ sở dữ liệu và in ra tất cả các mẩu tin trả lại.*

```
<?php # Doan ma 6.7 - view_users.php
$page_title = 'Xem nguoi dung hien tai';
include ('templates/header.inc');
```

```
require_once ('../mysql_connect.php');

$query = "SELECT CONCAT(last_name, ', ', first_name) AS name,
→ DATE_FORMAT(registration_date, '%M %d, %Y') AS dr FROM
→ users ORDER BY registration_date ASC";

$result = @mysql_query ($query);

if ($result) {

   echo '<table align="center" cellspacing="2" cellpadding="2">
   → <tr><td align="left"><b>Ten</b></td><td align="left">
   →<b>Ngay dang ky</b></td></tr>';

   while ($row = mysql_fetch_array($result, MYSQL_NUM)) {

      echo "<tr><td align=\"left\">$row[0]</td><td
      → align=\"left\">$row[1]</td></tr>\n";

   }

   echo '</table>';

   mysql_free_result ($result);

} else {

   echo '<p>Danh sach nguoi dung khong the hien thi vi mot loi
   → he thong. Chung toi xin loi vi su co nay.</p><p>' .
   → mysql_error() . '</p>';

}

mysql_close();

include ('templates/footer.inc');

?>
```

Hàm `mysql_fetch_row()` tương đương với hàm `mysql_fetch_array ($result, MYSQL_NUM)`.

Hàm `mysql_fetch_assoc()` tương đương với hàm `mysql_fetch_array ($result, MYSQL_ASSOC)`.

PHP cũng có hàm `mysql_fetch_object()` để lấy ra kết quả truy vấn. Nó trả lại một mẩu tin dưới dạng một đối tượng.

Khi lấy ra các mẩu tin từ cơ sở dữ liệu với mảng liên kết, bạn phải tham chiếu cột chính xác như khi chúng được định nghĩa trong cơ sở dữ liệu (do khóa trong mảng phân biệt chữ hoa và chữ thường).

Nếu gặp phải tình huống cần chạy một câu truy vấn thứ hai phía trong một vòng lặp, cần dùng biến khác để chứa kết quả (`$result1` và `$row1` thay cho `$result` và `$row`), nếu không bạn sẽ gặp phải các lỗi lôgic.

Khi sử dụng tên hiệu cho một cột trong các câu truy vấn, bạn phải dùng tên hiệu khi tham chiếu đến cột đó trong PHP.

An toàn

An toàn cơ sở dữ liệu với PHP có thể được chia thành hai nội dung chính:

- Bảo vệ thông tin truy xuất cơ sở dữ liệu.
- An toàn khi chèn dữ liệu vào cơ sở dữ liệu.

Chúng ta có thể thực hiện mục tiêu thứ nhất bằng cách đặt mã kịch bản kết nối MySQL nằm ngoài thư mục Web để nó không thể xem được thông qua trình duyệt Web. Vấn đề này đã được giới thiệu trong phần đầu của chương và sẽ tiếp tục được đề cập trong Chương 8.

Đối với mục tiêu thứ hai, chúng ta có một số tùy chọn. Đầu tiên, như đã thực hiện trong chương này là sử dụng mảng `$_POST` (hay `$HTTP_GET_VARS`, `$HTTP_POST_VARS` hoặc `$_GET`) thay cho các biến toàn cục. Thứ hai, sử dụng các biểu thức quy tắc để đảm bảo rằng các dữ liệu gửi phù hợp với những gì mong đợi. Các biểu thức quy tắc cũng sẽ được đề cập đến trong Chương 8. Tùy chọn thứ ba là sử dụng hàm `mysql_real_escape_string()`, ví dụ:

```
$data = mysql_real_escape_string($data).
```

Hàm này hoạt động giống như hàm `addslashes()` và nên được sử dụng với bất kỳ trường văn bản nào có trên biểu mẫu có liên quan đến cơ sở dữ liệu. Tuy nhiên, nó đòi hỏi phải có một kết nối với cơ sở dữ liệu để có thể làm việc, vì thế bạn cần tính đến yếu tố này khi viết mã kịch bản.

Magic Quotes và mysql_real_escape_string()

Magic Quotes (đã được đề cập trong Chương 2 "Lập trình với PHP") sẽ tự động mã hóa escape các ký tự có vấn đề trong các văn bản được nhập trên biểu mẫu. Điều này rất quan trọng khi truy vấn cơ sở dữ liệu vì các dấu nháy sẽ tạo ra các lỗi trong câu lệnh SQL. Tuy nhiên, hàm `mysql_real_escape_string()` là một cách tốt hơn để đạt được cùng một kết quả vì nó là hàm có sẵn của PHP và chuyên dùng cho cơ sở dữ liệu. Do đó, chúng ta sẽ sử dụng `mysql_real_escape_string()` thay cho Magic Quotes.

Giải pháp trong mã kịch bản này là kiểm tra trạng thái Magic Quotes với hàm `ini_get()`. Hàm này sẽ trả lại thiết lập của một tùy chọn cụ thể trong tập tin cấu hình PHP (`php.ini`). Nếu `ini_get('magic_quotes_gpc')` trả lại giá trị `TRUE` (nghĩa là Magic Quotes được kích hoạt), chúng ta sẽ loại bỏ các dấu chéo trước khi áp dụng `mysql_real_escape_string()`.

Trong các phiên bản hiện tại của PHP, Magic Quotes mặc định sẽ không được kích hoạt. Do đó, `ini_get('magic_quotes_gpc')` sẽ trả lại giá trị `FALSE` và chúng ta không cần phải loại bỏ các dấu chéo trong các chuỗi.

Thực hành sử dụng hàm `mysql_real_escape_string()` theo các bước sau:

1. Mở tập tin `register.php` (tham khảo lại đoạn mã 6.6) trong trình soạn thảo.
2. Sau điều kiện kiểm tra gửi biểu mẫu, bổ sung các dòng lệnh sau:

```
require_once ('../mysql_connect.php');
function escape_data ($data) {
   global $dbc;
   if (ini_get('magic_quotes_gpc')) {
      $data = stripslashes($data);
   }
   return mysql_real_escape_string($data, $dbc);
}
```

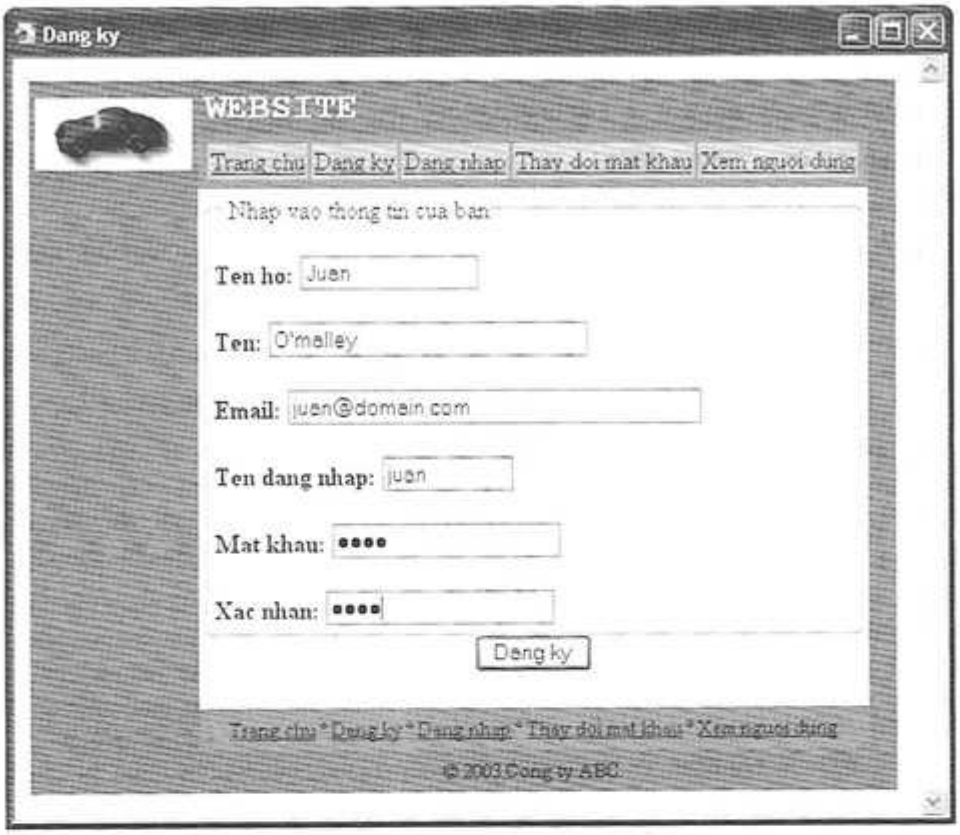
Hàm `escape_data()` nhận vào một chuỗi và áp dụng hàm `mysql_real_escape_string()` cho nó, sau đó trả lại kết quả. Nếu dữ liệu đã được chạy thông qua Magic Quotes (nghĩa là `ini_get('magic_quotes_gpc')` trả lại giá trị `TRUE`), dữ liệu sẽ được loại bỏ các ký tự chéo ngược trước để chúng không bị mã hóa escape nhiều lần.

3. Thay đổi thủ tục kiểm tra hợp lệ để sử dụng hàm này, thay thế các dòng `$var = $_POST['var']` bằng `$var = escape_data($_POST['var'])`.

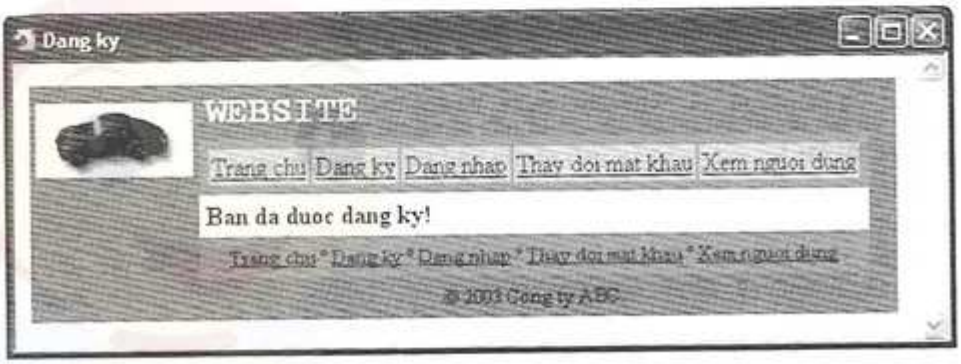
```
$fn = escape_data($_POST['first_name']);
$ln = escape_data($_POST['last_name']);
$e = escape_data($_POST['email']);
$u = escape_data($_POST['username']);
$p = escape_data($_POST['password1']);
```

Thay vì gán trực tiếp giá trị cho từng biến (`$fn`, `$ln`...), chúng ta sẽ chuyển dữ liệu cho hàm `escape_data()` và gán giá trị nhận được cho các biến này.

4. Xóa dòng `require_once('../mysql_connect.php')` gốc (tham khảo lại đoạn mã 6.6) vì chúng ta đã bổ sung dòng này ở đầu mã kịch bản.
5. Lưu tập tin với tên gọi **register.php**, tải nó lên máy chủ và thử trong trình duyệt Web (xem hình 6-17 và 6-18). Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 6.8.



Hình 6-17: *Các giá trị với dấu nháy phía trong sẽ được xử lý một cách thích hợp cho dù tùy chọn Magic Quotes có được kích hoạt hay không (xem thêm hình 6-18).*



Hình 6-18: *Giờ đây mã kịch bản đã làm việc mà không phụ thuộc vào Magic Quotes.*

Đoạn mã 6.8: *Mã kịch bản* ***register.php*** *giờ đây sử dụng hàm* ***mysql_real_escape_string()*** *để xử lý các dữ liệu được gửi tới.*

```
<?php # Doan ma 6.8 - register.php
$page_title = 'Dang ky';
include ('templates/header.inc');
if (isset($_POST['submit'])) {
   require_once ('../mysql_connect.php');
   function escape_data ($data) {
      global $dbc;
      if (ini_get('magic_quotes_gpc')) {
         $data = stripslashes($data);
      }
      return mysql_real_escape_string($data, $dbc);
   }
   $message = NULL;
   if (empty($_POST['first_name'])) {
      $fn = FALSE;
      $message .= '<p>Ban chua nhap ten ho!</p>';
   } else {
      $fn = escape_data($_POST['first_name']);
   }
   if (empty($_POST['last_name'])) {
      $ln = FALSE;
      $message .= '<p>Ban chua nhap ten!</p>';
   } else {
      $ln = escape_data($_POST['last_name']);
   }
   if (empty($_POST['email'])) {
      $e = FALSE;
      $message .= '<p>Ban chua nhap dia chi thu dien tu!</p>';
   } else {
      $e = escape_data($_POST['email']);
```

```
}
if (empty($_POST['username'])) {
   $u = FALSE;
   $message .= '<p>Ban chua nhap ten dang ky!</p>';
} else {
   $u = escape_data($_POST['username']);
}
if (empty($_POST['password1'])) {
   $p = FALSE;
   $message .= '<p>Ban chua nhap mat khau!</p>';
} else {
   if ($_POST['password1'] == $_POST['password2']) {
      $p = escape_data($_POST['password1']);
   } else {
      $p = FALSE;
      $message .= '<p>Mat khau khong khop voi phan xac
      -> nhan!</p>';
   }
}
if ($fn && $ln && $e && $u && $p) {
   $query = "INSERT INTO users (username, first_name,
   -> last_name, email, password, registration_date) VALUES
   -> ('$u', '$fn', '$ln', '$e', PASSWORD('$p'), NOW() )";
$result = @mysql_query ($query);
   if ($result) {
      // Gui thu dien tu neu can.
      echo '<p><b>Ban da duoc dang ky!</b></p>';
      include ('templates/footer.inc');
      exit();
   } else {
      $message = '<p>Ban khong the dang ky do mot loi he
      -> thong. Chung toi xin loi vi su co nay.</p><p>' .
      -> mysql_error() . '</p>';
```

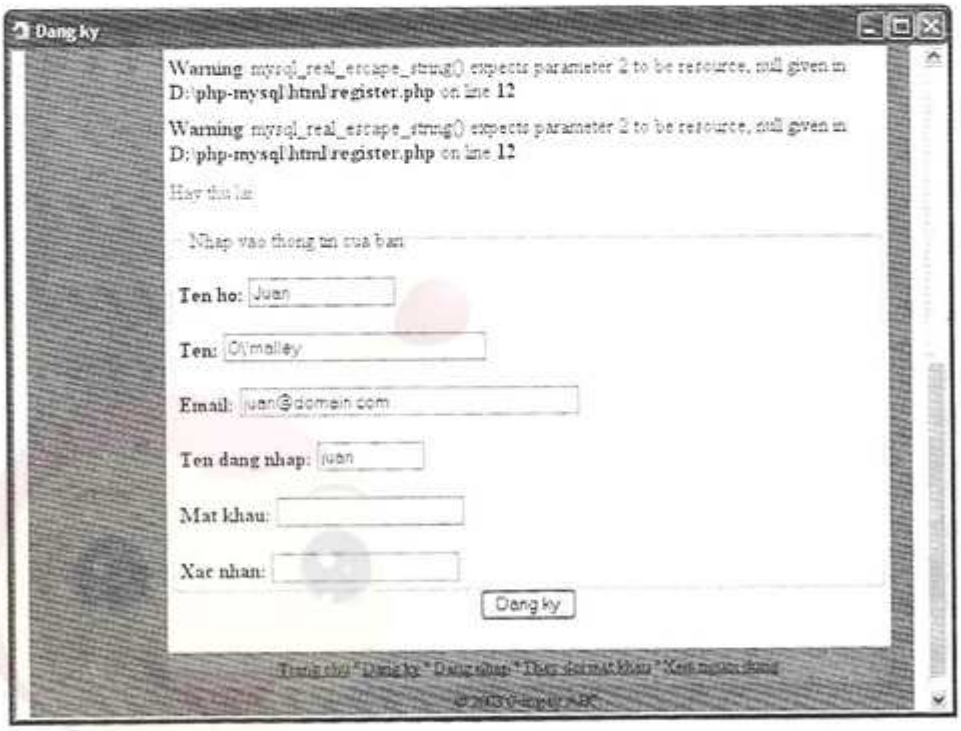
```
        }
        mysql_close();
    } else {
        $message .= '<p>Hay thu lai.</p>';
    }
}
if (isset($message)) {
    echo '<font color="red">', $message, '</font>';
}
?>
<form action="<?php echo $_SERVER['PHP_SELF']; ?>"
→ method="post">
<fieldset><legend>Nhap vao thong tin cua ban:</legend>
<p><b>Ten ho:</b> <input type="text" name="first_name"
→ size="15" maxlength="15" value="<?php if
→ (isset($_POST['first_name'])) echo $_POST['first_name'];
→ ?>" /></p>
<p><b>Ten:</b> <input type="text" name="last_name" size="30"
→ maxlength="30" value="<?php if (isset($_POST['last_name']))
→ echo $_POST['last_name']; ?>" /></p>
<p><b>Email:</b> <input type="text" name="email" size="40"
→ maxlength="40" value="<?php if (isset($_POST['email']))
→ echo $_POST['email']; ?>" /> </p>
<p><b>Ten dang nhap:</b> <input type="text" name="username"
→ size="10" maxlength="20" value="<?php if
→ (isset($_POST['username'])) echo $_POST['username']; ?>"
→ /></p>
<p><b>Mat khau:</b> <input type="password" name="password1"
→ size="20" maxlength="20" /></p>
<p><b>Xac nhan:</b> <input type="password" name="password2"
→ size="20" maxlength="20" /></p>
</fieldset>
<div align="center"><input type="submit" name="submit"
→ value="Dang ky" /></div>
</form>
```

```
<?php
include ('templates/footer.inc');
?>
```

*Hàm **mysql_real_escape_string()** sẽ mã hóa chuỗi theo ngôn ngữ hiện đang được sử dụng và đây là một ưu điểm của nó so với hàm **addslashes()** và hàm **mysql_escape_string()**.*

*Hàm **mysql_real_escape_string()** về bản chất là thay thế cho hàm **mysql_escape_string()** kể từ phiên bản 4.3. Nếu sử dụng các phiên bản trước đây của PHP, bạn nên sử dụng hàm **mysql_escape_string()**. Tham khảo tài liệu PHP để biết thêm chi tiết về cách sử dụng các hàm này.*

*Khi hàm **mysql_real_escape_string()** không thể truy cập dữ liệu, nó sẽ tạo kết quả như hình 6-19.*



Hình 6-19: *Vì hàm **mysql_real_escape_string()** đòi hỏi kết nối cơ sở dữ liệu, việc sử dụng nó có thể dẫn đến các lỗi như trong hình.*

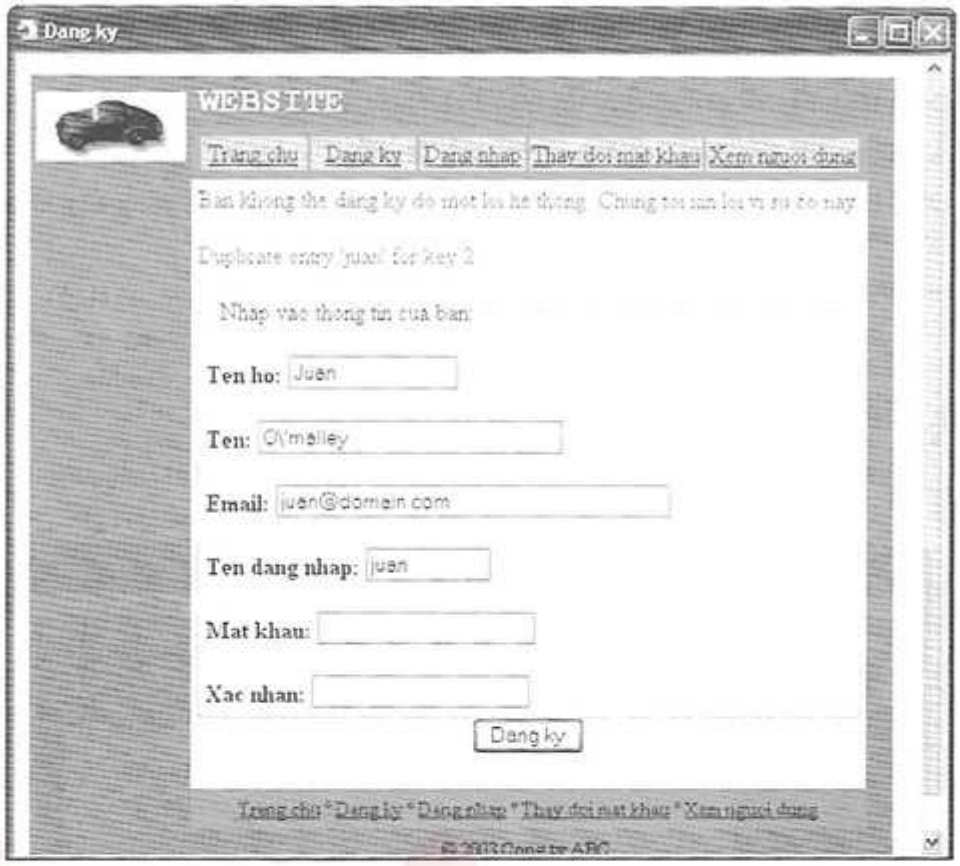
Nếu không sử dụng hàm này và Magic Quotes không được kích hoạt, các chuỗi như O'Malley sẽ phát sinh lỗi MySQL (xem hình 6-20) vì dấu nháy trong tên sẽ xung đột với các dấu nháy dùng trong câu truy vấn.

Hình 6-20: *Dấu nháy trong các giá trị trên biểu mẫu có thể gây ra vấn đề trong câu truy vấn.*

Với Magic Quotes được kích hoạt, bạn có thể sử dụng hàm **`stripslashes()`** *trước khi in giá trị được gửi ngược lại biểu mẫu (tham khảo trường "Ten" trong hình 6-21).*

Bạn nên sử dụng hàm **`trim()`** *cùng với hàm* **`escape_data()`** *để loại bỏ các khoảng trắng trước và sau dữ liệu gửi.*

Hàm **`get_magic_quotes_gpc()`** *cũng có thể được sử dụng để trả lại thiết lập Magic Quotes hiện tại.*



Hình 6-21: *Với Magic Quotes được kích hoạt, các giá trị được mã hóa được in ngược trở lại trong biểu mẫu sẽ không chính xác nếu không sử dụng hàm* ***stripslashes()****.*

Sử dụng mysql_num_rows()

Hàm kế tiếp được đề cập là **mysql_num_rows()**. Hàm này trả lại số mẩu tin được lấy ra khi thực hiện câu truy vấn **SELECT** và nhận vào kết quả truy vấn dưới dạng tham số.

Hàm **mysql_num_rows()** được sử dụng theo hai cách khác nhau. Thứ nhất, chúng ta dùng nó trong tập tin **view_users.php** để thể hiện tổng số người đăng ký. Thứ hai, ta dùng nó trong tập tin **register.php** để kiểm tra tên người dùng đã có trong cơ sở dữ liệu chưa.

Thực hành chỉnh sửa tập tin **view_users.php** theo các bước sau:

1. Mở tập tin **view_users.php** (tham khảo lại mã kịch bản 6.7) trong trình soạn thảo văn bản.

2. Trước điều kiện `if($result)`, bổ sung dòng lệnh sau:

```
$num = mysql_num_rows ($result);
```

Dòng lệnh này sẽ gán số mẩu tin trả về cho biến `$num`.

3. Thay đổi điều kiện gốc `$result` thành `if($num > 0) {`.

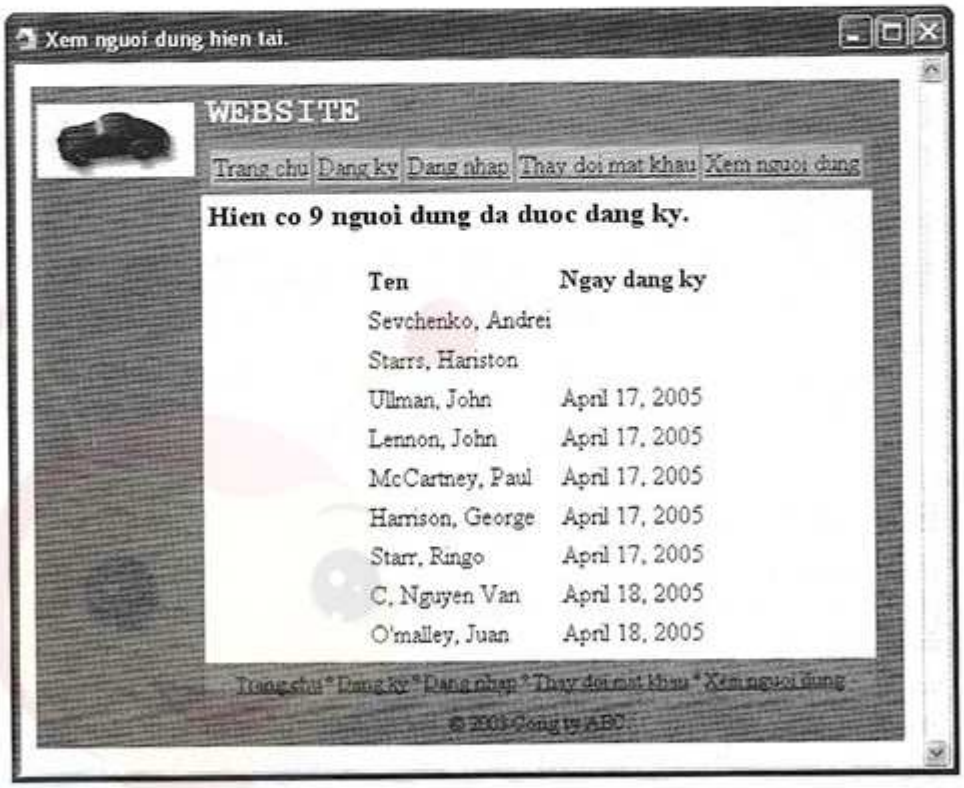
Điều kiện trước đây kiểm tra câu truy vấn có thực hiện thành công hay không, còn lúc này nó kiểm tra số mẩu tin được trả lại (chính xác hơn).

4. In số người dùng đã đăng ký.

```
echo "<p><big><b>Hien co $num nguoi dung da duoc dang
→ ky.</b></big></p>";
```

5. Thay đổi câu lệnh `echo()` gốc thành:

```
echo "<tr><td align=\"left\">" . stripslashes($row[0]) .
→ "</td><td align=\"left\">$row[1]</td></tr>\n";
```



Hình 6-22: *Số người dùng đăng ký được hiển thị ở đầu trang.*

Một thay đổi khác được thực hiện cho tập tin gốc là sử dụng hàm **stripslashes()** khi in ra tên người dùng. Việc dùng hàm này giúp loại bỏ ảnh hưởng của Magic Quotes nếu nó được kích hoạt.

6. Thay đổi phần else của câu lệnh điều kiện.

```
echo '<p>Hien chua co nguoi dung nao duoc dang ky.</p>';
```

Câu lệnh điều kiện gốc dựa vào việc câu truy vấn có thực hiện thành công hay không. Giờ đây, nó được dựa trên số người dùng đã đăng ký, vì thế chúng ta thay đổi câu lệnh **echo** này lại cho thích hợp.

7. Lưu tập tin với tên gọi **view_users.php**, tải nó lên máy chủ và thử trong trình duyệt Web (xem hình 6-22). Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 6.9.

Đoạn mã 6.9: *Giờ đây mã kịch bản **view_users.php** sẽ hiển thị tổng số người dùng đã đăng ký.*

```
<?php # Doan ma 6.9 - view_users.php

$page_title = 'Xem nguoi dung hien tai.';

include ('templates/header.inc');

require_once ('../mysql_connect.php');

$query = "SELECT CONCAT(last_name, ', ', first_name) AS name,
→ DATE_FORMAT(registration_date, '%M %d, %Y') AS dr FROM
→ users ORDER BY registration_date ASC";

$result = @mysql_query ($query);

$num = mysql_num_rows ($result);

if ($num > 0) {

   echo "<p><big><b>Hien co $num nguoi dung da duoc dang
   → ky.</b></big></p>";

   echo '<table align="center" cellspacing="2"
   → cellpadding="2">
   → <tr><td align="left"><b>Ten</b></td><td align="left">
   →<b>Ngay dang ky</b></td></tr>';

   while ($row = mysql_fetch_array($result, MYSQL_NUM)) {

      echo "<tr><td align=\"left\">" . stripslashes($row[0]) .
      → "</td><td align=\"left\">$row[1]</td></tr>\n";

   }
```

```
    echo '</table>';
    mysql_free_result ($result);
} else {
    echo '<p>Hien chua co nguoi dung nao duoc dang ky.</p>';
}
mysql_close();
include ('templates/footer.inc');
?>
```

Thực hành điều chỉnh mã kịch bản **register.php** theo các bước sau:

1. Mở tập tin **register.php** (tham khảo đoạn mã 6.8) trong trình soạn thảo văn bản.
2. Trước câu lệnh **INSERT**, bổ sung mã lệnh sau:

```
$query = "SELECT user_id FROM users WHERE username='$u'";
$result = @mysql_query ($query);
if (mysql_num_rows($result) == 0) {
```

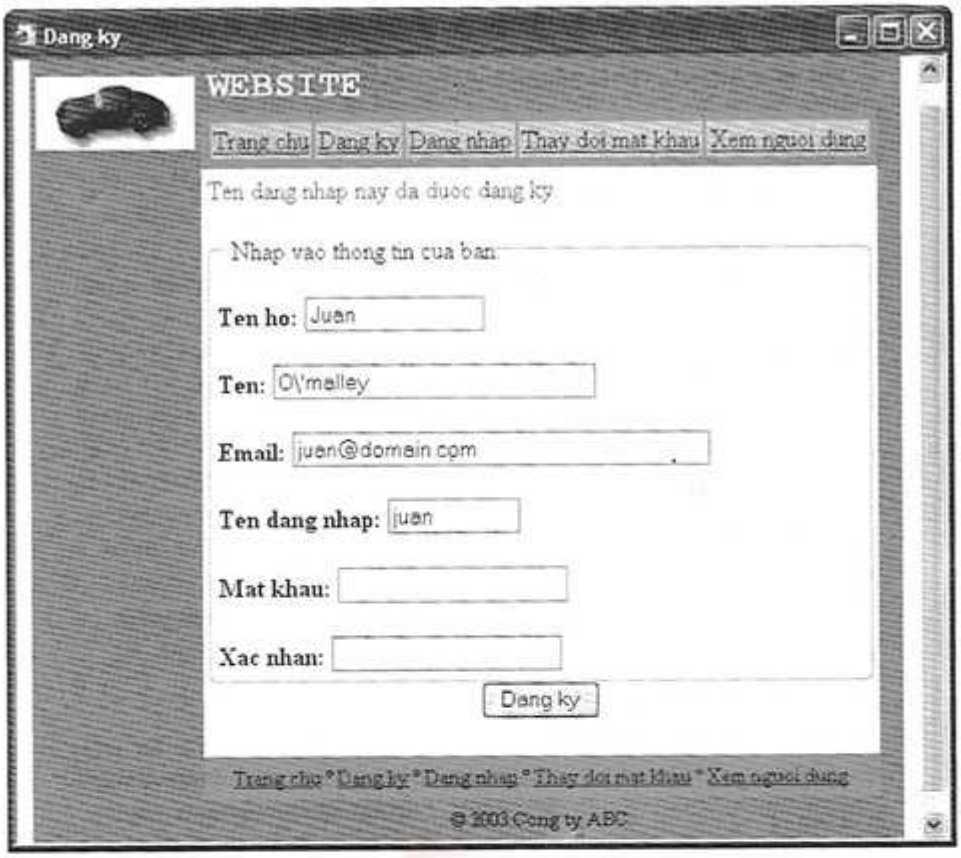
Câu truy vấn này sẽ kiểm tra xem tên người dùng gửi tới (**$u**) đã có trong cơ sở dữ liệu chưa bằng cách thử lấy ra mẩu tin với giá trị này. Nếu số mẩu tin trả lại là **0**, người dùng này chưa được đăng ký và ta sẽ đưa các thông tin của họ vào trong cơ sở dữ liệu.

3. Sau phần kết thúc của điều kiện **if($result)**, bổ sung mã lệnh sau:

```
} else {
    $message = '<p>Ten dang nhap nay da duoc dang ky.</p>';
}
```

Mệnh đề **else** này là phần kết luận của câu lệnh **if(mysql_num_rows ($result) == 0)**. Nó cho biết người dùng đã được đăng ký.

4. Lưu tập tin với tên gọi **register.php**, tải nó lên máy chủ và thử trong trình duyệt Web (xem hình 6-23). Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 6.10.



Hình 6-23: *Quá trình đăng ký sẽ không cho phép người dùng sử dụng một tên đăng ký đã được đăng ký trước đó (tên người dùng phải là duy nhất trong cơ sở dữ liệu).*

Đoạn mã 6.10: *Mã kịch bản* ***register.php*** *giờ đây sẽ kiểm tra xem tên người dùng đã có trong cơ sở dữ liệu chưa.*

```
<?php # Doan ma 6.10 - register.php
$page_title = 'Dang ky';
include ('templates/header.inc');
if (isset($_POST['submit'])) {
   require_once ('../mysql_connect.php');
   function escape_data ($data) {
      global $dbc;
      if (ini_get('magic_quotes_gpc')) {
         $data = stripslashes($data);
```

```
    }
    return mysql_real_escape_string($data, $dbc);
}
$message = NULL;
if (empty($_POST['first_name'])) {
    $fn = FALSE;
    $message .= '<p>Ban chua nhap ten ho!</p>';
} else {
    $fn = escape_data($_POST['first_name']);
}
if (empty($_POST['last_name'])) {
    $ln = FALSE;
    $message .= '<p>Ban chua nhap ten!</p>';
} else {
    $ln = escape_data($_POST['last_name']);
}
if (empty($_POST['email'])) {
    $e = FALSE;
    $message .= '<p>Ban chua nhap dia chi thu dien tu!</p>';
} else {
    $e = escape_data($_POST['email']);
}
if (empty($_POST['username'])) {
    $u = FALSE;
    $message .= '<p>Ban chua nhap ten dang ky!</p>';
} else {
    $u = escape_data($_POST['username']);
}
if (empty($_POST['password1'])) {
    $p = FALSE;
    $message .= '<p>Ban chua nhap mat khau!</p>';
} else {
    if ($_POST['password1'] == $_POST['password2']) {
        $p = escape_data($_POST['password1']);
    } else {
        $p = FALSE;
```

```
            $message .= '<p>Mat khau khong khop voi phan xac
            → nhan!</p>';
        }
    }
    if ($fn && $ln && $e && $u && $p) {
        $query = "SELECT user_id FROM users WHERE
        → username='$u'";
        $result = @mysql_query ($query);
        if (mysql_num_rows($result) == 0) {
            $query = "INSERT INTO users (username, first_name,
            → last_name, email, password, registration_date)
            → VALUES ('$u', '$fn', '$ln', '$e', PASSWORD('$p'),
            → NOW() )";
            $result = @mysql_query ($query);
            if ($result) {
                // Gui thu dien tu neu can.
                echo '<p><b>Ban da duoc dang ky!</b></p>';
                include ('templates/footer.inc');
                exit();
            } else {
                $message = '<p>Ban khong the dang ky do mot loi he
                → thong. Chung toi xin loi vi su co nay.</p><p>'.
                → mysql_error() . '</p>';
            }
        } else {
            $message = '<p>Ten dang nhap nay da duoc dang
            → ky.</p>';
        }
        mysql_close();
    } else {
        $message .= '<p>Hay thu lai.</p>';
    }
}
if (isset($message)) {
    echo '<font color="red">', $message, '</font>';
}
?>
```

```
<form action="<?php echo $_SERVER['PHP_SELF']; ?>"
→ method="post">
<fieldset><legend>Nhap vao thong tin cua ban:</legend>
<p><b>Ten ho:</b> <input type="text" name="first_name"
→ size="15" maxlength="15" value="<?php if
→ (isset($_POST['first_name'])) echo $_POST['first_name'];
→ ?>" /></p>
<p><b>Ten:</b> <input type="text" name="last_name" size="30"
→ maxlength="30" value="<?php if isset($_POST['last_name']))
→ echo $_POST['last_name']; ?>" /></p>
<p><b>Email:</b> <input type="text" name="email" size="40"
→ maxlength="40" value="<?php if (isset($_POST['email']))
→ echo $_POST['email']; ?>" /> </p>
<p><b>Ten dang nhap:</b> <input type="text" name="username"
→ size="10" maxlength="20" value="<?php if
→ (isset($_POST['username'])) echo $_POST['username']; ?>"
→ /></p>
<p><b>Mat khau:</b> <input type="password" name="password1"
→ size="20" maxlength="20" /></p>
<p><b>Xac nhan:</b> <input type="password" name="password2"
→ size="20" maxlength="20" /></p>
</fieldset>
<div align="center"><input type="submit" name="submit"
→ value="Dang ky" /></div>
</form>
<?php
include ('templates/footer.inc');
?>
```

Có thể lấy số mẩu tin bằng cách sử dụng câu lệnh SQL sau:

```
SELECT COUNT(*) FROM users;
```

Tuy nhiên, không nên sử dụng câu lệnh SQL này để lấy ra số mẩu tin sau khi thực hiện câu truy vấn SELECT. Hãy sử dụng `mysql_num_rows()` *và chỉ sử dụng* `COUNT(*)` *trong các trường hợp khác.*

Cập nhật mẩu tin với PHP

Kỹ thuật cuối cùng được giới thiệu trong chương này là cập nhật các mẩu tin cơ sở dữ liệu với mã kịch bản PHP. Việc này đòi hỏi sử dụng câu truy vấn `UPDATE` và kết quả của nó được kiểm tra với hàm `mysql_affected_ rows()` của PHP.

Trong khi hàm **mysql_num_rows()** trả lại số mẩu tin được lấy ra bởi câu truy vấn **SELECT**, thì hàm **mysql_affected_rows()** trả lại số mẩu tin bị ảnh hưởng bởi câu truy vấn **INSERT**, **UPDATE** hoặc **DELETE**. Nó được sử dụng như sau:

```
$num = mysql_affected_rows($dbc);
```

Tham số mà hàm này nhận vào là kết nối cơ sở dữ liệu (**$dbc**), chứ không phải là kết quả của câu truy vấn trước đó (**$result**).

Ví dụ sau là mã kịch bản cho phép người dùng đã đăng ký thay đổi mật khẩu của họ, đồng thời minh họa hai ý tưởng quan trọng:

- Kiểm tra tên đăng nhập và mật khẩu với các giá trị đã được đăng ký.
- Cập nhật mẩu tin trong cơ sở dữ liệu bằng cách dùng khóa chính làm điểm tham chiếu.

Thực hành cập nhật mẩu tin với PHP theo các bước sau:

1. Tạo một mã kịch bản PHP mới trong trình soạn thảo.

```
<?php # Doan ma 6.11 - change_password.php

$page_title = 'Thay doi mat khau.';

include ('templates/header.inc');
```

2. Tạo ra điều kiện chính.

```
if (isset($_POST['submit'])) {
```

Vì trang này vừa hiển thị vừa xử lý biểu mẫu, nên chúng ta sẽ sử dụng điều kiện gửi chuẩn.

3. Đưa vào kết nối cơ sở dữ liệu và viết hàm **escape_data()**.

```
require_once ('../mysql_connect.php');

function escape_data ($data) {

   global $dbc;

   if (ini_get('magic_quotes_gpc')) {

      $data = stripslashes($data);

   }

   return mysql_real_escape_string($data, $dbc);

}
```

Phần đầu của mã kịch bản này sẽ giả lập biểu mẫu đăng ký, bao gồm cả việc xử lý dữ liệu gửi với hàm **escape_data()**. Để cho quá trình đăng ký-đăng nhập làm việc được, dữ liệu phải được quản lý theo cùng cách thức cho cả hai bước.

4. Kiểm tra dữ liệu được gửi lên.

```
$message = NULL;
if (empty($_POST['username'])) {
   $u = FALSE;
   $message .= '<p>Ban chua nhap ten dang ky!</p>';
} else {
   $u = escape_data($_POST['username']);
}
if (empty($_POST['password'])) {
   $p = FALSE;
   $message .= '<p>Ban chua nhap mat khau hien tai!</p>';
} else {
   $p = escape_data($_POST['password']);
}
if (empty($_POST['password1'])) {
   $np = FALSE;
   $message .= '<p>Ban chua nhap mat khau moi!</p>';
} else {
   if ($_POST['password1'] == $_POST['password2']) {
      $np = escape_data($_POST['password1']);
   } else {
      $np = FALSE;
      $message .= '<p>Mat khau moi khong khop voi gia tri
      → xac nhan!</p>';
   }
}
```

Bản thân các quá trình cũng giống như trong mã kịch bản **register.php**. Biểu mẫu có bốn trường nhập: tên đăng nhập, mật khẩu hiện có và mật khẩu mới cùng với xác nhận của mật khẩu mới.

5. Nếu tất cả các bước kiểm tra đều thành công, chúng ta sẽ lấy ra giá trị ID của người dùng.

```
if ($u && $p && $np) {
   $query = "SELECT user_id FROM users WHERE (username='$u'
   → AND password=PASSWORD('$p') )";
```

```
$result = @mysql_query ($query);

$num = mysql_num_rows ($result);

if ($num == 1) {

   $row = mysql_fetch_array($result, MYSQL_NUM);
```

Câu truy vấn này sẽ trả lại trường user_id ứng với mẩu tin phù hợp với tên người dùng và mật khẩu được gửi lên. Để so sánh mật khẩu gửi lên với mật khẩu có trong cơ sở dữ liệu, chúng ta mã hóa nó với hàm **PASSWORD()**. Nếu người dùng đã đăng ký và nhập chính xác cả tên đăng ký lẫn mật khẩu truy cập, thì sẽ có một mẩu tin được trả lại (vì giá trị tên đăng ký là duy nhất trong toàn bộ cơ sở dữ liệu). Cuối cùng, mẩu tin duy nhất này sẽ được gán dưới dạng mảng (có một phần tử) cho biến **$row**.

6. Cập nhật cơ sở dữ liệu.

```
$query = "UPDATE users SET password=PASSWORD('$np') WHERE
→ user_id=$row[0]";

$result = @mysql_query ($query);
```

Câu truy vấn này sẽ thay đổi trường password với mật khẩu mới (chứa trong biến **$np**) của mẩu tin có trường user_id bằng với giá trị đã nhận được trong bước trên (chứa trong biến **$row[0]**).

7. Kiểm tra kết quả của câu truy-vấn.

```
if (mysql_affected_rows() == 1) {

   echo '<p><b>Mat khau da duoc thay doi thanh
   → cong.</b></p>';

   include ('templates/footer.inc');

   exit();

} else {

   $message = '<p>Mat khau cua ban khong the thay doi vi loi
   → he thong. Chung toi xin loi vi su co nay.</p><p>' .
   → mysql_error() . '</p>';

}
```

Phần này của mã kịch bản làm việc giống như trong tập tin **register.php**. Trong trường hợp này, nếu **mysql_affected_rows()** trả lại giá trị 1, mẩu tin đã được cập nhật, một thông điệp sẽ được in ra, tập tin cuối trang được đưa vào và mã kịch bản kết thúc. Nếu không, lỗi cơ sở dữ liệu được in ra.

8. Hoàn tất điều kiện.

```
   } else {
```

```
            $message = '<p>Ten dang nhap va mat khau cua ban
        → khong dung voi cac mau tin trong CSDL.</p>';
        }
        mysql_close();
    } else {
        $message .= '<p>Hay thu lai.</p>';
    }
}
```

9. In ra thông báo lỗi nếu có và hoàn tất mã lệnh PHP.

```
if (isset($message)) {
    echo '<font color="red">', $message, '</font>';
}
?>
```

10. Hiển thị biểu mẫu.

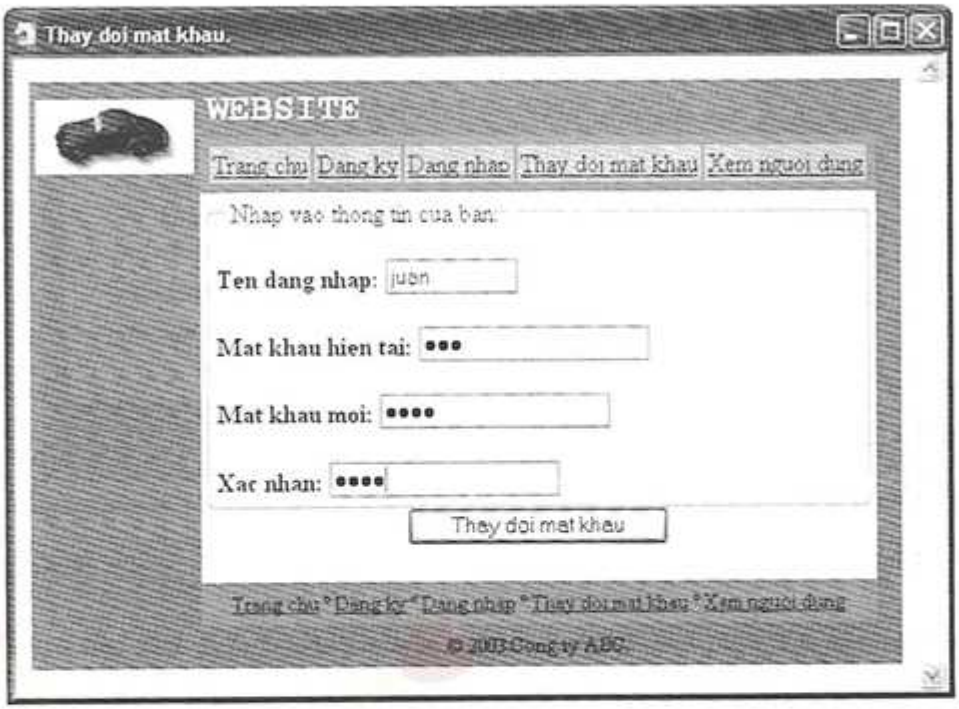
```
<form action="<?php echo $_SERVER['PHP_SELF']; ?>"
→ method="post">
<fieldset><legend>Nhap vao thong tin cua ban:</legend>
<p><b>Ten dang nhap:</b> <input type="text" name="username"
→ size="10" maxlength="20" value="<?php if
→ (isset($_POST['username'])) echo $_POST['username']; ?>"
→ /></p>
<p><b>Mat khau hien tai:</b> <input type="password"
→ name="password" size="20" maxlength="20" /></p>
<p><b>Mat khau moi:</b> <input type="password"
→ name="password1" size="20" maxlength="20" /></p>
<p><b>Xac nhan:</b> <input type="password" name="password2"
→ size="20" maxlength="20" /></p>
</fieldset>
<div align="center"><input type="submit" name="submit"
→ value="Thay doi mat khau" /></div>
</form>
```

11. Đưa vào tập tin cuối trang.

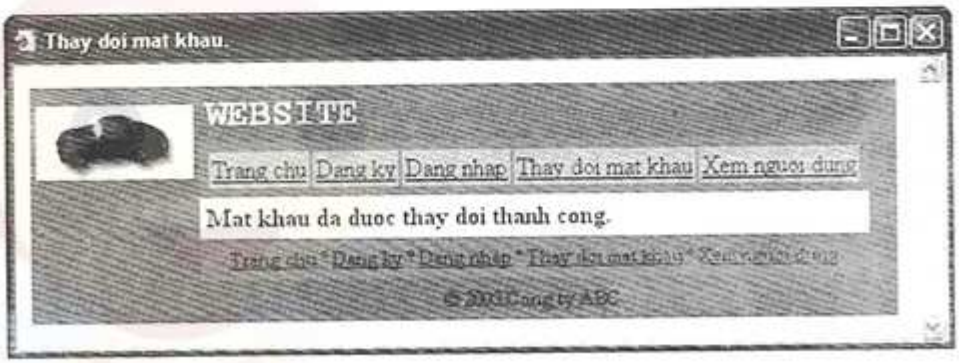
```
<?php
```

```
include ('templates/footer.inc');

?>
```

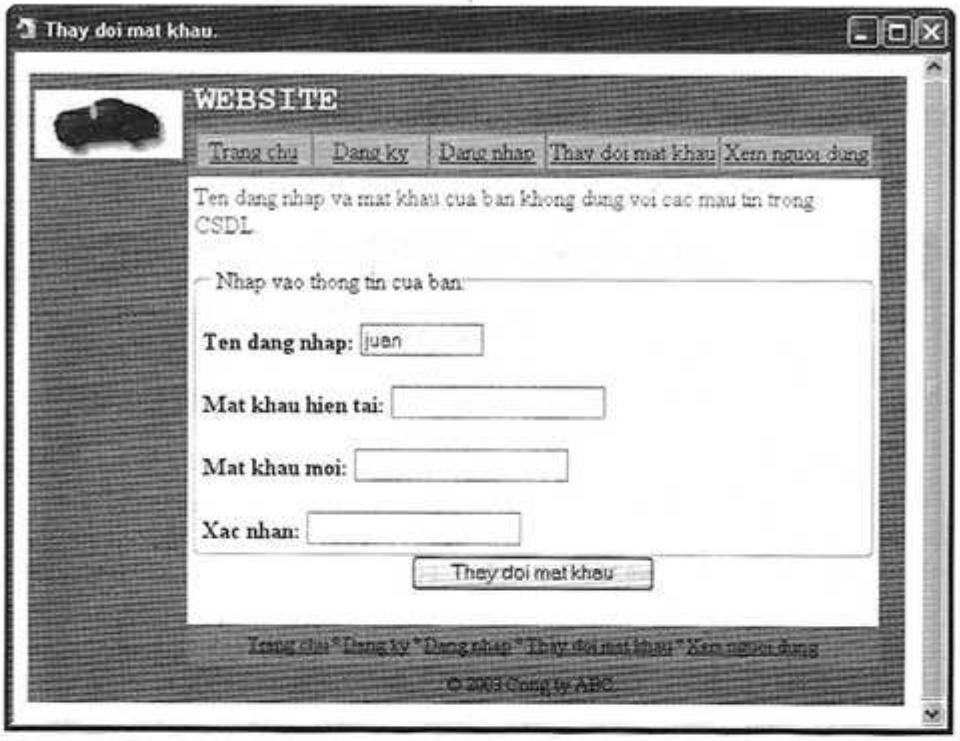
12. Lưu tập tin với tên gọi **change_password.php**, tải nó lên máy chủ và thử trong trình duyệt Web (xem hình 6-24, 6-25 và 6-26). Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 6.11.



Hình 6-24: *Biểu mẫu để thay đổi mật khẩu người dùng.*



Hình 6-25: *Mật khẩu đã được thay đổi trong cơ sở dữ liệu.*



Hình 6-26: *Nếu người dùng nhập tên đăng nhập và mật khẩu không đúng, mật khẩu mới sẽ không được cập nhật.*

Đoạn mã 6.11: *Mã kịch bản* ***change_password.php*** *sẽ thực hiện câu truy vấn* ***UPDATE*** *trên cơ sở dữ liệu và sử dụng hàm* ***mysql_affected_rows()*** *để xác nhận thay đổi.*

```
<?php # Doan ma 6.11 - change_password.php
$page_title = 'Thay doi mat khau.';
include ('templates/header.inc');
if (isset($_POST['submit'])) {
   require_once ('../mysql_connect.php');
   function escape_data ($data) {
      global $dbc;
      if (ini_get('magic_quotes_gpc')) {
         $data = stripslashes($data);
      }
      return mysql_real_escape_string($data, $dbc);
   }
   $message = NULL;
   if (empty($_POST['username'])) {
      $u = FALSE;
```

```
    $message .= '<p>Ban chua nhap ten dang ky!</p>';
  } else {
    $u = escape_data($_POST['username']);
  }
  if (empty($_POST['password'])) {
    $p = FALSE;
    $message .= '<p>Ban chua nhap mat khau hien tai!</p>';
  } else {
    $p = escape_data($_POST['password']);
  }
  if (empty($_POST['password1'])) {
    $np = FALSE;
    $message .= '<p>Ban chua nhap mat khau moi!</p>';
  } else {
    if ($_POST['password1'] == $_POST['password2']) {
      $np = escape_data($_POST['password1']);
    } else {
      $np = FALSE;
      $message .= '<p>Mat khau moi khong khop voi gia tri
      → xac nhan!</p>';
    }
  }
  if ($u && $p && $np) {
    $query = "SELECT user_id FROM users WHERE (username='$u'
    → AND password=PASSWORD('$p') )";
    $result = @mysql_query ($query);
    $num = mysql_num_rows ($result);
    if ($num == 1) {
      $row = mysql_fetch_array($result, MYSQL_NUM);
      $query = "UPDATE users SET password=PASSWORD('$np')
      → WHERE user_id=$row[0]";
      $result = @mysql_query ($query);
      if (mysql_affected_rows() == 1) {
        // Gui thu dien tu neu can.
        echo '<p><b>Mat khau da duoc thay doi thanh
        → cong.</b></p>';
        include ('templates/footer.inc');
        exit();
      } else {
        $message = '<p>Mat khau cua ban khong the thay doi
        → vi loi he thong. Chung toi xin loi vi su co
        → nay.</p><p>' . mysql_error() . '</p>';
      }
    } else {
      $message = '<p>Ten dang nhap va mat khau cua ban
```

```
        → khong dung voi cac mau tin trong CSDL.</p>';
      }
      mysql_close();
   } else {
      $message .= '<p>Hay thu lai.</p>';
   }
}
if (isset($message)) {
   echo '<font color="red">', $message, '</font>';
}
?>
<form action="<?php echo $_SERVER['PHP_SELF']; ?>"
→ method="post">
<fieldset><legend>Nhap vao thong tin cua ban:</legend>
<p><b>Ten dang nhap:</b> <input type="text" name="username"
→ size="10" maxlength="20" value="<?php if
→ (isset($_POST['username'])) echo $_POST['username']; ?>"
→ /></p>
<p><b>Mat khau hien tai:</b> <input type="password"
→ name="password" size="20" maxlength="20" /></p>
<p><b>Mat khau moi:</b> <input type="password"
→ name="password1" size="20" maxlength="20" /></p>
<p><b>Xac nhan:</b> <input type="password" name="password2"
→ size="20" maxlength="20" /></p>
</fieldset>
<div align="center"><input type="submit" name="submit"
→ value="Thay doi mat khau" /></div>
</form>
<?php
include ('templates/footer.inc');
?>
```

Nếu xóa tất cả mẩu tin trong một bảng, hàm ***mysql_affected_ rows()*** *sẽ trả lại* ***0*** *ngay cả khi câu truy vấn thành công và mọi mẩu tin đều bị xóa bỏ.*

Nếu câu truy vấn ***UPDATE*** *không thực sự thay đổi giá trị của một cột nào (ví dụ, mật khẩu được thay thế với cùng giá trị), hàm* ***mysql_affected_rows()*** *sẽ trả lại giá trị* ***0****.*

Hàm MySQL quan trọng cuối cùng, hàm ***mysql_insert_id()****, sẽ được đề cập đến trong Chương 11 "Ví dụ - Quản lý nội dung".*

Câu lệnh ***if(mysql_affected_rows()==1)*** *được sử dụng ở đây có thể được áp dụng cho mã kịch bản* ***register.php*** *vì nó chính xác hơn câu lệnh* ***if($result)****.*

Chương 7:

COOKIE VÀ SESSION

Giao thức HTTP (Hypertext Transfer Protocol) là một công nghệ không lưu giữ trạng thái, có nghĩa các trang Web là độc lập với nhau. HTTP không có cách nào để theo dõi người dùng hoặc giữ lại các biến khi họ truy xuất site. Bằng cách dùng ngôn ngữ kịch bản Web như PHP, chúng ta có thể khắc phục được việc không lưu giữ trạng thái này của Web. Có một số tùy chọn để áp dụng và hai tùy chọn phổ biến nhất là cookie và session (phiên truy cập).

Trước khi cookie ra đời, việc lướt Web chỉ là một cuộc du lịch không có ghi nhận quá trình. Mặc dù trình duyệt ghi lại những trang Web đã truy cập, cho phép sử dụng nút Back để quay trở lại trang đã truy cập trước đó, nhưng máy chủ không lưu giữ lại thông tin về người đã truy cập. Nếu máy chủ không có khả năng theo dõi truy cập, ta không thể có được giỏ mua hàng trên mạng cũng như tùy biến các Website theo ý mình được.

Session (phiên truy cập), được bổ sung trong PHP 4, được cải tiến từ cookie. Session cho phép các ứng dụng Web lưu giữ và lấy ra nhiều thông tin hơn so với cookie. Cả hai công nghệ này đều dễ dùng với PHP và đáng để tìm hiểu. Chương này sẽ giải thích từng kỹ thuật một với ví dụ là một hệ thống đăng nhập được bắt đầu từ chương trước.

Sử dụng cookie

Cookie là cách máy chủ lưu giữ thông tin trên máy tính người dùng để nhớ được người đã truy cập. Hãy xem cookie như là một thẻ tên: nó báo cho máy chủ tên người dùng và các thông tin liên quan.

Với đa số người dùng Web, cookie thường bị mang tiếng xấu vì họ nghĩ rằng cookie cho phép máy chủ biết được quá nhiều thông tin về họ. Tuy nhiên, cookie chỉ được sử dụng để lưu giữ thông tin mà máy chủ được cung cấp, vì thế nó không nguy hiểm bằng bất kỳ những gì có trên mạng. Thật không may, nhiều người vẫn còn hiểu sai về công nghệ này và không dám sử dụng. Điều này có thể làm suy yếu ứng dụng Web của họ.

Kiểm tra cookie

Để làm việc với cookie, bạn phải có khả năng kiểm tra chính xác sự hiện diện của nó. Cách tốt nhất để thực hiện việc này là yêu cầu trình duyệt thực hiện một điều gì đó khi nhận được một cookie. Ví dụ, cung cấp một hộp nhắc với thông tin nhất định mỗi khi PHP gửi một cookie.

Để thiết lập điều này với Internet Explorer trên Windows 2000, hãy chọn Internet Options trong menu Tools. Sau đó, nhấp vào thẻ Privacy, tiếp đến là nút Advanced phía trong mục Settings. Nhấp vào mục "Override automatic cookie handling" và sau đó chọn "Prompt" cho cả First và Third-party Cookies. Nếu bạn đang sử dụng một phiên bản Windows khác, chi tiết các bước trên có thể khác đi một chút (ví dụ, Windows XP quản lý cookie trong thẻ Security).

Với Nestcape hoặc Mozilla trên bất kỳ môi trường nào, hãy chọn Preferences phía trong menu Edit (hoặc trong menu Application trên Mac OS X). Sau đó, chọn Cookies phía trong menu con Privacy & Security và nhấp "Ask me before storing a cookie".

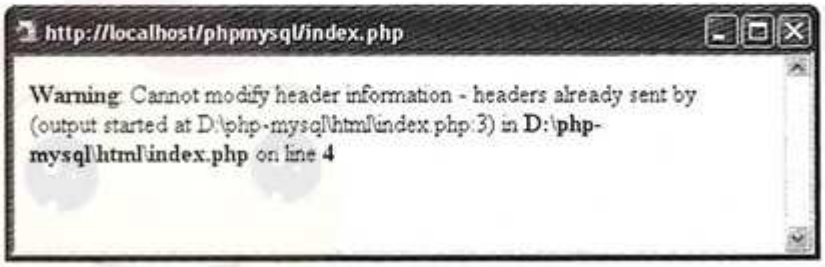
Đối với Opera trên bất kỳ môi trường nào, hãy chọn Preferences trong menu File (hoặc Application trên Mac OS X). Nhấp Privacy và sau đó chọn "Display... cookies" trong cả hai menu thả xuống.

Cuối cùng, nếu bạn đang sử dụng IE, OmniWeb hoặc Camino trên máy Mac, hãy chọn Preferences trong menu Edit (hoặc trong menu Application trên máy Mac OS X). Sau đó, nhấp vào Cookies và chọn "Prompt for each cookie" hoặc "Ask for each cookie".

Trong phần cookies của một số trình duyệt, có một tùy chọn để xem các cookie hiện tại. Đây cũng là một tùy chọn hữu ích cho các ví dụ trong chương này.

Thiết lập cookie

Điều quan trọng nhất cần hiểu về cookie là chúng phải được gửi từ máy chủ xuống máy khách hàng trước bất kỳ thông tin nào khác. Nếu máy chủ gửi một cookie sau khi trình duyệt Web đã nhận được HTML (cho dù là khoảng trắng), thì một thông báo lỗi sẽ xuất hiện và cookie sẽ không được gửi (xem hình 7-1). Đây là một lỗi phổ biến nhất về cookie cho đến lúc này.



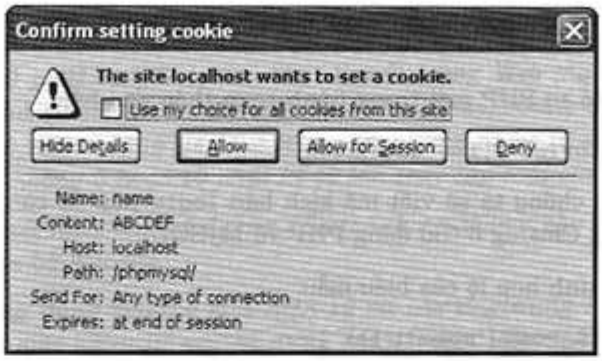
Hình 7-1: *Thông báo lỗi headers already sent... là lỗi phổ biến nhất khi tạo cookie.*

Cookie được gửi thông qua hàm **setcookie()**:

```
setcookie(ten, gia_tri);
```

```
setcookie('name', 'ABCDEF');
```

Dòng lệnh thứ hai sẽ gửi một cookie đến trình duyệt với tên là name và giá trị là ABCDEF (xem hình 7-2).



Hình 7-2: *Nếu thiết lập trình duyệt để thông báo khi nhận được cookie, bạn sẽ thấy một thông báo như trong hình khi nó nhận được một cookie.*

Có thể tiếp tục gửi nhiều cookie đến trình duyệt với hàm **setcookie()**:

```
setcookie('ID', '123');

setcookie('email','user@domain.com');
```

Cũng như tên biến trong PHP, tên của cookie có phân biệt dạng chữ và không cho phép có khoảng trắng hoặc dấu ngắt.

Sử dụng URL tuyệt đối với hàm header()

Trong chương này, chúng ta sẽ chuyển hướng trình duyệt bằng cách dùng hàm ***header()*** *như đã được đề cập trong Chương 3 "Tạo Website động". Bạn nên sử dụng URL tuyệt đối với hàm này (ví dụ http://www.domain.com/site.php) thay vì sử dụng địa chỉ tương đối (ví dụ site.php). Có hai cách để thực hiện điều này với mã kịch bản. Cách đầu tiên là mã hóa cố định URL:*

```
header('Location: http://www.domain.com/site.php');
```

Phương pháp này sẽ làm việc trừ khi thay đổi cấu hình máy chủ, URL hoặc di chuyển tập tin kịch bản.

Một phương pháp linh động hơn là sử dụng biến siêu toàn cục ***$_SERVER*** *và hàm* ***dirname()*** *(hàm này trả lại đường dẫn thư mục của mã kịch bản hiện tại). Tài liệu PHP đề nghị dùng cách thứ hai để chuyển hướng đến một trang Web khác nằm trong cùng thư mục với mã kịch bản hiện tại:*

```
header("Location: http://" . $_SERVER['HTTP_HOST'] .
→ dirname($_SERVER['HTTP_SELF']) . "/newpage.php");
```

Hàm này sẽ chuyển người dùng đến trang Web mới trong cùng miền (domain) và cùng thư mục với mã kịch bản hiện tại. Đây là phương pháp sẽ được sử dụng trong chương này.

Thực hành gửi một cookie theo các bước sau:

1. Tạo một hồ sơ PHP mới trong trình soạn thảo văn bản.

```
<?php # Script 7.1 - login.php
```

Với ví dụ này, chúng ta sẽ viết mã kịch bản **login.php** để cùng làm việc với mã kịch bản trong Chương 6 "Sử dụng PHP và MySQL".

2. Kiểm tra tính hợp lệ của biểu mẫu.

```
if (isset($_POST['submit'])) {
   require_once ('../mysql_connect.php');
   function escape_data ($data) {
      global $dbc;
      if (ini_get('magic_quotes_gpc')) {
         $data = stripslashes($data);
      }
      return mysql_real_escape_string($data, $dbc);
   }
   $message = NULL;
   if (empty($_POST['username'])) {
      $u = FALSE;
      $message .= '<p>Ban chua nhap ten dang nhap!</p>';
   } else {
      $u = escape_data($_POST['username']);
   }
   if (empty($_POST['password'])) {
      $p = FALSE;
      $message .= '<p>Ban chua nhap mat khau!</p>';
   } else {
      $p = escape_data($_POST['password']);
   }
```

Các bước kiểm tra này rất giống với các bước được thực hiện trong Chương 6. Điều kiện chính là kiểm tra xem biểu mẫu có được gửi hay không. Nếu đúng, thực hiện kết nối cơ sở dữ liệu bằng cách đưa vào mã kịch bản kết nối. Tiếp đến là hàm để mã hóa escape dữ liệu gửi. Cuối cùng, tên đăng nhập và mật khẩu sẽ được kiểm tra giá trị.

3. Lấy ra **user_id** và **first_name** của người dùng từ trong cơ sở dữ liệu.

```
if ($u && $p) {

   $query = "SELECT user_id, first_name FROM users WHERE
   → username='$u' AND password=PASSWORD('$p')";

   $result = @mysql_query ($query);

   $row = mysql_fetch_array ($result, MYSQL_NUM);
```

Nếu cả hai bước kiểm tra hợp lệ đều thành công, cơ sở dữ liệu sẽ được truy vấn, lấy ra các giá trị **user_id** và **first_name** ứng với mẩu tin có tên đăng nhập và mật khẩu trùng với các thông tin mà người dùng cung cấp.

4. Nếu người dùng nhập vào thông tin đúng, cho phép họ đăng nhập.

```
if ($row) {

   setcookie ('first_name', $row[1]);

   setcookie ('user_id', $row[0]);

    header ("Location:  http://" . $_SERVER['HTTP_HOST'] .
    → dirname($_SERVER['PHP_SELF']) . "/loggedin.php");

   exit();

} else {

    $message = '<p>Ten dang nhap va mat khau khong phu hop
    → voi du lieu trong CSDL.</p>';

}
```

Biến **$row** sẽ chỉ có giá trị nếu câu truy vấn trên trả lại ít nhất một mẩu tin (cho biết tên đăng nhập và mật khẩu phù hợp với thông tin có trong cơ sở dữ liệu). Trong trường hợp này, hai cookie sẽ được tạo và người dùng sẽ được chuyển đến mã kịch bản **loggedin.php**. Nếu biến **$row** không có giá trị (thông tin đăng nhập sai), thông báo lỗi sẽ được gán cho biến **$message**.

5. Đóng kết nối cơ sở dữ liệu và hoàn tất điều kiện chính.

```
    mysql_close();

} else {

    $message .= '<p>Hay thu lai.</p>';
```

```
    }
}
```

6. Đưa vào phần đầu HTML và in thông báo lỗi nếu có.

```
$page_title = 'Dang nhap';
include ('templates/header.inc');
if (isset($message)) {
    echo '<font color="red">', $message, '</font>';
}
?>
```

7. Hiển thị biểu mẫu HTML.

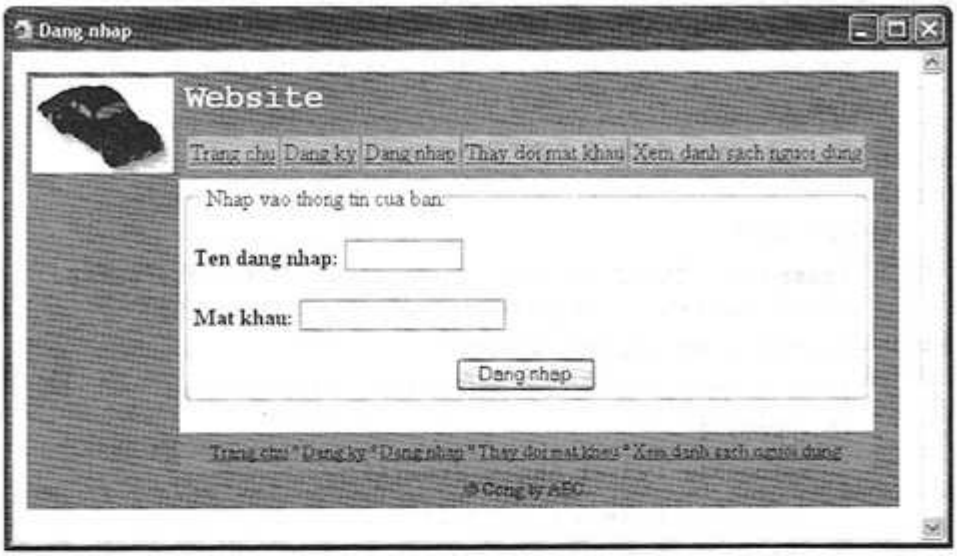
```
<form action="<?php echo $_SERVER['PHP_SELF']; ?>"
→ method="post">
<fieldset><legend>Nhap vao thong tin cua ban:</legend>
<p><b>Ten dang nhap:</b> <input type="text" name="username"
→ size="10" maxlength="20" value="<?php if
→ (isset($_POST['username'])) echo $_POST['username']; ?>"
→ /></p>
<p><b>Mat khau:</b> <input type="password" name="password"
→ size="20" maxlength="20" /></p>
<div align="center"><input type="submit" name="submit"
→ value="Dang nhap" /></div>
</fieldset>
</form>
```

Biểu mẫu HTML nhận vào hai giá trị nhập (tên người dùng và mật khẩu) và gửi dữ liệu cho cùng trang này (`$_SERVER['PHP_SELF']`).

8. Đưa vào phần cuối trang PHP.

```
<?php
include ('templates/footer.inc');
?>
```

9. Lưu tập tin với tên gọi **login.php**, tải nó lên máy chủ Web trong cùng thư mục với các tập tin đã được thực hiện trong Chương 6 và chạy thử trong trình duyệt (xem hình 7-3). Mã lệnh đầy đủ của tập tin `login.php` được thể hiện trong đoạn mã 7.1.



Hình 7-3: *Biểu mẫu đăng nhập.*

Đoạn mã 7.1: *Phiên bản mới của tập tin* ***login.php*** *sẽ tạo cookie khi đăng nhập thành công.*

```
<?php # Doan ma 7.1 - login.php
if (isset($_POST['submit'])) {
   require_once ('../mysql_connect.php');
   function escape_data ($data) {
      global $dbc;
      if (ini_get('magic_quotes_gpc')) {
         $data = stripslashes($data);
      }
      return mysql_real_escape_string($data, $dbc);
   }
   $message = NULL;
   if (empty($_POST['username'])) {
      $u = FALSE;
      $message .= '<p>Ban chua nhap ten dang nhap!</p>';
   } else {
      $u = escape_data($_POST['username']);
   }
   if (empty($_POST['password'])) {
```

```
      $p = FALSE;
      $message .= '<p>Ban chua nhap mat khau!</p>';
   } else {
      $p = escape_data($_POST['password']);
   }
   if ($u && $p) {
      $query = "SELECT user_id, first_name FROM users WHERE
      → username='$u' AND password=PASSWORD('$p')";
      $result = @mysql_query ($query);
      $row = mysql_fetch_array ($result, MYSQL_NUM);
      if ($row) {
         setcookie ('first_name', $row[1]);
         setcookie ('user_id', $row[0]);
         header ("Location:  http://" . $_SERVER['HTTP_HOST'] .
         → dirname($_SERVER['PHP_SELF']) . "/loggedin.php");
         exit();
      } else {
         $message = '<p>Ten dang nhap va mat khau khong
         → phu hop voi du lieu trong CSDL.</p>';
      }
      mysql_close();
   } else {
      $message .= '<p>Hay thu lai.</p>';
   }
}
$page_title = 'Dang nhap';
include ('templates/header.inc');
if (isset($message)) {
   echo '<font color="red">', $message, '</font>';
}
?>
<form action="<?php echo $_SERVER['PHP_SELF']; ?>"
→ method="post">
<fieldset><legend>Nhap vao thong tin cua ban:</legend>
<p><b>Ten dang nhap:</b> <input type="text" name="username"
→ size="10" maxlength="20" value="<?php if
→ (isset($_POST['username'])) echo $_POST['username']; ?>"
```

```
→ /></p>
<p><b>Mat khau:</b> <input type="password" name="password"
→ size="20" maxlength="20" /></p>
<div align="center"><input type="submit" name="submit"
→ value="Dang nhap" /></div>
</fieldset></form>
<?php
include ('templates/footer.inc');
?>
```

Cookie có kích thước giới hạn vào khoảng 4KB và mỗi trình duyệt Web có thể nhớ khoảng 20 cookie từ một máy chủ bất kỳ.

Vì các cookie phụ thuộc vào phần đầu HTTP, chúng ta có thể thiết lập chúng trong PHP bằng hàm ***header()****. Một điều quan trọng cần nhớ là hàm* ***setcookie()*** *và* ***header()*** *phải được gọi trước khi gửi bất kỳ dữ liệu nào cho trình duyệt Web.*

Hàm ***setcookie()*** *là một trong số ít hàm của PHP có thể tạo ra các kết quả khác nhau trong mỗi loại trình duyệt (vì các trình duyệt xử lý các cookie theo các cách khác nhau). Bạn nên kiểm tra kết quả của mình trên nhiều trình duyệt và môi trường để đảm bảo tính nhất quán.*

Chương 10 "Các chủ đề mở rộng", sẽ đề cập đến việc kiểm soát kết quả xuất ra để cookie có thể được gửi đi từ bất kỳ điểm nào trong mã kịch bản.

Truy xuất cookie

Để lấy một giá trị từ cookie, chúng ta chỉ cần tham chiếu đến biến siêu toàn cục `$_COOKIE` và dùng tên cookie làm khóa (giống như khi làm việc với mảng). Ví dụ, để lấy giá trị của cookie được thiết lập với dòng lệnh:

```
setcookie('username','Larry');
```

chúng ta sử dụng:

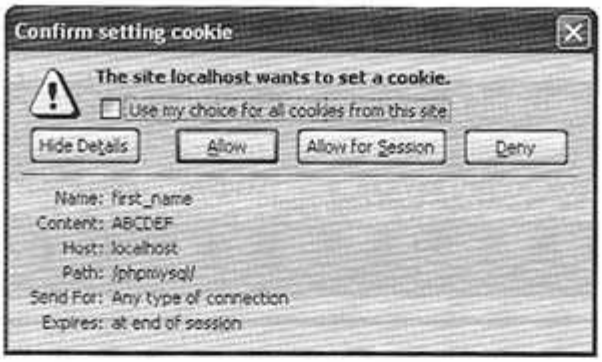
```
$_COOKIE['username'].
```

Với thiết lập `register_globals` được kích hoạt trong PHP, chúng ta có thể tham chiếu đến cookie trên bằng cách sử dụng cú pháp `$username`. Tuy nhiên, cách này kém an toàn hơn.

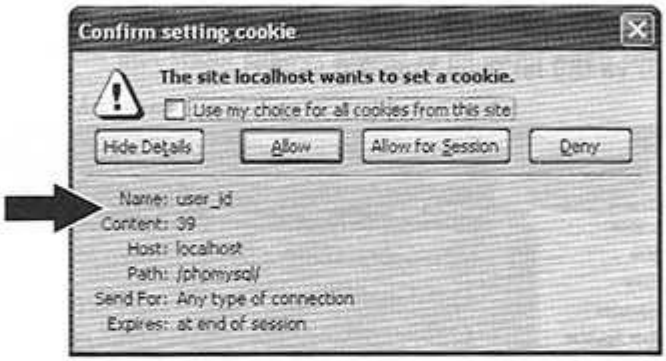
Thực hành truy xuất một cookie theo các bước sau:

1. Tạo một hồ sơ PHP mới trong trình soạn thảo văn bản.

```
<?php # Script 7.2 - loggedin.php
```



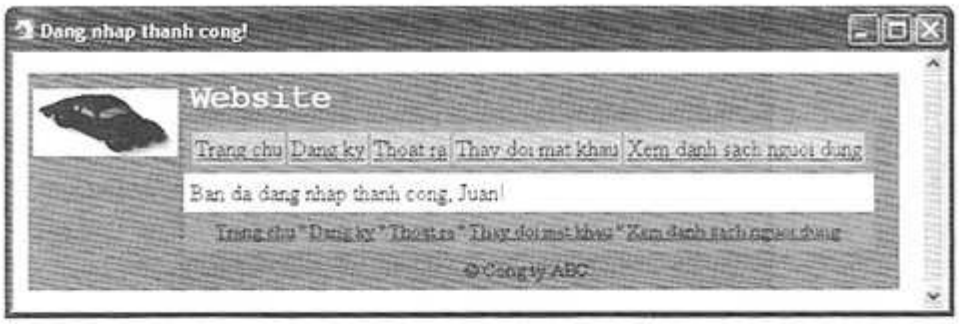
Hình 7-6: *Cookie* ***first_name*** *với giá trị ABCDEF.*



Hình 7-7: *Cookie* ***user_id*** *với giá trị 39.*

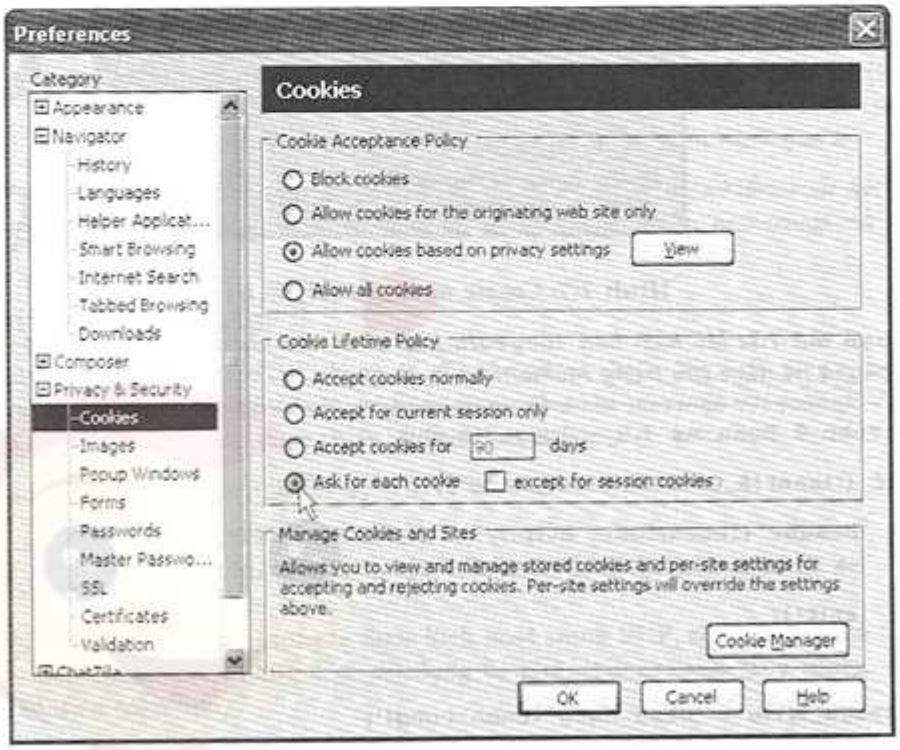
Đoạn mã 7.2: *Mã kịch bản* ***loggedin.php*** *sẽ in câu chào mừng người dùng dựa trên giá trị được lưu trong cookie.*

```
<?php # Doan ma 7.2 - loggedin.php

if (!isset($_COOKIE['first_name'])) {

   header ("Location:  http://" . $_SERVER['HTTP_HOST'] .
   → dirname($_SERVER['PHP_SELF']) . "/index.php");

   exit();

}

$page_title = 'Dang nhap thanh cong!';

include ('templates/header.inc');

echo "<p>Ban da dang nhap thanh cong,
→ {$_COOKIE['first_name']}!</p>";
```

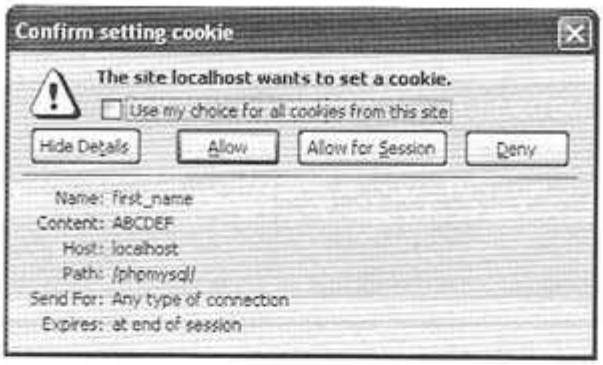


Hình 7-4: *Nếu sử dụng đúng tên và mật khẩu, người dùng sẽ được chuyển đến trang này sau khi đăng nhập.*

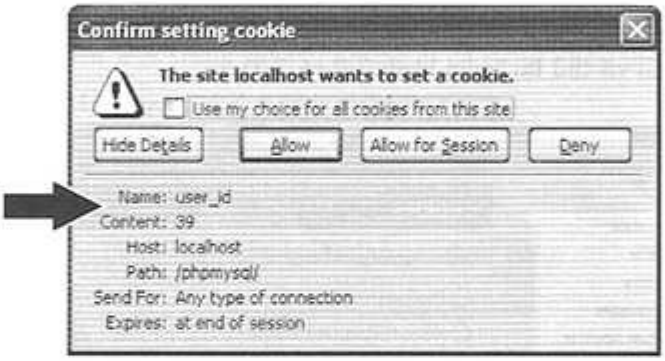
7. Nếu cần, hãy thay đổi các thiết lập cookie đối với trình duyệt của bạn (xem hình 7-5) và thử lại (xem hình 7-6 và 7-7).



Hình 7-5: *Để xem ảnh hưởng của hàm* `setcookie()`*, hãy thiết lập trình duyệt để nó yêu cầu xác nhận trước khi lưu giữ một cookie.*



Hình 7-6: *Cookie* `first_name` *với giá trị ABCDEF.*



Hình 7-7: *Cookie* `user_id` *với giá trị 39.*

Đoạn mã 7.2: *Mã kịch bản* `loggedin.php` *sẽ in câu chào mừng người dùng dựa trên giá trị được lưu trong cookie.*

```
<?php # Doan ma 7.2 - loggedin.php

if (!isset($_COOKIE['first_name'])) {

   header ("Location:  http://" . $_SERVER['HTTP_HOST'] .
   → dirname($_SERVER['PHP_SELF']) . "/index.php");

   exit();

}

$page_title = 'Dang nhap thanh cong!';

include ('templates/header.inc');

echo "<p>Ban da dang nhap thanh cong,
→ {$_COOKIE['first_name']}!</p>";
```

```
include ('templates/footer.inc');
?>
```

*Với thiết lập **register_globals** được kích hoạt, PHP sẽ tải các biến theo một trật tự nhất định (phụ thuộc vào thiết lập trong tập tin **php.ini**). Thường là theo trật tự: get, post, cookie, session. Nếu không sử dụng các mảng siêu toàn cục để tham chiếu đến các biến, thì giá trị của biến **$username** trong một biểu mẫu có thể bị ghi đè bởi giá trị của biến **$username** chứa trong một cookie.*

*Không thể truy xuất cookie khi trang thiết lập nó (**login.php**) chưa được tải lại hoặc một trang khác chưa được truy xuất (nói cách khác, không thể thiết lập và truy xuất một cookie trong cùng một lần xử lý mã kịch bản PHP).*

Nếu người dùng từ chối cookie hoặc thiết lập trình duyệt để không chấp nhận cookie, họ sẽ được chuyển đến trang chủ trong ví dụ này, ngay cả khi đăng nhập thành công. Vì lý do đó, bạn cần phải thông báo cho người dùng biết việc sử dụng cookie là bắt buộc.

Thiết lập các tham số cookie

Để tạo cookie, chúng ta chỉ cần truyền các tham số tên và giá trị cho hàm **setcookie()**. Tuy nhiên, bạn nên quan tâm đến một số tham số khác của nó. Hàm này có thể nhận vào bốn tham số nữa và chúng sẽ ảnh hưởng đến cách thiết lập cookie.

```
setcookie('tên','giá trị',thời hạn,'đường dẫn','tên miền',an toàn);
```

Tham số thời hạn được dùng để thiết lập thời gian tồn tại của một cookie. Thời gian này được tính theo số giây so với mốc thời gian 1-1-1970. Nếu không được thiết lập, cookie sẽ tiếp tục tồn tại cho đến khi người dùng đóng trình duyệt. Thông thường, thời gian tồn tại được xác định bằng cách cộng giờ hiện tại (sử dụng hàm **time()**) với một số phút hoặc giờ. Dòng lệnh dưới đây sẽ thiết lập thời gian tồn tại của cookie là 1 giờ (60 giây x 60 phút) tính từ thời điểm hiện tại:

```
setcookie('tên', 'giá trị', time() + 3600);
```

Các tham số đường dẫn và tên miền được dùng để giới hạn một cookie với một thư mục cụ thể trên Website (đường dẫn) hoặc với một địa chỉ nhất định. Ví dụ, chúng ta có thể giới hạn một cookie để nó chỉ tồn tại phía trong thư mục admin (hoặc thư mục con của nó) của một miền (domain).

```
setcookie('tên', 'giá trị', time() + 3600, '/admin/');
```

Cuối cùng, tham số an toàn cho biết cookie có được gửi trên một kết nối HTTPS an toàn hay không. Giá trị **1** cho biết kết nối an toàn phải được sử dụng và giá trị **0** cho biết một kết nối thông thường là đủ.

```
setcookie('tên', 'giá trị', time() + 3600, '/admin/', 1);
```

Với tất cả các hàm có nhận vào tham số, bạn phải truyền các giá trị theo đúng thứ tự. Để bỏ qua một tham số nào đó, hãy sử dụng giá trị NULL hoặc một chuỗi trống. Các giá trị thời hạn và an toàn là các số nguyên nên không cần đặt trong dấu nháy.

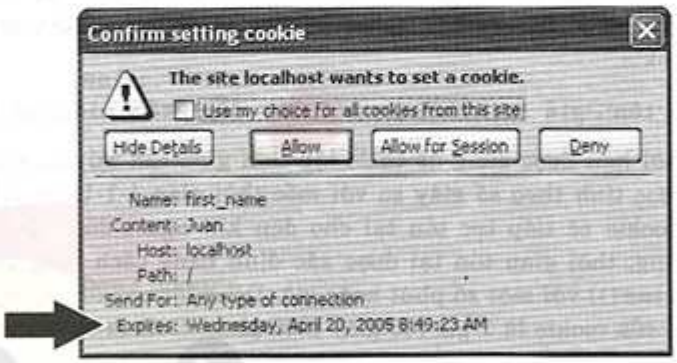
Để minh họa, chúng ta sẽ bổ sung một thiết lập thời hạn cho các cookie để chúng chỉ tồn tại trong vòng 1 giờ như sau:

1. Mở tập tin **login.php** trong trình soạn thảo văn bản (tham khảo lại đoạn mã 7.1).
2. Thay đổi hai dòng **setcookie()** để đưa vào ngày hết hạn (60 phút tính từ thời điểm hiện tại).

```
setcookie('first_name',$row[1],time()+3600,'/','',0);

setcookie('user_id',$row[0],time()+3600,'/','',0);
```

Bằng cách thiết lập ngày hết hạn theo **time() + 3600**, cookie sẽ tồn tại 1 giờ sau khi nó được thiết lập. Ở đây, chúng ta cũng sử dụng các tham số khác.

3. Lưu lại mã kịch bản, tải nó lên máy chủ Web và chạy thử trong trình duyệt Web (xem hình 7-8). Mã lệnh đầy đủ của tập tin **login.php** được thể hiện trong đoạn mã 7.3.



Hình 7-8: *Một khi ngày hoặc giờ hết hạn đã được thiết lập, nó sẽ được phản ánh trong cookie được gửi tới trình duyệt.*

Đoạn mã 7.3: *Mã kịch bản **login.php** giờ đây sử dụng tất cả các tham số mà hàm **setcookie()** có thể nhận.*

```
<?php # Doan ma 7.3 - login.php
if (isset($_POST['submit'])) {
   require_once ('../mysql_connect.php');
   function escape_data ($data) {
```

```
    global $dbc;
    if (ini_get('magic_quotes_gpc')) {
       $data = stripslashes($data);
    }
    return mysql_real_escape_string($data, $dbc);
}
$message = NULL;
if (empty($_POST['username'])) {
    $u = FALSE;
    $message .= '<p>Ban chua nhap ten dang nhap!</p>';
} else {
    $u = escape_data($_POST['username']);
}
if (empty($_POST['password'])) {
    $p = FALSE;
    $message .= '<p>Ban chua nhap mat khau!</p>';
} else {
    $p = escape_data($_POST['password']);
}

if ($u && $p) {
    $query = "SELECT user_id, first_name FROM users WHERE
    -> username='$u' AND password=PASSWORD('$p')";
    $result = @mysql_query ($query);
    $row = mysql_fetch_array ($result, MYSQL_NUM);
    if ($row) {
       setcookie ('first_name', $row[1], time()+3600, '/',
       -> '', 0);
       setcookie ('user_id', $row[0], time()+3600, '/', '',
       -> 0);
       header ("Location:  http://" . $_SERVER['HTTP_HOST'] .
       -> dirname($_SERVER['PHP_SELF']) . "/loggedin.php");
       exit();
    } else {
       $message = '<p>Ten dang nhap va mat khau khong
       -> phu hop voi du lieu trong CSDL.</p>';
    }
```

```
        mysql_close();
    } else {
        $message .= '<p>Hay thu lai.</p>';
    }
}
$page_title = 'Dang nhap';
include ('templates/header.inc');
if (isset($message)) {
    echo '<font color="red">', $message, '</font>';
}
?>
<form action="<?php echo $_SERVER['PHP_SELF']; ?>"
→ method="post">
<fieldset><legend>Nhap vao thong tin cua ban:</legend>
<p><b>Ten dang nhap:</b> <input type="text" name="username"
→ size="10" maxlength="20" value="<?php if
→ (isset($_POST['username'])) echo $_POST['username']; ?>"
→ /></p>
<p><b>Mat khau:</b> <input type="password" name="password"
→ size="20" maxlength="20" /></p>
<div align="center"><input type="submit" name="submit"
→ value="Dang nhap" /></div>
</fieldset>
</form>
<?php
include ('templates/footer.inc');
?>
```

Một số trình duyệt gặp khó khăn với cookie khi ta không cung cấp đầy đủ tham số. Việc xác định tất cả các tham số (cho dù chuỗi rỗng, như chúng ta đã thực hiện) sẽ đưa lại kết quả đáng tin cậy trên nhiều trình duyệt.

Một số hướng dẫn về cách thiết lập thời hạn của cookie:

- *Nếu muốn cookie tồn tại cùng với một phiên truy cập, bạn không cần thiết lập thời hạn cụ thể.*
- *Nếu muốn cookie tiếp tục tồn tại sau khi người dùng đóng và mở lại trình duyệt, bạn cần thiết lập một thời hạn hết hạn là một thời điểm trong tương lai (ví dụ, một tháng kể từ ngày hiện tại).*

- *Nếu sự tồn tại của cookie là một điều nguy hiểm, bạn nên thiết lập thời hạn khoảng một giờ hoặc ít hơn để nó không tồn tại quá lâu sau khi người dùng đóng trình duyệt.*

Với mục đích an toàn, bạn nên thiết lập thời hạn hết hạn của cookie là 5 hoặc 10 phút và gửi lại cookie với mỗi trang Web mà người dùng truy xuất (giả sử cookie đã tồn tại). Theo cách này, cookie sẽ tiếp tục tồn tại miễn là người dùng tiếp tục truy cập, nhưng nó sẽ tự động mất hiệu lực trong khoảng 5 hoặc 10 phút kể từ lần truy cập cuối cùng.

Việc thiết lập tham số đường dẫn thành '/' sẽ làm cho cookie có hiệu lực với toàn bộ Website.

Các ứng dụng thương mại điện tử và các ứng dụng khác có liên quan đến bí mật riêng tư nên sử dụng kết nối SSL (Secure Socket Layer) cho tất cả các giao dịch, kể cả cookie.

Xóa cookie

Điều cuối cùng cần biết về sử dụng cookie là xóa nó. Tùy theo thiết lập, cookie sẽ tự động hết hạn khi người dùng đóng trình duyệt hoặc hết thời hạn tồn tại. Tuy nhiên, có những trường hợp chúng ta cần xóa nó một cách thủ công. Ví dụ, trong một Website có khả năng đăng ký người dùng và đăng nhập, chúng ta cần phải xóa cookie khi người dùng thoát ra.

Mặc dù hàm **setcookie()** có thể nhận tới sáu tham số nhưng chỉ có một tham số là bắt buộc: tên của cookie. Nếu gửi một cookie chỉ có phần tên mà không có giá trị, nó sẽ có tác dụng như xóa cookie hiện có với cùng tên. Ví dụ, để tạo cookie first_name, chúng ta sử dụng dòng lệnh:

```
setcookie('first_name', 'Larry');
```

Để xóa cookie first_name, chúng ta sử dụng dòng lệnh:

```
setcookie('first_name');
```

Cũng có thể xóa cookie bằng cách thiết lập thời hạn hết hạn là một thời điểm trong quá khứ:

```
setcookie('first_name','',time() - 300);
```

Để minh họa, chúng ta sẽ bổ sung chức năng thoát khỏi site. Chức năng này chỉ có hiệu lực với những ai đã đăng nhập.

Thực hành xóa một cookie theo các bước sau:

1. Tạo ra một hồ sơ PHP mới trong trình soạn thảo văn bản.

   ```
   <?php # Script 7.4 - logout.php
   ```

2. Kiểm tra sự tồn tại của cookie first_name. Nếu nó tồn tại, xóa toàn bộ cookie.

```
if (!isset($_COOKIE['first_name'])) {
    header ("Location:  http://" . $_SERVER['HTTP_HOST']
    → dirname($_SERVER['PHP_SELF']) . "/index.php");
    exit();
} else {
    setcookie ('first_name', '', time()-300, '/', '', 0);
    setcookie ('user_id', '', time()-300, '/', '', 0);
}
```

Cũng như với trang **loggedin.php**, nếu người dùng chưa đăng nhập, trang này sẽ chuyển họ đến trang chủ. Nếu họ đã đăng nhập, mã kịch bản sẽ xóa hai cookie đã có.

3. Thực hiện phần còn lại của trang PHP.

```
$page_title = 'Thoat ra!';
include ('templates/header.inc');
echo "<p>Ban da thoat ra.</p>";
include ('templates/footer.inc');
?>
```

4. Lưu tập tin với tên gọi **logout.php**. Mã lệnh đầy đủ của tập tin này được thể hiện trong đoạn mã 7.4.

Đoạn mã 7.4: *Mã kịch bản **logout.php** sẽ xóa các cookie đã được thiết lập trước đây.*

```
<?php # Doan ma 7.4 - logout.php
if (!isset($_COOKIE['first_name'])) {
    header ("Location:  http://" . $_SERVER['HTTP_HOST']
    → dirname($_SERVER['PHP_SELF']) . "/index.php");
    exit();
} else {
    setcookie ('first_name', '', time()-300, '/', '', 0);
    setcookie ('user_id', '', time()-300, '/', '', 0);
}
$page_title = 'Thoat ra!';
```

```
include ('templates/header.inc');
echo "<p>Ban da thoat ra.</p>";
include ('templates/footer.inc');
?>
```

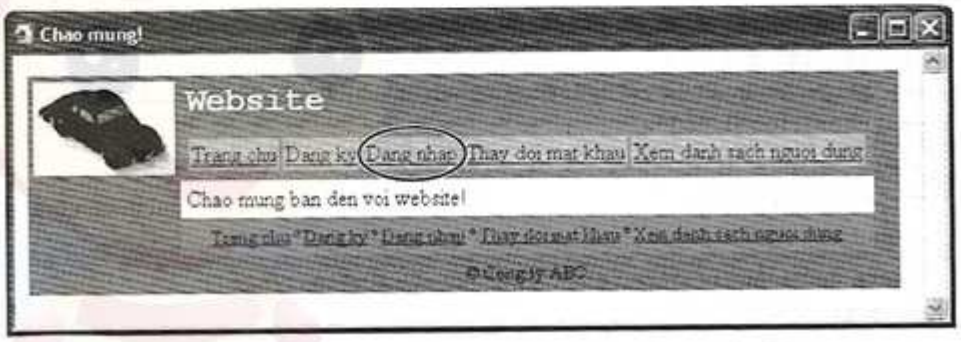
Thực hành tạo liên kết thoát khỏi site theo các bước sau:

1. Mở tập tin **header.inc** (tham khảo lại đoạn mã 6.1 ở Chương 6) trong trình soạn thảo.
2. Thay đổi liên kết đăng nhập thành:

```
<td width="20%" align="center" bgcolor="#FFCC66">
<?php if (isset($_COOKIE['user_id']) AND
→ (substr($_SERVER['PHP_SELF'], -10) != 'logout.php')) {
→ echo '<a href="logout.php">Thoat ra</a>';
→} else {
→ echo '<a href="login.php">Dang nhap</a>';
→}?>
</td>
```

Thay vì giữ cố định liên kết đăng nhập trong khuôn mẫu, chúng ta sẽ hiển thị liên kết thoát ra (Logout) nếu người dùng đã đăng nhập, hoặc hiển thị liên kết đăng nhập (Login) nếu họ chưa đăng nhập. Điều này sẽ được thực hiện dựa trên sự hiện diện của cookie. Vì mã kịch bản **logout.php** sẽ chỉ hiển thị liên kết thoát ra (cookie đã tồn tại khi trang lần đầu tiên được truy xuất), chúng ta phải bổ sung điều kiện để kiểm tra xem trang hiện tại có phải là **logout.php** không.

3. Lặp lại quá trình cho tập tin **footer.inc** (so sánh đoạn mã 6.2 và 7.6).
4. Lưu cả hai tập tin, tải chúng lên máy chủ Web và thử quá trình đăng nhập hoặc thoát ra trên trình duyệt (xem hình 7-9, 7-10 và 7-11).



Hình 7-9: *Trang chủ với liên kết đăng nhập (Login).*

```
<td bgcolor="#999966"> </td>
<td bgcolor="#FFFFFF">
```

Đoạn mã 7.6: *Tập tin* ***footer.inc*** *sẽ hiển thị liên kết đăng nhập hoặc thoát ra tùy theo trạng thái hiện tại của người dùng.*

```
<!-- Doan ma 7.6 - footer.inc -->
</td>
<td width="10" bgcolor="#999966"> </td>
</tr>
<tr>
<td bgcolor="#999966"> </td>
<td bgcolor="#CC9933"><div align="center"><small>
→ <a href="index.php">Trang chu</a> &ordm;
→ <a href="register.php">Dang ky</a> &ordm;
<?php
if (isset($_COOKIE['user_id']) AND
→ (substr($_SERVER['PHP_SELF'], -10) != 'logout.php')) {
   echo '<a href="logout.php">Thoat ra</a>';
} else {
   echo '<a href="login.php">Dang nhap</a>';
}
?>
&ordm; <a href="change_password.php">Thay doi mat khau</a>
→ &ordm; <a href="view_users.php">Xem danh sach nguoi dung
→ </a></small></div></td>
<td width="10" bgcolor="#999966"> </td>
</tr>
<tr>
<td bgcolor="#999966"> </td>
<td bgcolor="#999966">
<div align="center"><font size="-1">
→ © Cong ty ABC.</font></div></td>
<td bgcolor="#999966"> </td>
</tr>
</table>
</body>
</html>
```

```
<body>
<!-- Doan ma 7.5 - header.inc -->
<table border="0" cellspacing="0" cellpadding="4">
<tr>
<td rowspan="2" bgcolor="#999966"><img src="images/logo.jpg"
→ alt="trout" width="100" height="47" /></td>
<td width="*" bgcolor="#999966"><font color="#FFFFFF"
→ size="+2" face="Courier New, Courier, mono">
→ <strong>Website</strong></font></td>
<td width="10" rowspan="2" bgcolor="#999966"> </td>
</tr>
<tr>
<td bgcolor="#CC9933">
<table width="100%" border="0" cellspacing="2"
→ cellpadding="2">
<tr>
<td width="20%" align="center" bgcolor="#FFCC66">
→ <a href="index.php">Trang chu</a></td>
<td width="20%" align="center" bgcolor="#FFCC66">
→ <a href="register.php">Dang ky</a></td>
<td width="20%" align="center" bgcolor="#FFCC66"><?php
→ if (isset($_COOKIE['user_id']) AND
→ (substr($_SERVER['PHP_SELF'], -10) != 'logout.php')) {
→ echo '<a href="logout.php">Thoat ra</a>';
→ } else {
→ echo '<a href="login.php">Dang nhap</a>';
→ }?>
</td>
<td width="20%" align="center" bgcolor="#FFCC66"
→ nowrap="nowrap"><a href="change_password.php">Thay doi mat
→ khau</a></td>
<td width="20%" align="center" bgcolor="#FFCC66"
→ nowrap="nowrap"><a href="view_users.php">Xem danh sach
→ nguoi dung</a></td>
</tr>
</table>
</td>
</tr>
<tr>
```

```
<td bgcolor="#999966"> </td>
<td bgcolor="#FFFFFF">
```

Đoạn mã 7.6: *Tập tin* ***footer.inc*** *sẽ hiển thị liên kết đăng nhập hoặc thoát ra tùy theo trạng thái hiện tại của người dùng.*

```
<!-- Doan ma 7.6 - footer.inc -->
</td>
<td width="10" bgcolor="#999966"> </td>
</tr>
<tr>
<td bgcolor="#999966"> </td>
<td bgcolor="#CC9933"><div align="center"><small>
→ <a href="index.php">Trang chu</a> &ordm;
→ <a href="register.php">Dang ky</a> &ordm;
<?php
if (isset($_COOKIE['user_id']) AND
→ (substr($_SERVER['PHP_SELF'], -10) != 'logout.php')) {
   echo '<a href="logout.php">Thoat ra</a>';
} else {
   echo '<a href="login.php">Dang nhap</a>';
}
?>
&ordm; <a href="change_password.php">Thay doi mat khau</a>
→ &ordm; <a href="view_users.php">Xem danh sach nguoi dung
→ </a></small></div></td>
<td width="10" bgcolor="#999966"> </td>
</tr>
<tr>
<td bgcolor="#999966"> </td>
<td bgcolor="#999966">
<div align="center"><font size="-1">
→ © Cong ty ABC.</font></div></td>
<td bgcolor="#999966"> </td>
</tr>
</table>
</body>
</html>
```

Để xem kết quả của những lời gọi hàm `setcookie()` *trong mã kịch bản này, bạn nên thiết lập chế độ xác nhận cookie trong trình duyệt.*

Để xóa một cookie, bạn phải sử dụng cùng các tham số như khi thiết lập cookie.

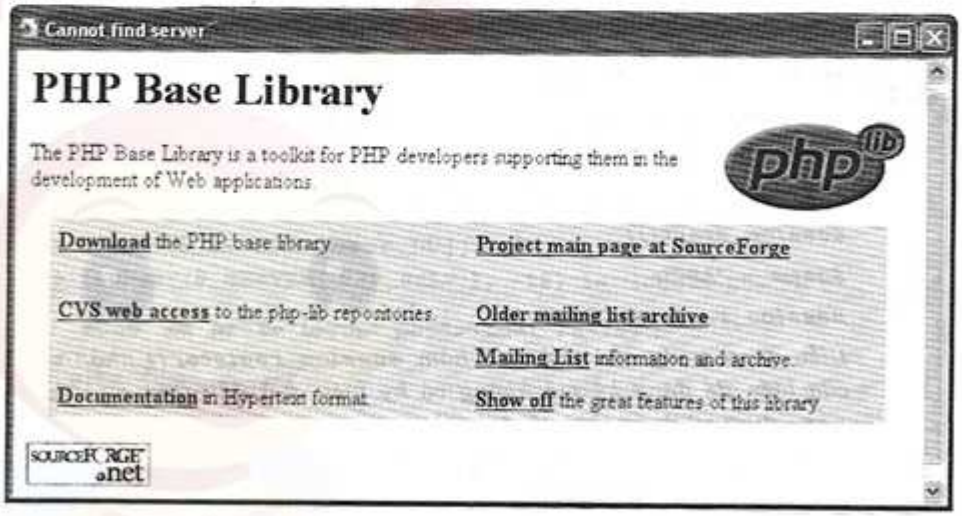
Việc xóa một cookie sẽ không có hiệu lực chừng nào trang chưa được tải lại hoặc một trang khác được truy xuất (nói cách khác, cookie sẽ vẫn tồn tại sau khi mã kịch bản trong trang đã xóa nó). Đó là lý do tại sao chúng ta cần bổ sung mệnh đề `AND (substr($_SERVER['PHP_SELF'], -10) != 'logout.php')` *vào điều kiện trong các tập tin* `header.inc` *và* `footer.inc`.

Sử dụng Session

Một trong những bổ sung hữu dụng cho PHP là kể từ phiên bản 4 nó đã có hỗ trợ session. Nói ngắn gọn, session là dữ liệu được chứa trên máy chủ, không phải trên trình duyệt và định danh session (session identifier) được dùng để xác định mẩu tin người dùng (dữ liệu session).

Trước PHP 4, nếu muốn tạo session, nhà lập trình phải viết toàn bộ thư viện mã lệnh để tạo ra định danh session duy nhất, chứa thông tin trong cơ sở dữ liệu hoặc tập tin văn bản, gửi và đọc cookie, lấy ra dữ liệu đã được lưu giữ...

Mặc dù PHP Base Library (xem hình 7-12) đã đơn giản quá trình này ở một mức độ nhất định, sự hỗ trợ session của phiên bản PHP 4 mới thực sự là sự bổ sung tuyệt vời.



Hình 7-12: *PHP Base Library hay ngắn gọn là PHPLib, có tại địa chỉ http://phplib.sourceforge.net, cho phép các nhà phát triển sử dụng session với các phiên bản PHP trước 4.0.*

Bạn có thể đặt câu hỏi: Tại sao phải sử dụng session trong khi cookie cũng làm việc tốt? Trước hết, session an toàn hơn ở chỗ toàn bộ thông tin được chứa trên máy chủ và không phải gửi liên tục qua lại giữa máy chủ và máy khách hàng. Thứ hai, một số người dùng không chấp nhận cookie hoặc họ hoàn toàn bỏ đặc điểm cookie. Mặc dù session được thiết kế để làm việc với cookie, nhưng nó có thể hoạt động mà không cần đến chúng, như bạn sẽ thấy trong phần tiếp theo của chương này.

Để minh họa session và để so sánh nó với cookie, chúng ta sẽ viết lại các mã kịch bản trước đây.

Thiết lập biến session

Quy tắc quan trọng nhất với session là mỗi trang sử dụng chúng phải bắt đầu với hàm **session_start()**. Hàm này thông báo cho PHP biết để bắt đầu một session mới hoặc sử dụng một session đã có.

Lần đầu tiên được gọi, hàm **session_start()** sẽ gửi một cookie với tên PHPSESSID (tên session) và giá trị đại loại như a61f8670baa8d90a30c875df98a207f4 (32 chữ cái hệ thập lục phân - định danh của session). Vì gửi một cookie, nên hàm **session_start()** phải được gọi trước khi gửi dữ liệu bất kỳ cho trình duyệt (giống như khi sử dụng các hàm **setcookie()** và **header()**).

Sau khi session đã được khởi tạo, các giá trị có thể được đăng ký với session:

```
$_SESSION['khóa'] = 'giá trị';
```

```
$_SESSION['name'] = 'Jessica';
```

Chúng ta sẽ viết lại mã kịch bản **login.php** với các thông tin này.

Sử dụng session_register()

*Trước PHP 4.1 (khi biến siêu toàn cục **$_SESSION** có hiệu lực), các biến session thường được thiết lập bằng cách sử dụng hàm đặc biệt **session_register()**. Cú pháp là:*

```
session_start();
$name = 'ABC';
session_register('name');
```

*Điều quan trọng cần chú ý là hàm **session_register()** nhận vào tên của biến để đăng ký mà không có ký hiệu **$** ở đầu (trong ví dụ trên, **name** được sử dụng thay vì **$name**).*

*Để viết ví dụ mã kịch bản đăng nhập với phương pháp này, bạn cần thay thế hai dòng **$_SESSION** với:*

```
$first_name = $row[1];
$user_id = $row[0];
session_register['first_name'];
```

```
session_register['user_id'];
```

Thực hành bắt đầu một session theo các bước sau:

1. Mở tập tin **login.php** (tham khảo lại đoạn mã 7.3) trong trình soạn thảo văn bản.
2. Thay thế các dòng **setcookies()** với các dòng sau:

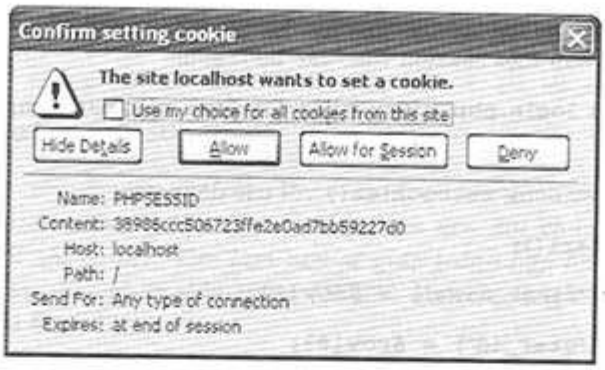
```
session_start();

$_SESSION['first_name'] = $row[1];

$_SESSION['user_id'] = $row[0];
```

Bước đầu tiên là bắt đầu một session. Vì không có lệnh **echo**, các lời gọi hàm hoặc HTML nào phía trước trong mã kịch bản, nên chúng ta có thể sử dụng **session_start()** một cách an toàn. Sau đó, ta bổ sung hai cặp *khóa-giá trị* cho mảng siêu toàn cục **$_SESSION** để đăng ký tên và ID của người dùng với session.

3. Lưu tập tin, tải nó lên máy chủ và chạy thử trong trình duyệt (xem hình 7-13). Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 7.7.

Hình 7-13: *Biểu mẫu đăng nhập giữ nguyên không thay đổi với người dùng, nhưng tính năng ẩn phía trong đã được thay đổi để sử dụng session.*

Mặc dù các mã kịch bản **loggedin.php**, phần đầu và cuối trang cũng cần phải viết lại. Tuy nhiên, bạn vẫn có thể thử mã kịch bản đăng nhập và xem cookie được tạo (xem hình 7-14).



Hình 7-14: *Cookie được tạo bởi hàm* ***session_start()*** *của PHP để chứa định danh session.*

Đoạn mã 7.7: *Mã kịch bản* ***login.php*** *giờ đây sử dụng session thay cho cookie.*

```
<?php # Doan ma 7.7 - login.php
if (isset($_POST['submit'])) {
   require_once ('../mysql_connect.php');
   function escape_data ($data) {
      global $dbc;
      if (ini_get('magic_quotes_gpc')) {
         $data = stripslashes($data);
      }
      return mysql_real_escape_string($data, $dbc);
   }
   $message = NULL;
   if (empty($_POST['username'])) {
      $u = FALSE;
      $message .= '<p>Ban chua nhap vao ten dang nhap!</p>';
   } else {
      $u = escape_data($_POST['username']);
   }
   if (empty($_POST['password'])) {
      $p = FALSE;
      $message .= '<p>Ban chua nhap vao mat khau!</p>';
   } else {
      $p = escape_data($_POST['password']);
   }
```

```
   if ($u && $p) {
      $query = "SELECT user_id, first_name FROM users WHERE
      → username='$u' AND password=PASSWORD('$p')";
      $result = @mysql_query ($query);
      $row = mysql_fetch_array ($result, MYSQL_NUM);
      if ($row) {
         session_start();
         $_SESSION['first_name'] = $row[1];
         $_SESSION['user_id'] = $row[0];
         header ("Location:  http://" . $_SERVER['HTTP_HOST'] .
         → dirname($_SERVER['PHP_SELF']) . "/loggedin.php");
         exit();
      } else {
         $message = '<p>Ten dang nhap va mat khau khong
         → phu hop voi du lieu trong CSDL.</p>';
      }
      mysql_close();
   } else {
      $message .= '<p>Hay thu lai.</p>';
   }
}
$page_title = 'Login';
include ('templates/header.inc');
if (isset($message)) {
   echo '<font color="red">', $message, '</font>';
}
?>
<form action="<?php echo $_SERVER['PHP_SELF']; ?>"
→ method="post">
<fieldset><legend>Nhap vao thong tin cua ban:</legend>
<p><b>Ten dang nhap:</b> <input type="text" name="username"
→ size="10" maxlength="20" value="<?php if
→ (isset($_POST['username'])) echo $_POST['username']; ?>"
→ /></p>
<p><b>Mat khau:</b> <input type="password" name="password"
→ size="20" maxlength="20" /></p>
<div align="center"><input type="submit" name="submit"
→ value="Dang nhap" /></div>
```

```
</fieldset>
</form>
<?php
include ('templates/footer.inc');
?>
```

Vì session thường gửi và đọc cookie, bạn nên đặt chúng ở đầu mã kịch bản mỗi khi có thể. Việc làm này sẽ giúp tránh gặp phải các vấn đề gửi cookie sau khi phần đầu trang (HTML hoặc khoảng trắng) đã được gửi (xem lại hình 7-1).

Nếu muốn, bạn có thể thiết lập ***session.auto_start()*** *trong tập tin* ***php.ini*** *thành 1. Việc làm này giúp khỏi phải đưa vào dòng lệnh* ***session_start()*** *trên mỗi trang PHP.*

Có thể chứa các đối tượng hoặc mảng trong session, giống như các chuỗi hoặc số. Tuy nhiên, để chứa đối tượng, các trang có tham chiếu tới đối tượng đó phải truy xuất được trang định nghĩa lớp đối tượng đó.

Truy xuất biến session

Sau khi session đã được khởi tạo và các biến đã được đăng ký, chúng ta có thể viết các mã kịch bản khác để truy xuất các biến này. Muốn làm như vậy, mã kịch bản trước hết phải tham chiếu đến session bằng cách gọi hàm **session_start()**.

Việc này sẽ cho phép mã kịch bản hiện tại truy xuất đến session đã được khởi tạo trước đó (nếu nó có thể truy xuất giá trị PHPSESSID chứa trong cookie) hoặc tạo ra một session mới nếu nó không thể truy xuất được giá trị này (trong trường hợp này, ta không thể truy xuất giá trị đã được lưu giữ).

Sau đó, để tham chiếu đến một biến session, bạn có thể sử dụng **$_SESSION['tên biến']** như với bất kỳ mảng nào khác.

Thực hành truy xuất các biến session theo các bước sau:

1. Mở tập tin **loggedin.php** (tham khảo lại đoạn mã 7.2) trong trình soạn thảo văn bản.
2. Đưa vào lời gọi hàm **session_start()**.

```
session_start();
```

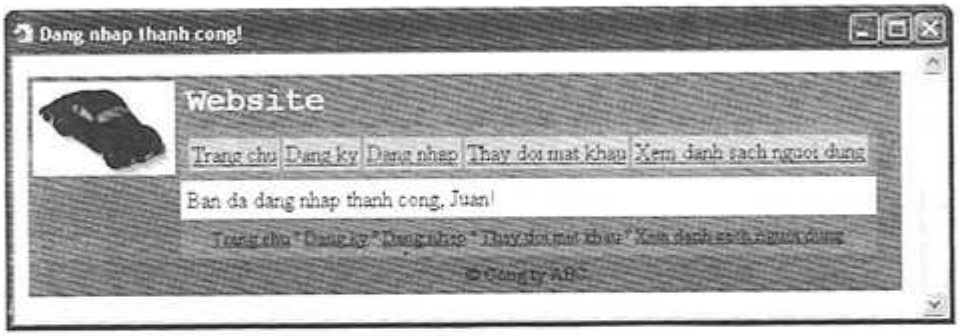
Mỗi mã kịch bản PHP có thiết lập hoặc lấy ra giá trị của các biến session đều phải gọi hàm **session_start()**. Dòng lệnh này phải được gọi trước khi tập tin **header.inc** được đưa vào.

3. Thay thế các tham chiếu đến **$_COOKIE** bằng **$_SESSION**.

```
if (!isset($_SESSION['first_name'])) {

   header ("Location:  http://" . $_SERVER['HTTP_HOST'] .
   → dirname($_SERVER['PHP_SELF']) . "/index.php");

   exit();

}
```

Để chuyển mã kịch bản từ sử dụng cookie sang session, bạn chỉ cần chuyển `$_COOKIE` thành `$_SESSION`. Cũng có thể sử dụng `$HTTP_SESSION_VARS` hoặc đơn giản là `$first_name` nếu thông số `register_globals` đã được kích hoạt. Tuy nhiên, tùy chọn cuối cùng không được tốt lắm.

4. Lưu tập tin, tải nó lên máy chủ và thử trong trình duyệt Web (xem hình 7-15). Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 7.8.



Hình 7-15: *Sau khi đăng nhập, người dùng sẽ được chuyển đến trang* ***loggedin.php****. Trang này sẽ thể hiện câu chào mừng với các thông tin chứa trong session.*

5. Thay thế các tham chiếu đến `$_COOKIE` bằng `$_SESSION` trong tập tin đầu trang (`header.inc`) - (chuyển từ đoạn mã 7.5 thành 7.9).

```
<td width="20%" align="center" bgcolor="#FFCC66"><?php

if (isset($_SESSION['user_id']) AND
→ (substr($_SERVER['PHP_SELF'], -10) != 'logout.php')) {

   echo '<a href="logout.php">Thoat ra</a>';

} else {

   echo '<a href="login.php">Dang nhap</a>';

}

?></td>
```

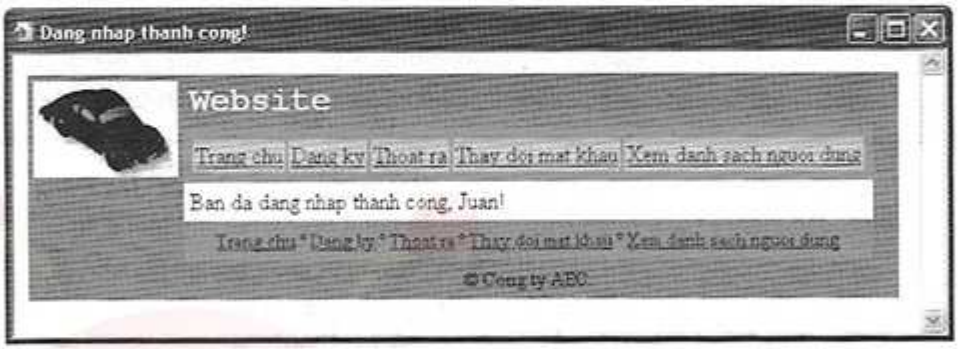
Để các liên kết đăng nhập hoặc thoát ra hoạt động đúng (chú ý đến liên kết sai trong hình 7-15), tham chiếu đến các biến cookie trong tập tin đầu và cuối trang

phải được chuyển sang để tham chiếu đến session. Không trang nào trong hai trang này sử dụng **session_start()** vì cả hai đều được đưa vào trong các trang đã gọi hàm này.

6. Lặp lại cùng quá trình (chuyển **$_COOKIE** thành **$_SESSION**) cho tập tin **footer.inc** (từ đoạn mã 7.6 thành 7.10).

```
<?php

if (isset($_SESSION['user_id']) AND
→ (substr($_SERVER['PHP_SELF'], -10) != 'logout.php')) {

   echo '<a href="logout.php">Thoat ra</a>';

} else {

   echo '<a href="login.php">Dang nhap</a>';

}

?>
```

7. Lưu lại hai tập tin này, tải chúng lên máy chủ và chạy thử trong trình duyệt Web (xem hình 7-16).



Hình 7-16: *Với các tập tin đầu và cuối trang đã được chỉnh sửa để sử dụng session, các liên kết đăng nhập hoặc thoát ra sẽ được hiển thị một cách thích hợp (so sánh với hình 7-15).*

Đoạn mã 7.8: *Cập nhật tập tin* ***loggedin.php*** *để nó tham chiếu đến biến* ***$_SESSION*** *thay cho* ***$_COOKIE***

```
<?php # Doan ma 7.8 - loggedin.php
session_start();
if (!isset($_SESSION['first_name'])){
   header ("Location:  http://" .
   → $_SERVER['HTTP_HOST'] . dirname($_SERVER['PHP_SELF']) .
   → "/index.php");
   exit();
```

```
}
$page_title = 'Dang nhap thanh cong!';
include ('templates/header.inc');
echo "<p>Ban da dang nhap thanh cong,
→ {$_SESSION['first_name']}!</p>";
include ('templates/footer.inc');
?>
```

Đoạn mã 7.9: *Tập tin* ***header.inc*** *giờ đây cũng tham chiếu đến* ***$_SESSION***

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
→ Transitional//EN"
→ "http://www.w3.org/TR/2000/REC-xhtml1-20000126/DTD/xhtml1-
→transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">
<head>
<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />
<title><?php echo $page_title; ?></title>
</head>
<body>
<!-- Doan ma 7.9 - header.inc -->
<table border="0" cellspacing="0" cellpadding="4">
<tr>
<td rowspan="2" bgcolor="#999966"><img src="images/logo.jpg"
→ alt="LOGO" width="100" height="47" /></td>
<td width="*" bgcolor="#999966"><font color="#FFFFFF"
→ size="+2" face="Courier New, Courier, mono">
→<strong>Website</strong></font></td>
<td width="10" rowspan="2" bgcolor="#999966"> </td>
</tr>
<tr>
<td bgcolor="#CC9933">
<table width="100%" border="0" cellspacing="2"
→ cellpadding="2">
<tr>
<td width="20%" align="center" bgcolor="#FFCC66">
→<a href="index.php">Trang chu</a></td>
```

```
<td width="20%" align="center" bgcolor="#FFCC66">
→<a href="register.php">Dang ky</a></td>
<td width="20%" align="center" bgcolor="#FFCC66">
<?php
if (isset($_SESSION['user_id']) AND
→ (substr($_SERVER['PHP_SELF'], -10) != 'logout.php')) {
   echo '<a href="logout.php">Thoat ra</a>';
} else {
   echo '<a href="login.php">Dang nhap</a>';
}
?>
</td>
<td width="20%" align="center" bgcolor="#FFCC66"
→ nowrap="nowrap"><a href="change_password.php">Thay doi mat
→ khau</a></td>
<td width="20%" align="center" bgcolor="#FFCC66"
→ nowrap="nowrap"><a href="view_users.php">Xem danh sach
→ nguoi dung</a></td>
</tr>
</table>
</td>
</tr>
<tr>
<td bgcolor="#999966"> </td>
<td bgcolor="#FFFFFF">
```

Đoạn mã 7.10: *Trang footer.inc đã được chỉnh sửa.*

```
<!-- Doan ma 7.10 - footer.inc -->
</td>
<td width="10" bgcolor="#999966"> </td>
</tr>
<tr>
<td bgcolor="#999966"> </td>
<td bgcolor="#CC9933"><div align="center"><small>
→<a href="index.php">Trang chu</a> &ordm;
→<a href="register.php">Dang ky</a> &ordm;
<?php
if (isset($_SESSION['user_id']) AND
```

```
→ (substr($_SERVER['PHP_SELF'], -10) != 'logout.php')) {
   echo '<a href="logout.php">Thoat ra</a>';
} else {
   echo '<a href="login.php">Dang nhap</a>';
}
?>
&ordm; <a href="change_password.php">Thay doi mat khau</a>
→ &ordm; <a href="view_users.php">Xem danh sach nguoi dung
→ </a></small></div></td>
<td width="10" bgcolor="#999966"> </td>
</tr>
<tr>
<td bgcolor="#999966"> </td>
<td bgcolor="#999966">
<div align="center"><font size="-1">
→© Cong ty ABC.</font></div></td>
<td bgcolor="#999966"> </td>
</tr>
</table>
</body>
</html>
```

*Để các liên kết đăng nhập hoặc thoát ra làm việc với các trang khác (**register.php**, **index.php**, **view_users.php** và **change_password.php**), bạn cần bổ sung lời gọi hàm **session_start()** vào đầu các mã kịch bản này.*

Xóa các biến session

Khi sử dụng session, đặc biệt với một hệ thống đăng nhập hoặc thoát ra như đang đề cập ở đây, chúng ta cần phải tạo ra một cách thức để xóa các biến session. Trong ví dụ hiện tại, điều này sẽ được thực hiện khi người dùng thoát ra.

Trong khi hệ thống sử dụng cookie chỉ cần gửi một cookie khác để xóa cookie hiện có, thì với hệ thống sử dụng session, chúng ta phải quan tâm đến cả cookie trên máy khách hàng và session trên máy chủ.

Để xóa một biến session nào đó, chúng ta sử dụng hàm **unset()**:

```
unset($_SESSION['var']);
```

Để xóa tất cả các biến session, chúng ta thiết lập lại toàn bộ mảng **$_SESSION**:

```
$_SESSION = array();
```

Cuối cùng, để loại bỏ tất cả dữ liệu session trên máy chủ, chúng ta sử dụng hàm **session_destroy()**:

```
session_destroy();
```

Chú ý, trước khi sử dụng một trong các phương pháp này, trang phải được bắt đầu với **session_start()** để có thể truy xuất được session hiện có.

Thực hành xóa một session theo các bước sau:

1. Tạo ra một mã kịch bản PHP mới trong trình soạn thảo văn bản.

```
<?php # Doan ma 7.11 - logout.php
```

Mã kịch bản **logout.php** sẽ kết thúc phiên truy cập của người dùng và xóa tất cả thông tin liên quan đến phiên truy cập đó.

2. Gọi hàm **session_start()**.

```
session_start();
```

Mỗi khi sử dụng session, bạn phải sử dụng hàm **session_start()**. Thông thường, chúng ta đặt lời gọi hàm này ở đầu mã kịch bản. Điều này cũng áp dụng khi muốn xóa session.

3. Kiểm tra sự hiện diện của biến first_name.

```
if (!isset($_SESSION['first_name'])) {
    header ("Location:  http://" . $_SERVER['HTTP_HOST'] .
    → dirname($_SERVER['PHP_SELF']) . "/index.php");
    exit();
```

Cũng giống như mã kịch bản **logout.php** trong ví dụ sử dụng cookie, nếu người dùng không còn ở trạng thái đăng nhập, chúng ta sẽ chuyển họ sang trang chủ.

4. Xóa tất cả thông tin liên quan đến session.

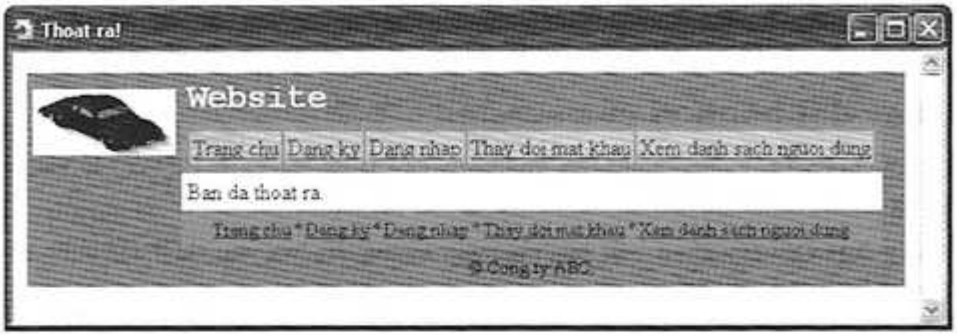
```
} else {
    $_SESSION = array();
    session_destroy();
    setcookie ('PHPSESSID', '', time()-300, '/', '', 0);
}
```

Dòng lệnh thứ hai trên đây sẽ thiết lập lại biến **$_SESSION** thành một mảng mới, xóa toàn bộ giá trị hiện có của nó. Dòng lệnh thứ ba sẽ loại bỏ dữ liệu khỏi máy chủ. Dòng lệnh thứ tư gửi một cookie để thay thế cho cookie chứa thông tin session hiện có trong trình duyệt.

5. Tạo trang HTML và in thông báo.

```
$page_title = 'Thoat ra!';
include ('templates/header.inc');
echo "<p>Ban da thoat ra.</p>";
include ('templates/footer.inc');
?>
```

6. Lưu tập tin với tên gọi **logout.php**, tải nó lên máy chủ và thử trong trình duyệt (xem hình 7-17). Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 7.11.



Hình 7-17: *Trang thoát ra.*

Đoạn mã 7.11: *Xóa một session đòi hỏi một cú pháp đặc biệt.*

```
<?php # Doan ma 7.11 - logout.php
session_start();
if (!isset($_SESSION['first_name'])) {
   header ("Location:  http://" . $_SERVER['HTTP_HOST'] .
   → dirname($_SERVER['PHP_SELF']) . "/index.php");
   exit();
} else {
   $_SESSION = array();
   session_destroy();
   setcookie ('PHPSESSID', '', time()-300, '/', '', 0);
}
$page_title = 'Thoat ra!';
include ('templates/header.inc');
echo "<p>Ban da thoat ra.</p>";
include ('templates/footer.inc');
?>
```

Nếu sử dụng một phiên bản PHP cũ và mảng $_SESSION không có hiệu lực, bạn nên dùng hàm session_unset() thay cho dòng $_SESSION = array().

Không thiết lập $_SESSION thành NULL vì điều đó sẽ gây ra trục trặc trên một số máy chủ.

Để xóa một biến session, bạn nên dùng unset($_SESSION['biến']). Trong các phiên bản cũ của PHP, sử dụng session_unregister('biến').

Để kiểm tra xem một biến nào đó có được thiết lập hay chưa, bạn có thể dùng isset($_SESSION['biến']). Trong các phiên bản cũ của PHP, sử dụng session_is_registered('biến').

Thay đổi hành vi của session

Có khoảng 20 tùy chọn cấu hình khác nhau để thiết lập cách PHP xử lý session. Bảng 7.1 liệt kê các tùy chọn quan trọng nhất trong số này.

Bảng 7.1: *Các tùy chọn cấu hình session của PHP, với các thiết lập mặc định được nêu trong cột ví dụ.*

Thiết lập	Ví dụ	Ý nghĩa
`session.auto_start`	`0`	Session có được sử dụng tự động hay không (0 có nghĩa là không).
`session.cookie_domain`	`www.domain.com`	URL mà từ đó có thể truy xuất được cookie gắn với session.
`session.cookie_lifetime`	`0`	Thời gian (tính theo giây) tồn tại của cookie gắn với session. 0 có nghĩa là tồn tại theo thời gian sử dụng trình duyệt.
`session.cookie_path`	`/`	Đường dẫn tên miền mà từ đó cookie có thể được truy xuất.
`session.cookie_secure`	`0`	Cookie phải được truyền trên một kết nối an toàn hay không an toàn (0 có nghĩa là không an toàn).
`session.gc_probability`	`1`	Thông số quy định thu dọn, từ 1 đến 100.
`session.gc_maxlifetime`	`1440`	Chu kỳ sống (tính theo giây) của một cookie.
`session.name`	`PHPSESSID`	Tên được gán cho mọi session.

session.save_handler	files	Cách lưu trữ dữ liệu session.
session.save_path	/tmp	Nơi lưu trữ dữ liệu session.
session.serialize_handler	php	Phương pháp được dùng để tổ chức các biến session.
session.use_cookies	1	Có chứa ID của session trong cookie hay không. 1 có nghĩa là có.
session.use_only_cookies	0	ID của session phải được chứa trong cookie hay là không. 0 có nghĩa là không bắt buộc.
session.use_trans_sid	0	ID của session sẽ được PHP đưa vào trong mỗi liên kết trong ứng dụng hay là không. 0 có nghĩa là không.

Để thiết lập tên của session (để tạo ra một dạng thân thiện hơn với người dùng), bạn có thể sử dụng hàm **ini_set()** hoặc đơn giản hơn là hàm **session_name()**.

```
session_name('MySession');
```

Có hai ưu điểm của việc tạo tên cho session: nó an toàn hơn và dễ được chấp nhận hơn bởi người dùng cuối (vì tên session chính là tên cookie mà người dùng sẽ thấy). Chú ý, để **session_name()** hoạt động, nó phải được gọi trước khi sử dụng hàm **session_start()** trong các ứng dụng.

Thực hành tạo tên cho session theo các bước sau:

1. Mở tập tin **login.php** (tham khảo lại đoạn mã 7.7) trong trình soạn thảo văn bản.
2. Trước lời gọi **session_start()**, bổ sung dòng lệnh sau:

```
session_name ('YourVisitID');
```

Thay vì để session có tên là PHPSESSID, chúng ta sử dụng một tên gọi thân thiện hơn (YourVisitID).

3. Lặp lại quá trình này cho tập tin **loggedin.php** (so sánh đoạn mã 7.8 với đoạn mã 7.13).

Vì mọi trang đều phải sử dụng cùng tên session, dòng lệnh này phải được thêm vào trong mã kịch bản **loggedin.php** để nó hoạt động đúng.

4. Bổ sung dòng lệnh sau cho tập tin **logout.php** (so sánh đoạn mã 7.11 với đoạn mã 7.14):

```
session_name ('YourVisitID');
```

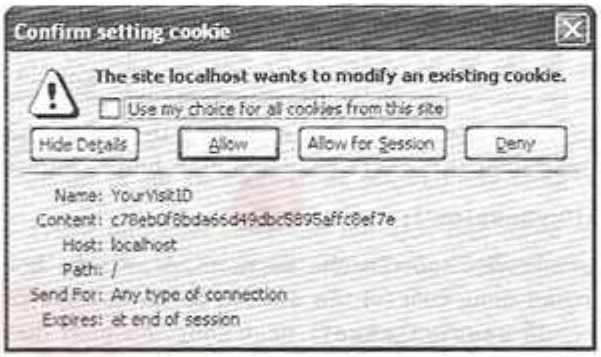
5. Thay đổi dòng **setcookie()** của tập tin **logout.php** để nó sử dụng hàm **session_name()**:

```
setcookie (session_name(), '', time()-300, '/', '', 0);
```

Hàm **session_name()** sẽ thiết lập tên session hoặc trả lại tên của session hiện tại (nếu không có tham số nào được truyền vào). Chúng ta muốn gửi cookie với cùng tên như khi tạo và hàm **session_name()** giúp thiết lập đúng giá trị này.

6. Lưu lại các tập tin, tải chúng lên máy chủ Web và thử trong trình duyệt. Mã lệnh đầy đủ của các tập tin **login.php**, **loggedin.php**, **logout.php** được thể hiện trong các đoạn mã 7.12, 7.13 và 7.14.

7. Nếu cần, hãy xem cookie được thiết lập trong quá trình đăng nhập (xem hình 7-18).



Hình 7-18: *Tên cookie sẽ ứng với tên của session.*

Đoạn mã 7.12: *Mã kịch bản* ***login.php*** *giờ đây sử dụng tên session do bạn quy định.*

```
<?php # Doan ma 7.12 - login.php
if (isset($_POST['submit'])) {
   require_once ('../mysql_connect.php');
   function escape_data ($data) {
      global $dbc;
      if (ini_get('magic_quotes_gpc')) {
         $data = stripslashes($data);
```

```
    }
    return mysql_real_escape_string($data, $dbc);
}
$message = NULL;
if (empty($_POST['username'])) {
    $u = FALSE;
    $message .= '<p>Ban chua nhap ten dang nhap!</p>';
} else {
    $u = escape_data($_POST['username']);
}
if (empty($_POST['password'])) {
    $p = FALSE;
    $message .= '<p>Ban chua nhap mat khau!</p>';
} else {
    $p = escape_data($_POST['password']);
}
if ($u && $p) {
    $query = "SELECT user_id, first_name FROM users WHERE
    -> username='$u' AND password=PASSWORD('$p')";
    $result = @mysql_query ($query);
    $row = mysql_fetch_array ($result, MYSQL_NUM);
    if ($row) {
        session_name ('YourVisitID');
        session_start();
        $_SESSION['first_name'] = $row[1];
        $_SESSION['user_id'] = $row[0];
        header ("Location:  http://" . $_SERVER['HTTP_HOST'] .
        -> dirname($_SERVER['PHP_SELF']) . "/loggedin.php");
        exit();
    } else {
        $message = '<p>Ten dang nhap va mat khau khong
        -> phu hop voi du lieu trong CSDL.</p>';
    }
    mysql_close();
} else {
    $message .= '<p>Hay thu lai.</p>';
```

```
    }
}
$page_title = 'Dang nhap';
include ('templates/header.inc');
if (isset($message)) {
    echo '<font color="red">', $message, '</font>';
}
?>
<form action="<?php echo $_SERVER['PHP_SELF']; ?>"
→ method="post">
<fieldset><legend>Nhap vao thong tin cua ban:</legend>
<p><b>Ten dang nhap:</b> <input type="text" name="username"
→ size="10" maxlength="20" value="<?php if
→ (isset($_POST['username'])) echo $_POST['username']; ?>"
→ /></p>
<p><b>Mat khau:</b> <input type="password" name="password"
→ size="20" maxlength="20" /></p>
<div align="center"><input type="submit" name="submit"
→ value="Dang nhap" /></div>
</fieldset>
</form>
<?php
include ('templates/footer.inc');
?>
```

Đoạn mã 7.13: *Tên của session phải được sử dụng nhất quán trong mọi tập tin chứa mã kịch bản.*

```
<?php # Doan ma 7.13 - loggedin.php
session_name ('YourVisitID');
session_start();
if (!isset($_SESSION['first_name'])) {
    header ("Location:  http://" . $_SERVER['HTTP_HOST'] .
    → dirname($_SERVER['PHP_SELF']) . "/index.php");
    exit();
}
$page_title = 'Dang nhap thanh cong!';
include ('templates/header.inc');
```

```
echo "<p>Ban da dang nhap thanh cong,
{$_SESSION['first_name']}!</p>";
include ('templates/footer.inc');
?>
```

Đoạn mã 7.14: *Trang* ***logout.php*** *sử dụng hàm* ***session_name()*** *để xác định tên của cookie sẽ được gửi.*

```
<?php # Doan ma 7.14 - logout.php
session_name ('YourVisitID');
session_start();
if (!isset($_SESSION['first_name'])) {
   header ("Location:  http://" . $_SERVER['HTTP_HOST'] .
   → dirname($_SERVER['PHP_SELF']) . "/index.php");
   exit();
} else {
   $_SESSION = array();
   session_destroy();
   setcookie (session_name(), '', time()-300, '/', '', 0);
}
$page_title = 'Thoat ra!';
include ('templates/header.inc');
echo "<p>Ban da thoat ra.</p>";
include ('templates/footer.inc');
?>
```

Nếu dùng chung một máy chủ với các tên miền khác, bạn nên thay đổi tham số ***save_path*** *từ /tmp (mọi người đều có thể truy xuất) sang một vị trí khác an toàn hơn.*

Hàm ***session_set_cookie_params()*** *cho phép thay đổi các thuộc tính khác nhau của cookie gắn với session: thời hạn hết hiệu lực, đường dẫn, tên miền. Nó sẽ được minh họa trong phần sau của chương này.*

Dọn dẹp

Với session, dọn dẹp là một quá trình xóa các tập tin session chứa trong thư mục /tmp trên phần lớn máy chủ. Việc tạo ra một hệ thống thoát ra để xóa các session là một việc lý tưởng, nhưng không có gì đảm bảo là người dùng

sẽ thoát ra theo đúng cách mà họ phải làm. Vì lý do này, PHP có đưa vào một quá trình dọn dẹp.

Mỗi khi hàm ***session_start()*** *được gọi, việc dọn dẹp của PHP sẽ được kích hoạt, kiểm tra thời gian sửa đổi sau cùng của mỗi session (một session được sửa đổi mỗi khi các biến trong nó được thiết lập hoặc lấy ra) và loại bỏ các session đã hết hạn. Hiệu năng của máy chủ có thể bị ảnh hưởng đáng kể trên các site có nhiều truy cập, vì thế bạn có thể cần phải điều chỉnh lại cách hoạt động của PHP.*

Hai thông số có liên quan đến việc dọn dẹp: ***session.gc_maxlifetime*** *và* ***session.gc_probability****. Thông số đầu cho biết sau khoảng bao nhiêu giây không được kích hoạt, một session sẽ được coi là nhàn rỗi và do đó sẽ bị xóa. Thông số thứ hai xác định xác xuất mà việc dọn dẹp được thực hiện với giá trị từ 1 đến 100. Với thiết lập mặc định, mỗi lần gọi hàm* ***session_start()****, quá trình dọn dẹp có 1% cơ hội được thực hiện. Nếu PHP khởi động quá trình dọn dẹp, bất kỳ session nào không được sử dụng trong phạm vi 1440 giây sẽ bị xóa.*

Với các thông tin này, bạn có thể thay đổi cách hoạt động của quá trình dọn dẹp để phù hợp hơn với ứng dụng của mình. Giá trị 24 phút thường là một khoảng thời gian hợp lý, tuy nhiên bạn có thể tăng xác xuất đến giá trị khoảng 30% để có một sự cân bằng tốt giữa tính vận hành và tính "bừa bộn".

Session và cookie

Trong các ví dụ trước, chúng ta đã thực hiện cùng các nhiệm vụ (đăng nhập và thoát ra) bằng cách sử dụng cookie và session. Rõ ràng, cả hai đều dễ dùng trong PHP, nhưng câu hỏi đặt ra là khi nào thì sử dụng cookie và khi nào sử dụng session.

Session có một số ưu điểm so với cookie như sau:

- Nói chung là an toàn hơn (vì dữ liệu được lưu trên máy chủ).
- Cho phép lưu trữ nhiều dữ liệu hơn.
- Có thể được sử dụng mà không cần dùng tới cookie.

Trong khi đó, cookie lại có một số ưu điểm so với session:

- Dễ lập trình.
- Không cần nhiều đến máy chủ.

Nói chung, để lưu và lấy ra chỉ một ít thông tin, bạn nên sử dụng cookie. Tuy nhiên, đối với phần lớn ứng dụng Web, hãy sử dụng session. Do session dựa trên cookie theo mặc định, chúng ta sẽ đề cập đến cách quản lý mối quan hệ này tốt hơn.

An toàn session

Việc lưu tên session trong một cookie được xem là phương pháp an toàn nhất trong sử dụng cookie. Các dạng khác được đề cập đến trong chương này (đưa tên session vào trong URL hoặc chứa nó trong các trường ẩn trên biểu mẫu) sẽ không an toàn vì dễ bị người có dụng ý xấu chiếm đoạt.

Một phương pháp ngăn ngừa việc chiếm đoạt là chứa giá trị HTTP_USER_AGENT (một tổ hợp của trình duyệt và hệ điều hành đang được sử dụng) trong một session và sau đó so sánh giá trị này với HTTP_USER_AGENT của người dùng hiện tại. Một cách tương tự, bạn có thể sử dụng địa chỉ IP của người dùng (có thể thay đổi trong ngày nhưng trong một phiên truy cập thì không).

Để sử dụng các session quan trọng, cần sử dụng cookie và truyền chúng trên một kết nối an toàn nếu có thể. Bạn có thể thiết lập PHP để chỉ sử dụng cookie bằng cách thiết lập **`session.use_only_cookies`** *thành 1.*

Trên máy chủ, bản thân dữ liệu session có thể được chứa trong một cơ sở dữ liệu thay vì một tập tin văn bản. Cách này an toàn hơn, tuy nhiên nó đòi hỏi phải lập trình nhiều hơn. Ngoài ra, bạn có thể thay đổi thư mục lưu dữ liệu session mặc định để những người dùng khác trên máy chủ không thể truy xuất được các tập tin session của mình.

Thay đổi các thiết lập cookie liên quan đến session

Như đã nói, cookie được gửi bởi hàm **`session_start()`** và sử dụng một số tham số mặc định: thời hạn hết hạn được thiết lập là **`0`** (nghĩa là cookie sẽ tồn tại khi trình duyệt còn mở), đường dẫn là **`'/'`** (cookie có hiệu lực trong thư mục hiện tại và tất cả các thư mục con của nó) và không có tên miền. Để thay đổi các thiết lập mặc định này, chúng ta sử dụng hàm **`session_set_cookie_params()`**:

```
session_set_cookie_params(thời hạn, 'đường dẫn', 'tên miền', an toàn);
```

Thiết lập thời hạn là giá trị duy nhất phải có. Tham số này được tính theo giây với thiết lập mặc định là **`0`**. Giá trị thời hạn không phải là số giây tính từ mốc 1-1-1970 (như trong trường hợp của hàm **`setcookie()`**), do đó chúng ta sẽ sử dụng 300 (5 phút) thay vì time() + 300 (5 phút kể từ thời điểm này).

Thực hành thay đổi thiết lập cookie của session theo các bước sau:

1. Mở tập tin **`login.php`** (tham khảo đoạn mã 7.12) trong trình soạn thảo văn bản.
2. Trước lời gọi hàm **`session_start()`**, bổ sung dòng lệnh sau:

```
session_set_cookie_params (900,'/folder/', →'www.yourdomain.com');
```

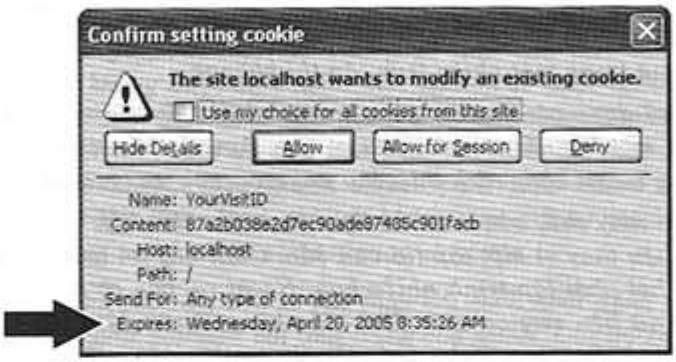
Hàm **`session_set_cookie_params()`** phải được gọi trước khi hàm **`session_start()`** có hiệu lực. Bạn có thể thay đổi đường dẫn và tên miền với các giá trị phù hợp với ứng dụng của mình hoặc bỏ qua chúng để sử dụng giá trị

mặc định. Trong ví dụ này, chúng ta thiết lập thời gian hết hạn của cookie là 15 phút kể từ thời điểm hiện tại.

3. Lưu tập tin với tên **login.php**, tải nó lên máy chủ và chạy thử trong trình duyệt Web. Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 7.15.

Sau 15 phút, cookie sẽ hết hiệu lực và các mã kịch bản PHP sẽ không thể truy xuất các giá trị của session (first_name và user_id).

4. Xem cookie được gửi xuống trình duyệt (xem hình 7-19).



Hình 7-19: *Cookie của session giờ đây đã có thời hạn hết hiệu lực.*

5. Nếu cần, hãy thay đổi tập tin **logout.php** để hàm **setcookie()** sử dụng cùng các tham số như trong **login.php**.

Đoạn mã 7.15: *Phiên bản này của mã kịch bản đăng nhập sẽ thiết lập cụ thể các tham số cookie.*

```
<?php # Doan ma 7.15 - login.php
if (isset($_POST['submit'])) {
   require_once ('../mysql_connect.php');
   function escape_data ($data) {
      global $dbc;
      if (ini_get('magic_quotes_gpc')) {
         $data = stripslashes($data);
      }
      return mysql_real_escape_string($data, $dbc);
   }
   $message = NULL;
   if (empty($_POST['username'])) {
      $u = FALSE;
```

```
        $message .= '<p>Ban chua nhap ten dang nhap!</p>';
    } else {
        $u = escape_data($_POST['username']);
    }
    if (empty($_POST['password'])) {
        $p = FALSE;
        $message .= '<p>Ban chua nhap mat khau!</p>';
    } else {
        $p = escape_data($_POST['password']);
    }
    if ($u && $p) {
        $query = "SELECT user_id, first_name FROM users WHERE
        → username='$u' AND password=PASSWORD('$p')";
        $result = @mysql_query ($query);
        $row = mysql_fetch_array ($result, MYSQL_NUM);
        if ($row) {
            session_name ('YourVisitID');
            session_set_cookie_params (900, '/folder/',
            → 'www.yourdomain.com');
            session_start();
            $_SESSION['first_name'] = $row[1];
            $_SESSION['user_id'] = $row[0];
            header ("Location:  http://" . $_SERVER['HTTP_HOST'] .
            → dirname($_SERVER['PHP_SELF']) . "/loggedin.php");
            exit();
        } else {
            $message = '<p>Ten dang nhap va mat khau khong
            → phu hop voi du lieu trong CSDL.</p>';
        }
        mysql_close();
    } else {
        $message .= '<p>Hay thu lai.</p>';
    }
}
$page_title = 'Dang nhap';
include ('templates/header.inc');
```

```
if (isset($message)) {
   echo '<font color="red">', $message, '</font>';
}
?>
<form action="<?php echo $_SERVER['PHP_SELF']; ?>"
→ method="post">
<fieldset><legend>Nhap vao thong tin cua ban:</legend>
<p><b>Ten dang nhap:</b> <input type="text" name="username"
→ size="10" maxlength="20" value="<?php if
→ (isset($_POST['username'])) echo $_POST['username']; ?>"
→ /></p>
<p><b>Mat khau:</b> <input type="password" name="password"
→ size="20" maxlength="20" /></p>
<div align="center"><input type="submit" name="submit"
→ value="Dang nhap" /></div>
</fieldset>
</form>
<?php
include ('templates/footer.inc');
?>
```

*Hàm **session_get_cookie_params()** trả lại một mảng các thiết lập cookie hiện tại.*

*Các tham số cookie của session cũng có thể được thay đổi bằng hàm **ini_set()**.*

Thời hạn hết hiệu lực của cookie chỉ nói đến thời gian tồn tại của cookie trong trình duyệt Web chứ không nói đến thời gian tồn tại của dữ liệu session được chứa trên máy chủ.

Sử dụng session mà không cần đến cookie

Theo mặc định, để làm việc chính xác, session phải phụ thuộc vào cookie. Khi một session bắt đầu, nó gửi một cookie xuống trình duyệt Web của người dùng. Sau đó, tất cả trang phải sử dụng hàm **session_start()** để sử dụng cookie (chứa tên và ID của session). Vấn đề là ở chỗ người dùng có thể tắt cookie trong trình duyệt hoặc không chấp nhận cookie vì họ không hiểu mục đích của nó. Trong trường hợp như vậy, PHP sẽ tạo ra một session mới cho mỗi trang và không một biến đăng ký nào có thể truy xuất được.

Chúng ta có thể sử dụng session mà không cần đến cookie bằng cách truyền tên và định danh của session từ trang này sang trang khác. Rất dễ để thực hiện kỹ

thuật này. Tuy nhiên, nếu bạn chỉ quên truyền session trong một lần nào đó, toàn bộ quá trình sẽ bị mất.

Để truyền tên session từ trang này sang trang khác, bạn có thể sử dụng hằng SID (Session ID). Hằng này có giá trị dạng session_name=session_ID. Nếu giá trị này được nối vào cuối mỗi URL trong site, session sẽ tiếp tục làm việc cho dù người dùng có chấp nhận cookie hay không.

Thực hành sử dụng session mà không cần đến cookie theo các bước sau:

1. Mở tập tin **login.php** (tham khảo đoạn mã 7.15) trong trình soạn thảo văn bản.
2. Thay thế dòng **session_set_cookie_params()** với mã lệnh sau:

```
ini_set ('session.use_cookies', 0);
```

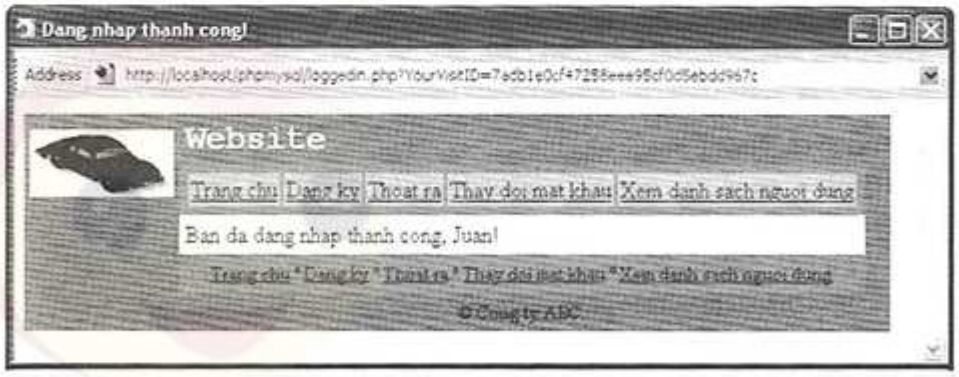
Mã lệnh này thông báo cho PHP không sử dụng cookie.

3. Thay đổi dòng chuyển hướng trình duyệt (**header()**) thành:

```
header ("Location:  http://" . $_SERVER['HTTP_HOST'] .
→ dirname($_SERVER['PHP_SELF']) . "/loggedin.php?" . SID);
```

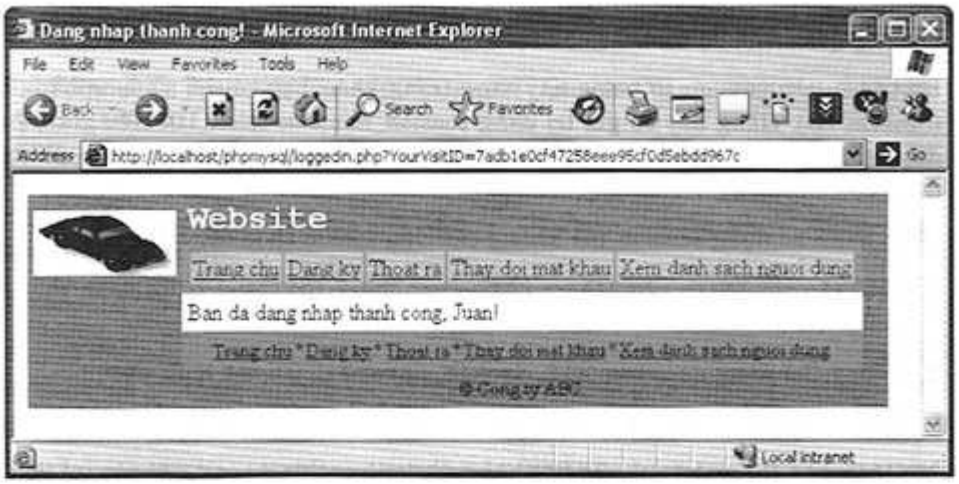
Việc bổ sung dấu chấm hỏi (?) và hằng SID vào mã lệnh chuyển hướng trình duyệt sẽ đưa **?session_name=session_ID** vào trong URL. Theo cách này, định danh của session sẽ được truyền một cách hiệu quả cho mã kịch bản **loggedin.php**.

4. Lưu lại tập tin, tải nó lên máy chủ Web và thử trong trình duyệt (xem hình 7-20). Mã lệnh đầy đủ của tập tin **login.php** được thể hiện trong đoạn mã 7.16.



Hình 7-20: *Khi trình duyệt được chuyển hướng tới trang* ***loggedin.php****, tên và định danh của session sẽ được đưa vào trong URL.*

5. Sao chép URL từ trình duyệt và dán vào trong một trình duyệt khác (xem hình 7-21).



Hình 7-21: *Bằng cách sử dụng định danh của session hiện có trong một trình duyệt mới, chúng ta có thể sử dụng session của người dùng khác và truy xuất đến tất cả dữ liệu session đã được đăng ký của họ.*

Đây được gọi là "ăn trộm session" và là một trong những lý do tại sao phải sử dụng cookie mỗi khi có thể.

6. Chỉnh sửa tập tin **header.inc** và **footer.inc** để tất cả các liên kết đều có kèm theo ?session_name=session_ID.

Một vấn đề khác với việc sử dụng session[illegible]okie là
húng ta phải quan tâm đến tất cả các liên kết trong toàn bộ ứng dụng của mình

hông sử ng cách

Đoạn mã 7.16: *Phiên bản cuối cùng của mã kịch bản đăng nhập này dụng cookie, thay vào đó việc duy trì trạng thái sẽ được thực hiện truyền định danh của session theo URL.*

```
<?php # Doan ma 7.16 - login.php
if (isset($_POST['submit'])) {
   require_once ('../mysql_connect.php');
   function escape_data ($data) {
      global $dbc;
      if (ini_get('magic_quotes_gpc')) {
         $data = stripslashes($data);
      }
      return mysql_real_escape_string($data, $dbc);
   }
   $message = NULL;
   if (empty($_POST['username'])) {
```

```
        $u = FALSE;
        $message .= '<p>Ban chua nhap ten dang nhap!</p>';
    } else {
        $u = escape_data($_POST['username']);
    }
    if (empty($_POST['password'])) {
        $p = FALSE;
        $message .= '<p>Ban chua nhap mat khau!</p>';
    } else {
        $p = escape_data($_POST['password']);
    }
    if ($u && $p) {
        $query = "SELECT user_id, first_name FROM users WHERE
        → username='$u' AND password=PASSWORD('$p')";
        $result = @mysql_query ($query);
        $row = mysql_fetch_array ($result, MYSQL_NUM);
        if ($row) {
            session_name ('YourVisitID');
            ini_set ('session.use_cookies', 0);
            session_start();
            $_SESSION['first_name'] = $row[1];
            $_SESSION['user_id'] = $row[0];
            header ("Location:  http://" . $_SERVER['HTTP_HOST'] .
            → dirname($_SERVER['PHP_SELF']) . "/loggedin.php?" .
            → SID);
            exit();
        } else {
            $message = '<p>Ten dang nhap va mat khau khong
            → phu hop voi du lieu trong CSDL.</p>';
        }
        mysql_close();
    } else {
        $message .= '<p>Hay thu lai.</p>';
    }
}
$page_title = 'Dang nhap';
include ('templates/header.inc');
if (isset($message)) {
    echo '<font color="red">', $message, '</font>';
```

```
}
?>
<form action="<?php echo $_SERVER['PHP_SELF']; ?>"
→ method="post">
<fieldset><legend>Nhap vao thong tin cua ban:</legend>
<p><b>Ten dang nhap:</b> <input type="text" name="username"
→ size="10" maxlength="20" value="<?php if
→ (isset($_POST['username'])) echo $_POST['username']; ?>"
→ /></p>
<p><b>Mat khau:</b> <input type="password" name="password"
→ size="20" maxlength="20" /></p>
<div align="center"><input type="submit" name="submit"
→ value="Dang nhap" /></div>
</fieldset>
</form>
<?php
include ('templates/footer.inc');
?>
```

*Nếu có quyền truy xuất tập tin **php.ini** và đang sử dụng PHP 4, bạn có thể thiết lập **session.use_trans_sid** thành 1 hoặc On. Điều này sẽ thông báo cho PHP tự động nối SID vào trong mỗi URL thay vì phải thực hiện thủ công như trong ví dụ trên. Tuy nhiên, nó sẽ làm giảm tốc độ thực hiện của các mã kịch bản vì PHP phải kiểm tra các URL trong từng trang.*

*Hàm **session_id()** trả lại giá trị session hiện tại (hoặc cho phép bạn xác định session để sử dụng).*

Chúng ta có thể truyền hằng SID từ trang này sang trang khác bằng cách chứa nó trong một trường ẩn trên biểu mẫu.

Tùy vào trình duyệt Web đang được sử dụng bởi khách hàng, một session có thể theo trình duyệt (browser-specific) hoặc theo cửa sổ (window-specific). Nếu session theo cửa sổ, một cửa sổ pop-up trong site sẽ không phải là thành phần của cùng một session trừ khi nó nhận được định danh của session.

Chú ý, việc sử dụng phương pháp chứa và truyền định danh của session trong URL kém an toàn hơn so với việc sử dụng cookie.

Chương 8:

AN TOÀN

An toàn của các ứng dụng Web là một chủ đề quan trọng mà chúng ta thường xuyên nhắc đến trong mỗi chương, khi đề cập đến một chủ đề nhất định trong cuốn sách này. Mặc dù đã đề cập các vấn đề liên quan đến an toàn nhiều nơi trong cuốn sách, nhưng chương 8 sẽ tập trung vào chủ đề này, giúp bạn bổ sung những kiến thức còn thiếu về nó.

Các chủ đề được đề cập ở đây bao gồm kiểm tra xác thực HTTP (HTTP Authentication) với PHP, kiểm tra tính hợp lệ của biểu mẫu với JavaScript, sử dụng biểu thức quy tắc và an toàn cơ sở dữ liệu. Không giống với hai chương trước, trong đó sử dụng một loạt các ví dụ gắn liền với nhau, chương này sẽ sử dụng các ví dụ cụ thể cho từng chủ đề.

Xác thực HTTP

Một đặc điểm mới của PHP 4 là khả năng kiểm tra xác thực HTTP, mặc dù nó chỉ có hiệu lực khi chạy PHP dưới dạng môđun của Apache. Xác thực HTTP là cách để bảo vệ tập tin hoặc thư mục bằng cách yêu cầu người dùng nhập vào tên đăng nhập và mật khẩu trong một cửa sổ pop-up để có thể truy xuất được chúng (xem hình 8-1).

Hình 8-1: *Hộp nhắc kiểm tra xác thực HTTP.*

Hộp nhắc kiểm tra xác thực HTTP được tạo bằng dòng lệnh sau:

```
header('WWW-Authenticate: Basic realm="Some Caption"');
```

Bạn có thể thay đổi thuộc tính realm để hộp nhắc thể hiện điều muốn thông báo với người dùng.

Thông thường, chúng ta bổ sung dòng lệnh dưới đây vào sau dòng lệnh này:

```
header('HTTP/1.0 401 Unauthorized');
```

Dòng lệnh này sẽ có tác dụng khi người dùng nhấp vào nút Cancel trong cửa sổ kiểm tra xác thực.

Giá trị được nhập vào trong cửa sổ kiểm tra xác thực có thể được truy xuất thông qua `$_SERVER['PHP_AUTH_USER']` và `$_SERVER['PHP_AUTH_PW']` (hoặc thông qua `$HTTP_SERVER_VARS` hay đơn giản là `$PHP_AUTH_PW` nếu thông số `register_globals` được kích hoạt). Sau đó, các giá trị này được sử dụng để so sánh với các giá trị được lưu giữ trong một hồ sơ văn bản, một tập tin `.htaccess` hoặc cơ sở dữ liệu.

Các ưu điểm của xác thực HTTP gồm:

- Ít yêu cầu mã lệnh PHP.
- Tên đăng nhập và mật khẩu được ghi nhớ mà không phải dùng PHP để truyền cookie hoặc thiết lập session.
- Không cần phải có hộp đăng nhập trên trang Web.
- Xác thực có thể được kiểm tra bằng cách sử dụng cơ sở dữ liệu.
- Xác thực có thể được thiết lập trên cơ sở từng trang (thay vì toàn bộ thư mục).

Nhược điểm gồm:

- Giới hạn sử dụng (chỉ làm việc dạng môđun với Apache, tuy Apache là máy chủ Web phổ biến nhất và PHP là môđun phổ biến nhất của máy chủ này).
- Không thể thiết lập nhóm người dùng hoặc xác định mức truy cập.

Với các thông tin này, chúng ta sẽ viết hai mã kịch bản để minh họa cách sử dụng kiểm tra xác thực HTTP trong ứng dụng Web.

Thực hành sử dụng kiểm tra xác thực HTTP thực hiện theo các bước sau:

1. Tạo một hồ sơ PHP mới trong trình soạn thảo văn bản.

```
<?php # Doan ma 8.1 - authentication.php
```

Mã kịch bản đầu tiên sẽ thực hiện việc kiểm tra xác thực. Mã kịch bản thứ hai sẽ đưa mã kịch bản này vào để buộc kiểm tra xác thực trước khi cho phép truy xuất nó.

2. Tạo một biến mới và kiểm tra thông tin gửi để kiểm tra xác thực.

```
$authorized = FALSE;

if ( (isset($_SERVER['PHP_AUTH_USER']) AND
→ isset($_SERVER['PHP_AUTH_PW'])) ) {
```

Biến **$authorized** được dùng để xác định xem người dùng có được phép truy xuất trang này hay không. Sau đó, phần điều kiện sẽ kiểm tra xem cả hai biến yêu cầu có giá trị không (nghĩa là hộp nhắc xác thực đã được gửi).

3. Kết nối với cơ sở dữ liệu.

```
define ('DB_USER', 'username');
define ('DB_PASSWORD', 'password');
define ('DB_HOST', 'localhost');
define ('DB_NAME', 'sitename');
$dbc = @mysql_connect (DB_HOST, DB_USER, DB_PASSWORD) OR
→ die ('Could not connect to MySQL: ' . mysql_error() );
mysql_select_db (DB_NAME) OR
→ die('Could not select the database: ' . mysql_error());
```

Các dòng lệnh này tương tự như trong Chương 6 "Sử dụng PHP và MySQL". Bạn có thể thay đổi các giá trị để phù hợp với cơ sở dữ liệu của mình (hoặc chỉ cần điều chỉnh quyền truy xuất được viết trong mã kịch bản **mysql_connect.php**).

4. Truy vấn cơ sở dữ liệu và hoàn thành phần điều kiện.

```
    $query = "SELECT first_name FROM users WHERE
    → username='{$_SERVER['PHP_AUTH_USER']}' and
    → password=PASSWORD('{$_SERVER['PHP_AUTH_PW']}')";
    $result = mysql_query ($query);
    $row = @mysql_fetch_array ($result);
    if ($row) {
        $authorized = TRUE;
    }
}
```

Các dòng lệnh này tương tự như trong Chương 7 "Cookie và Session", trong đó ta sử dụng tên đăng nhập và mật khẩu được gửi lên để so sánh chúng với các giá trị được chứa trong cơ sở dữ liệu. Trong thực tế, mã kịch bản này sử dụng chung cơ sở dữ liệu (*sitename*) và bảng (*users*) như trong Chương 7.

5. Kiểm tra xem người dùng có được quyền truy xuất không và hoàn tất trang PHP.

```
if (!$authorized) {
    header('WWW-Authenticate: Basic realm="Website"');
    header('HTTP/1.0 401 Unauthorized');
```

```
}
?>
```

Mã lệnh này kiểm tra biến **$authorized**. Nếu nó có giá trị **FALSE** (vì người dùng chưa thấy hộp nhắc xác thực, chưa gửi thông tin xác thực hoặc vì thông tin họ gửi không phù hợp với thông tin trong cơ sở dữ liệu), hộp nhắc kiểm tra xác thực sẽ được hiển thị. Lời gọi hàm **header()** thứ hai được dùng trong trường hợp người dùng nhấp nút Cancel trong hộp nhắc kiểm tra xác thực.

6. Lưu trang với tên gọi **authentication.php**, tải nó lên máy chủ Web (bạn nên đặt nó ngoài phạm vi của thư mục Web). Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 8.1.

Đoạn mã 8.1: *Mã kịch bản này xử lý việc kiểm tra xác thực bằng cách dùng HTTP và một cơ sở dữ liệu.*

```
<?php # Doan ma 8.1 - authentication.php
$authorized = FALSE;
if ( (isset($_SERVER['PHP_AUTH_USER']) AND
→ isset($_SERVER['PHP_AUTH_PW'])) ) {
   define ('DB_USER', 'username');
   define ('DB_PASSWORD', 'password');
   define ('DB_HOST', 'localhost');
   define ('DB_NAME', 'sitename');
   $dbc = @mysql_connect (DB_HOST, DB_USER, DB_PASSWORD) OR
   → die ('Could not connect to MySQL: ' . mysql_error() );
   mysql_select_db (DB_NAME) OR die ('Could not select the
   → database: ' . mysql_error() );
   $query = "SELECT first_name FROM users WHERE
   → username='{$_SERVER['PHP_AUTH_USER']}' and
   → password=PASSWORD('{$_SERVER['PHP_AUTH_PW']}')";
   $result = mysql_query ($query);
   $row = @mysql_fetch_array ($result);
   if ($row) {
      $authorized = TRUE;
   }
}
if (!$authorized) {
   header('WWW-Authenticate: Basic realm="Website"');
   header('HTTP/1.0 401 Unauthorized');
}
?>
```

Thực hành bổ sung kiểm tra xác thực cho một trang theo các bước sau:

1. Tạo một hồ sơ PHP mới trong trình soạn thảo văn bản và đưa vào tập tin kiểm tra xác thực.

```
<?php # Script 8.2 - index.php
require_once ('../authentication.php');
?>
```

Mỗi trang bắt đầu với **require_once('../authentication.php')** đều tự động kiểm tra xác thực người dùng trước khi cho phép họ truy xuất. Trang kiểm tra xác thực được đặt ngoài thư mục gốc của Web vì mục đích an toàn.

2. Tạo phần đầu HTML.

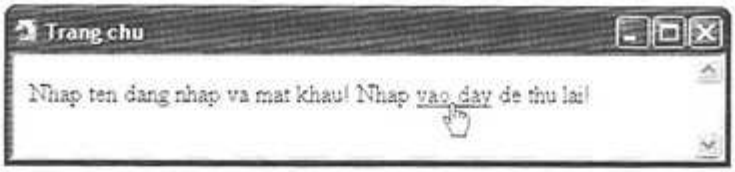
```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
→ Transitional//EN" "http://www.w3.org/TR/2000/REC-
→xhtml1-20000126/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">
<head>
<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />
<title>Trang chu</title>
</head>
<body>
```

3. Kiểm tra xem người dùng đã được kiểm tra xác thực chưa.

```
<?php
if (!$authorized) {
echo '<p>Nhap ten dang nhap va mat khau! Nhap
→<a href="index.php">vao day</a> de thu lai!</p>';
} else {
   echo '<p>Ban da duoc xac thuc!</p>';
}
?>
```

Điều kiện này sẽ in hai thông báo khác nhau dựa trên giá trị của biến **$authorized**. Nếu nó có giá trị **FALSE**, người dùng sẽ được yêu cầu nhập lại tên đăng nhập và mật khẩu hợp lệ (xem hình 8-2). Đây cũng là trường hợp khi người dùng nhấp nút Cancel.

Nếu biến **$authorized** có giá trị **TRUE**, khi đó thông báo cho biết người dùng đã được kiểm tra xác thực thành công sẽ được hiển thị (xem hình 8-3).



Hình 8-2: *Nếu người dùng không vượt qua bước kiểm tra xác thực hoặc nhấp nút Cancel, họ sẽ thấy thông báo như trong hình (trên một số trình duyệt).*

Hình 8-3: *Kết quả của kiểm tra xác thực thành công.*

4. Hoàn tất mã HTML.

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

5. Lưu tập tin với tên gọi **index.php**, tải nó lên máy chủ Web và chạy thử trong trình duyệt (xem hình 8-4). Mã lệnh đầy đủ của tập tin được thể hiện đầy đủ trong đoạn mã 8.2.

Hình 8-4: *Các trường trong hộp nhắc kiểm tra xác thực được tự động điền bằng các thông tin đăng ký đã được thực hiện trong Chương 7.*

Vì mã kịch bản này sử dụng cùng cơ sở dữ liệu như các ví dụ trong chương trước, nên bạn có thể sử dụng tên đăng nhập và mật khẩu đã đăng ký.

Đoạn mã 8.2: *Trang này sử dụng tập tin kiểm tra xác thực (tham khảo lại đoạn mã 8.1) để bảo vệ.*

```
<?php # Doan ma 8.2 - index.php
require_once ('../authentication.php');
?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
→ Transitional//EN"
→ "http://www.w3.org/TR/2000/REC-xhtml1-20000126/DTD/xhtml1-
→ transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">
<head>
<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />
<title>Trang chu</title>
</head>
<body>
<?php
if (!$authorized) {
   echo '<p>Nhap ten dang nhap va mat khau! Nhap
   →<a href="index.php">vao day</a> de thu lai!</p>';
} else {
   echo '<p>Ban da duoc xac thuc!</p>';
}
?>
</body>
</html>
```

Một tùy chọn an toàn khác để bảo vệ các tập tin và thư mục là dùng tập tin ***.htaccess*** *(chỉ áp dụng đối với Apache). Đọc các tài liệu về Apache để biết thêm chi tiết.*

Để tương thích tốt nhất với tất cả các trình duyệt, bạn nên viết mã lệnh trong mã kịch bản này một cách chính xác (Basic với chữ cái B ở dạng in hoa, realm trong dấu nháy kép và có một khoảng trắng giữa 1.0 và 401 Unauthorized).

Có thể truyền tên đăng nhập và mật khẩu tới mã kịch bản bằng cách sử dụng định dạng:
header("Location:http://$PHP_AUTH_USER:$PHP_AUTH_PW@site.com").
Tuy nhiên, làm như vậy sẽ không an toàn.

Trên một số trình duyệt, mã kịch bản kiểm tra xác thực sẽ tiếp tục nhắc người dùng cho đến khi họ nhập vào thông tin đăng nhập đúng hoặc

nhấp nút Cancel. Trên các trình duyệt khác, ngay lần đầu nhập vào thông tin sai, họ sẽ thấy ngay thông báo Please enter a valid username...

Trên một số trình duyệt, chúng ta có thể tạo một hệ thống thoát ra bằng cách gửi xuống trình duyệt mã lệnh sau: `header('HTTP/1.0 401 Unauthorized')`. *Mã lệnh này sẽ xóa tên đăng nhập và mật khẩu trong bộ đệm của trình duyệt.*

Kiểm tra biểu mẫu với JavaScript

Mặc dù JavaScript không phải là một biện pháp an toàn thực sự, nhưng nó giúp bổ sung một mức độ an toàn nhất định và cung cấp sự tiện lợi cho người dùng. Vì JavaScript là một công nghệ sử dụng phía máy khách hàng (trong khi PHP là một trong những công nghệ phía máy chủ), việc tích hợp nó vào trong trang Web sẽ giúp người dùng tránh được việc phải đụng đến mã kịch bản PHP trước khi họ thấy có vấn đề. Thay vì vậy, chúng ta sử dụng JavaScript để ngay lập tức thực hiện một số kiểm tra, sau đó nếu dữ liệu đáp ứng yêu cầu, gửi biểu mẫu lên cho mã kịch bản PHP.

Bản thân JavaScript không an toàn vì nó dễ dàng bị mất hiệu lực trong trình duyệt (xem hình 8-5). Do đó, bạn nên sử dụng PHP như là một biện pháp an toàn chính của mình.

Hình 8-5: *Phần lớn trình duyệt Web ngày nay cho phép người dùng làm mất hiệu lực của JavaScript.*

Để minh họa, chúng ta sẽ tạo một biểu mẫu để gửi các URL (sẽ được sử dụng trong Chương 11 "Ví dụ - Quản lý nội dung"). Phần kế tiếp của chương này sẽ tiếp tục kiểm tra dữ liệu được gửi bằng cách dùng biểu thức quy tắc.

Thực hành kiểm tra tính hợp lệ của các biểu mẫu theo các bước sau:

1. Tạo một hồ sơ HTML mới trong trình soạn thảo văn bản.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
→ Transitional//EN" "http://www.w3.org/TR/2000/REC-xhtml1-
→ 20000126/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">
<head>
<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />
<title>Gui URL</title>
```

Trong ví dụ này, chúng ta sẽ tạo một trang HTML để hiển thị biểu mẫu và một mã kịch bản PHP riêng để xử lý nó.

2. Tạo phần JavaScript và bắt đầu hàm.

```
<script type="text/javascript" language="Javascript">
<!--
function CheckData() {
   var problem = 'No';
```

JavaScript có thể được đặt nơi bất kỳ trong hồ sơ HTML với thẻ <script>, nhưng bạn nên đặt chúng ở đầu trang.

Hàm `CheckData()` sẽ được gọi khi biểu mẫu được gửi đi. Mục đích duy nhất của nó là để kiểm tra tất cả các giá trị nhập. Biến `problem` (các biến trong JavaScript không sử dụng ký tự `$` ở đầu tên biến) sẽ được dùng như một cờ để báo hiệu nếu có bất kỳ một vấn đề gì.

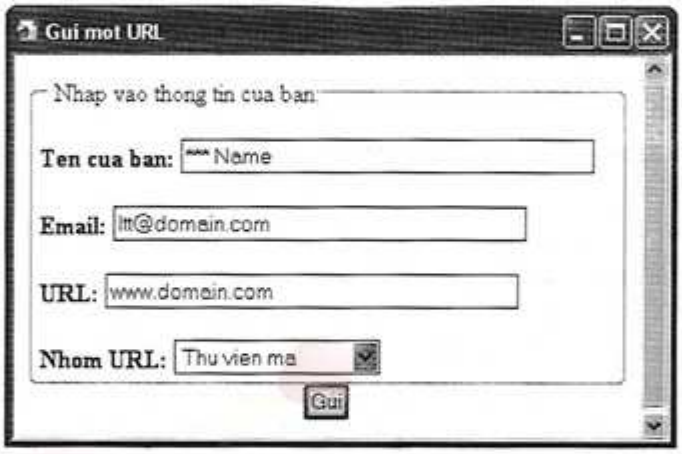
3. Kiểm tra xem người dùng có nhập vào tên không.

```
if (document.url_form.name.value.length <= 0) {
   alert ("Nhap vao ten cua ban.");
   document.url_form.name.value = "*** Name";
   document.url_form.name.focus();
   problem = 'Yes';
}
```

Kết cấu `document.tenbieumau.tentruongnhap.value.length` cho biết chiều dài của chuỗi được nhập vào trường *tentruongnhap* của biểu mẫu *tenbieumau* trong hồ sơ (hoặc cửa sổ trình duyệt). Vì thế, để kiểm tra xem người dùng có nhập phần tên không, chúng ta sử dụng `document.url_form.name.value.length` và đảm bảo giá trị phải lớn hơn `0`.

Nếu người dùng không nhập tên, khi đó một hộp cảnh báo sẽ được tạo (xem hình 8-6) bởi lệnh **alert()**. Kế đến, trường *name* sẽ được cung cấp giá trị ***Name* để gây chú ý và tiêu điểm sẽ được chuyển đến trường *name* trong trình duyệt Web (xem hình 8-7) thông qua chức năng **focus()** của JavaScript.

Hình 8-6: *Hàm **alert()** của JavaScript sẽ tạo ra hộp thông báo này.*



Hình 8-7: *Hàm **focus()** của JavaScript sẽ làm nổi bật một phần tử giống như hộp văn bản trong biểu mẫu này (kết quả thực tế còn phụ thuộc vào trình duyệt).*

Nếu có một vấn đề nào đó, biến **problem** sẽ được thiết lập giá trị Yes.

4. Lặp lại quá trình này cho địa chỉ thư điện tử và URL.

```
if (document.url_form.email.value.length <= 0) {
   alert ("Nhap vao dia chi thu dien tu cua ban.");
   document.url_form.email.value = "*** Email Address";
   document.url_form.email.focus();
   problem = 'Yes';
}
if (document.url_form.url.value.length <= 0) {
   alert ("Nhap vao URL.");
```

```
        document.url_form.url.value = "*** URL";
        document.url_form.url.focus();
        problem = 'Yes';
    }
```

Hai thủ tục kiểm tra này là bản điều chỉnh của mã lệnh kiểm tra tên trong bước 3, với việc thay thế *name* bằng *email* và *url* một cách tương ứng.

5. Kiểm tra để đảm bảo một nhóm URL đã được chọn.

```
    if ((document.url_form.url_category.value == 0) ||
    → (document.url_form.url_category.selectedIndex == 0)) {
        alert ("Hay chon mot nhom cho URL.");
        problem = 'Yes';
    }
```

Việc sử dụng JavaScript với menu thả xuống trong Netscape Navigator và Internet Explorer (hai trình duyệt hàng đầu hiện nay) là khác nhau. Mã lệnh `...url_category.value == 0` sẽ làm việc với IE, còn `...url_category.selectedIndex == 0` sẽ làm việc với Netscape. Chúng ta đưa cả hai biểu thức này vào trong phần điều kiện của mình với một liên kết OR, miễn là một trong hai điều kiện này có giá trị true, bước kiểm tra sẽ thành công.

6. Kiểm tra xem có vấn đề nào không và hoàn tất mã JavaScript.

```
        if (problem == 'No') {
            return true;
        } else {
            return false;
        }
    }
    //-->
    </script>
```

Biến `problem` ban đầu được gán giá trị No. Nếu biểu mẫu vượt qua tất cả bốn bước kiểm tra trên, nó vẫn có giá trị là No. Nếu biểu mẫu không vượt qua được một bước kiểm tra nào đó, nó sẽ có giá trị Yes.

Nếu hàm `CheckData()` trả lại giá trị true, nghĩa là biểu mẫu đã vượt qua được các bước kiểm tra và nó sẽ được gửi lên cho mã kịch bản PHP. Nếu hàm này trả về giá trị false, biểu mẫu sẽ không được gửi đi và người dùng vẫn ở lại với trang hiện tại.

7. Hoàn tất phần đầu HTML, tiếp tục với phần thân và biểu mẫu.

```
    </head>
    <body>
```

```
<form name="url_form" action="handle_submit_url.php"
→ method="post" onsubmit="return CheckData()">
```

Khi kiểm tra biểu mẫu với JavaScript, phần quan trọng nhất chính là mã lệnh `<form ...>`. Trong đó, chúng ta đặt tên cho biểu mẫu là **url_form**. Tên này đã được tham chiếu bởi mã lệnh JavaScript trong các bước trên. Biểu mẫu cũng được thiết lập để gọi hàm `CheckData()` khi nút submit được nhấn.

8. Hoàn tất biểu mẫu.

```
<fieldset><legend>Nhap vao thong tin cua ban:</legend>
<p><b>Ten cua ban:</b> <input type="text" name="name"
→ size="40" maxlength="60" /></p>
<p><b>Email:</b> <input type="text" name="email" size="40"
→ maxlength="60" /> </p>
<p><b>URL:</b> <input type="text" name="url" size="40"
→ maxlength="80" /></p>
<p><b>Nhom URL:</b> <select name="url_category">
<option>Chon mot</option>
<option value="3">Thu vien ma</option>
<option value="6">CSDL noi chung</option>
<option value="5">MySQL noi chung</option>
<option value="1">PHP noi chung</option>
<option value="4">Lap trinh</option>
<option value="2">Phat trien web</option>
</select></p>
</fieldset>
<div align="center"><input type="submit" name="submit"
→ value="Gui" /></div>
</form>
```

Mã kịch bản này sẽ được dùng trong Chương 11, để tạo một menu thả xuống với dữ liệu trong cơ sở dữ liệu *content*. Còn bây giờ, chúng ta mã hóa cố định các giá trị *url_category*.

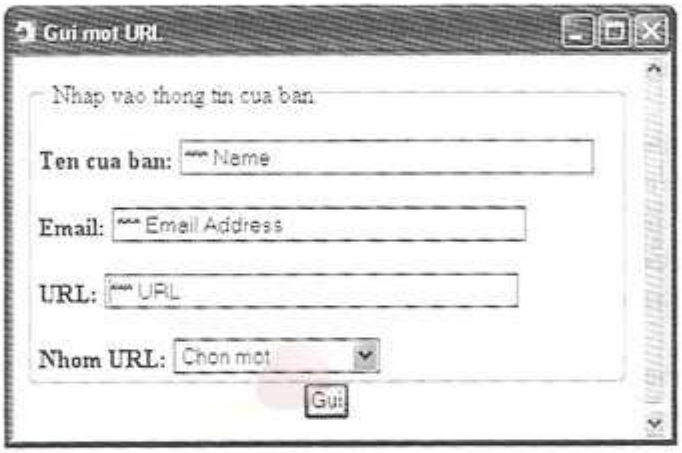
9. Hoàn tất trang HTML.

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

10. Lưu tập tin với tên gọi **submit_url.html** và chạy thử nó trong trình duyệt (xem hình 8-8 và 8-9). Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 8.3.

Hình 8-8: *Hộp cảnh báo sẽ được tạo nếu có một trường nào đó bị bỏ qua.*

Bạn không cần phải tải tập tin này lên máy chủ Web vì đây chỉ là một trang HTML thuần túy. Trong bất kỳ trường hợp nào (chạy thông qua máy chủ hay chạy trên máy cục bộ), nếu biểu mẫu vượt qua được các kiểm tra, bạn sẽ thấy một thông báo lỗi *Page Not Found* (vì mã kịch bản xử lý biểu mẫu vẫn chưa được viết).



Hình 8-9: *Các trường bị bỏ qua sẽ được đánh dấu bằng các dấu hoa thị.*

Đoạn mã 8.3: *Trang* ***submit_url.html*** *sử dụng JavaScript để kiểm tra tính hợp lệ của biểu mẫu.*

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
→ Transitional//EN"
→ "http://www.w3.org/TR/2000/REC-xhtml1-20000126/DTD/xhtml1-
→transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">
<head>
<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />
<title>Gui mot URL</title>
```

```
<script type="text/javascript" language="Javascript">
<!--
function CheckData() {
   var problem = 'No';
   if (document.url_form.name.value.length <= 0) {
      alert ("Nhap vao ten cua ban.");
      document.url_form.name.value = "*** Name";
      document.url_form.name.focus();
      problem = 'Yes';
   }
   if (document.url_form.email.value.length <= 0) {
      alert ("Nhap vao dia chi thu dien tu cua ban.");
      document.url_form.email.value = "*** Email Address";
      document.url_form.email.focus();
      problem = 'Yes';
   }
   if (document.url_form.url.value.length <= 0) {
      alert ("Nhap vao URL.");
      document.url_form.url.value = "*** URL";
      document.url_form.url.focus();
      problem = 'Yes';
   }
   if ((document.url_form.url_category.value == 0) ||
   → (document.url_form.url_category.selectedIndex == 0)) {
      alert ("Hay chon mot nhom cho URL.");
      problem = 'Yes';
   }
   if (problem == 'No') {
      return true;
   } else {
      return false;
   }
}
//-->
</script>
</head>
<body>
<!-- Doan ma 8.3 - submit_url.html -->
```

```
<form name="url_form" action="handle_submit_url.php"
→ method="post" onsubmit="return CheckData()">
<fieldset><legend>Nhap vao thong tin cua ban:</legend>
<p><b>Ten cua ban:</b> <input type="text" name="name"
→ size="40" maxlength="60" /></p>
<p><b>Email:</b> <input type="text" name="email" size="40"
→ maxlength="60" /> </p>
<p><b>URL:</b> <input type="text" name="url" size="40"
→ maxlength="80" /></p>
<p><b>Nhom URL:</b> <select name="url_category">
<option>Chon mot</option>
<option value="3">Thu vien ma</option>
<option value="6">CSDL noi chung</option>
<option value="5">MySQL noi chung</option>
<option value="1">PHP noi chung</option>
<option value="4">Lap trinh</option>
<option value="2">Phat trien web</option>
</select></p>
</fieldset>
<div align="center"><input type="submit" name="submit"
→ value="Gui" /></div>
</form>
</body>
</html>
```

Kiểm tra biểu mẫu với PHP

Chúng ta đã nhiều lần đề cập đến việc kiểm tra biểu mẫu với PHP trong cuốn sách này. Các quy tắc kiểm tra cần tuân thủ:

Sử dụng các biến siêu toàn cục (ví dụ `$_POST['tênbiến']`*) thay cho biến toàn cục (*`$tênbiến`*).*

- *Kiểm tra các trường nhập văn bản, mật khẩu và vùng văn bản bằng hàm* `empty()`*.*
- *Kiểm tra các trường nhập khác bằng hàm* `isset()`*.*
- *Kiểm tra một trường nhập bất kỳ bằng cách kiểm tra chiều dài của chúng có phải là số dương không.*

Một phương pháp kiểm tra chính xác hơn với PHP là sử dụng biểu thức quy tắc. Chúng ta sẽ đề cập đến phương pháp này trong phần kế tiếp. Từ phiên bản 1.5, JavaScript cũng hỗ trợ biểu thức quy tắc. Để mã lệnh được tương thích hơn, bạn nên tránh sử dụng biểu thức quy tắc trong JavaScript.

*Sử dụng phương pháp **POST** với biểu mẫu sẽ an toàn hơn so với việc sử dụng phương pháp **GET**. Nếu biểu mẫu có một trường mật khẩu hoặc chứa các thông tin nhạy cảm, bạn nên sử dụng phương pháp **POST**.*

- *Chú ý, các trường ẩn trong biểu mẫu vẫn có thể xem được trong mã nguồn HTML.*
- *Việc sử dụng JavaScript cho vấn đề an toàn cũng tương tự như việc gắn tờ giấy ghi "Ngôi nhà này được bảo vệ bởi..." trên cửa nhà bạn vậy. Nó chỉ có thể bảo vệ ở một mức độ nhất định.*
- *Các nghiên cứu trực tuyến cho thấy có khoảng 10-20% người sử dụng không kích hoạt khả năng hỗ trợ JavaScript trong trình duyệt của mình.*

Biểu thức quy tắc

Biểu thức quy tắc (regular expresssion) là một trong những công cụ rất mạnh, nhưng khá tẻ nhạt trong phần lớn ngôn ngữ lập trình hiện nay. Hãy xem các biểu thức quy tắc như một hệ thống so khớp mẫu phức tạp (system of matching pattern). Đầu tiên, chúng ta viết mẫu (pattern), sau đó sử dụng một trong những hàm cài sẵn của PHP để áp dụng mẫu đó cho một chuỗi văn bản (biểu thức quy tắc được dùng chủ yếu với chuỗi).

PHP có hai hàm sử dụng biểu thức quy tắc để so khớp mẫu (một phân biệt dạng chữ, một không phân biệt) và hai hàm để so khớp, thay thế văn bản phù hợp bằng văn bản khác (một phân biệt dạng chữ và một không phân biệt). Ngoài ra, PHP còn hỗ trợ hai dạng biểu thức quy tắc: POSIX Extended và PCRE (tương thích PERL). Phiên bản POSIX hơi yếu và chậm hơn so với PCRE, nhưng lại dễ dùng hơn. Vì lý do này, chúng ta sẽ để cập đến biểu thức POSIX. Nếu đã quen với cú pháp tương thích PERL, bạn chỉ cần thay thế tên hàm POSIX với dạng PCRE tương đương trong các ví dụ.

Một số trình soạn thảo văn bản, chẳng hạn như BBEdit và emacs, cho phép sử dụng các biểu thức quy tắc để so khớp và thay thế các mẫu trong một hồ sơ hoặc trên nhiều hồ sơ.

Một khác biệt giữa biểu thức quy tắc POSIX và PCRE là dạng sau có thể sử dụng cho kiểu dữ liệu nhị phân trong khi dạng trước lại không thể.

Định nghĩa mẫu

Trước khi sử dụng bất kỳ hàm về biểu thức quy tắc nào của PHP, chúng ta phải định nghĩa mẫu mà hàm đó sẽ dùng để so khớp. PHP có một số quy tắc để tạo ra một mẫu so khớp và phần lớn trong số chúng tương tự như các quy tắc được dùng với Perl, C, Java. Chúng ta có thể sử dụng các quy tắc một cách riêng biệt hoặc ở dạng tổ hợp, tạo ra các mẫu so khớp từ đơn giản đến phức tạp.

Để giải thích cách tạo ra một mẫu so khớp, chúng ta bắt đầu với các ký hiệu sẽ được sử dụng trong các biểu thức quy tắc, sau đó sẽ nói đến cách gom nhóm các ký tự lại với nhau và kết thúc với các lớp ký tự. Sau khi tất cả những điều này đã được đề cập, bạn có thể sử dụng chúng với các hàm PHP. Cũng như với quy tắc định dạng, chúng ta sẽ định nghĩa mẫu so khớp trong dấu nháy (`""`) và cho biết những gì mẫu này sẽ đáp ứng ở dạng in nghiêng.

Dạng ký tự đầu tiên được sử dụng để định nghĩa mẫu là hằng chữ. Hằng chữ là một giá trị được viết chính xác như khi nó được biên dịch. Ví dụ, mẫu "`a`" sẽ khớp với chữ cái *a*, "`ab`" sẽ khớp với *ab*... Do đó, giả sử một phép tìm kiếm không phân biệt dạng chữ được thực hiện, "`rom`" sẽ khớp với bất kỳ chuỗi nào dưới đây:

- CD-ROM.
- Rommel crossed the desert.
- I'm writing a roman a clef.

Cùng với các hằng chữ, mẫu so khớp còn sử dụng các ký tự meta. Đây là các ký hiệu đặc biệt có ý nghĩa khác với hằng chữ (xem bảng 8.1). Trong khi "`a`" chỉ có nghĩa là *a*, dấu chấm (.) lại khớp với một ký tự bất kỳ ("." khớp với *a, b, c, _*, khoảng trắng...). Để so khớp một ký tự meta bất kỳ, chúng ta cần phải mã hóa escape chúng, giống như khi mã hóa escape dấu nháy trong một chuỗi để in nó ra vậy. Khi đó "`\.`" sẽ khớp với dấu chấm.

Có hai ký tự meta quy định nơi ký tự phải được tìm thấy. Ký tự thứ nhất là dấu mũ (`^`) - so khớp một chuỗi bắt đầu với chữ cái đi sau dấu mũ. Ký tự thứ hai là dấu đô la (`$`) - so khớp với bất kỳ chuỗi nào kết thúc với chữ cái đi trước ký hiệu này. Ví dụ, mẫu "`^a`" sẽ khớp với bất kỳ chuỗi nào bắt đầu bằng chữ cái *a*, trong khi đó mẫu "`a$`" sẽ khớp với bất kỳ chuỗi nào kết thúc với chữ cái *a*. Do đó, "`^a$`" sẽ chỉ khớp với chữ cái *a* (một chuỗi bắt đầu và kết thúc với chữ cái *a* và chỉ có một ký tự).

Các biểu thức quy tắc cũng sử dụng ký tự chuyển hướng (`|`) với ý nghĩa tương tự như từ *hoặc*. Do đó, "`a|b`" sẽ khớp với *a* hoặc *b*.

Kế đến, có ba ký tự meta để cập đến số lần xuất hiện: "`a*`" sẽ khớp với 0 đến nhiều chữ cái *a* (*a, aa, aaa, aaaa, ...*); "`a+`" sẽ khớp với 1 đến nhiều chữ cái *a* (*a, aa, aaa...* nhưng phải có ít nhất một chữ cái); "`a?`" khớp với 0 hoặc 1 chữ cái *a* (*a* hoặc không có chữ *a* nào). Các ký tự meta này là những ký tự định lượng trong các mẫu so khớp, ngoài ra ta còn có thể sử dụng các dấu ngoặc móc để xác định con số cụ thể.

Để so khớp với một số lượng chữ cái nhất định, chúng ta đặt số lần xuất hiện trong dấu ngoặc móc (`{}`). Khai báo này cho biết số lượng tối thiểu, hoặc tối thiểu và tối đa. Ví dụ, "`a{3}`" sẽ khớp với *aaa* và "`a{3,5}`" sẽ khớp với *aaa, aaaa, aaaaa* (từ ba đến năm chữ cái *a*). Bảng 8.2 liệt kê tất cả các ký tự meta định lượng này.

Bảng 8.1: *Các ký tự meta có ý nghĩa riêng trong các biểu thức quy tắc.*

Ký tự	Tên	Ý nghĩa
^	Dấu mũ	Bắt đầu một chuỗi.
$	Đô la	Kết thúc một chuỗi.
.	Dấu chấm	Một ký tự bất kỳ.
|	Chuyển hướng	Tương đương (hoặc).

Bảng 8.2: *Các ký tự định lượng cho phép xác định số lần xuất hiện của một ký tự nào đó.*

Ký tự	Ý nghĩa
?	0 hoặc một lần xuất hiện.
*	0 hoặc nhiều lần xuất hiện.
+	Một hoặc nhiều lần xuất hiện.
{x}	Chính xác *x* lần xuất hiện.
{x,y}	Từ *x* đến *y* lần xuất hiện.
{x,}	Tối thiểu *x* lần xuất hiện.

Sau khi đã hiểu được các ký hiệu cơ bản, bạn có thể sử dụng các dấu ngoặc để gom nhóm các ký tự thành các mẫu có liên quan nhau. Ví dụ: "`(abc)`" sẽ khớp với *abc*, "`(trout)`" sẽ khớp với *trout*. Hãy xem các dấu ngoặc như là một hằng chữ với nhiều ký tự hơn. Vì thế, trong khi "`yes|no`" sẽ khớp với *yeso* hoặc *yeno*, thì "`(yes)|(no)`" lại khớp với *yes* hoặc *no*.

Cho dù bạn có kết hợp các chữ cái vào các nhóm như thế nào thì chúng cũng chỉ có tác dụng để so khớp các từ nhất định. Nhưng nếu muốn so khớp một từ gồm bốn chữ cái hoặc một chuỗi số bất kỳ thì sao? Để thực hiện điều này, chúng ta sẽ định nghĩa và sử dụng các lớp ký tự.

Các lớp được tạo bằng cách đặt các ký tự trong dấu ngoặc vuông (`[]`). Ví dụ, so khớp một nguyên âm với "`[aeiou]`" (để so sánh, "`(aeiou)`" sẽ khớp với một chuỗi gồm năm chữ cái có trong mẫu). Sử dụng dấu gạch ngang (-) để chỉ ra một phạm vi ký tự: "`[a-z]`" khớp với một chữ cái bất kỳ ở dạng in thường, "`[A-Z]`" khớp với một chữ cái bất kỳ ở dạng in hoa, "`[a-zA-Z]`" khớp với một chữ cái bất kỳ và "`[0-9]`" khớp với một chữ số bất kỳ. Ví dụ, "`[a-z]{3}`" sẽ khớp với *abc*, *def*, *ijk*...

PHP định nghĩa sẵn một số lớp hữu dụng nhất để sử dụng khi lập trình. Các lớp này sử dụng cú pháp `[[:name:]]`. Ví dụ, lớp `[[:alpha:]]` khớp với các chữ cái và tương đương với "`[a-zA-Z]`".

Bằng cách định nghĩa các lớp của riêng mình và sử dụng các lớp được định nghĩa sẵn trong PHP, chúng ta có thể định nghĩa ra các mẫu so khớp tốt hơn cho các biểu thức quy tắc (xem bảng 8.3).

Bảng 8.3: *Các lớp ký tự là các công cụ linh động để định nghĩa mẫu so khớp.*

Lớp	Ý nghĩa
`[a-z]`	Chữ cái bất kỳ ở dạng in thường.
`[A-Z]`	Chữ cái bất kỳ ở dạng in hoa.
`[a-zA-Z]`	Chữ cái bất kỳ.
`[0-9]`	Chữ số bất kỳ.
`[ \f\r\t\n\v]`	Khoảng trắng bất kỳ.
`[aeiou]`	Nguyên âm bất kỳ.
`[[:alnum:]]`	Chữ cái hoặc chữ số bất kỳ.
`[[:alpha:]]`	Chữ cái bất kỳ (tương tự như `[a-zA-Z]`).
`[[:blank:]]`	Tab hoặc khoảng trắng.
`[[:digit:]]`	Chữ số bất kỳ (tương tự `[0-9]`).
`[[:lower:]]`	Chữ cái bất kỳ ở dạng in thường.
`[[:upper:]]`	Chữ cái bất kỳ ở dạng in hoa.
`[[:punct:]]`	Ký tự ngắt bất kỳ (.,;:-).
`[[:space:]]`	Khoảng trắng bất kỳ.

Khi sử dụng các dấu ngoặc móc để xác định số ký tự, bạn luôn phải xác định số tối thiểu. Giá trị tối đa là tùy chọn: "`a{3}`" và "`a{3,}`" được chấp nhận, trong khi "`a{,3}`" thì không được chấp nhận.

Để đưa các ký tự đặc biệt (`^.[]$()|*?{}\`) vào trong mẫu, chúng cần phải được mã hóa escape (đặt một dấu chéo ngược phía trước ký tự).

Ký hiệu đô la ($) và dấu chấm (.) không có ý nghĩa gì đặc biệt khi đặt phía trong lớp ký tự.

Phía trong dấu ngoặc vuông, ký hiệu dấu mũ (^) được sử dụng để loại trừ một ký tự. Vì thế "`[^aeiou]`" sẽ khớp với bất kỳ ký tự nào không phải nguyên âm.

Để so khớp với một từ bất kỳ không sử dụng dấu ngắt, bạn nên sử dụng "`^[[:alpha:]]+$`" (nghĩa là, từ phải bắt đầu và kết thúc chỉ với các chữ cái).

So khớp mẫu

Có hai hàm được cài sẵn trong PHP cho mục đích so khớp một mẫu với một chuỗi: **ereg()** và **eregi()** (các hàm tương thích PERL sẽ sử dụng **preg_match()**). Khác biệt duy nhất giữa hai hàm này là hàm **ereg()** có phân biệt dạng chữ trong các mẫu, trong khi hàm **eregi()** lại không phân biệt. Hàm sau thường được dùng cho các mục đích nói chung (không phân biệt dạng chữ), còn hàm đầu chỉ dùng trong các trường hợp cần phân biệt dạng chữ (như so khớp mật khẩu). Cả hai hàm sẽ trả lại giá trị TRUE nếu mẫu trùng khớp và FALSE nếu ngược lại. Dưới đây là hai cách sử dụng của hai hàm này:

```
ereg("pattern","string");
```

hoặc:

```
$pattern = "pattern";
$string = "string";
eregi($pattern, $string);
```

Phương pháp thứ hai dễ sử dụng hơn, nhưng phương pháp thứ nhất lại giúp tiết kiệm được một hoặc hai bước. Nếu thấy các ví dụ dưới đây phức tạp, bạn nên bắt đầu bằng việc đặt các mẫu so khớp trong các biến.

Thực hành so khớp một mẫu theo các bước sau:

1. Tạo một hồ sơ PHP mới trong trình soạn thảo văn bản.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
→ Transitional//EN" "http://www.w3.org/TR/2000/REC-xhtml1-
→20000126/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
lang="en">
<head>
<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />
<title>Gui URL</title>
</head>
<body>
<?php # Doan ma 8.4 - handle_submit_url.php
```

Mã kịch bản này sẽ nhận dữ liệu của biểu mẫu từ trang **submit_url.html** (tham khảo đoạn mã 8.3).

2. Tạo các biến kiểm tra lỗi.

```
$message = '<font color="red">Da xay ra cac loi sau:<br />';
$problem = FALSE;
```

Biến **$message** được dùng để chứa thông báo lỗi nếu có. Biến **$problem** (cũng tương tự như biến được sử dụng trong JavaScript) được dùng để kiểm tra xem có vấn đề gì không.

3. Kiểm tra tên được gửi lên.

```
if (!eregi ("^[[:alpha:].' -]{4,}$",
→ stripslashes(trim($_POST['name'])))) {
   $problem = TRUE;
   $message .= '<p>Hay nhap vao ten hop le.</p>';
}
```

Điều kiện này kiểm tra tên được gửi lên với một mẫu so khớp. Nếu giá trị gửi lên không đáp ứng được yêu cầu của biểu thức quy tắc, biến **$problem** sẽ có giá trị TRUE.

Mẫu trong điều kiện trên là một lớp bao gồm **[:alpha:]** (tất cả các chữ cái), dấu chấm, dấu móc lửng ('), khoảng trắng và dấu gạch ngang. Mẫu này yêu cầu tên chỉ được phép chứa các ký tự trên và phải có ít nhất bốn ký tự.

Tất cả các giá trị nhập sẽ được loại bỏ các dấu chéo và khoảng trắng trước và sau chuỗi.

4. Kiểm tra địa chỉ thư điện tử.

```
if (!eregi ("^[[:alnum:]][a-z0-9_.-]*@[a-z0-9.-]+\.[a-z]
→{2,4}$", stripslashes(trim($_POST['email'])))) {
   $problem = TRUE;
   $message .= '<p>Hay nhap vao dia chi thu dien tu hop
   → le.</p>';
}
```

Địa chỉ thư điện tử và URL là các thành phần khó kiểm tra với độ chính xác tuyệt đối. Mẫu so khớp được sử dụng ở đây yêu cầu địa chỉ thư điện tử phải bắt đầu với một chữ cái hoặc một chữ số, sau đó tiếp tục với tổ hợp của các chữ cái, số, dấu gạch dưới, dấu chấm và dấu gạch ngang. Địa chỉ thư điện tử phải có một ký tự @, tiếp theo sau bởi một tổ hợp các chữ, số, dấu gạch dưới, dấu chấm và dấu gạch ngang. Cuối cùng, phải có một dấu chấm, tiếp theo sau bởi một chuỗi từ hai đến bốn ký tự (.com, .edu, .uk, .vn...).

5. Kiểm tra URL.

```
if (!eregi ("^((http|https|ftp)://)?([[:alnum:]-])+
→ (\.)([[:alnum:]]){2,4}([[:alnum:]/+=%&_.~?-]*)$",
→stripslashes(trim($_POST['url'])))) {
   $problem = TRUE;
   $message .= '<p>Hay nhap vao URL hop le.</p>';
}
```

Với URL, trước hết chúng ta kiểm tra chuỗi http://, https://, ftp://. Sau đó, là các chữ cái, chữ số hoặc dấu chéo, tiếp theo sau bởi một dấu chấm, tiếp đến là một chuỗi từ hai đến bốn chữ cái (com, edu, vn, info...). Cuối cùng, cho phép URL có thể có một số ký tự khác để xác định một tập tin cụ thể, các tham số được gửi theo URL...

6. Kiểm tra nhóm loại URL.

```
if (!isset ($_POST['url_category']) OR
→ !is_numeric ($_POST['url_category'])) {
   $message .= '<p>Hay chon mot nhom URL hop le.</p>';
   $problem = TRUE;
}
```

Vì **url_category** được chọn từ menu thả xuống và phải là một con số, chúng ta có thể kiểm tra nó mà không cần sử dụng biểu thức quy tắc.

7. Tạo phần điều kiện dựa vào kết quả của các bước kiểm tra trên.

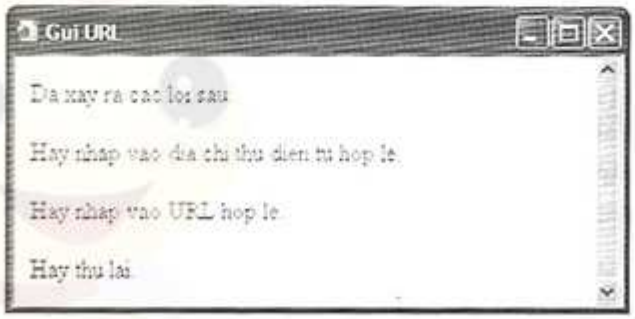
```
if (!$problem) {
   echo '<p>Cam on ban da gui URL.</p>';
} else {
   echo $message;
   echo '</font><p>Hay thu lai.</p>';
}
```

Nếu không có vấn đề gì xảy ra, một lời cảm ơn sẽ được hiển thị (trong Chương 11, thông tin này sẽ được chứa trong cơ sở dữ liệu). Nếu có bất kỳ một vấn đề gì, một thông báo lỗi sẽ xuất hiện.

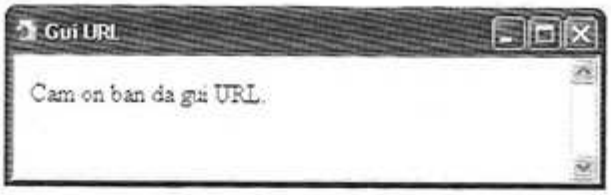
8. Hoàn tất mã lệnh PHP và kết thúc trang HTML.

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

9. Lưu tập tin với tên gọi **handle_submit_url.php**, tải nó lên máy chủ (vào trong cùng thư mục với tập tin **submit_url.html**) và chạy thử trong trình duyệt (xem hình 8-10 và 8-11). Mã lệnh dầy đủ của tập tin được thể hiện trong đoạn mã 8.4.



Hình 8-10: *Nếu có một dữ liệu nào đó không phù hợp với biểu thức quy tắc tương ứng, một thông báo lỗi sẽ được hiển thị.*



Hình 8-11: *Nếu dữ liệu gửi đáp ứng được các điều kiện, một thông điệp cảm ơn sẽ được thể hiện.*

Đoạn mã 8.4: *Mã kịch bản này xử lý biểu mẫu bằng cách sử dụng các biểu thức quy tắc để kiểm tra dữ liệu gửi lên.*

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
→ Transitional//EN" "http://www.w3.org/TR/2000/REC-xhtml1-
→ 20000126/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">
<head>
<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />
<title>Gui URL</title>
</head>
<body>
<?php # Doan ma 8.4 - handle_submit_url.php
$message = '<font color="red">Da xay ra cac loi sau:<br />';
$problem = FALSE;
if (!eregi ("^[[:alpha:].' -]{4,}$",
→ stripslashes(trim($_POST['name'])))) {
   $problem = TRUE;
   $message .= '<p>Hay nhap vao ten hop le.</p>';
}
if (!eregi ("^[[:alnum:]][a-z0-9_.-]*@[a-z0-9.-]+\.
→ [a-z]{2,4}$", stripslashes(trim($_POST['email'])))) {
   $problem = TRUE;
   $message .= '<p>Hay nhap vao dia chi thu dien tu hop
   → le.</p>';
}

if (!eregi ("^((http|https|ftp)://)?([[:alnum:]-])+
→ (\.)([[:alnum:]]){2,4}([[:alnum:]/+=%&_.~?-]*)$",
→ stripslashes(trim($_POST['url'])))) {
   $problem = TRUE;
   $message .= '<p>Hay nhap vao URL hop le.</p>';
```

```
}
if (!isset ($_POST['url_category']) OR
→!is_numeric ($_POST['url_category'])) {
   $message .= '<p>Hay chon mot nhom URL hop le.</p>';
   $problem = TRUE;
}
if (!$problem) {
   echo '<p>Cam on ban da gui URL.</p>';
} else {
   echo $message;
   echo '</font><p>Hay thu lai.</p>';
}
?>
</body>
</html>
```

Chú ý, các biểu thức quy tắc trong PHP phân biệt dạng chữ theo mặc định. Hàm ***ereg()*** *sẽ bỏ qua đặc điểm này.*

Nếu đang tìm kiếm một chuỗi con trong một chuỗi khác, bạn nên dùng hàm ***strstr()*** *vì nó nhanh hơn biểu thức quy tắc. Trong thực tế, bạn chỉ nên sử dụng biểu thức quy tắc nếu việc cần làm không thể thực hiện bằng một hàm hoặc một kỹ thuật nào khác.*

So khớp và thay thế mẫu

Trong khi các hàm `ereg()` và `eregi()` phù hợp cho việc kiểm tra một chuỗi, bạn có thể đưa khả năng lập trình của mình tiến thêm một bước nữa bằng cách so khớp một mẫu và thay thế nó bằng một mẫu hoặc một văn bản khác. Cú pháp để thực hiện điều này như sau:

```
ereg_replace("pattern","replace","string");
```

hoặc:

```
$pattern = "pattern";
$replace = "replace";
$string = "string";
ereg_replace($pattern, $replace, $string);
```

Một ví dụ sử dụng hàm này là chuyển URL mà người dùng nhập vào thành liên kết HTML có thể nhấp chuột, bằng cách đặt nó phía trong thẻ `<a href="url"></a>`. Hàm `ereg_replace()` có phân biệt dạng chữ, trong khi hàm `eregi_replace()` thì không.

Có một khái niệm cần để cập đến với hai hàm này: Tham chiếu ngược (back referencing).

Trong mẫu so khớp mã ZIP (`"^([0-9]{5})(-[0-9]{4})?$"`), có hai nhóm được đặt trong các dấu ngoặc (nhóm đầu biểu thị cho năm con số bắt đầu và nhóm thứ

hai biểu thị cho phần mở rộng tùy chọn, gồm một dấu gạch và bốn chữ số). Trong một biểu thức quy tắc, PHP sẽ tự động đánh số các nhóm được đặt trong dấu ngoặc, bắt đầu bằng 1. Tham chiếu ngược cho phép tham chiếu đến từng phần cụ thể bằng cách sử dụng hai dấu chéo ngược (\\) phía trước con số tương ứng. Ví dụ, nếu so khớp một mã ZIP 94710-0001 với mẫu so khớp này, tham chiếu ngược **\\1** sẽ ứng với 94710. Mã **\\0** tham chiếu đến toàn bộ chuỗi ban đầu.

Thực hành so khớp và thay thế mẫu theo các bước sau:

1. Mở tập tin **handle_submit_url.php** (tham khảo lại đoạn mã 8.4) trong trình soạn thảo văn bản.
2. Bổ sung đoạn mã sau vào phần kiểm tra địa chỉ thư điện tử.

```
} else {
   $email = eregi_replace ("^[[:alnum:]][a-z0-9_.-]*@
   → [a-z0-9.-]+\.[a-z]{2,4}$",'<a href="mailto:\\0">
   → Email</a>',stripslashes(trim($_POST['email'])));
```

Nếu địa chỉ thư điện tử đáp ứng được biểu thức quy tắc, chúng ta sẽ sử dụng hàm **ereg_replace()** bằng cách dùng lại cùng biểu thức quy tắc. Hàm này sẽ chuyển địa chỉ thư điện tử (ví dụ *email@domain.com*) thành mã HTML:

```
<a href="mailto:email@domain.com">email@domain.com</a>.
```

3. Thay thế phần kiểm tra URL bằng khối lệnh sau:

```
if (eregi ("^((http|https|ftp)://)([[:alnum:]-])+
→ (\.)([[:alnum:]-]){2,4}([[:alnum:]/+=%&_.~?-]*)$",
→ stripslashes(trim($_POST['url']))) {
   $url = eregi_replace ("^((http|https|ftp)://)
   → ([[:alnum:]-])+(\.)([[:alnum:]-]){2,4}
   → ([[:alnum:]/+=%&_.~?-]*)$", '<a href="\\0">
   →\\0</a>',stripslashes(trim($_POST['url'])));
} elseif (eregi ("^([[:alnum:]-])+(\.)([[:alnum:]-]){2,4}
→ ([[:alnum:]/+=%&_.~?-]*)$",
→ stripslashes(trim($_POST['url']))) {
   $url = eregi_replace ("^([[:alnum:]-])+(\.)
   → ([[:alnum:]-]){2,4}([[:alnum:]/+=%&_.~?-]*)$",
   →'<a href="http://\\0">\\0</a>',
   →stripslashes(trim($_POST['url'])));
} else {
   $problem = TRUE;
   $message .= '<p>Hay nhap vao URL hop le.</p>';
}
```

Phần mở rộng này phức tạp hơn so với ví dụ trước. Ở đây, chúng ta kiểm tra xem chuỗi có bắt đầu bằng http://, https:// hay ftp:// hay không. Nếu có, (URL đáp ứng toàn bộ biểu thức quy tắc) thì toàn bộ URL sẽ được sử dụng để tạo liên kết HTML.

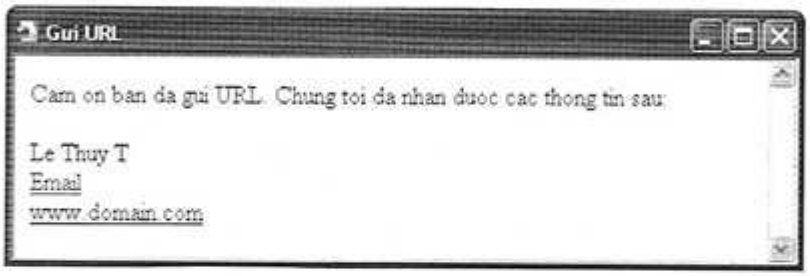
Nếu chuỗi bắt đầu không phải là một trong các giá trị trên, thì liên kết HTML sẽ tự động được thêm vào, tiếp theo là giá trị URL được gửi lên.

4. Thay đổi điều kiện **if($problem)...** để phần đầu của nó như sau:

```
echo "<p>Cam on ban da gui URL. Chung toi da nhan duoc cac
→ thong tin sau:</p>\n{$_POST['name']}<br />\n$email<br />
→ \n$url";
```

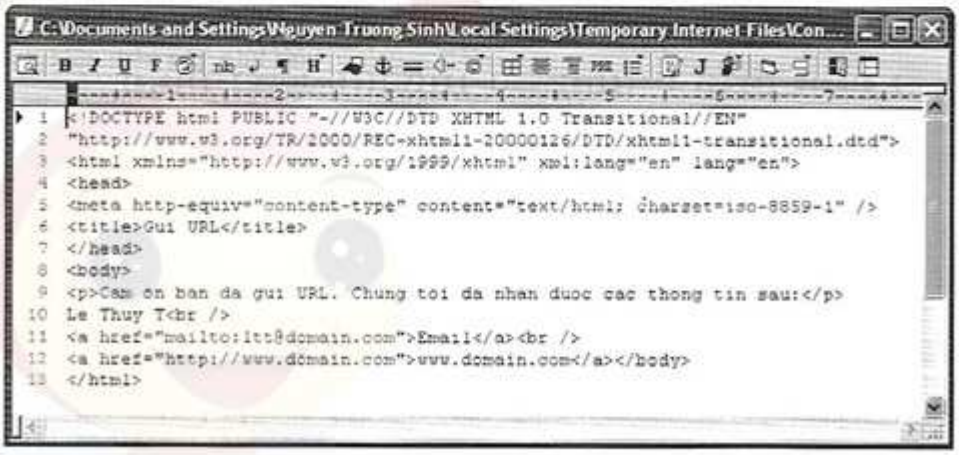
Thông điệp cảm ơn giờ đây sẽ in lại các giá trị được gửi, đồng thời định dạng lại địa chỉ thư điện tử và URL.

5. Lưu lại tập tin, tải nó lên máy chủ Web và chạy thử trong trình duyệt (xem hình 8-12). Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 8.5.



Hình 8-12: *Biểu mẫu giờ đây sẽ in ra các giá trị đã được gửi lên, đồng thời chỉnh sửa địa chỉ thư điện tử và URL để tạo các liên kết.*

6. Kiểm tra mã nguồn của trang Web để xem kết quả của hàm **ereg_replace()**. Xem hình 8-13.



Hình 8-13: *Mã nguồn HTML của trang cho biết các thay đổi đã được thực hiện với các dữ liệu được gửi lên.*

Đoạn mã 8.5: *Phiên bản sửa đổi này của mã kịch bản* ***handle_submit_url.php*** *sử dụng hàm* ***ereg_replace()*** *để tạo ra các chuỗi mới dựa trên các mẫu được so khớp.*

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
→ Transitional//EN" "http://www.w3.org/TR/2000/REC-xhtml1-
→20000126/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">
<head>
<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />
<title>Gui URL</title>
</head>
<body>
<?php # Doan ma 8.5 - handle_submit_url.php
$message = '<font color="red">Da xay ra cac loi sau:<br />';
$problem = FALSE;
if (!eregi ("^[[:alpha:].' -]{4,}$",
→ stripslashes(trim($_POST['name'])))) {
   $problem = TRUE;
   $message .= '<p>Hay nhap vao ten hop le.</p>';
}
if (!eregi ("^[[:alnum:]][a-z0-9_.-]*@[a-z0-9.-]+\.
→ [a-z]{2,4}$", stripslashes(trim($_POST['email'])))) {
   $problem = TRUE;
   $message .= '<p>Hay nhap vao dia chi thu dien tu hop
   → le.</p>';
} else {
   $email = eregi_replace ("^[[:alnum:]][a-z0-9_.-]*@
   → [a-z0-9.-]+\.[a-z]{2,4}$", '<a href="mailto:\\0">
   →Email</a>',stripslashes(trim($_POST['email'])));
}
if (eregi ("^((http|https|ftp)://)([[:alnum:]-])+
→ (\.)([[:alnum:]-]){2,4}([[:alnum:]/+=%&_.~?-]*)$",
→ stripslashes(trim($_POST['url'])))) {
   $url = eregi_replace ("^((http|https|ftp)://)([[:alnum:]-])+
   → (\.)([[:alnum:]-]){2,4}([[:alnum:]/+=%&_.~?-]*)$", '<a
   → href="\\0">\\0</a>',stripslashes(trim($_POST['url'])));
} elseif (eregi ("^([[:alnum:]-])+(\.)([[:alnum:]-]){2,4}
→ ([[:alnum:]/+=%&_.~?-]*)$",
```

```
→ stripslashes(trim($_POST['url']))) {
   $url = eregi_replace ("^([[:alnum:]-])+(\.)([[:alnum:]-])
→{2,4}([[:alnum:]/+=%&_.~?-]*)$", '<a href="http://\\0">
→\\0</a>',stripslashes(trim($_POST['url'])));
} else {
   $problem = TRUE;
   $message .= '<p>Hay nhap vao URL hop le.</p>';
}
if (!isset ($_POST['url_category']) OR
→ !is_numeric ($_POST['url_category'])) {
   $message .= '<p>Hay chon mot nhom URL hop le.</p>';
   $problem = TRUE;
}
if (!$problem) {
   echo "<p>Cam on ban da gui URL. Chung toi da nhan duoc cac
   → thong tin sau:</p>\n{$_POST['name']}<br />\n$email
   →<br />\n$url";
} else {
   echo $message;
   echo '</font><p>Hay thu lai.</p>';
}
?>
</body>
</html>
```

Các hàm **ereg()** *và* **eregi()** *trả lại các mẫu được so khớp dưới dạng tham số thứ ba, nghĩa là mã lệnh trong ví dụ này có thể được viết lại bằng cách dùng hai hàm này.*

Hàm **split()** *của PHP làm việc giống như hàm* **explode()** *ở chỗ nó chuyển một chuỗi thành một mảng, nhưng nó cho phép sử dụng các biểu thức quy tắc để định nghĩa phần phân cách.*

Phiên bản tương thích PERL của hàm **ereg_replace()** *là* **preg_replace()**.

An toàn dữ liệu

Để kết thúc chương này, chúng ta sẽ đề cập một chút về các vấn đề an toàn cần được xem xét khi sử dụng và quản trị cơ sở dữ liệu. Sau đó, ta minh họa hai hàm MySQL cuối cùng có thể được sử dụng để mã hóa dữ liệu.

Thực hành an toàn

Nếu có quyền quản trị cơ sở dữ liệu của mình, bạn cần ghi nhớ những điều sau:

- Không cho người dùng không rõ danh tính truy cập cơ sở dữ liệu.
- Luôn bắt buộc phải có mật khẩu khi kết nối MySQL.
- Yêu cầu người dùng xác định địa chỉ máy. Điều này giới hạn nơi người dùng có thể truy xuất cơ sở dữ liệu MySQL (mặc dù nó có thể phiền phức).
- Khi chứa các thông tin nhạy cảm trong một bảng, đặc biệt là mật khẩu, hãy bảo vệ dữ liệu bằng cách sử dụng hàm `PASSWORD()` hoặc `ENCODE()`. Chúng ta sẽ đề cập đến hai hàm này trong phần sau.
- Gán cho mỗi người dùng một quyền truy cập nhất định ở mức tối thiểu.
- Kiểm tra dữ liệu được gửi lên trước khi chèn chúng vào trong cơ sở dữ liệu, như đã được đề cập trong phần đầu của chương này.
- Chỉ cho phép tài khoản root được truy xuất từ máy cục bộ (localhost).
- Xóa cơ sở dữ liệu `test`. Theo mặc định, đây là cơ sở dữ liệu mà ai cũng có thể truy xuất được.
- Xóa các tài khoản không sử dụng.

Phần lớn các vấn đề nêu ra có liên quan đến quyền và nội dung này sẽ được đề cập đến trong Phụ lục A "Cài đặt".

Mã hóa

MySQL có một số hàm mã hóa và một hàm giải mã. Chúng ta đã sử dụng hàm `PASSWORD()` để mã hóa mật khẩu trước khi chứa nó trong cơ sở dữ liệu.

Một hàm khác là `ENCRYPT()`. Giống như hàm `PASSWORD()`, nó cũng mã hóa một chuỗi, nhưng khác với hàm `PASSWORD()` ở chỗ bạn có thể đưa vào tham số salt để làm cho quá trình mã hóa có tính ngẫu nhiên. Ví dụ:

```
INSERT INTO users (username, password) VALUES ('trout',
→ ENCRYPT('password', 'salt'));
```

`ENCRYPT()` sử dụng chức năng `crypt()` của UNIX để tạo một chuỗi mã hóa mà không thể giải mã ngược trở lại. Đây là một đặc điểm an toàn, vì dữ liệu mã hóa không thể lấy ra ở dạng có thể đọc được.

Nếu cần chứa dữ liệu ở dạng mã hóa và sau đó có thể giải mã ngược trở lại, bạn cần sử dụng các hàm `ENCODE()` và `DECODE()`. Các hàm này cũng nhận vào một tham số salt để tạo ngẫu nhiên cho quá trình mã hóa. Ví dụ:

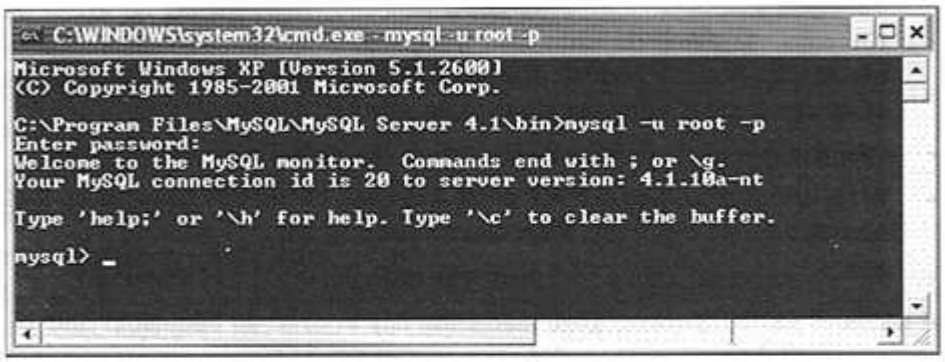
```
INSERT INTO users (username, password) VALUES ('trout',
→ ENCODE('password', 'salt'));
```

```
SELECT DECODE(password, 'salt') AS passwd FROM users
→ WHERE username='trout';
```

Để minh họa, chúng ta sẽ tạo một cơ sở dữ liệu mới và thực hiện một số câu truy vấn trên nó bằng cách sử dụng trình quản lý mysql.

Thực hành mã hóa và giải mã dữ liệu theo các bước sau:

1. Đăng nhập vào trình quản lý mysql (xem hình 8-14).

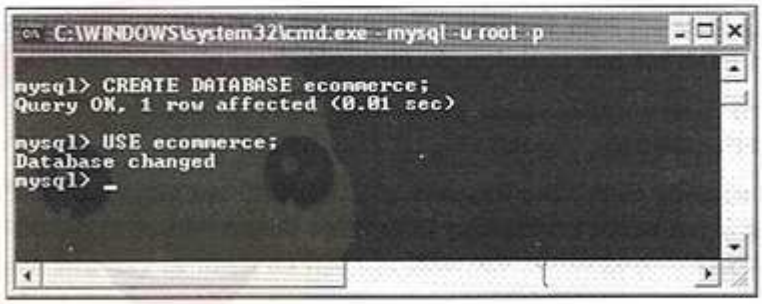


Hình 8-14: *Các ví dụ sau sẽ được thực hiện trong trình quản lý mysql (tham khảo Chương 4 "Giới thiệu về SQL và MySQL").*

```
cd /usr/local/mysql

bin/mysql -u root -p
```

Bạn cũng có thể sử dụng chương trình phpMyAdmin hoặc một công cụ tương tự.

2. Tạo và chọn cơ sở dữ liệu mới (xem hình 8-15).



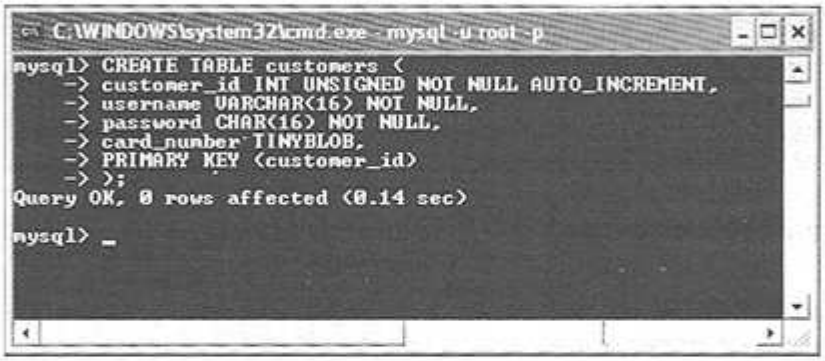
Hình 8-15: *Bước đầu tiên là tạo và chọn cơ sở dữ liệu mới.*

```
CREATE DATABASE ecommerce;

USE ecommerce;
```

Cơ sở dữ liệu *ecommerce* cũng được sử dụng trong Chương 13 "Ví dụ - Thương mại điện tử". Để hoàn thành bước này, bạn cần đăng nhập vào với một tài khoản có quyền tạo cơ sở dữ liệu.

3. Tạo bảng *customers* mới (xem hình 8-16).

```
CREATE TABLE customers (
customer_id INT UNSIGNED NOT NULL AUTO_INCREMENT,
username VARCHAR(16) NOT NULL,
password CHAR(16) NOT NULL,
card_number TINYBLOB,
PRIMARY KEY (customer_id)
);
```



Hình 8-16: *Bảng customers được đưa vào trong cơ sở dữ liệu.*

Bảng *customers* sẽ chứa `customer_id`, `username`, `password`, `card_number`. Trường `password` sẽ chứa một chuỗi mã hóa bằng hàm `PASSWORD()`. Vì hàm `PASSWORD()` luôn tạo ra một chuỗi 16 ký tự (với phiên bản MySQL 4.0), chúng ta sẽ thiết lập chiều dài của trường này là `CHAR(16)`. Trường `card_number` sẽ được mã hóa bằng hàm `ENCODE()`. Hàm này trả lại một giá trị nhị phân nên phải được chứa trong trường có kiểu `BLOB` (hoặc `TINYBLOB`).

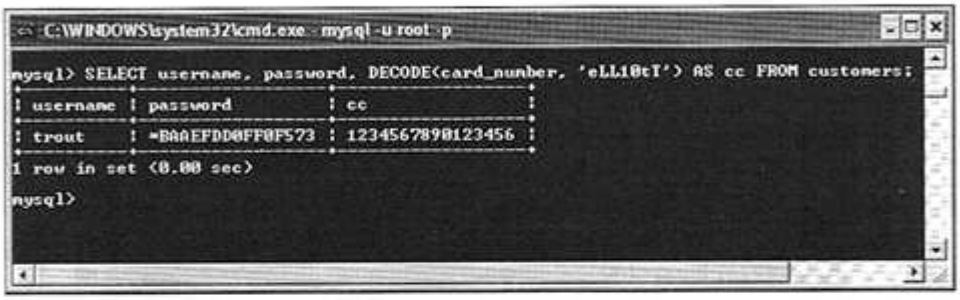
4. Chèn vào một khách hàng mới (xem hình 8-17).

Hình 8-17: *Mẩu tin khách hàng được chèn vào.*

```
INSERT INTO customers (username, password, card_number)
→ VALUES ('trout', PASSWORD('MYpassword'),
→ ENCODE(1234567890123456, 'eLL10tT'));
```

Ở đây, chúng ta đưa một khách hàng mới vào bảng, sử dụng hàm **PASSWORD()** để mã hóa mật khẩu (*MYpassword*) và hàm **ENCODE()** với tham số salt (*eLL10tT*) để mã hóa thẻ tín dụng. Bạn nên sử dụng một giá trị salt duy nhất với các hàm mã hóa của mình.

5. Lấy ra thông tin của khách hàng (xem hình 8-18).



Hình 8-18: *Mẩu tin của khách hàng được lấy ra cùng với việc giải mã thẻ tín dụng.*

```
SELECT username, password, DECODE(card_number, 'eLL10tT')
→ AS cc FROM customers;
```

Câu truy vấn này trả lại tất cả các thông tin gồm: Tên đăng nhập, mật khẩu và thẻ tín dụng của khách hàng. Chú ý, cột **password** không thể giải mã được. Bất kỳ giá trị nào được mã hóa bằng hàm **ENCODE()** đều có thể lấy ra và giải mã bằng hàm **DECODE()**, miễn là sử dụng chung giá trị của tham số salt (ở đây chúng ta dùng *eLL10tT*).

*Khi sử dụng hàm **ENCRYPT()**, **ENCODE()** hoặc **DECODE()** trong mã kịch bản PHP, bạn nên đặt giá trị salt tại một nơi an toàn.*

*Hàm **PASSWORD()** được dùng cho các thông tin không thể xem lại được (như mật khẩu và có thể là cả tên đăng nhập nữa). Sử dụng hàm **ENCODE()** đối với các thông tin cần bảo vệ nhưng lại muốn xem được sau này (như số thẻ tín dụng, số chứng minh nhân dân, địa chỉ...).*

*Hàm **ENCRYPT()** sẽ trả lại giá trị **NULL** nếu nó không có hiệu lực trên hệ điều hành của bạn.*

Để thực hiện mã hóa trong PHP, bạn có thể sử dụng một thư viện ngoài như Mcrypt hoặc một ứng dụng riêng biệt như GnuPG.

Chương 9:

PHÁT TRIỂN ỨNG DỤNG WEB

Giống như hai chương trước, chương này cũng để cập đến một số chủ đề về phát triển Website và đưa chúng lên một mức cao hơn: Dò lỗi, quản lý lỗi và cải thiện khả năng vận hành.

Khi phát triển ứng dụng Web với PHP và MySQL, bạn sẽ gặp phải lỗi về HTML, lỗi lập trình PHP, lỗi SQL hoặc lỗi chính tả MySQL. Để khắc phục lỗi, trước hết chúng ta phải xác định nguyên nhân chính gây ra chúng. Phần đầu của chương sẽ đưa ra các nguyên lý dò lỗi PHP và MySQL. Hai phần quản lý lỗi (một cho PHP và một cho MySQL) sẽ chỉ cho bạn một số cách để đưa ứng dụng từ dạng phát triển sang dạng sản phẩm. Phần cuối của chương sẽ nêu chi tiết những chính sách và thủ tục mà bạn có thể sử dụng để cải thiện khả năng vận hành của Website.

Các kỹ thuật dò lỗi PHP

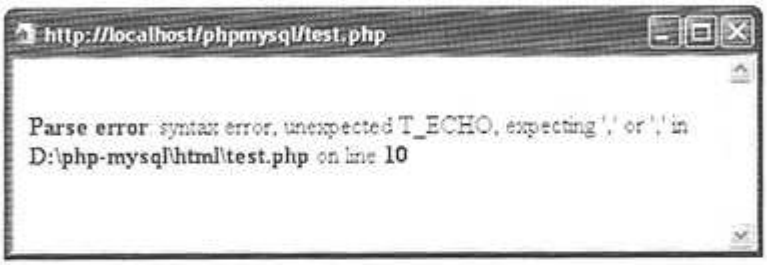
Để dò lỗi một mã kịch bản PHP (ít nhất là dò lỗi một cách hiệu quả), bạn cần phải hiểu rõ các nguyên nhân gây lỗi thông thường. Bảng 9.1 liệt kê các lỗi thường xảy ra nhất với mã kịch bản PHP.

Bảng 9.1: *Một số dạng lỗi phổ biến trong PHP, cùng với các nguyên nhân gây ra lỗi.*

Lỗi	Nguyên nhân
Lỗi biên dịch	Thiếu dấu chấm phẩy, thiếu dấu ngoặc, thiếu dấu nháy hoặc sử dụng ký tự nháy không được mã hóa escape trong chuỗi.
Giá trị của biến rỗng	Quên phần tiền tố (`$`) của biến, sai tên biến hoặc dạng chữ dùng trong tên biến, phạm vi biến không thích hợp, thiết lập `register_globals` bị tắt.
Biến không xác định	Sử dụng biến trước khi nó được gán một giá trị.
Gọi hàm không xác định	Sai tên hàm, hàm được định nghĩa sau khi nó được sử dụng (PHP3) hoặc hồ sơ PHP chứa định nghĩa hàm chưa được kèm theo.
Không thể khai báo lại hàm	Hồ sơ chứa định nghĩa hàm được đưa vào hai lần.
Phần đầu trang đã được gửi	Khoảng trắng tồn tại trong mã kịch bản phía trước thẻ PHP, dữ liệu đã được in ra hoặc một tập tin kèm theo đã được đưa vào.

Dạng lỗi đầu tiên và phổ biến nhất là lỗi về cú pháp. Loại lỗi này sẽ ngăn không cho mã kịch bản hoạt động và dẫn đến các thông báo như trong hình 9-1. Để tránh gặp phải chúng khi lập trình, bạn cần đảm bảo:

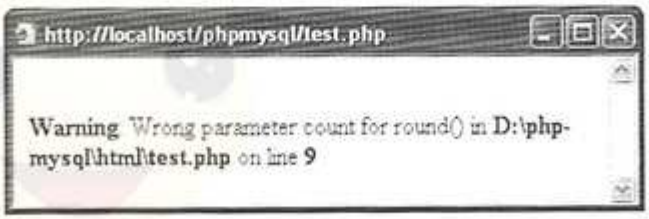
- Kết thúc từng câu lệnh (không áp dụng cho các kết cấu ngôn ngữ như vòng lặp, điều kiện) với dấu chấm phẩy.
- Điền đủ dấu nháy, dấu ngoặc (phải có phần mở và phần đóng).
- Nhất quán trong việc sử dụng dấu nháy (dấu nháy đơn mở phải được kết thúc với dấu nháy đơn đóng).
- Mã hóa escape (sử dụng dấu chéo ngược) cho tất cả các ký tự nháy đơn và nháy kép trong lệnh `print()`.



Hình 9-1: *Các lỗi biên dịch là dạng lỗi phổ biến, đặc biệt đối với các nhà lập trình mới.*

Chú ý, đôi khi thông báo cho biết lỗi xảy ra trên một dòng nào đó, nhưng thực tế nó lại nằm trên một dòng khác. Điều này là do quan niệm khác nhau về số dòng khi PHP thông báo với số dòng thể hiện trong trình soạn thảo. Vì thế, bạn nên xem số dòng gây lỗi trong thông báo của PHP như là một thông tin tham khảo, thay vì xem đó là một con số tuyệt đối.

Dạng lỗi thứ hai cũng thường gặp là sử dụng hàm sai. Ví dụ, lỗi xảy ra khi gọi hàm `setcookie()` hoặc `header()` sau khi trình duyệt đã nhận được mã lệnh HTML hoặc khi một hàm được gọi mà không cung cấp đủ tham số. Dạng lỗi này sẽ được phát hiện bởi PHP khi chạy thử mã lệnh (xem hình 9-2).



Hình 9-2: *Việc sử dụng hàm sai (gọi nó không đúng lúc hoặc sai tham số) sẽ gây lỗi trong quá trình thực hiện.*

Dạng lỗi thứ ba không liên quan đến PHP, mà là lỗi của nhà lập trình hay lỗi lôgic (thật ra, tất cả các lỗi đều do nhà lập trình). Một trong những lý do làm phát sinh lỗi này là dùng sai tên biến. Trong trường hợp này, chúng ta sẽ không thấy các

thông báo lỗi như trong các hình trên, mà sẽ thấy các kết quả lạ hoặc không thể đoán được. Đây là một trong số các dạng lỗi khó phát hiện nhất, vì PHP không thể cung cấp một manh mối nào cho biết lỗi đó ở đâu và làm sao để giải quyết.

Để khắc phục lỗi, chúng ta cần xác định lỗi nào đã xảy ra và ở đâu. Có một số cách để xác định lỗi:

- Xem mã nguồn HTML.

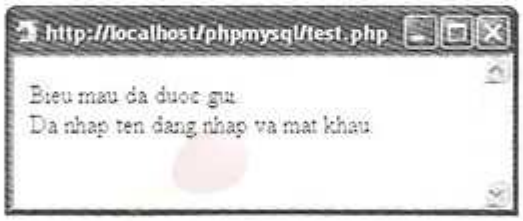
Một số trình duyệt sẽ thể hiện trang trắng khi có lỗi PHP. Việc xem mã nguồn có thể là cách duy nhất để xem vấn đề gì đã xảy ra: Lỗi hoàn toàn về PHP hay chỉ là HTML.

- Sử dụng ghi chú.

Cũng giống như việc sử dụng ghi chú cho mã kịch bản, chúng ta có thể sử dụng ghi chú mã lệnh để tìm ra dòng gây lỗi. Nếu PHP thông báo lỗi nằm trên dòng 12, bạn nên đánh dấu ghi chú dòng lệnh này xem có hết lỗi không. Nếu không, chúng ta biết rằng lỗi xảy ra trên một dòng khác.

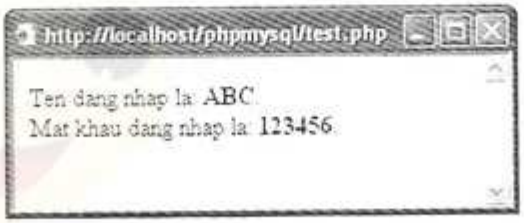
- Gọi hàm `print()` và `echo()`.

Trong các mã kịch bản phức tạp, bạn nên sử dụng hàm `print()` để in ra các thông báo nhằm biết được những gì đang xảy ra khi thực hiện mã kịch bản (xem hình 9-3). Khi mã kịch bản có một số bước, rất khó để biết được vấn đề xảy ra trong bước nào. Bằng cách dùng hàm `print()`, ta có thể thu hẹp phạm vi gây ra vấn đề.



Hình 9-3: *Việc dò lỗi có thể được thực hiện bằng cách in các thông báo cho biết mã lệnh đang thực hiện đến đâu.*

- Theo dõi biến (xem hình 9-4).



Hình 9-4: *Việc sử dụng hàm `print()`, `echo()`, `print_r()` hoặc `var_dump()` là cách dễ nhất để theo dõi giá trị của các biến trong suốt quá trình thực hiện mã kịch bản.*

Rất dễ để làm cho một mã kịch bản không hoạt động chỉ vì tham chiếu sai biến, đúng biến nhưng sai tên hoặc biến không có đúng giá trị mong đợi. Để kiểm tra các khả năng này, bạn nên sử dụng hàm **print()** hoặc **echo()** để in giá trị của biến tại các điểm quan trọng trong mã kịch bản.

Đối với các biến phức tạp hơn như mảng hoặc đối tượng, hàm **print_r()** và **var_dump()** sẽ in ra giá trị của chúng mà không cần dùng tới vòng lặp. Cả hai hàm đều thực hiện một nhiệm vụ, tuy nhiên hàm **var_dump()** sẽ thể hiện thông tin chi tiết hơn hàm **print_r()**.

Nhiều trình soạn thảo có kèm theo tiện ích để kiểm tra tính cân xứng của các dấu ngoặc và dấu nháy.

*Nếu không thể tìm thấy lỗi biên dịch trong một mã kịch bản phức tạp, bạn nên bắt đầu bằng việc ghi chú toàn bộ mã lệnh PHP (sử dụng /**/). Sau đó, dần dần thu hẹp phần ghi chú (bằng cách di chuyển phần mở hoặc đóng của nó) rồi chạy lại mã kịch bản cho đến khi xác định được dòng gây lỗi.*

Một số lập trình viên sử dụng phương pháp kiểm tra ngược để giảm thiểu sự hiện diện của các lỗi lôgic. Kiểm tra ngược là chuyển đổi thứ tự trong điều kiện. Ví dụ:

```
if($variable = 5){...
```

Dòng lệnh này sẽ vẫn gây ra vấn đề vì điều kiện luôn luôn đúng. Việc thay đổi điều kiện thành:

```
if(5 = $variable) {...
```

sẽ ngay lập tức phát hiện ra lỗi vì không thể gán một giá trị cho hằng số 5 được.

*Để kết quả in ra bởi hàm **print_r()** hoặc **var_dump()** được dễ đọc hơn trong trình duyệt Web, bạn nên đặt chúng trong phạm vi của thẻ **<pre>**:*

```
echo "<pre>";
print_r($var);
echo "</pre>";
```

Các kỹ thuật dò lỗi SQL và MySQL

Các lỗi SQL phổ biến nhất thường được gây ra bởi:

- Sử dụng thiếu cân xứng các dấu nháy hoặc dấu ngoặc.
- Không mã hóa escape ký tự nháy trong các giá trị cột.
- Sai chính tả tên cột, tên bảng hoặc hàm.
- Đặt mệnh đề của câu truy vấn không đúng thứ tự.

Ngoài ra, khi sử dụng MySQL, chúng ta còn có thể gặp phải các lỗi như:

- Kết quả truy vấn không như mong đợi.

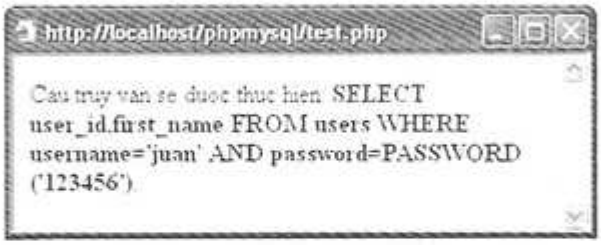
- Không thể truy xuất cơ sở dữ liệu.
- Khó khởi động MySQL.

Khi tạo các ứng dụng bằng PHP, cách tốt nhất để xác định lỗi SQL hoặc MySQL là sử dụng hàm **mysql_error()**, như đã được giới thiệu trong Chương 6 "Sử dụng PHP và MySQL". Phần tiếp theo sẽ giới thiệu một phương pháp khác để giải quyết các lỗi SQL và MySQL. Tiếp đến, chúng ta cùng xem xét chủ đề về truy xuất.

Dò lỗi các vấn đề về SQL

Thực hành dò lỗi câu truy vấn SQL theo các bước sau:

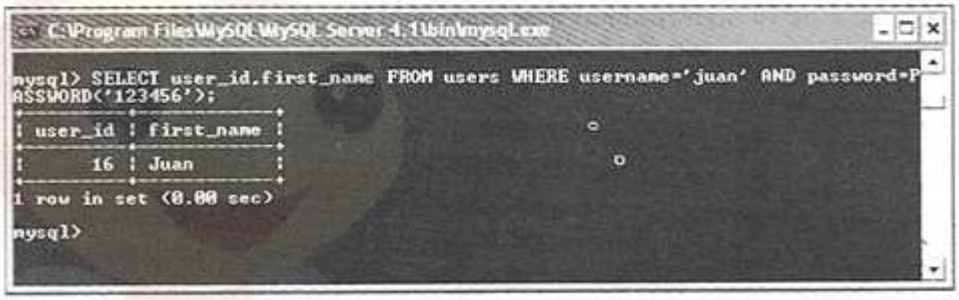
- In ra những câu truy vấn có vấn đề trong mã kịch bản (xem hình 9-5).



Hình 9-5: *Việc biết chính xác câu truy vấn mà mã kịch bản thực hiện là bước đầu tiên để giải quyết các vấn đề SQL và MySQL.*

Bằng cách dùng mã lệnh **echo $query** (hoặc một dạng tương đương) trong mã kịch bản PHP, chúng ta có thể in ra trình duyệt chính xác câu truy vấn sẽ được thực hiện. Đôi khi chỉ điều này thôi cũng đã giúp xác định được vấn đề.

- Thực hiện câu truy vấn trong trình quản lý mysql hoặc một công cụ khác (xem hình 9-6).



Hình 9-6: *Để xem kết quả mà mã kịch bản sẽ nhận được là gì, hãy thực hiện câu truy vấn trong một công cụ khác.*

Phương pháp hiệu quả nhất để xác định lỗi SQL hoặc MySQL là thực hiện cùng câu truy vấn được sử dụng với mã kịch bản PHP trong một ứng dụng độc lập: Trình quản lý mysql, phpMyAdmin... Việc này sẽ cung cấp cho ta các kết quả giống với những gì mà mã kịch bản PHP sẽ nhận được.

Nếu ứng dụng độc lập trả lại kết quả đúng như mong đợi trong khi mã kịch bản vẫn cho ra kết quả không đúng, khi đó chúng ta biết rằng lỗi nằm ở bản thân mã kịch bản chứ không phải trong câu lệnh SQL hoặc cơ sở dữ liệu MySQL.

- Sử dụng lệnh **EXPLAIN** để xem MySQL xử lý câu truy vấn ra sao.

Ở cuối chương này, chúng ta sẽ để cập đến câu lệnh **EXPLAIN** chi tiết hơn và tìm hiểu cách dùng nó để cải thiện kết quả cho câu truy vấn.

- Viết lại câu truy vấn ở dạng cơ bản nhất, sau đó bổ sung từ từ các thành phần cho đến khi phát hiện ra vấn đề.

Đôi khi, rất khó để dò lỗi một câu truy vấn vì có quá nhiều thứ cùng diễn ra. Cũng giống như ghi chú mã lệnh trong mã kịch bản PHP, việc thu gọn câu truy vấn xuống mức tối thiểu, sau đó mở rộng dần ra, cũng là một cách dễ dàng để dò lỗi một câu truy vấn phức tạp.

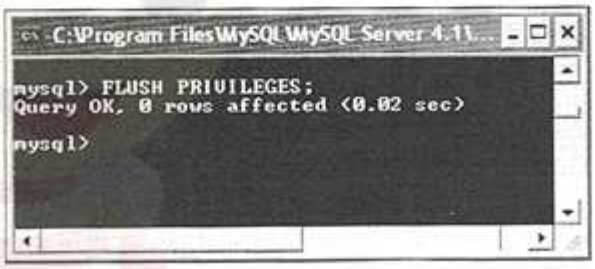
Thay vì in câu truy vấn ra trình duyệt, bạn có thể in nó ở dạng ghi chú HTML (chỉ xem được ở dạng mã nguồn HTML), bằng cách dùng mã lệnh `echo "<!-- $query -->";`.

Trong giai đoạn phát triển của một site, bạn nên sử dụng định dạng `OR die (mysql_error() . ": $query")` *để quản lý lỗi MySQL. Theo cách này, nếu có một lỗi xảy ra, mã kịch bản sẽ ngừng lại và lỗi MySQL sẽ được in ra.*

Dò các lỗi liên quan đến truy xuất

Các thông báo lỗi từ chối truy xuất cũng là dạng phổ biến mà các nhà phát triển mới thường gặp khi sử dụng PHP để tương tác với MySQL. Các giải pháp cho vấn đề này bao gồm:

- Tải lại MySQL sau khi thay đổi thiết lập quyền để chúng có tác dụng. Sử dụng công cụ MySQLadmin hoặc chạy lệnh **FLUSH PRIVILEGES** trong trình quản lý mysql (xem hình 9-7). Bạn cần phải đăng nhập với tài khoản có quyền thực hiện lệnh này.



Hình 9-7: *Lệnh* `FLUSH PRIVILEGES` *cần được thực hiện để cập nhật các thay đổi.*

- Kiểm tra kỹ mật khẩu sử dụng. Thông báo lỗi *Access denied for user: 'user@localhost' (Using password: YES)* thường là do mật khẩu bị sai. Đây không phải là lỗi phổ biến, tuy nhiên là nguyên nhân cần phải kiểm tra đầu tiên.
- Đừng quên sử dụng hàm `PASSWORD()` khi thiết lập quyền hoặc cập nhật mật khẩu.
- Thông báo lỗi *Can't connect to...* (lỗi số 2005) cho biết rằng MySQL không chạy hoặc chạy sai cổng.

Một vấn đề phổ biến khác liên quan đến truy cập là quên mật khẩu của tài khoản root. Việc thiết lập lại mật khẩu thường kèm theo việc khởi động lại MySQL mà không kích hoạt hệ thống tài khoản hoặc phân quyền. Khi đó, bất kỳ ai cũng có thể truy xuất cơ sở dữ liệu, thực hiện các thay đổi cần thiết và khởi động lại MySQL.

Thực hành thiết lập lại mật khẩu của tài khoản root theo các bước sau:

1. Ngừng máy chủ MySQL bằng cách thực hiện lệnh sau trên hệ điều hành UNIX hoặc tương đương:

```
kill -TERM 'cat /path/to/mysql/data/hostname.pid'
```

Bạn sẽ gặp khó khăn trong việc ngừng máy chủ MySQL theo cách thông thường nếu không đăng nhập với tài khoản root của hệ thống. Vì lý do đó, bạn nên sử dụng kỹ thuật có tính ép buộc này.

Trên các hệ thống Windows, với MySQL đang chạy dưới dạng một dịch vụ, bạn có thể ngừng máy chủ bằng cách nhấp vào biểu tượng đèn tín hiệu trên thanh tác vụ (xem hình 9-8) và thực hiện theo các hộp nhắc.

Hình 9-8: *Trên phần lớn bản cài đặt cho Windows NT, bạn có thể quản lý máy chủ MySQL thông qua một biểu tượng đèn tín hiệu trên thanh tác vụ của Window.*

2. Khởi động lại MySQL với `--skip-grant-table`.

```
bin/safe_mysqld --user=mysql --skip-grant-table &
```

Tùy chọn `--skip-grant-table` báo cho MySQL thực hiện mà không cần quan tâm đến người dùng và phân quyền. Đây rõ ràng là một rủi ro rất lớn về an toàn, nhưng lại là cách duy nhất để thiết lập lại mật khẩu cho tài khoản root.

Người dùng Windows cũng có thể thực hiện bước tương tự bằng công cụ WinMySQLadmin.

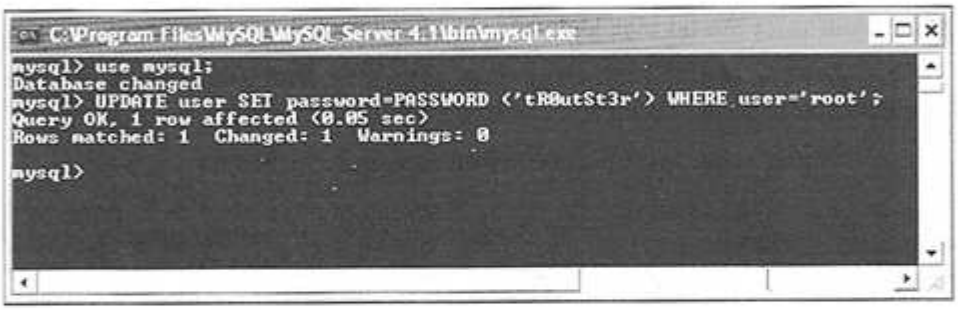
3. Đăng nhập vào trình quản lý mysql và chọn cơ sở dữ liệu MySQL.

```
bin/mysql -u root
USE mysql;
```

Cơ sở dữ liệu mysql (được tạo ra khi thực hiện mã kịch bản **mysql_install_db** trong quá trình cài đặt) được sử dụng để xác định ai có thể truy xuất cơ sở dữ liệu nào, cũng như với quyền nào.

4. Cập nhật mật khẩu của tài khoản root (xem hình 9-9):

```
UPDATE user SET password=PASSWORD('tR0utSt3r')
→ WHERE user='root';
```



Hình 9-9: *Sử dụng câu lệnh* ***UPDATE*** *để thay đổi mật khẩu của tài khoản root (hoặc một tài khoản bất kỳ).*

Câu truy vấn này sẽ cập nhật bảng **user**, nơi chứa các tên đăng nhập và mật khẩu cho tất cả các tài khoản truy xuất MySQL.

5. Thoát khỏi trình quản lý mysql, ngừng máy chủ MySQL và sau đó khởi động lại như bình thường.

```
exit;
bin/mysqladmin -u root shutdown
bin/safe_mysql --user=mysql &
```

Vì MySQL hiện đang chạy mà không quan tâm đến việc phân quyền, bạn cần ngừng và khởi động lại thay vì chỉ thực hiện lệnh **FLUSH PRIVILEGES**.

Người dùng Windows có thể ngừng và khởi động lại máy chủ MySQL bằng công cụ WinMySQLadmin.

Để thiết lập lại mật khẩu của root bằng công cụ MySQLadmin, hãy thực hiện các bước từ 1 đến 2 như trong bài thực hành trên, sau đó nhập

bin/mysqladmin -u root password 'mật khẩu mới'. Cuối cùng, khởi động lại MySQL như bình thường.

Quản lý lỗi PHP

PHP có kèm theo một số hàm hữu ích cho việc quản lý lỗi, cho phép chúng ta xử lý site chuyên nghiệp hơn trong quá trình phát triển.

Với mục đích an toàn và thẩm mỹ, không nên để cho người dùng thấy được các thông báo lỗi, đặc biệt là các thông báo có liên quan đến cơ sở dữ liệu, vì chúng sẽ để lộ ra các thông tin cần phải bảo vệ. Chúng ta đều mong rằng mọi chuyện sẽ hoạt động tốt trong quá trình phát triển, nhưng điều đó rất hiếm. Việc sử dụng các hàm được đề cập ở đây cho phép xử lý các thông báo lỗi của PHP theo một cách thích hợp và an toàn.

Thông báo lỗi

Hàm `error_reporting()` trong PHP dùng để thiết lập loại lỗi nào PHP sẽ thông báo. Hàm nhận vào một con số hoặc một hằng (xem bảng 9.2) - (tài liệu PHP có thể liệt kê một số hằng khác liên quan đến bản thân PHP).

Bảng 9.2: *Các thiết lập thông báo lỗi của PHP được dùng với hàm* ***error_reporting()*** *hoặc tập tin* ***php.ini****.*

Giá trị	Hằng	Thông báo
1	E_ERROR	Lỗi nghiêm trọng lúc chạy (ngừng thực hiện mã kịch bản).
2	E_WARNING	Cảnh báo lúc chạy (lỗi không nghiêm trọng).
4	E_PARSE	Lỗi biên dịch.
8	E_NOTICE	Thông báo (có thể nghiêm trọng hoặc không nghiêm trọng).
256	E_USER_ERROR	Thông báo lỗi phát sinh bởi hàm `trigger_error()`.
512	E_USER_WARNING	Cảnh báo phát sinh bởi hàm `trigger_error()`.
1024	E_USER_NOTICE	Thông báo phát sinh bởi hàm `trigger_error()`.
2047	E_ALL	Tất cả các lỗi và cảnh báo.

Thiết lập 0 sẽ tắt toàn bộ thông báo lỗi (các lỗi vẫn xảy ra, chỉ có điều là chúng ta không còn thấy chúng nữa). Ngược lại, `error_reporting(E_ALL)` sẽ làm cho

PHP thông báo tất cả các lỗi xảy ra. Chúng ta có thể kết hợp các con số để tùy biến mức thông báo lỗi bằng cách dùng các toán tử bit như | (or), ~ (not), & (and).

Thực hành điều chỉnh mức thông báo theo các bước sau:

1. Tạo mã kịch bản PHP trong trình soạn thảo văn bản.

```
<?php # Doan ma 9.1 - config.inc
```

2. Thiết lập thông báo lỗi ở mức cao nhất.

```
error_reporting (E_ALL);
```

Trong quá trình phát triển, bạn nên để cho PHP thông báo tất cả các lỗi và cảnh báo. Dòng lệnh này sẽ thực hiện điều đó.

3. Đóng mã kịch bản PHP và lưu tập tin với tên gọi **config.inc**. Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 9.1.

```
?>
```

4. Để sử dụng tập tin này trong ứng dụng Web, hãy đưa vào dòng lệnh sau ở đầu mỗi mã kịch bản.

```
include ('config.inc');
```

Tập tin cấu hình này sẽ thiết lập thông báo lỗi cho toàn bộ ứng dụng. Việc đưa nó vào đầu mỗi mã kịch bản PHP (thậm chí trước cả các lời gọi **session_start()** sẽ áp dụng các thông báo lỗi cho cả lời gọi này).

Đoạn mã 9.1: *Phiên bản này của tập tin cấu hình thiết lập các thông báo lỗi ở mức cao nhất để phục vụ quá trình phát triển.*

```
<?php # Doan ma 9.1 - config.inc

error_reporting (E_ALL);

?>
```

Sử dụng bộ xử lý lỗi tùy biến

Một tùy chọn khác cho việc quản lý lỗi với các Website (ngoài việc điều chỉnh mức thông báo lỗi) là thay đổi cách xử lý lỗi của PHP. Bạn có thể tạo ra các hàm của riêng mình để thay cho cách xử lý lỗi mặc định của PHP (in ra trong các thẻ HTML). Ví dụ:

```
function ReportErrors($number, $message){
    // Xử lý lỗi
```

```
set_error_handler('ReportErrors');
```

Hàm **set_error_handler()** *xác định hàm sẽ xử lý lỗi, đó có thể là hàm do bạn tự định nghĩa. Việc cung cấp cho hàm* **set_error_handler()** *tên gọi của một hàm khác sẽ làm cho hàm đó (trong ví dụ trên là hàm* **ReportErrors**) *được gọi mỗi khi có một lỗi nào đó xảy ra. Hàm xử lý lỗi (***ReportErrors**) *sẽ nhận vào hai tham số: Số hiệu lỗi và thông báo lỗi ở dạng văn bản.*

Hàm này cũng có thể được viết để nhận vào năm tham số, trong đó ba tham số còn lại lần lượt là tập tin, dòng và ngữ cảnh nơi xảy ra lỗi.

Nếu sử dụng cả hai hàm **error_reporting()** *và* **set_error_handler()**, *các thiết lập của hàm trước sẽ chiếm ưu thế hơn so với hàm sau trong tất cả trường hợp, ngoại trừ các thông báo và cảnh báo.*

Trong phiên bản hiện tại của PHP, thông báo lỗi được thiết lập mặc định là **E_ALL & ~E_NOTICE** *(thông báo tất cả lỗi ngoại trừ các thông báo).*

Chương 6 "Sử dụng PHP và MySQL", đã đề cập đến biểu tượng @ để tắt các thông báo lỗi PHP và hàm **die()** *để kết thúc mã kịch bản. Bạn nên sử dụng @ cho các hàm để khi thực hiện không thành công cũng không ảnh hưởng đến toàn bộ mã kịch bản và sử dụng hàm* **die()** *cho các lời gọi hàm vì tác động của nó có tính quyết định đến mã kịch bản.*

Biểu tượng @ chỉ làm việc với các biểu thức, không thể sử dụng @ trước các điều kiện, vòng lặp, định nghĩa hàm...

Nếu tập tin **php.ini** *kích hoạt thông số* **track_errors** *(thiết lập thành 1 hoặc on), mọi thông báo lỗi sẽ được chứa trong mảng có tên* **$php_errormsg**.

Thiết lập **display_errors** *trong tập tin* **php.ini** *(nó cũng có thể được thiết lập với hàm* **ini_set()**) *cho biết các thông báo lỗi có được gửi xuống trình duyệt hay không.*

Các mã kịch bản được viết trong cuốn sách này cũng được thiết lập với thông báo lỗi ở mức tối đa để đón bắt tất cả các lỗi phát sinh.

Ghi nhận lỗi

Hàm **error_reporting()** báo cho PHP biết những lỗi nào cần thông báo, trong khi đó hàm **error_log()** sẽ hướng dẫn cho PHP làm gì khi các lỗi xảy ra:

```
error_log("message", type [, destination [, extra headers]]);
```

Bảng 9.3 liệt kê các loại ghi nhận lỗi mà bạn có thể dùng. Tham số *destination* có thể là tên của một tập tin (dạng ghi nhận lỗi 3) hoặc một địa chỉ thư điện tử (dạng 1). Tham số *extra headers* chỉ được dùng khi gửi thư điện tử (dạng 1).

Bảng 9.3: *PHP có thể ghi nhận các lỗi bằng cách sử dụng bốn kỹ thuật này.*

Số hiệu	Ý nghĩa
0	Thông báo được ghi nhận bằng cách sử dụng chức năng mặc định của hệ thống.
1	Thông báo được gửi đến một địa chỉ thư điện tử.
2	Thông báo được gửi đến một chương trình dò lỗi (ví dụ, một máy chủ khác). Tùy chọn này có hiệu lực trong phiên bản 3 của PHP nhưng không có trong phiên bản 4.
3	Thông báo được ghi nhận trong một tập tin.

Thực hành ghi nhận lỗi theo các bước sau:

1. Mở tập tin **config.inc** (tham khảo lại đoạn mã 9.1) trong trình soạn thảo văn bản.
2. Thay đổi dòng **error_reporting()** để không có thông báo lỗi nào được in ra.

```
error_reporting (0);
```

Một khi Website đã được đưa lên mạng, bạn cần phải thiết lập để không có thông báo lỗi nào được gửi xuống trình duyệt của người dùng. Việc thay đổi thông báo lỗi thành 0 sẽ thực hiện điều này.

3. Tạo một thông báo lỗi cần được ghi nhận.

```
$message = 'Co mot loi xay ra trong ma kich ban ' . __FILE__ .
→ ' tai dong ' . __LINE__ . "\n";
```

Các hằng **__FILE__** và **__LINE__** nói đến mã kịch bản và số hiệu dòng nơi xảy ra lỗi. Chúng được dùng để tạo các thông báo lỗi cụ thể hơn.

Bạn cũng có thể làm cho thông báo lỗi này có tính thông tin hơn bằng cách bổ sung ngày, giờ hoặc mô tả lỗi.

4. Ghi nhận thông báo lỗi vào một tập tin văn bản.

```
error_log ($message, 3, 'errors.txt');
```

Hàm này sẽ ghi giá trị của biến **$message** vào tập tin **errors.txt** mỗi khi xảy ra lỗi. Nếu muốn chứa tập tin văn bản này trong một thư mục khác, bạn cần sử dụng đường dẫn tuyệt đối (ví dụ: *C:\logs\errors.txt*).

5. Lưu lại tập tin **config.inc**. Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 9.2.
6. Tạo một hồ sơ văn bản có tên **errors.txt** và thiết lập quyền để tài khoản chạy máy chủ Web ghi được tập tin này.

7. Nếu cần thiết, hãy tạo ra một số lỗi và kiểm tra tập tin để xem các thiết lập có hoạt động chính xác không.

Đoạn mã 9.2: *Phiên bản sửa đổi của tập tin cấu hình này sẽ tắt các thông báo lỗi và sử dụng một tập tin văn bản để ghi nhận lỗi.*

```
<?php # Doan ma 9.2 - config.inc

error_reporting (0);

$message = 'Co mot loi xay ra trong ma kich ban ' . __FILE__ →
. ' tai dong ' . __LINE__ . "\n";

error_log ($message, 3, 'errors.txt');

?>
```

Nếu muốn ghi nhận lại các lỗi mỗi khi mã kịch bản không thể kết nối cơ sở dữ liệu (một lỗi nghiêm trọng trong các ứng dụng Web dựa vào cơ sở dữ liệu), bạn có thể sử dụng:

```
mysql_connect(DB_HOST, DB_USER, DB_PASS) or
→ error_log('Could not connect to database.', 1,
→ 'me@domain.com');
```

Thiết lập `log_errors` *trong tập tin* `php.ini` *(cũng có thể được thiết lập thông qua hàm* `ini_set()`*) xác định việc các lỗi mã kịch bản có được ghi nhận hay không. Tài liệu PHP khuyên nên tắt tùy chọn* `display_errors` *và bật* `log_errors` *cho các site khi đưa vào hoạt động.*

Với PHP 4.0.1, hai hàm `trigger_error()` *và* `user_error()` *có thể được dùng để phát sinh thông báo lỗi một cách thủ công (hai hàm này tương đương nhau).*

Quản lý lỗi MySQL

Cũng như việc ghi nhận lỗi trong PHP, chúng ta cũng có thể thực hiện với MySQL. Có hai dạng ghi nhận trong MySQL: Ghi nhận lỗi (error log) để ghi lại các vấn đề và ghi nhận theo dõi (tracking log) để ghi lại thông tin của những lần thay đổi cơ sở dữ liệu.

Ghi nhận lỗi của MySQL sẽ được chứa trong thư mục data và được đặt tên là **hostname.err**. Thông thường, đường dẫn đầy đủ đến tập tin này là `/usr/local/mysql/data/hostname.err` (trên các hệ thống UNIX) hoặc `c:\mysql\data\hostname.err` (trên Windows). Tập tin ghi nhận lỗi sẽ ghi lại từng lỗi phát sinh bởi phần mềm, chủ yếu là các lỗi xảy ra khi khởi động máy chủ MySQL.

Ghi nhận theo dõi (còn gọi là ghi nhận cập nhật - update log) sẽ ghi nhận lại từng thay đổi được thực hiện cho một cơ sở dữ liệu bất kỳ. Nói cách khác, nó sẽ

ghi lại tất cả các câu lệnh SQL đã được thực hiện mà có ảnh hưởng đến cấu trúc hoặc nội dung của một bảng (**INSERT**, **UPDATE**, **DELETE**, **DROP**, **ALTER** và **TRUNCATE**). Dạng ghi nhận này cũng rất cần thiết cho việc thực hiện các sao lưu cơ sở dữ liệu. Bản thân các tập tin ghi nhận có tên gọi hostname001 và tiếp tục với các con số tuần tự được thay đổi mỗi khi máy chủ MySQL khởi động lại (hostname002, hostname003...).

Thực hành sử dụng tập tin ghi nhận thay đổi theo các bước sau:

1. Ngừng MySQL nếu nó đang chạy.

Vì muốn thay đổi cách hoạt động của MySQL (ghi nhận lại các thay đổi được thực hiện với cơ sở dữ liệu), chúng ta cần ngừng và khởi động lại máy chủ MySQL.

2. Khởi động lại máy chủ MySQL với tùy chọn **--log-update**.

```
bin/safe_mysqld --user=mysql --log-update &
```

Tùy chọn **--log-update** báo cho MySQL ghi nhận lại mỗi hành động làm thay đổi cấu trúc hoặc nội dung của một cơ sở dữ liệu. Mặc định, MySQL sẽ không khởi động với tùy chọn này.

Trên Windows, bạn có thể kích hoạt đặc điểm này với công cụ WinMySQLadmin.

3. Đăng nhập vào trình quản lý mysql và thực hiện một thay đổi nào đó với cơ sở dữ liệu.

Bạn cũng có thể thực hiện mã kịch bản PHP được tạo trong cuốn sách này để đưa các mẩu tin vào trong cơ sở dữ liệu.

4. Thoát ra khỏi trình quản lý mysql và xem nội dung của các tập tin ghi nhận thay đổi (xem hình 9-10).

Hình 9-10: *Tập tin ghi nhận lỗi sẽ ghi lại các vấn đề xảy ra.*

Bạn có thể xem các ghi nhận này trong một trình soạn thảo văn bản bất kỳ, bao gồm *vi* (trên hệ thống Unix và Mac OS X), *BBEdit* (Mac OS X) hoặc *Notepad* (Windows).

Không nên quá lạm dụng việc sử dụng ghi nhận các thông tin cập nhật MySQL vì chúng sẽ làm chậm việc xử lý của cơ sở dữ liệu và chiếm nhiều dung lượng ổ đĩa.

Để thiết lập lại trạng thái của một cơ sở dữ liệu từ tập tin ghi nhận cập nhật, hãy chạy nó như một tập tin lệnh:

```
mysql -u root -p < localhost.001
```

Ghi nhận lỗi sẽ ghi lại bất kỳ lỗi nào xảy ra khi khởi động MySQL, đây chính là nơi tốt nhất để kiểm tra các vấn đề ứng với lỗi xảy ra.

Cải thiện khả năng vận hành của ứng dụng Web

Để kết thúc chương này, chúng ta sẽ đề cập đến một số vấn đề liên quan đến khả năng vận hành của ứng dụng Web. Một site càng "bận rộn" thì càng cần được điều chỉnh để nâng cao khả năng vận hành.

Khả năng vận hành của các ứng dụng Web được quyết định bởi nhiều yếu tố:

- Bản thân máy chủ vật lý (máy tính chạy MySQL).
- Mã kịch bản PHP.
- MySQL và SQL.

Cách thức hiển nhiên và cũng tốn kém nhất để cải thiện khả năng vận hành của các ứng dụng Web là nâng cấp bản thân máy chủ: Tốc độ bộ xử lý, bộ nhớ, ổ cứng (loại, tốc độ, kích thước...), kết nối Internet...

Một cách khác để cải thiện khả năng vận hành của các ứng dụng Web là điều chỉnh cách làm việc của MySQL. Bạn có thể thực hiện điều này bằng cách xác định các tùy chọn khác nhau, chẳng hạn như `key_buffer` *(bộ nhớ được dành cho chỉ mục),* `max_connections` *(số kết nối tối đa được xử lý cùng lúc) và* `table_cache` *(bộ đệm bảng). Thông tin chi tiết về các tham số này có thể tìm thấy trong tài liệu MySQL.*

Máy chủ Apache có kèm theo một tiện ích đo khả năng vận hành có tên là ab *(viết tắt của ApacheBench). Bạn có thể sử dụng tiện ích này để thử khả năng vận hành của các trang Web, kể cả mã kịch bản PHP.*

Trong Chương 10 "Các chủ đề mở rộng", chúng ta sẽ sử dụng lớp Benchmark của PEAR để tính thời gian thực hiện mã kịch bản PHP.

Cải thiện khả năng vận hành của PHP

Việc cải thiện khả năng vận hành tổng thể của các mã kịch bản PHP sẽ dẫn đến việc phân tích các xử lý và các biến. Các ứng dụng sẽ hoạt động tốt hơn nếu giảm thời gian thực hiện mã kịch bản và giảm bộ nhớ đòi

hỏi. Nói chung, điều này có nghĩa là nên bỏ qua những biến không cần thiết và giới hạn số lần gọi hàm nếu như có thể. Ngoài ra, bạn nên tách mã HTML dưới dạng một tập tin kèm theo. Điều này giúp mã lệnh HTML không phải biên dịch bởi PHP.

Một số đề nghị khác bao gồm:

- *Sử dụng các hàm xử lý chuỗi thay cho các biểu thức quy tắc mỗi khi có thể.*
- *Sử dụng các biểu thức quy tắc tương thích PERL thay cho POSIX.*
- *Truyền các biến lớn (chuỗi dài, mảng, đối tượng...) cho hàm dưới dạng tham chiếu thay vì giá trị (tham khảo tài liệu PHP để biết thêm chi tiết về khái niệm này).*
- *Thiết kế ứng dụng Web sao cho các trang chỉ được đưa vào các hàm yêu cầu.*
- *Thiết lập một biến thành* **`NULL`** *(hoặc dùng* **`unset()`***) khi không cần đến nó nữa.*

Tối ưu bảng

Một trong những cách để cải thiện khả năng vận hành của MySQL là thực hiện câu truy vấn **`OPTIMIZE`** trên các bảng. Câu truy vấn này sẽ giải phóng các bảng khỏi các phần không cần thiết, do đó sẽ làm tăng tốc độ tương tác với chúng.

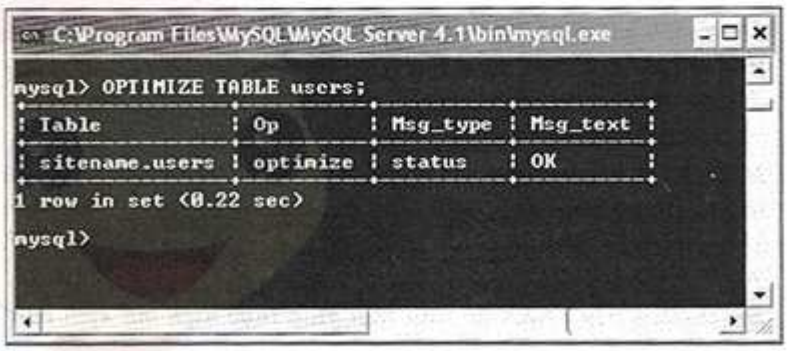
Thực hành tối ưu hóa một bảng theo các bước sau:

1. Đăng nhập vào trình quản lý mysql và chọn cơ sở dữ liệu *sitename*.

```
mysql -u root -p
USE sitename;
```

2. Tối ưu bảng users (xem hình 9-11).

```
OPTIMIZE TABLE users;
```



Hình 9-11: *Câu truy vấn OPTIMIZE TABLE sẽ cải thiện khả năng vận hành của các bảng trong MySQL.*

Sau khi chạy câu truy vấn này, MySQL sẽ cho biết trạng thái của bảng.

3. Thoát khỏi trình quản lý mysql.

```
exit
```

Các bảng MySQL có dạng MyISAM (kiểu mặc định trong các phiên bản hiện tại của phần mềm) có thể được tối ưu bằng cách sử dụng tiện ích myisamchk.

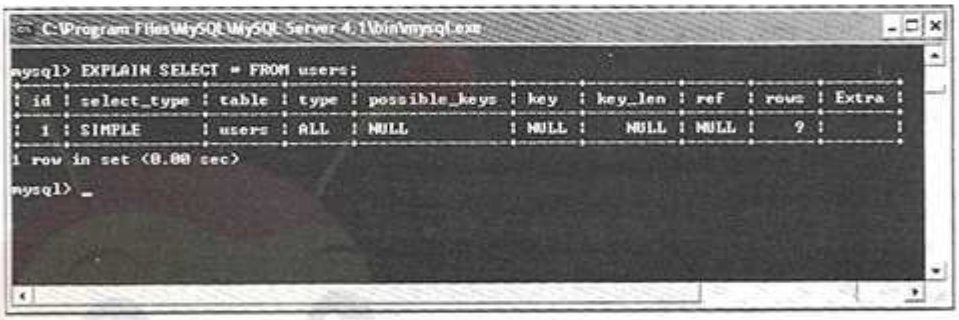
Bạn cũng có thể thiết lập cron (trên Unix) hoặc một dịch vụ (trên Windows) để tối ưu hóa tự động các bảng theo một lịch cụ thể, với việc sử dụng tiện ích myisamchk hoặc một mã kịch bản PHP thực hiện câu truy vấn ***OPTIMIZE***

Nên sao lưu cơ sở dữ liệu trước khi thực hiện bất kỳ một hành động nào, kể cả việc tối ưu chúng.

Diễn giải các câu truy vấn

Trung tâm của các ứng dụng Web dựa trên nền MySQL là các câu truy vấn cơ sở dữ liệu được thực hiện từ các mã kịch bản PHP. Do đó, việc làm cho các câu truy vấn này hiệu quả nhất có thể là một điều rất quan trọng. Để cải thiện tính hiệu quả của câu truy vấn, chúng ta cần hiểu rõ cách MySQL thực hiện nó ra sao. Điều này có thể được thực hiện bằng lệnh **EXPLAIN**. Ví dụ (xem hình 9-12):

```
EXPLAIN SELECT * FROM users;
```



Hình 9-12: *Câu truy vấn EXPLAIN sẽ cho biết cách MySQL xử lý câu lệnh SQL ra sao.*

Phần diễn giải được cung cấp bởi MySQL ứng với một câu truy vấn sẽ giúp ta biết được cần phải tạo chỉ mục ở đâu và điều chỉnh câu lệnh SQL thế nào cho tốt hơn. Phần diễn giải cũng liệt kê các bảng được sử dụng, loại liên kết (xem bảng 9.4), các chỉ mục hoặc khóa mà MySQL sử dụng cho câu truy vấn và số dòng mà MySQL phải xử lý khi thực hiện câu truy vấn này.

Bảng 9.4: *Các loại liên kết này (được trả lại bởi câu truy vấn* ***EXPLAIN****) được liệt kê theo chiều thứ tự giảm dần.*

Loại	Ý nghĩa
`System`	Bảng chỉ chứa một dòng.
`Const`	Bảng chứa dòng phù hợp nhất.
`eq_ref`	Một dòng được đọc từ bảng này ứng với mỗi tổ hợp các dòng từ các bảng liên kết.
`Ref`	Một tập các dòng sẽ được đọc từ bảng này với mỗi tổ hợp của các dòng từ các bảng liên kết.
`Range`	Một tập các dòng sẽ được trả lại, dựa trên một trường chỉ mục.
`Index`	Chỉ mục sẽ được xem xét với mỗi dòng trong bảng.
`All`	Tất cả các dòng trong bảng đều được xem xét.

Thực hành sử dụng lệnh **EXPLAIN** theo các bước sau:

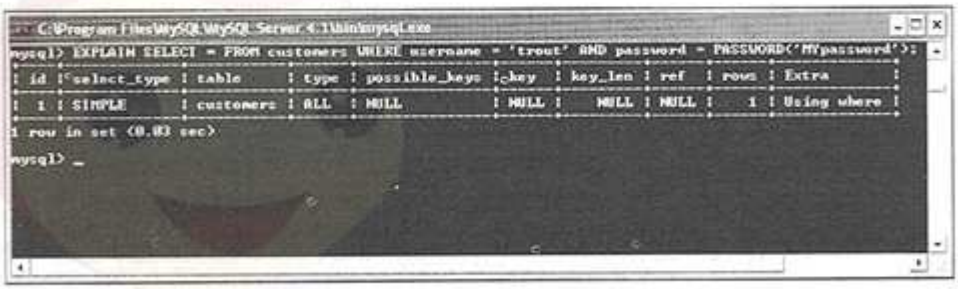
1. Đăng nhập vào trình quản lý mysql và chọn cơ sở dữ liệu *ecommerce*.

```
mysql -u username -p

USE ecommerce;
```

Trong ví dụ này, chúng ta sử dụng cơ sở dữ liệu *ecommerce* đã được tạo ra trong chương trước.

2. Thực hiện câu truy vấn **EXPLAIN** (xem hình 9-13).

```
EXPLAIN SELECT * FROM customers WHERE username = 'trout'
→ AND password = PASSWORD('MYpassword');
```



Hình 9-13: *Việc sử dụng câu truy vấn* ***EXPLAIN*** *sẽ giúp ta hiểu được cách làm việc của MySQL.*

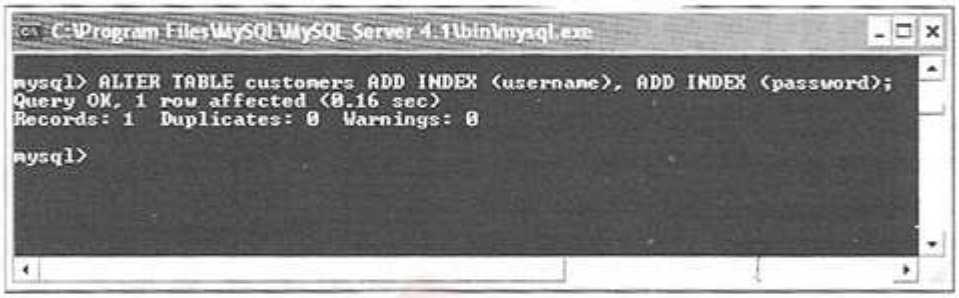
Đây là một câu truy vấn lôgic có thể được thực hiện với một bảng, lấy ra tất cả các thông tin về một khách hàng cụ thể. Sau khi thực hiện câu truy vấn **EXPLAIN**, chúng ta sẽ có được các thông tin sau:

- Dạng liên kết là *all*, cho biết đây là dạng kém hiệu quả nhất.
- Không có khóa.
- Tất cả các dòng đều được quét qua.

3. Nếu cần, hãy thay đổi câu truy vấn hoặc bổ sung các chỉ mục cần thiết (xem hình 9-14):

```
ALTER TABLE customers ADD INDEX (username), ADD INDEX
→ (password);
```

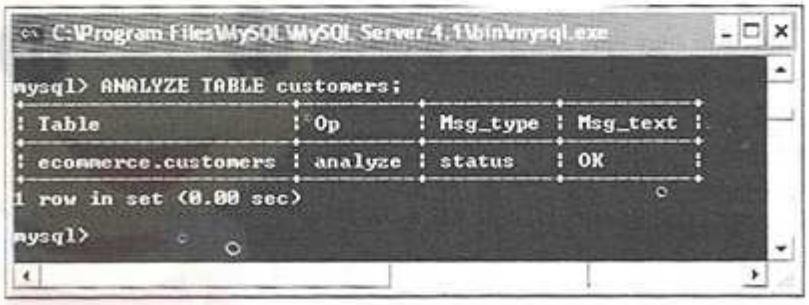
Để cải thiện tính hiệu quả cho bảng này, chúng ta sẽ tạo chỉ mục cho hai cột thường được tham chiếu (trường **customer_id** được tự động tạo chỉ mục dưới dạng khóa chính).



Hình 9-14: *Để cải thiện khả năng vận hành của một câu truy vấn, chúng ta bổ sung hai chỉ mục cho bảng customers.*

4. Thực hiện câu truy vấn **ANALYZE TABLE** (xem hình 9-15):

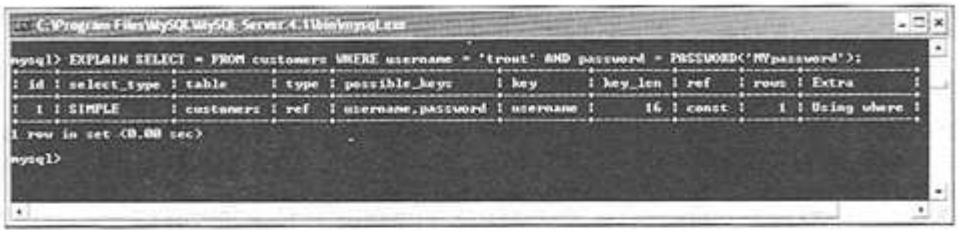
```
ANALYZE TABLE customers;
```



Hình 9-15: *Việc phân tích các bảng cần được thực hiện một cách thường xuyên để cải thiện tính vận hành của MySQL.*

Câu truy vấn **ANALYZE TABLE** buộc MySQL cập nhật các số liệu thống kê về một bảng. Điều này sẽ giúp cải thiện tính hiệu quả của các câu truy vấn và **EXPLAIN**.

5. Thực hiện câu truy vấn **EXPLAIN** lần thứ hai (xem hình 9-16):



Hình 9-16: *Thực hiện lại câu truy vấn **EXPLAIN** sẽ cho thấy ảnh hưởng của các thay đổi.*

```
EXPLAIN SELECT * FROM customers WHERE username = 'trout'
→ AND password = PASSWORD('MYpassword');
```

Khả năng vận hành của câu truy vấn này đã được cải thiện đáng kể sau khi bổ sung của hai chỉ mục. Kết quả của câu truy vấn **EXPLAIN** giờ đây cho thấy:

- Loại liên kết là *ref*, có nghĩa là tốt hơn *all*.
- Có hai khóa được sử dụng.
- Khóa *username* được sử dụng.
- Chỉ có một dòng được quét qua.

Để cải thiện khả năng vận hành của các cơ sở dữ liệu, bạn nên sử dụng kích thước cột tối thiểu khi định nghĩa bảng.

*Câu truy vấn **EXPLAIN <tên bảng>** tương tự như **DESCRIBE <tên bảng>**.*

*Bạn có thể buộc MySQL sử dụng một khóa cụ thể trong một câu truy vấn với mệnh đề **USE KEY**.*

*Các câu truy vấn **INSERT** sẽ chạy nhanh hơn nếu sử dụng các giá trị mặc định, do đó bạn chỉ nên chèn vào những giá trị khác với mặc định.*

Cơ sở dữ liệu liên quan đến các con số sẽ nhanh hơn so với chuỗi, do đó không nên đặt các con số trong dấu nháy.

Chương 10:

CÁC CHỦ ĐỀ MỞ RỘNG

Chương này giới thiệu phần "nội dung" cuối cùng của cuốn sách và tập trung vào các thông tin mới (ba chương còn lại sẽ làm việc với các ứng dụng cụ thể). Các chủ đề mở rộng được đề cập ở đây không thể áp dụng cho mọi ứng dụng nhưng rất đáng quan tâm. Chúng bao gồm: Xuất ra bộ nhớ đệm, bộ đệm trang Web, phát hiện trình duyệt Web, sử dụng PHP với JavaScript và PEAR (một framework dạng PHP).

Xuất ra bộ đệm

Một đặc điểm mới rất hấp dẫn trong phiên bản 4 của PHP là khả năng kiểm soát đầu ra (output control). Đặc điểm này (hoặc xuất ra bộ đệm) cho phép viết và thực hiện các mã kịch bản như bình thường nhưng chỉ gửi dữ liệu xuống trình duyệt Web tại những điểm nhất định. Ưu điểm chính của hệ thống này là ta có thể gọi các hàm **header()**, **setcookie()** và **session_start()** gần như tại mọi điểm trong mã kịch bản mà không phải lo lắng về các thông báo *headers already sent*.

Sử dụng hàm **ob_start()** để bắt đầu xuất ra bộ đệm. Sau khi gọi hàm này, mỗi lời gọi hàm **echo()**, **print()** và các hàm tương tự sẽ gửi dữ liệu vào bộ nhớ đệm thay vì gửi trực tiếp xuống trình duyệt Web. Trong khi đó, các hàm **header()** và **setcookie()** sẽ không đưa dữ liệu vào bộ đệm mà vẫn hoạt động như bình thường.

Ở cuối mã kịch bản, chúng ta gọi hàm **ob_end_flush()** để gửi dữ liệu trong bộ đệm xuống trình duyệt Web. Ta sử dụng hàm **ob_end_clean()** để xóa nội dung trong bộ đệm mà không gửi chúng xuống trình duyệt. Cả hai hàm cũng đồng thời tắt chế độ xuất ra bộ đệm.

Với quan điểm của người lập trình, việc xuất ra bộ đệm cho phép chúng ta cấu trúc mã kịch bản ở dạng lôgic hơn mà không phải bận tâm đến các phần đầu HTTP (HTTP header). Chúng ta sẽ tạo ra các phiên bản mới của các mã kịch bản đăng nhập đã được xây dựng trong Chương 7 "Cookie và Session" làm ví dụ cho chương này.

Thực hành sử dụng xuất ra bộ đệm theo các bước sau:

1. Tạo một hồ sơ PHP mới trong trình soạn thảo văn bản.

```
<?php # Doan ma 10.1 - login.php
```

2. Bắt đầu xuất ra bộ đệm.

```
ob_start();
```

Chìa khóa để sử dụng kỹ thuật xuất ra bộ đệm là gọi hàm **ob_start()** ở đầu mã kịch bản.

3. Đưa vào phần đầu HTML.

```
$page_title = 'Login';
include ('templates/header.inc');
```

Vì tập tin được đưa vào sau khi chế độ xuất ra bộ đệm đã khởi động, các nội dung trong tập tin **header.inc** sẽ được chứa trong bộ đệm thay vì gửi trực tiếp xuống trình duyệt.

4. Kiểm tra xem biểu mẫu có được gửi không và tạo một hàm để xử lý dữ liệu biểu mẫu.

```
if (isset($_POST['submit'])) {
   require_once ('../mysql_connect.php');
   function escape_data ($data) {
      global $dbc;
      if (ini_get('magic_quotes_gpc')) {
         $data = stripslashes($data);
      }
      return mysql_real_escape_string(trim($data), $dbc);
   }
```

Phần này của mã kịch bản chỉ có một thay đổi so với phiên bản gốc của nó. Chúng ta sử dụng hàm **trim()** để loại bỏ các khoảng trắng dư ra khỏi các giá trị.

5. Kiểm tra tên đăng nhập và mật khẩu.

```
if (empty($_POST['username'])) {
   $u = FALSE;
   echo '<p>Ban chua nhap ten dang nhap!</p>';
} else {
   $u = escape_data($_POST['username']);
}
if (empty($_POST['password'])) {
   $p = FALSE;
   echo '<p>Ban chua nhap mat khau!</p>';
} else {
   $p = escape_data($_POST['password']);
}
```

Thay vì sử dụng biến **$message** để xử lý các lỗi như đã thực hiện trước đây, bây giờ chúng ta có thể in trực tiếp các thông báo này. Các chuỗi sẽ được đưa vào trong bộ đệm thay vì được gửi ngay xuống trình duyệt.

6. Truy vấn cơ sở dữ liệu.

```
if ($u && $p) {

   $query = "SELECT user_id, first_name FROM users
   → WHERE  username='$u' AND password=PASSWORD('$p')";

   $result = @mysql_query ($query);

   $row = mysql_fetch_array ($result, MYSQL_NUM);
```

7. Khi tìm thấy mẩu tin thích hợp, mã lệnh sẽ chuyển người dùng sang một trang Web khác. Nếu không, nó sẽ hiển thị thông báo:

```
if ($row) {

   session_start();

   $_SESSION['first_name'] = $row[1];

   $_SESSION['user_id'] = $row[0];

   ob_end_clean();

   header ("Location:  http://" . $_SERVER['HTTP_HOST'] .
   → dirname($_SERVER['PHP_SELF']) . "/loggedin.php");

   exit();

} else {

   echo '<p>Ten dang nhap va mat khau khong phu hop voi
   → du lieu trong CSDL.</p>';

}
```

Nếu tên đăng nhập và mật khẩu gửi lên phù hợp với giá trị trong cơ sở dữ liệu thì biến **$row** sẽ có một giá trị. Khi đó, chúng ta có thể đăng ký **user_id**, password với session và chuyển hướng trình duyệt Web. Nếu không, các thông báo lỗi sẽ được in ra.

Vì sử dụng kỹ thuật xuất ra bộ đệm, nên có thể sử dụng hàm **header()** ở đây, dù đã đưa vào tập tin đầu trang và đã sử dụng hàm **echo()** để xuất ra kết quả trước đó. Do không sử dụng bộ nhớ đệm nếu chuyển hướng trình duyệt, chúng ta sẽ gọi hàm **ob_end_clean()** trước khi gọi hàm **header()**. Việc này sẽ làm xóa toàn bộ dữ liệu hiện có trong bộ đệm và ngừng quá trình xuất ra bộ đệm.

8. Đóng kết nối cơ sở dữ liệu và hoàn tất phần điều kiện cũng như mã lệnh PHP.

```
        mysql_close();
           } else {
        echo '<p>Hay thu lai.</p>';
     }
  }
  ?>
```

9. Xây dựng biểu mẫu HTML.

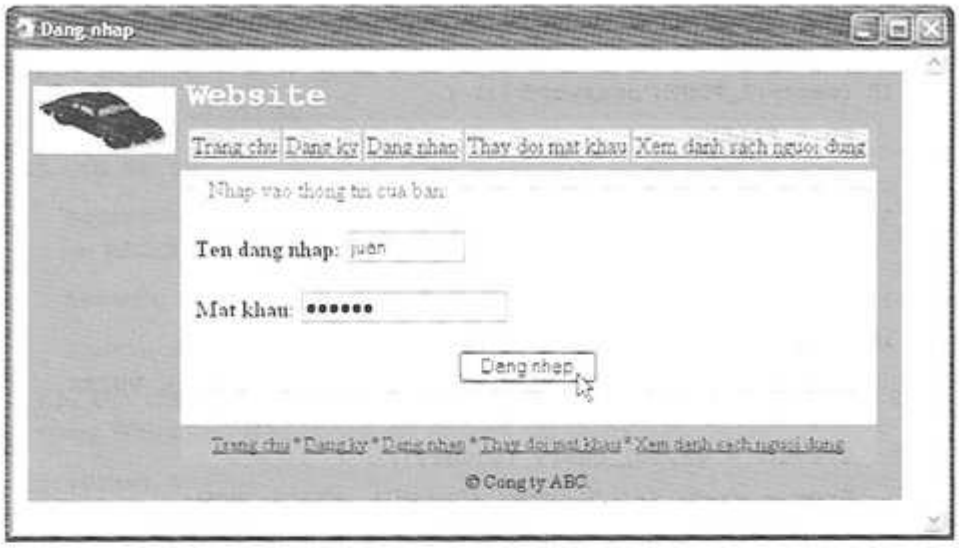
```
<form action="<?php echo $_SERVER['PHP_SELF']; ?>"
→ method="post">
<fieldset><legend>Nhap vao thong tin cua ban:</legend>
<p><b>Ten dang nhap:</b> <input type="text" name="username"
→ size="10" maxlength="20" value="<?php if
→ (isset($_POST['username'])) echo $_POST['username']; ?>"
→ /></p>
<p><b>Mat khau:</b> <input type="password" name="password"
→ size="20" maxlength="20" /></p>
<div align="center"><input type="submit" name="submit"
→ value="Dang nhap" /></div>
</fieldset>
</form>
```

10. Đưa vào phần cuối trang HTML và đẩy dữ liệu trong bộ đệm xuống trình duyệt.

```
<?php
include ('templates/footer.inc');
ob_end_flush();
?>
```

Dòng thứ hai trong khối lệnh trên sẽ gửi tất cả dữ liệu hiện có trong bộ đệm xuống trình duyệt, đồng thời tắt chế độ xuất ra bộ đệm.

11. Lưu tập tin với tên gọi **login.php**, tải nó lên máy chủ Web (trong cùng thư mục với các tập tin khác) và chạy thử trong trình duyệt (xem hình 10-1). Mã lệnh đầy đủ của tập tin này được thể hiện trong đoạn mã 10.1.



Hình 10-1: *Tính năng của mã kịch bản đăng nhập không thay đổi.*

Đoạn mã 10.1: *Mã kịch bản* ***login.php*** *có thể được viết ở dạng cấu trúc lôgic hơn mà không phải bận tâm đến phần đầu HTTP (HTTP header), nhờ việc sử dụng kỹ thuật xuất ra bộ đệm.*

```
<?php # Doan ma 10.1 - login.php
ob_start();
$page_title = 'Dang nhap';
include ('templates/header.inc');
if (isset($_POST['submit'])) {
   require_once ('../mysql_connect.php');
   function escape_data ($data) {
      global $dbc;
      if (ini_get('magic_quotes_gpc')) {
         $data = stripslashes($data);
      }
      return mysql_real_escape_string(trim($data), $dbc);
   }
   if (empty($_POST['username'])) {
      $u = FALSE;
      echo '<p>Ban chua nhap ten dang nhap!</p>';
   } else {
```

```
    $u = escape_data($_POST['username']);
  }
  if (empty($_POST['password'])) {
    $p = FALSE;
    echo '<p>Ban chua nhap mat khau!</p>';
  } else {
    $p = escape_data($_POST['password']);
  }
  if ($u && $p) {
    $query = "SELECT user_id, first_name FROM users WHERE
    → username='$u' AND password=PASSWORD('$p')";
    $result = @mysql_query ($query);
    $row = mysql_fetch_array ($result, MYSQL_NUM);
    if ($row) {
      session_start();
      $_SESSION['first_name'] = $row[1];
      $_SESSION['user_id'] = $row[0];
      ob_end_clean();
      header ("Location:  http://" . $_SERVER['HTTP_HOST'] .
      → dirname($_SERVER['PHP_SELF']) . "/loggedin.php");
      exit();
    } else {
      echo '<p>Ten dang nhap va mat khau khong phu hop voi
      → du lieu trong CSDL.</p>';
    }
    mysql_close();
  } else {
    echo '<p>Hay thu lai.</p>';
  }
}
?>
<form action="<?php echo $_SERVER['PHP_SELF']; ?>"
→ method="post">
<fieldset><legend>Nhap vao thong tin cua ban:</legend>
<p><b>Ten dang nhap:</b> <input type="text" name="username"
→ size="10" maxlength="20" value="<?php if
→ (isset($_POST['username'])) echo $_POST['username']; ?>"
```

```
→ /></p>
<p><b>Mat khau:</b> <input type="password" name="password"
→ size="20" maxlength="20" /></p>
<div align="center"><input type="submit" name="submit"
→ value="Dang nhap" /></div>
</fieldset>
</form>
<?php
include ('templates/footer.inc');
ob_end_flush();
?>
```

- Thực hành tạo mã kịch bản **loggedin.php** theo các bước sau:

1. Tạo một hồ sơ PHP mới trong trình soạn thảo văn bản.

```
<?php # Doan ma 10.2 - loggedin.php
```

Với mục đích minh họa cho kỹ thuật xuất ra bộ đệm, chúng ta sẽ viết một phiên bản mới của mã kịch bản **loggedin.php** (mã kịch bản mà người dùng sẽ được chuyển đến sau khi đăng nhập thành công).

2. Bắt đầu chế độ xuất ra bộ đệm, khởi tạo session và đưa vào phần đầu HTML.

```
ob_start();
session_start();
$page_title = 'Logged In!';
include ('templates/header.inc');
```

3. Kiểm tra xem tên đã được đăng ký chưa.

```
if (isset($_SESSION['first_name'])) {
    echo "<p>Ban da dang nhap thanh cong,
    → {$_SESSION['first_name']}!</p>";
} else {
    ob_end_clean();
    header ("Location:  http://" . $_SERVER['HTTP_HOST'] .
    → dirname($_SERVER['PHP_SELF']) . "/index.php");
    exit();
}
```

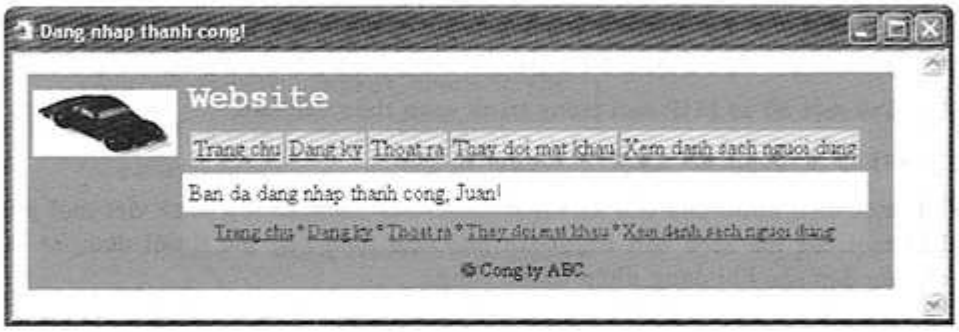
Thay vì phải có hai quá trình riêng biệt (kiểm tra sự hiện diện của biến **$_SESSION['first_name']** và chào mừng người dùng), chúng ta viết nó trong

cùng một điều kiện. Nếu người dùng đã đăng nhập, họ sẽ được chào mừng. Nếu không, bộ nhớ đệm sẽ bị xóa và người dùng sẽ được chuyển đến trang **index.php**.

4. Hoàn tất phần cuối trang HTML và PHP.

```
include ('templates/footer.inc');

ob_end_flush();

?>
```

5. Lưu tập tin với tên gọi **loggedin.php**, tải nó lên máy chủ Web và chạy thử trong trình duyệt (xem hình 10-2). Mã lệnh đầy đủ của tập tin này được thể hiện trong đoạn mã 10.2.



Hình 10-2: *Trình duyệt Web sau khi người dùng đăng nhập.*

Để thử mã kịch bản, có thể truy xuất trực tiếp trang **loggedin.php** mà không đăng nhập trước. Khi đó, bạn sẽ được chuyển trở lại trang đăng nhập.

Đoạn mã 10.2: *Phiên bản này của mã kịch bản* ***loggedin.php*** *sử dụng kỹ thuật xuất ra bộ đệm cùng với session.*

```
<?php # Doan ma 10.2 - loggedin.php
ob_start();
session_start();
$page_title = 'Dang nhap thanh cong!';
include ('templates/header.inc');
if (isset($_SESSION['first_name'])) {
   echo "<p>Ban da dang nhap thanh cong,
   → {$_SESSION['first_name']}!</p>";
} else {
   ob_end_clean();
   header ("Location:  http://" . $_SERVER['HTTP_HOST'] .
   → dirname($_SERVER['PHP_SELF']) . "/index.php");
```

```
    exit();
}
include ('templates/footer.inc');
ob_end_flush();
?>
```

Kích thước bộ đệm tối đa được thiết lập trong tập tin **php.ini**. *Giá trị mặc định là 4096 ký tự.*

- *Từ phiên bản PHP 4.0.4, chúng ta có thể gửi dữ liệu nén xuống trình duyệt bằng cách kiểm soát kết quả xuất ra với* **ob_start('ob_gzhandler')**. *Điều này sẽ giúp làm giảm kích thước dữ liệu tải xuống trình duyệt. Hàm* **ob_gzhandler()** *xác định kiểu nén thích hợp sẽ được sử dụng (gzip, giải nén, hoặc không nén) dựa trên những gì trình duyệt có thể tiếp nhận. Chú ý, việc nén chỉ có hiệu quả với các trang có kích thước lớn và để sử dụng nó, bạn cần tắt thiết lập* **output_buffering** *trong tập tin* **php.ini**.
- *Hàm* **ob_get_length()** *trả lại chiều dài (tính theo số ký tự) của bộ đệm hiện tại.*
- *Hàm* **ob_get_contents()** *sẽ trả lại nội dung bộ đệm hiện tại để gán cho một biến nếu có yêu cầu.*
- *Hàm* **ob_flush()** *(một hàm mới của PHP 4.2) sẽ gửi nội dung hiện tại trong bộ đệm xuống trình duyệt và sau đó loại bỏ chúng ra khỏi bộ đệm, cho phép khởi tạo một bộ đệm mới. Hàm này cho phép mã kịch bản duy trì kích thước bộ đệm một cách hợp lý.*
- *Hàm* **ob_clean()** *- cũng là một hàm mới của PHP 4.2. Nó xóa toàn bộ nội dung hiện có trong bộ đệm mà không làm ngừng quá trình đưa dữ liệu vào trong bộ đệm.*
- *PHP sẽ tự động thực hiện hàm* **ob_end_flush()** *khi kết thúc mã kịch bản nếu không có một hành động nào khác được thực hiện.*

Đưa trang Web vào bộ đệm

Vì các trang HTML (phát sinh bởi mã kịch bản PHP) sẽ được tạo ra lúc chạy, chúng thường không được đưa vào trong bộ đệm bởi các trình duyệt Web và máy chủ ủy quyền (proxy) (việc đưa trang Web vào bộ đệm sẽ giúp cho việc truy cập lại trang Web đã được truy cập trước đó nhanh hơn vì nó tham chiếu đến phiên bản trong bộ đệm thay vì phải lấy xuống từ máy chủ mỗi khi có yêu cầu). Tuy nhiên, các ISP thường đưa vào bộ đệm các site phổ biến để cải thiện tốc độ vận hành và người dùng đôi khi điều chỉnh chính sách bộ đệm trên trình duyệt theo

nhu cầu riêng của họ (xem hình 10-3). Là nhà phát triển PHP, chúng ta có thể sử dụng hàm **header()** để kiểm soát chế độ này một cách cụ thể.

Hình 10-3: *Các thiết lập về bộ đệm có thể được thay đổi theo nhiều cách khác nhau.*

Thực hành ngăn không đưa một trang vào bộ đệm theo các bước sau:

1. Tạo một hồ sơ PHP mới.

```
<?php # Doan ma 10.3 - nocache.php
```

2. Tạo phần đầu trang HTML đầu tiên.

```
header("Expires: Mon, 26 Jul 1997 05:00:00 GMT");

header("Last-Modified: " . gmdate("D, d M Y H:i:s")
→ " GMT");
```

Phần đầu trang thứ nhất cho biết nội dung sẽ mất hiệu lực tại một thời điểm trong quá khứ (bất kỳ giá trị nào trong quá khứ đều được), nghĩa là trình duyệt cần phải gửi yêu cầu đến máy chủ để có được phiên bản mới nhất. Phần đầu trang thứ hai thiết lập ngày chỉnh sửa của mã kịch bản này là ngày hiện tại (bằng cách sử dụng **gmdate()** - giống như hàm **date()** - nhưng tính theo giờ GMT). Nói cách khác, không chỉ thông tin của trang này đã hết hiệu lực, mà bản thân trang cũng mới được điều chỉnh.

3. Tạo các phần đầu trang HTTP 1.1 và 1.0.

```
header ("Cache-Control: no-store, no-cache,
→ must-revalidate");

header ("Cache-Control: post-check=0, pre-check=0",
→ false);

header ("Pragma: no-cache");
```

Ba dòng lệnh này thực hiện theo các giao thức 1.0 và 1.1 trong việc kiểm soát chế độ ghi vào bộ đệm. Hai dòng đầu áp dụng theo giao thức của 1.1, trong khi đó dòng lệnh cuối cùng áp dụng cho giao thức 1.0.

4. Đóng mã lệnh PHP và lưu tập tin với tên **nocache.php**. Mã lệnh đầy đủ của tập tin này được thể hiện trong đoạn mã 10.3.

```
?>
```

5. Để ngăn trình duyệt đưa một trang PHP nào đó vào trong bộ đệm, hãy đưa mã kịch bản này vào dòng đầu tiên trong mã kịch bản đó.

- `include('nocache.php')`

Đoạn mã 10.3: *Mã kịch bản này sẽ ghi đè các thiết lập bộ đệm của ISP và trình duyệt.*

```
<?php # Doan ma 10.3 - nocache.php

header ("Expires: Mon, 26 Jul 1997 05:00:00 GMT");

header ("Last-Modified: " . gmdate("D, d M Y H:i:s") .
→ " GMT");

header ("Cache-Control: no-store, no-cache,
→ must-revalidate");

header ("Cache-Control: post-check=0, pre-check=0", false);

header ("Pragma: no-cache");

?>
```

Không nên nhầm lẫn việc ghi bộ đệm ở trình duyệt với các công cụ ghi bộ đệm ở phía máy chủ (như Zend Optimizer). Dạng ghi bộ đệm trước thay đổi khi trình duyệt truy xuất các trang Web, trong khi dạng sau sẽ mã hóa và tăng tốc cho các mã kịch bản trên máy chủ.

Nếu ứng dụng có sử dụng session, bạn có thể điều chỉnh chế độ ghi bộ đệm session với hàm ***session_cache_limit()****. Xem các tài liệu liên quan để biết thêm chi tiết.*

Mặc dù ví dụ này buộc trình duyệt phải luôn luôn lấy URL từ máy chủ, bạn cũng có thể dùng các giá trị về thời gian có hiệu lực và chỉnh sửa để trình duyệt chỉ truy xuất máy chủ khi nó được cập nhật.

*Việc ghi bộ đệm trang Web cũng có thể bị ảnh hưởng bằng cách sử dụng các thẻ **META** được đặt trong phần đầu (trong thẻ <head>) của trang HTML.*

Phát hiện trình duyệt

Việc xem xét một số trình duyệt hiện có (cùng với các thiết lập khác nhau mà người dùng có thể điều chỉnh) nhằm đảm bảo tính nhất quán cho tất cả người dùng là một vấn đề rất khó. JavaScript (một công nghệ phía khách hàng), có thể phát hiện loại trình duyệt, phiên bản và các thông tin khác một cách khá dễ dàng. Trong PHP cũng có một tùy chọn để thực hiện điều này, đó là hàm **get_browser()**.

```
$browser = get_browser();
```

Mặc định, hàm này sẽ sử dụng biến **$HTTP_USER_AGENT** (hoặc **$_SERVER ['HTTP_USER_AGENT']**). Sau đó, nó sẽ so sánh giá trị có được với giá trị trong tập tin **browscap.ini** (một tập tin văn bản liệt kê các mô tả trình duyệt theo phiên bản và môi trường). Tập tin này có thể được tải xuống miễn phí và cần được cài đặt trên máy chủ để hàm **get_browser()** có thể hoạt động.

Hàm **get_browser()** trả lại một đối tượng (một kiểu biến chưa đề cập đến trong cuốn sách này) với các thuộc tính được nêu trong hình 10-4. Để tham chiếu đến một giá trị cụ thể, chúng ta sử dụng cú pháp:

```
$doi_tuong->thuoc_tinh;
```

PHP và CSS

Cascading Style Sheet (CSS), đang phát triển như là một thành phần của Web, định nghĩa lại các khía cạnh định dạng và tạo phong cách cho trang Web. Trong khi không có một quan hệ trực tiếp nào giữa PHP và CSS, chúng ta vẫn có thể sử dụng PHP để viết ra mã lệnh CSS (cũng như dùng PHP để viết ra mã lệnh JavaScript vậy). Hơn nữa, vì CSS thường được chứa trong một tập tin ngoài và được đưa vào trong hồ sơ HTML, nên có thể sử dụng PHP để xác định tập tin CSS nào sẽ được sử dụng tùy theo trình duyệt.

*Ngoài việc cần biết loại trình duyệt (Mozilla, Netscape, IE), phiên bản trình duyệt (4, 5, 6) và hệ điều hành (Windows, Linux, Mac...) đang được sử dụng, chúng ta còn cần biết phiên bản nào của CSS được hỗ trợ bởi trình duyệt bằng cách tham chiếu đến biến **$browser->css** sau khi gọi hàm **get_browser()** - (xem hình 10-4).*

Hình 10-4: *Các giá trị được trả về bởi hàm* ***get_browser()***.

Thực hành cài đặt tập tin **browscap.ini** theo các bước sau:

1. Tải tập tin từ địa chỉ: *http://www.garykeith.com/browsers/downloads.asp*.

Để cài đặt tập tin **browscap.ini**, trước hết bạn phải tải nó xuống từ URL này (tập tin này miễn phí).

2. Đưa tập tin vào một vị trí thích hợp.

Tập tin **browscap.ini** cần được đặt vào một vị trí thích hợp trên máy chủ. Đây có thể là một nơi trong thư mục gốc của Web hoặc vị trí nào đó trên ổ cứng.

Trên hệ điều hành Unix hoặc Mac OS X, bạn có thể sử dụng dòng lệnh:

```
mv browscap.ini /usr/local/lib
```

Trên Windows, bạn có thể sử dụng Windows Explorer để đưa nó vào thư mục thích hợp, chẳng hạn như **c:\inetpub** (hoặc trong thư mục của máy chủ Web).

3. Chỉnh sửa tập tin **php.ini** để nó thể hiện được vị trí của tập tin **browscap.ini**. Ví dụ:

```
browscap = c:\inetpub\browscap.ini
```

Sử dụng trình soạn thảo văn bản để thay đổi thiết lập **browscap**, phản ánh vị trí của tập tin **browscap.ini** trên máy chủ (đã được xác định trong bước 2).

4. Khởi động lại máy chủ Web để PHP sử dụng tập tin **browscap.ini**.

Trên Unix hoặc Mac OS X, chúng ta có thể khởi động lại Apache với lệnh **apachectl graceful**. Trên Windows, ta có thể khởi động lại phần lớn máy chủ Web (Apache, Xitami, IIS) thông qua lệnh Restart.

Thực hành việc phát hiện trình duyệt theo các bước sau:

1. Tạo một mã kịch bản PHP mới trong trình soạn thảo.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
→ Transitional//EN" "http://www.w3.org/TR/2000/REC-
→xhtml1-20000126/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">
<head>
<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />
<title>Phat hien trinh duyet</title>
</head>
<body>
<?php # Doan ma 10.4 - browser.php
```

2. Gọi hàm **get_browser()** và in ra giá trị của **HTTP_USER_AGENT**.

```
$browser = get_browser();
echo "<p>HTTP_USER_AGENT is
→ {$_SERVER['HTTP_USER_AGENT']}.</p>";
```

Chúng ta in giá trị của **HTTP_USER_AGENT** để so sánh giá trị này với giá trị được trả về bởi hàm **get_browser()**.

3. In các thông tin liên quan đến trình duyệt.

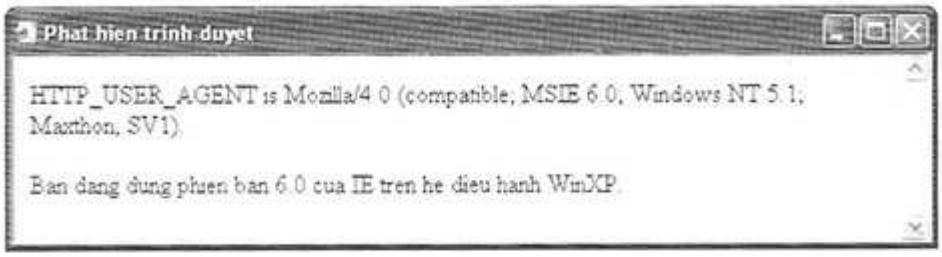
```
echo "<p>Ban dang dung phien ban $browser->version cua
→ $browser->browser tren he dieu hanh
→ $browser->platform.</p>";
```

Với mục đích của mình, chúng ta in tên trình duyệt, phiên bản và hệ điều hành. Thông tin này thường được sử dụng để tùy biến một Website cho phù hợp với một trình duyệt hoặc một hệ điều hành cụ thể.

4. Hoàn tất mã lệnh PHP và trang HTML.

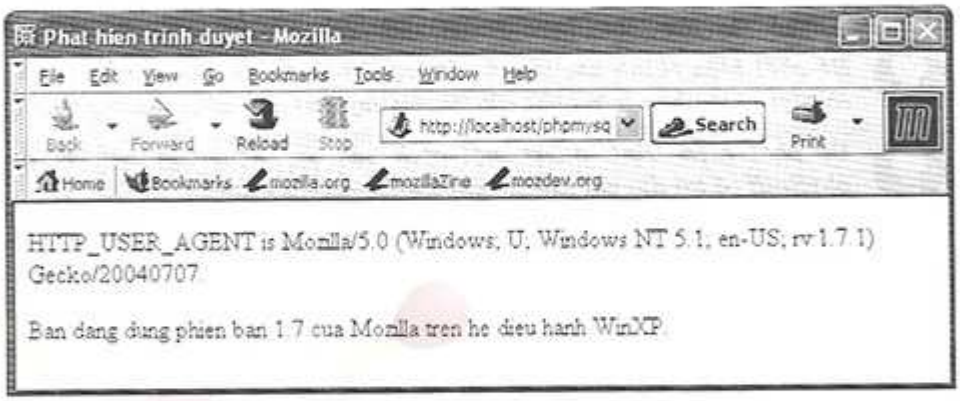
```
?>
</body>
</html>
```

5. Lưu tập tin với tên gọi **browser.php**, tải nó lên máy chủ và chạy thử trong trình duyệt (xem hình 10-5). Mã lệnh đầy đủ của tập tin này được thể hiện trong đoạn mã 10.4.



Hình 10-5: *Trang* ***browser.php*** *được xem trong trình duyệt IE trên Windows XP.*

6. Nếu có thể, chạy thử mã kịch bản này trong một trình duyệt khác (xem hình 10-6).



Hình 10-6: *Trang* ***browser.php*** *được xem trong trình duyệt Firefox trên Windows XP.*

Đoạn mã 10.4: *Mã kịch bản thực hiện việc phát hiện trình duyệt.*

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
→ Transitional//EN" "http://www.w3.org/TR/2000/REC-xhtml1-
→20000126/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">

<head>

<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />
```

```
<title>Phat hien trinh duyet</title>
</head>
<body>
<?php # Doan ma 10.4 - browser.php
$browser = get_browser();
echo "<p>HTTP_USER_AGENT is
→ {$_SERVER['HTTP_USER_AGENT']}.</p>";
echo "<p>Ban dang dung phien ban $browser->version cua
→ $browser->browser tren he dieu hanh
→ $browser->platform.</p>";
?>
</body>
</html>
```

Để phát hiện chính xác tất cả các loại trình duyệt, bạn cần phải tải và cài đặt phiên bản mới nhất của ***browscap.ini*** *một cách thường xuyên.*

Để phát hiện trình duyệt một cách sâu sắc hơn, bạn nên sử dụng phpSniff có tại địa chỉ http://phpsniff.sourceforge.net.

Chỉ với các kỹ thuật được nêu ở đây, chúng ta không thể kiểm tra được việc trình duyệt có chấp nhận cookie hay không, JavaScript có được hỗ trợ hay không hoặc các plug-in nào đã được cài đặt.

Để vận hành tối ưu, bạn chỉ cần phát hiện trình duyệt một lần và sau đó sử dụng thông tin này cho tất cả các trang (không phát hiện trình duyệt cho mọi trang).

Phụ thuộc vào yêu cầu, chúng ta có thể phát hiện một số thông tin mà không cần đến ***browscap.ini****, bằng cách biên dịch giá trị* ***HTTP_USER_AGENT*** *với các biểu thức quy tắc hoặc hàm xử lý chuỗi.*

PHP và JavaScript

Mặc dù còn có những giới hạn nhất định và các vấn đề về tương thích trình duyệt, JavaScript vẫn là một công nghệ thông dụng trong các trang Web ngày nay. Thật không may, nhiều nhà lập trình PHP mới gặp nhiều khó khăn trong việc tương tác giữa PHP và JavaScript.

Khác biệt đáng kể nhất giữa hai ngôn ngữ ở chỗ JavaScript là một công nghệ phía khách hàng (chạy trong trình duyệt), trong khi đó PHP là công nghệ phía máy chủ. JavaScript có thể phát hiện kích thước trình duyệt theo pixel, tạo ra

cửa sổ pop-up và thực hiện các hiệu ứng Rollover, trong khi đó PHP lại không thể thực hiện những điều này. Mặc dù PHP không thể thực hiện những điều mà JavaScript có thể, nhưng nó lại được dùng để tạo ra mã lệnh JavaScript.

Gửi dữ liệu JavaScript đến PHP

Như một ví dụ trong phần này sẽ thể hiện, việc gửi dữ liệu từ PHP đến JavaScript khá dễ dàng: Chỉ cần tạo HTML. Tuy nhiên, việc truyền dữ liệu từ JavaScript đến PHP lại khá phức tạp. Có một số phương pháp để thực hiện điều này:

- *Sử dụng JavaScript để thiết lập cookie và sau đó lấy ra giá trị này từ một mã kịch bản PHP.*
- *Đưa biến và giá trị vào URL (ví dụ: script.php?var=val) với JavaScript, sử dụng JavaScript để chuyển trình duyệt Web đến URL đó.*
- *Dùng JavaScript để thiết lập giá trị của trường ẩn trong biểu mẫu HTML, rồi gửi biểu mẫu lên cho mã kịch bản PHP.*

Ví dụ, nếu muốn xác định kích thước cửa sổ trình duyệt, trước hết cần phát hiện thông tin này bằng JavaScript (không thể thực hiện với PHP). Nếu muốn truyền thông tin này đến một mã kịch bản PHP (để chứa trong một cơ sở dữ liệu), chúng ta sử dụng một trong những cách thức nêu trên để thực hiện công việc này. Chú ý, trong mỗi trường hợp, cần phải có một trang HTML sử dụng JavaScript để phát hiện trình duyệt và sử dụng một mã kịch bản PHP để lấy ra thông tin đó.

Thực hành tạo JavaScript với PHP theo các bước sau:

1. Bắt đầu một hồ sơ PHP mới trong trình soạn thảo văn bản.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
→ Transitional//EN" "http://www.w3.org/TR/2000/REC-
→xhtml1-20000126/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">

<head>

<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />

<title>Hinh anh</title>
```

Đầu tiên, mã kịch bản này sẽ liệt kê một danh sách hình ảnh, kích thước của chúng và liên kết để xem hình ảnh thực trong một cửa sổ pop-up. Cửa sổ pop-up này sẽ được tạo ra bởi JavaScript, tuy nhiên PHP sẽ được sử dụng để tạo ra một số tham số nhất định.

2. Bắt đầu hàm JavaScript.

```
<script language="JavaScript">

<!--

function create_window (image, width, height) {

   width = width + 25;

   height = height + 50;
```

Hàm JavaScript `create_window()` sẽ nhận vào ba tham số: Tên hình ảnh, chiều rộng và chiều cao của nó. Ba tham số này sẽ được gửi đến cho hàm mỗi khi người dùng nhấp vào một liên kết. Các giá trị này sẽ được xác định bởi mã lệnh PHP.

Một lượng pixel nhất định được bổ sung vào chiều rộng và chiều cao để làm cho cửa sổ pop-up rộng hơn hình ảnh một chút.

3. Thay đổi kích thước của cửa sổ pop-up nếu như nó đã được mở trước đó.

```
if (window.popup_window && !window.popup_window.closed) {

   window.popup_window.resizeTo (width, height);

}
```

Đầu tiên, mã lệnh sẽ kiểm tra xem cửa sổ pop-up có tồn tại hay không và nếu có thì nó đã bị đóng lại chưa. Nếu vượt qua được cả hai phép kiểm tra, cửa sổ này sẽ được điều chỉnh theo kích thước của hình ảnh mới. Mục đích của mã lệnh này là để thay đổi kích thước của cửa sổ hiện có cho phù hợp với các hình ảnh khác nhau.

4. Xác định các thuộc tính của cửa sổ pop-up và URL, sau đó tạo ra cửa sổ.

```
var window_specs = "location=no, scrollbars=no, menubars=no,
→ toolbars=no, resizable=yes, left=0, top=0, width=" +
→ width + ", height=" + height;

var url = "image_window.php?image=" + image;

popup_window = window.open(url, "PictureWindow",
→ window_specs);

popup_window.focus();
```

Dòng lệnh đầu tiên sẽ thiết lập các thuộc tính của cửa sổ pop-up (cửa sổ không có thanh công cụ, thanh cuộn, menu; nó có thể thay đổi kích thước và được đặt ở góc trên phía trái của màn hình; nó cũng có chiều rộng và chiều cao xác định). Dấu cộng được sử dụng để thực hiện việc kết hợp trong JavaScript, cho phép đưa giá trị các biến vào trong một chuỗi.

Dòng lệnh thứ hai thiết lập URL của cửa sổ pop-up (image_ window.php?image= cộng với tên của hình ảnh).

Cuối cùng, cửa sổ pop-up được tạo dựa trên các thuộc tính đã xác định. Tiêu điểm sẽ chuyển đến cửa sổ này, nghĩa là nó sẽ xuất hiện phía trên cửa sổ hiện tại.

5. Hoàn tất hàm JavaScript và phần đầu HTML.

```
}
//-->
</script>
</head>
```

6. Tạo phần văn bản hướng dẫn và bắt đầu bảng.

```
<body>
<div align="center">Nhap vao tieu de de xem hinh anh.
→ </div><br />
<table align="center" cellspacing="3" cellpadding="3"
→ border="1">
<tr>
<td align="center"><b>Tieu de</b></td>
<td align="center"><b>Kich thuoc</b></td>
</tr>
```

7. Bắt đầu mã lệnh PHP và tạo một mảng hình ảnh.

```
<?php # Doan ma 10.5 - images.php
$images['trixie.jpg'] = 'Con cho Trixie.';
$images['guggenheim.jpg'] = 'Vien bao tang.';
```

Mảng `$images` chứa tất cả các hình ảnh sẽ được hiển thị trên trang Web. Nó nhận vào tên hình ảnh làm khóa và một tiêu đề làm giá trị. Như vậy, tiêu đề *"Con cho Trixie."* sẽ được áp dụng cho hình ảnh *trixie.jpg*. Bằng cách này, bạn có thể bổ sung tùy ý các hình ảnh mình muốn vào trong mảng.

8. Duyệt qua mảng `$image`, liên kết từng hình ảnh với hàm JavaScript.

```
foreach ($images as $i => $c) {
  $image_size = getimagesize ('images/' . $i);
  $file_size = round ( (filesize ('images/' . $i)) /
  → 1024) . "k";
  echo "<tr>
  → <td><a href=\"javascript: create_window ('$i',
  → $image_size[0], $image_size[1])\">$c</a></td>
```

```
    → <td>$file_size</td>
    → </tr>";
}
```

Vòng lặp này sẽ duyệt qua từng hình ảnh trong mảng và tạo ra một dòng trong bảng. Có hai hàm PHP chưa được sử dụng trước đây (tham khảo tài liệu PHP để biết thêm chi tiết): **getimagesize()** và **filesize()**. Hàm **getimagesize()** trả lại một mảng cho biết chiều rộng (chỉ số 0) và chiều cao (chỉ số 1), loại hình ảnh (chỉ số 2) cùng với một chuỗi mã lệnh HTML ở dạng height="xx" width="yy" (chỉ số 3) của một hình ảnh. Các giá trị trả về bởi hàm này sẽ được dùng để thiết lập chiều rộng và chiều cao được gửi cho hàm JavaScript **create_window()**.

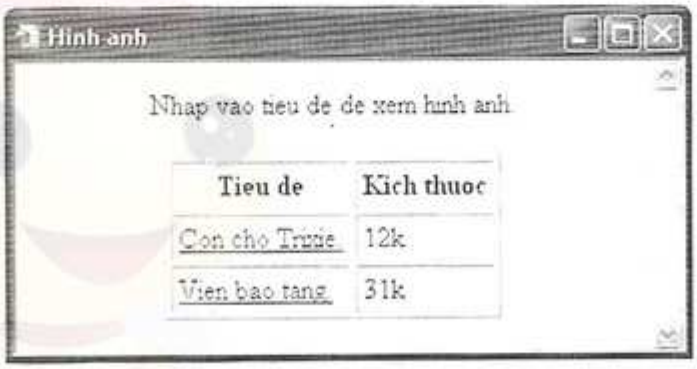
Hàm **filesize()** trả lại kích thước của một tập tin tính theo byte. Để tính theo đơn vị kylobyte, chúng ta chỉ cần chia con số này cho 1024 và làm tròn nó.

Cuối cùng, vòng lặp tạo ra một hàng trong bảng để thể hiện tiêu đề của hình ảnh và kích thước của nó. Tiêu đề sẽ được liên kết với lời gọi hàm JavaScript **create_window()** để khi người dùng nhấp vào nó, hàm này sẽ được thực hiện.

9. Hoàn tất mã lệnh PHP và trang HTML.

```
?>

</table>

</body>

</html>
```

10. Lưu lại tập tin với tên **images.php**, tải nó lên máy chủ Web cùng với các hình ảnh vào trong thư mục images rồi chạy thử nó trong trình duyệt (xem hình 10-7). Mã lệnh đầy đủ của tập tin này được thể hiện trong đoạn mã 10.5.



Hình 10-7: *Kết quả của mã kịch bản* ***images.php*** *(tham khảo đoạn mã 10.5).*

Đoạn mã 10.5: *Mã kịch bản* ***images.php*** *sẽ hiển thị các liên kết (dưới dạng chữ hoa) đến các hình ảnh chứa trên máy chủ.*

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
→ Transitional//EN" "http://www.w3.org/TR/2000/REC-xhtml1-
→20000126/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">
<head>
<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />
<title>Hinh anh</title>
<script language="JavaScript">
<!--
function create_window (image, width, height) {
   width = width + 25;
   height = height + 50;
   if (window.popup_window && !window.popup_window.closed){
      window.popup_window.resizeTo (width, height);
   }
   var window_specs = "location=no, scrollbars=no,
   → menubars=no, toolbars=no, resizable=yes, left=0, top=0,
   → width=" + width + ", height=" + height;
   var url = "image_window.php?image=" + image;
   popup_window = window.open(url, "PictureWindow",
   → window_specs);
   popup_window.focus();
}
//-->
</script>
</head>
<body>
<div align="center">Nhap vao tieu de de xem hinh anh.</div>
<br />
<table align="center" cellspacing="3" cellpadding="3"
→ border="1">
<tr>
<td align="center"><b>Tieu de</b></td>
```

```
<td align="center"><b>Kich thuoc</b></td>
</tr>
<?php # Doan ma 10.5 - images.php
$images['trixie.jpg'] = 'Con cho Trixie.';
$images['guggenheim.jpg'] = 'Vien bao tang.';
foreach ($images as $i => $c) {
   $image_size = getimagesize ('images/' . $i);
   $file_size = round ( (filesize ('images/' . $i)) / 1024) .
   → "k";
   echo "<tr>
   <td><a href=\"javascript: create_window ('$i',
   → $image_size[0], $image_size[1])\">$c</a></td>
   →<td>$file_size</td></tr>";
}
?>
</table>
</body>
</html>
```

Thực hành tạo mã kịch bản **image_window.php** theo các bước sau:

1. Bắt đầu một hồ sơ PHP mới trong trình soạn thảo văn bản.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
→ Transitional//EN" "http://www.w3.org/TR/2000/REC-
→xhtml1-20000126/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">
<head>
<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />
<title>Xem hinh anh</title>
</head>
<body>
```

2. Bắt đầu phần PHP và tạo ra một biến mới.

```
<?php # Doan ma 10.6 - image_window.php
$okay = FALSE;
```

Biến **$okay** được dùng để kiểm tra hình ảnh sẽ hiển thị. Nếu không có hình ảnh (do mã kịch bản không nhận được tên hình ảnh hoặc vì hình ảnh không thể truy xuất được), **$okay** được thiết lập thành FALSE và một thông báo lỗi sẽ hiển thị.

3. Kiểm tra tên hình ảnh sẽ được hiển thị.

```
if (isset($_GET['image'])) {
   $ext = substr ($_GET['image'], -4);
   if ($ext == '.png' OR $ext == '.jpg' OR $ext == 'jpeg'
   → OR $ext == '.gif') {
```

Trước khi truy xuất hình ảnh trên máy chủ, chúng ta sẽ kiểm tra nó hai lần. Điều kiện thứ nhất kiểm tra xem tên hình ảnh có được gửi đến thông qua URL không. Điều kiện thứ hai đảm bảo tên hình ảnh (tức là tên tập tin) phải có phần mở rộng hợp lệ (`.png`, `.jpg`, `.jpeg` hoặc `.gif`) bằng cách kiểm tra bốn ký tự cuối của tên tập tin nhận được (chú ý, bốn ký tự cuối bao gồm cả *jpeg* mà không có dấu chấm).

4. Hiển thị hình ảnh.

```
if ($image = @getimagesize ('images/' . $_GET['image'])){
   echo "<img src=\"images/{$_GET['image']}\" $image[3]
   → border=\"2\" />";
   $okay = TRUE;
}
```

Nếu mã kịch bản có thể lấy ra được các thông tin, hình ảnh sẽ được hiển thị (bằng cách tạo ra mã lệnh HTML thích hợp) và biến `$okay` được thiết lập thành TRUE.

5. Hoàn tất phần điều kiện và in ra thông báo lỗi nếu có.

```
   }
}
if (!$okay) {
   echo '<div align="center"><font color="red" size="+1">
   → Ma kich ban nay phai nhan duoc ten hinh anh hop le!
   → </font></div>';
}
```

Nếu biến `$okay` có giá trị FALSE, nghĩa là có một vấn đề nào đó trong quá trình xử lý, khi đó mã kịch bản sẽ in thông báo lỗi.

6. Hoàn tất mã lệnh PHP và trang HTML.

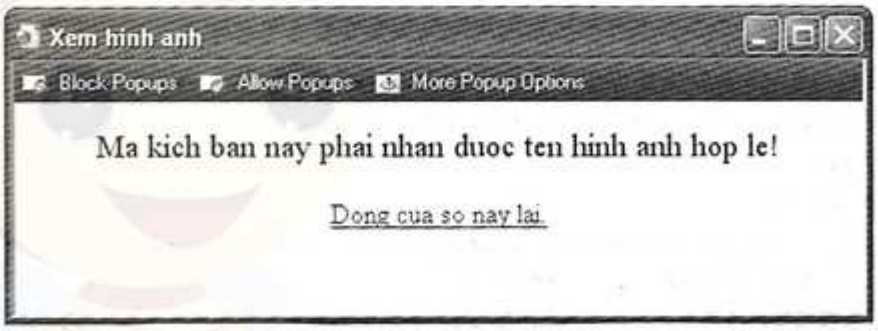
```
?>
<br />
<div align="center"><a href="javascript:self.close();">Dong
→ cua so nay lai.</a></div>
</body>
</html>
```

Phần cuối cùng này sẽ tạo ra một liên kết, mà khi được nhấp sẽ đóng cửa sổ pop-up lại.

7. Lưu tập tin với tên gọi **image_window.php**, tải nó lên máy chủ Web (trong cùng thư mục với **images.php**) và chạy thử trong trình duyệt (xem hình 10-8). Mã lệnh đầy đủ của tập tin này được thể hiện trong đoạn mã 10.6.

Hình 10-8: *Một hình ảnh được hiển thị trong cửa sổ pop-up.*

Nếu có bất kỳ vấn đề nào phát sinh trong quá trình xử lý, mã kịch bản **image_window.php** sẽ hiển thị một thông báo như trong hình 10-9.



Hình 10-9: *Mã kịch bản* ***image_window.php*** *hiển thị thông báo lỗi khi không nhận được tên hình ảnh hợp lệ.*

Đoạn mã 10.6: *Mã kịch bản này sẽ hiển thị một hình ảnh bằng cách sử dụng hàm* ***getimagesize()*** *để xác định kích thước của nó.*

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
→ Transitional//EN" "http://www.w3.org/TR/2000/REC-xhtml1-
→20000126/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">
<head>
<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />
<title>Xem hinh anh</title>
</head>
<body>
<?php # Doan ma 10.6 - image_window.php
$okay = FALSE;
if (isset($_GET['image'])) {
   $ext = substr ($_GET['image'], -4);
   if ($ext == '.png' OR $ext == '.jpg' OR $ext == 'jpeg' OR
   → $ext == '.gif') {
      if ($image = @getimagesize ('images/'
      → $_GET['image'])){
         echo "<img src=\"images/{$_GET['image']}\" $image[3]
         → border=\"2\" />";
         $okay = TRUE;
      }
   }
}
if (!$okay) {
   echo '<div align="center"><font color="red" size="+1">Ma
   → kich ban nay phai nhan duoc ten hinh anh hop le!
   →</font></div>';
}
?>
<br />
<div align="center"><a href="javascript:self.close();">Dong
→ cua so nay lai.</a></div>
</body>
</html>
```

Không phải tất cả những gì có thể thực hiện với JavaScript (thay đổi kích thước, di chuyển cửa sổ trình duyệt) đều có thể thực hiện được với PHP.

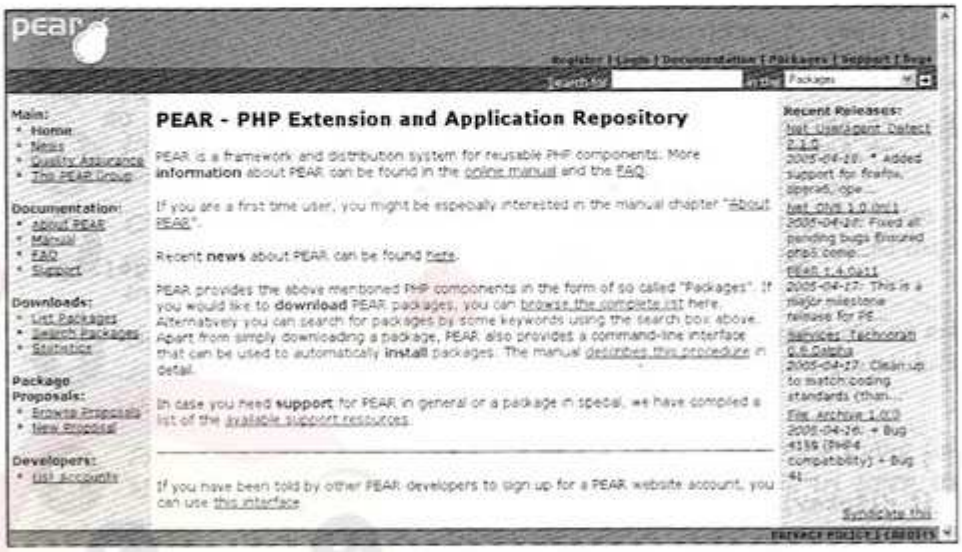
Có một số trùng lặp giữa hai ngôn ngữ PHP và JavaScript. Cả hai đều có thể thiết lập và đọc cookie, tạo HTML và phát hiện trình duyệt.

Hàm `file_exists()` của PHP (kiểm tra sự tồn tại của một tập tin trên máy chủ), cũng có thể được sử dụng trong cả hai mã kịch bản này để kiểm tra sự hiện diện của hình ảnh trước khi tạo liên kết hoặc hiển thị nó.

Về mặt kỹ thuật, JavaScript đã được đổi tên thành ECMAScript, tuy nhiên mọi người vẫn quen gọi nó là JavaScript.

Sử dụng PEAR

PEAR (viết tắt của PHP Extension and Application Repository) đóng vai trò như là một thư viện mã lệnh dùng chung, cho phép các nhà phát triển có được các tính năng phức tạp mà không cần phải viết mã PHP nhiều. Website chính thức của PEAR là *http://pear.php.net* (xem hình 10-10).



Hình 10-10: *Trang chủ của PEAR.*

Nhiều điều thực hiện trong cuốn sách này có thể được thực hiện bằng cách sử dụng các môđun PEAR, bao gồm:

- Tương tác với MySQL
- Ghi trang vào bộ đệm
- Gửi thư điện tử
- Kiểm tra xác thực HTTP

Mỗi gói PEAR (hoặc lớp) đều có chứa các mô tả về những gì mà nó thực hiện. Chúng cũng thường cung cấp các ví dụ về cách sử dụng phù hợp cho các nhà lập trình PHP trung bình, ngay cả khi họ chưa được đào tạo chính thức về lập trình hướng đối tượng. Để minh họa cho việc sử dụng PEAR, chúng ta sẽ bổ sung một chức năng đo tốc độ thực hiện của mã kịch bản hiện có để xem nó diễn ra trong bao lâu.

Cài đặt PEAR

PEAR có một trình quản lý cài đặt riêng. Để cài đặt nó trên hệ thống Unix, chúng ta có thể thực hiện một trong hai lệnh sau từ chế độ dòng lệnh:

```
lynx -source http://go-pear.org | php
```

```
curl -source http://go-pear.org | php
```

Một khi trình quản lý của PEAR đã được cài đặt, chúng ta có thể cài đặt các gói cụ thể bằng cách dùng dòng lệnh:

```
pear install <tên gói>
```

Thật không may, điều này chỉ có thể thực hiện nếu đã cài đặt phiên bản nhị phân của PHP và hiện tại nó không có hiệu lực trên Windows.

Mặc dù việc cài đặt các gói bằng cách sử dụng trình quản lý gói PEAR rất dễ dàng, nhưng cách an toàn nhất lại là tải các gói thích hợp từ Website PEAR và đưa các tập tin vào trong cùng thư mục với mã kịch bản PHP có sử dụng chúng. Để cài đặt tập tin cần cho ví dụ của phần này, bạn hãy truy xuất vào địa chỉ http://pear.php.net, nhấp vào liên kết List Packages và sau đó nhấp vào Benchmark. Trên trang kết quả, nhấp vào mục Download và tải xuống phiên bản mới nhất. Tiếp đến, xả nén và chép các tập tin vào thư mục Web của mình.

Thực hành sử dụng PEAR theo các bước sau:

1. Tạo một hồ sơ PHP mới trong trình soạn thảo văn bản.

```
<?php # Doan ma 10.7 - view_users.php
```

Phần lớn mã kịch bản này giống với đoạn mã **view_users.php** đã viết trong Chương 6. Chúng ta sẽ bổ sung một số dòng lệnh để tính thời gian thực hiện của mã kịch bản.

2. Tạo ra một bộ tính giờ.

```
require_once ('Benchmark/Timer.php');

$t = new Benchmark_Timer;

$t->start();
```

Để sử dụng lớp PEAR, đầu tiên chúng ta cần có tập tin thích hợp (xem phần nói về cách cài đặt và sử dụng PEAR). Sau đó, thực hiện theo ví dụ được thể hiện

trong mã kịch bản **Timer.php** (là một phần của gói **Benchmark**), để tạo một đối tượng với tên là **$t**, rồi khởi động bộ tính giờ này.

3. Đưa vào tập tin đầu trang HTML và thiết lập một mốc tính giờ.

```
$page_title = 'Xem nguoi dung hien tai';

include ('templates/header.inc');

$t->setMarker ('Dua vao tap tin header.inc!');
```

Đầu tiên, chúng ta sẽ tính xem việc đưa tập tin đầu trang vào hết bao lâu. Để thực hiện điều này, ta thiết lập một mốc tính giờ bằng cách sử dụng một tên mốc thích hợp cùng với hàm **setMarker()**.

4. Truy vấn cơ sở dữ liệu và thiết lập một mốc tính giờ khác.

```
require_once ('../mysql_connect.php');

$query = "SELECT CONCAT(last_name, ', ', first_name)
→ AS name, DATE_FORMAT(registration_date, '%M %d, %Y')
→ AS dr FROM users ORDER BY registration_date ASC";

$result = @mysql_query ($query);

$num = mysql_num_rows ($result);

$t->setMarker ('Truy van CSDL!');
```

Điểm mốc thứ hai sẽ được dùng để xác định thời gian kết nối cơ sở dữ liệu và thực hiện câu truy vấn.

5. Lấy ra kết quả truy vấn và thiết lập một mốc tính giờ khác.

```
if ($num > 0) {

   echo "<p><big><b>Hien co $num nguoi dung.</b>
   →</big></p>";

   echo '<table align="center" cellspacing="2"
   → cellpadding="2">

   <tr><td align="left"><b>Ten</b></td><td align="left">
   →<b>Ngay dang ky</b></td></tr>';

   while ($row = mysql_fetch_array($result, MYSQL_NUM)) {

      echo "<tr><td align=\"left\">" .
      → stripslashes($row[0])
      → . "</td><td align=\"left\">$row[1]</td></tr>\n";

}

echo '</table>';

mysql_free_result ($result);
```

```
$t->setMarker ('In ra ket qua!');
```

Phần điều kiện và phần lấy ra kết quả truy vấn giống như đã thực hiện trong Chương 6. Sau đó, chúng ta thiết lập một mốc tính giờ nữa để xem quá trình này diễn ra trong bao lâu.

6. Hoàn tất điều kiện và ngừng bộ tính giờ.

```
} else {
   echo '<p>Hien chua co ai dang ky.</p>';
}
mysql_close();
include ('templates/footer.inc');
$t->stop();
$result = $t->getProfiling();
```

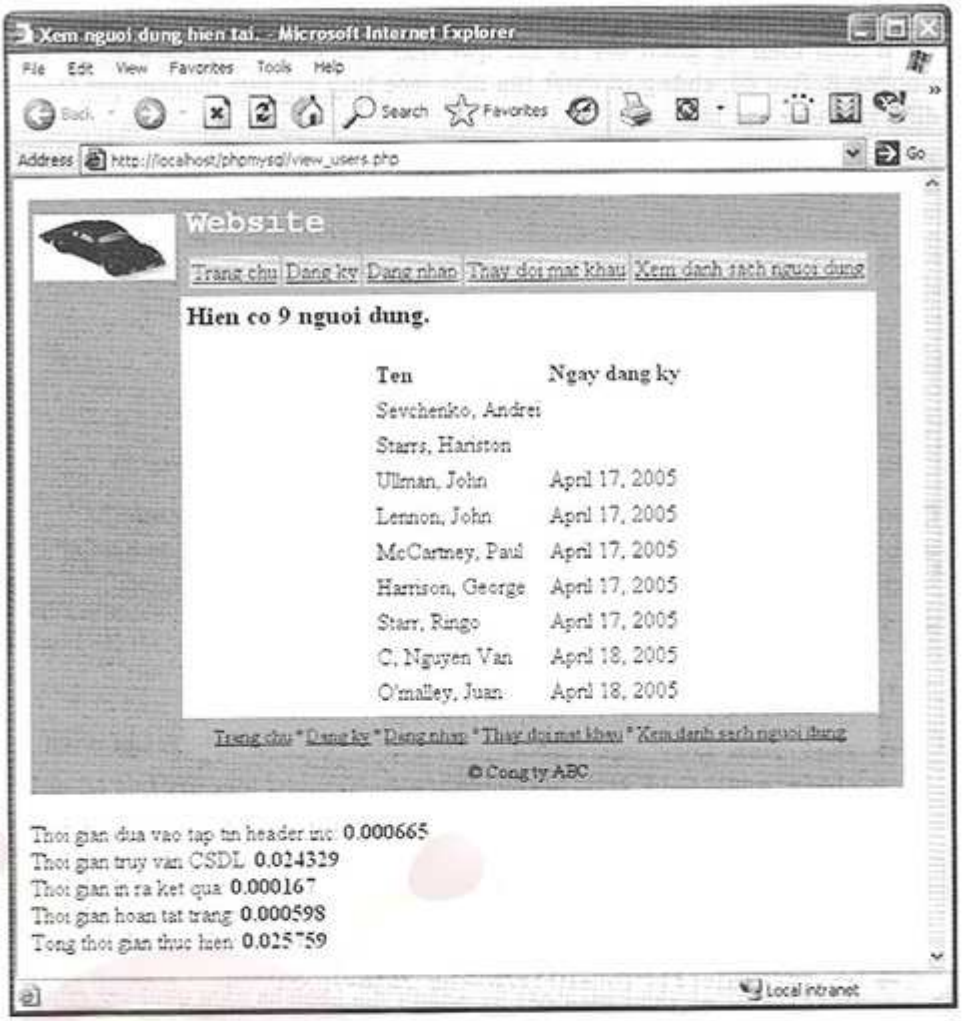
Chức năng `stop()` của bộ tính giờ sẽ làm ngưng quá trình. Sau khi đã ngừng bộ tính giờ, một lời gọi chức năng `getProfiling()` sẽ trả lại một mảng nhiều chiều với kết quả tính giờ của mã kịch bản.

7. In kết quả và hoàn tất trang PHP.

```
echo "<p>Thoi gian dua vao tap tin header.inc:
→ <b>{$result[1]['diff']}</b><br/>\n
→ Thoi gian truy van CSDL:
→ <b>{$result[2]['diff']}</b><br/>\n
→ Thoi gian in ra ket qua:
→ <b>{$result[3]['diff']}</b><br/>\n
→ Thoi gian hoan tat trang:
→ <b>{$result[4]['diff']}</b><br/>\n
→ Tong thoi gian thuc hien:
→ <b>{$result[4]['total']}</b><br />\n</p>";

?>
```

Biến `$result` là một mảng nhiều chiều, trong đó phần tử đầu tiên (ở chỉ số 0) ứng với thời điểm ban đầu và các phần tử tiếp theo ứng với các mốc tính thời gian. Vì thế `$result[1]` sẽ ứng với mốc **header.inc**, `$result[2]` ứng với mốc truy vấn cơ sở dữ liệu... Bản thân mỗi phần tử là một mảng gồm bốn phần tử: *name* (tên của mốc thời gian), *time* (thời gian thiết lập mốc), *diff* (hiệu số thời gian lúc thiết lập mốc với thời điểm thiết lập mốc trước đó hoặc so với lúc bắt đầu bộ tính giờ) và *total* (tổng thời gian tính từ lúc bắt đầu bộ tính giờ).

Với các thông tin này, chúng ta sẽ in lượng thời gian ứng với từng mốc thời gian bằng cách sử dụng `$result[x]['diff']`, trong đó x là chỉ số của mốc tính giờ. Thời gian ứng với mốc tính giờ cuối cùng cũng chính là tổng thời gian cần để thực hiện toàn bộ mã kịch bản.



Hình 10-11: *Kết quả đo thời gian được in ra ở cuối màn hình.*

8. Lưu tập tin với tên gọi **view_users.php**, tải nó lên máy chủ (trong cùng thư mục với các tập tin khác trong Chương 6) và chạy thử trong trình duyệt (xem hình 10-11). Mã lệnh đầy đủ của tập tin này được thể hiện trong đoạn mã 10.7.

Đoạn mã 10.7: *Lớp Benchmark của PEAR được dùng để xác định thời gian thực hiện một mã kịch bản PHP.*

```
<?php # Doan ma 10.7 - view_users.php

require_once ('Benchmark/Timer.php');
```

```
$t = new Benchmark_Timer;

$t->start();

$page_title = 'Xem nguoi dung hien tai.';

include ('templates/header.inc');

$t->setMarker ('Dua vao tap tin header.inc!');

require_once ('../mysql_connect.php');

$query = "SELECT CONCAT(last_name, ', ', first_name) AS
→ name, DATE_FORMAT(registration_date, '%M %d, %Y') AS dr
→ FROM users ORDER BY registration_date ASC";

$result = @mysql_query ($query);

$num = mysql_num_rows ($result);

$t->setMarker ('Truy van CSDL!');

if ($num > 0) {

   echo "<p><big><b>Hien co $num nguoi dung.</b></big></p>";

   echo '<table align="center" cellspacing="2"
   → cellpadding="2"><tr><td align="left"><b>Ten</b></td>
   →<td align="left"><b>Ngay dang ky</b></td></tr>';

   while ($row = mysql_fetch_array($result, MYSQL_NUM)) {

      echo "<tr><td align=\"left\">" . stripslashes($row[0]) .
      → "</td><td align=\"left\">$row[1]</td></tr>\n";

   }

   echo '</table>';

   mysql_free_result ($result);

   $t->setMarker ('In ra ket qua!');

} else {

   echo '<p>Hien chua co ai dang ky.</p>';

}

mysql_close();

include ('templates/footer.inc');

$t->stop();

$result = $t->getProfiling();
```

```
echo "<p>Thoi gian dua vao tap tin header.inc:
→ <b>{$result[1]['diff']}</b><br />\n
→ Thoi gian truy van CSDL: <b>{$result[2]['diff']}</b><br/>\n
→ Thoi gian in ra ket qua: <b>{$result[3]['diff']}</b><br/>\n
→ Thoi gian hoan tat trang: <b>{$result[4]['diff']}</b><br/>\n
→ Tong thoi gian thuc hien: <b>{$result[4]['total']}</b>
→ <br/>\n</p>";

?>
```

PEAR là một đặc điểm được hỗ trợ chính thức từ PHP 4.3.

Để xem danh sách các gói hiện có, bạn có thể chạy lệnh **`pear list-remote-packages`** *trong chế độ dòng lệnh (cần phải cài đặt trình quản lý PEAR trước).*

Một số gói PEAR sẽ không làm việc trên hệ điều hành Windows vì chúng sử dụng các tính năng riêng của Unix. subset

Có một tập con của PEAR có tên là PECL ("pickle" - PHP Extension Code Library). Đây là các môđun chuẩn của PEAR được viết trực tiếp bằng C và C+ (giúp cải thiện đáng kể tốc độ), thay vì bằng PHP. Khi PHP 5 ra đời, phần lớn PEAR đều là PECL.

Một điểm quan trọng của PEAR là các môđun được viết theo một chuẩn nhất định (tên biến, cấu trúc hồ sơ...). Ngay cả khi bạn không có ý định bổ sung vào thư viện PEAR, việc thực hiện theo các chuẩn này cũng là một điều đáng quan tâm.

Hồ sơ PEAR trực tuyến có trên các Website về PEAR, các nhóm trao đổi thư điện tử (mailing list) và các tài liệu liên quan.

Nếu thực sự quan tâm đến việc điều chỉnh để cải thiện tính vận hành cho các ứng dụng, bạn nên sử dụng lớp Benchmark để tính thời gian thực hiện cho mỗi mã kịch bản. Sau đó, tập trung vào chỉnh sửa những phần chiếm nhiều thời gian thực hiện nhất (ví dụ, truy vấn cơ sở dữ liệu trong mã kịch bản).

Thông tin chi tiết về OOP PHP (nền tảng của PEAR và một đặc điểm quan trọng của PHP 5) có thể tìm thấy tại nhiều địa chỉ được nêu trong Phụ lục D.

Lớp về cơ sở dữ liệu của PEAR là một trong những công cụ phổ biến nhất, cho phép phát triển các ứng dụng Web mà không phụ thuộc vào loại cơ sở dữ liệu.

Chương 11:

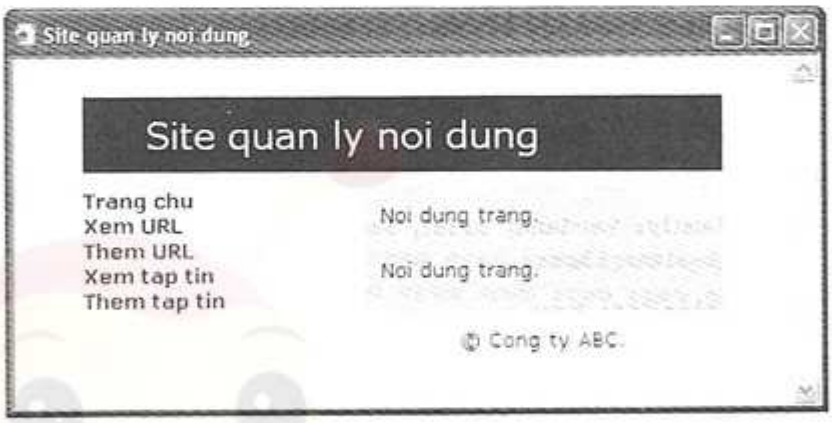
VÍ DỤ - QUẢN LÝ NỘI DUNG

Ứng dụng đầu tiên được trình bày trong cuốn sách là một site quản lý nội dung. Ứng dụng này quản lý các URL và những tập tin mà người dùng có thể bổ sung, hiển thị và truy cập. Ví dụ này không tạo phần quản trị của site, tuy nhiên có đưa vào những chú thích để hoàn thành nó.

Mặc dù trọng tâm trong chương này là về ví dụ cụ thể, nhưng một số hàm và kỹ thuật mới cũng được đề cập, bao gồm hàm `mysql_insert_id()` và `list()`, cũng như cách xử lý tải lên tập tin.

Tạo khuôn mẫu

Bước đầu tiên trong phát triển ứng dụng này là tạo một hệ thống khuôn mẫu để xử lý thiết kế HTML. Kết quả cuối cùng (xem hình 11-1) sẽ dùng bảng và Cascading Style Sheets (CSS).



Hình 11-1: *Bố cục mặc định được dùng cho ứng dụng trong chương này.*

Thực hành tạo tập tin **header.html** theo các bước sau:

1. Tạo hồ sơ HTML mới trong trình soạn thảo văn bản.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
→ Transitional//EN" "http://www.w3.org/TR/2000/REC-xhtml1-
20000126/DTD/xhtml1-transitional.dtd">
```

```
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">

<head>

<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />

<title><?php echo $page_title; ?></title>
```

2. Tạo mã CSS.

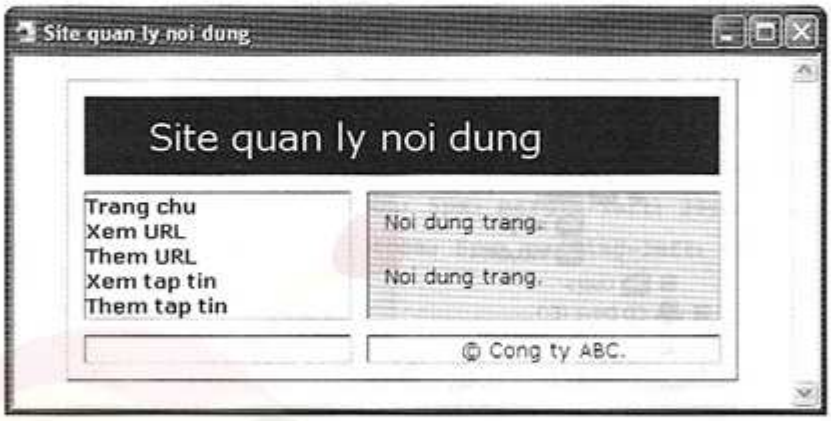
```
<style type="text/css" media="screen">

body { background-color: #ffffff; }

.content {

   background-color: #f5f5f5;

   padding-top: 10px; padding-right: 10px; padding-bottom:
   → 10px; padding-left: 10px;

   margin-top: 10px; margin-right: 10px; margin-bottom:
   → 10px; margin-left: 10px;

}

a.navlink:link { color: #003366; text-decoration: none; }

a.navlink:visited { color: #003366; text-decoration: none; }

a.navlink:hover { color: #cccccc; text-decoration: none; }

td {

   font-family: Verdana, Arial, Helvetica, sans-serif;
   → font-size: 13px;

   vertical-align: top;

}

.title {

   font-size: 24px; font-weight: normal; color: #ffffff;

   margin-top: 5px; margin-bottom: 5px; margin-left: 20px;

   padding-top: 5px; padding-bottom: 5px; padding-left:
   → 20px;

}

</style>
```

Trong ứng dụng này, chúng ta sẽ dùng một số mã lệnh CSS để điều chỉnh định dạng văn bản. Vì không có nhiều mã lệnh CSS, nên ta lưu nó trực tiếp trong phần **HEAD** của mã HTML, chứ không đặt trong một tập tin riêng.

3. Hoàn tất phần đầu trang, tạo hàng tiêu đề.

```
</head>

<body>

<table width="90%" border="0" cellspacing="10"
→ cellpadding="0" align="center">

<tr>

<td colspan="2" bgcolor="#003366"><p class="title">Site
→ quan ly noi dung</p></td>

</tr>
```

Bố cục cơ bản của site là bảng gồm ba hàng: hàng tiêu đề, hàng chứa các liên kết di chuyển và nội dung, cuối cùng hàng bản quyền (xem hình 11-2).



Hình 11-2: *Bố cục trang được hiển thị đường biên để thể hiện hàng và cột của bảng.*

4. Tạo vùng di chuyển và khởi đầu ô nội dung.

```
<tr>
<td valign="top" nowrap="nowrap">
<b><a href="index.php" class="navlink">Trang chu</a><br />
<a href="view_urls.php" class="navlink">Xem URL</a><br />
<a href="add_url.php" class="navlink">Them URL</a><br />
<a href="view_files.php" class="navlink">Xem tap tin</a>
→ <br/>
```

```
<a href="add_file.php" class="navlink">Them tap tin</a></b>
</td>
<td valign="top" class="content">
<!-- Doan ma 11.1 - header.html -->
```

5. Lưu tập tin với tên **header.html** và tải lên máy chủ Web (đưa vào thư mục *includes*). Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 11.1.

Tham khảo hình 11-3 để biết trước cấu trúc hồ sơ Web.

Hình 11-3: *Cấu trúc của ứng dụng Web, với thư mục gốc là html.*

Đoạn mã 11.1: *Tập tin đầu trang khởi đầu bố cục HTML và đưa vào mã CSS cần thiết.*

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
→ Transitional//EN" "http://www.w3.org/TR/2000/REC-xhtml1-
→ 20000126/DTD/xhtml1-Transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">
<head>
<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />
```

```
<title><?php echo $page_title; ?></title>
<style type="text/css" media="screen">
body { background-color: #ffffff; }
.content {
   background-color: #f5f5f5;
   padding-top: 10px; padding-right: 10px; padding-bottom:
   → 10px; padding-left: 10px;
   margin-top: 10px; margin-right: 10px; margin-bottom: 10px;
   → margin-left: 10px;
}
a.navlink:link { color: #003366; text-decoration: none; }
a.navlink:visited { color: #003366; text-decoration: none; }
a.navlink:hover { color: #cccccc; text-decoration: none; }
td {
   font-family: Verdana, Arial, Helvetica, sans-serif;
   → font-size: 13px;
   vertical-align: top;
}
.title {
   font-size: 24px; font-weight: normal; color: #ffffff;
   margin-top: 5px; margin-bottom: 5px; margin-left: 20px;
   padding-top: 5px; padding-bottom: 5px; padding-left: 20px;
}
</style>
</head>
<body>
<table width="90%" border="0" cellspacing="10"
→ cellpadding="0" align="center">
<tr>
<td colspan="2" bgcolor="#003366"><p class="title">Site quan
→ ly noi dung</p></td>
</tr>
<tr>
<td valign="top" nowrap="nowrap">
<b><a href="index.php" class="navlink">Trang chu</a><br />
<a href="view_urls.php" class="navlink">Xem URL</a><br />
<a href="add_url.php" class="navlink">Them URL</a><br />
```

```
<a href="view_files.php" class="navlink">Xem tap tin</a>
<br/>
<a href="add_file.php" class="navlink">Them tap tin</a></b>
</td>
<td valign="top" class="content">
<!-- Doan ma 11.1 - header.html -->
```

Ví dụ trong chương sau sẽ dùng mã CSS phức tạp hơn. Khi đó mã CSS sẽ được lưu trong tập tin ngoài và sẽ được đưa vào tập tin ***header.html****.*

Thực hành tạo tập tin **footer.html** theo các bước sau:

1. Tạo hồ sơ HTML mới trong trình soạn thảo văn bản.

```
<!-- Doan ma 11.2 - footer.html -->
```

2. Hoàn tất hàng giữa bảng.

```
</td>
</tr>
```

Toàn bộ nội dung trang được đặt trong hàng giữa, được bắt đầu từ trong tập tin **header.html**. Mã lệnh này sẽ hoàn tất hàng đó trước khi bắt đầu hàng thứ ba (hàng cuối cùng trong bảng).

3. Tạo hàng cuối và hoàn tất trang HTML.

```
<tr>
<td> </td>
<td align="center">&copy; Cong ty ABC.</td>
</tr>
</table>
</body>
</html>
```

4. Lưu tập tin với tên **footer.html** và tải lên máy chủ Web (đặt trong thư mục *includes)*. Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 11.2.

Đoạn mã 11.2: *Tập tin cuối trang kết thúc bố cục và toàn bộ trang HTML.*

```
<!-- Doan ma 11.2 - footer.html -->
</td>
</tr>
<tr>
```

```
<td> </td>
<td align="center">&copy; Cong ty ABC.</td>
</tr>
</table>
</body>
</html>
```

Tạo các trang cơ bản

Trước khi đi vào phần trọng tâm của ứng dụng (phần quản trị nội dung), chúng ta cần tạo thêm hai trang. Trang thứ nhất (**index.php**) đóng vai trò trang chủ của site. Trang thứ hai (**mysql_connect.php**) là đoạn mã kết nối MySQL và chọn cơ sở dữ liệu cần dùng (xem phần *"Lược đồ cơ sở dữ liệu"* để biết thêm thông tin về cơ sở dữ liệu dùng cho chương này).

Thực hành tạo trang chủ theo các bước sau:

1. Tạo đoạn mã PHP mới trong trình soạn thảo văn bản.

```
<?php # Doan ma 11.3 - index.php
```

2. Thiết lập tiêu đề trang và đưa vào phần đầu HTML.

```
$page_title = 'Site quan ly noi dung';

include_once ('includes/header.html');

?>
```

3. Tạo nội dung.

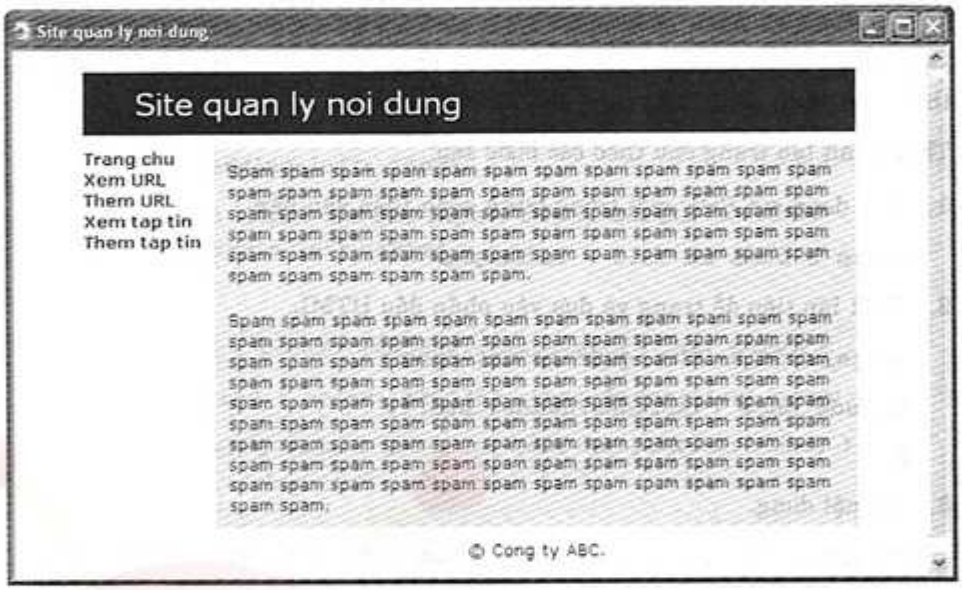
```
<p>Spam spam spam spam spam spam spam spam spam spam spam
spam spam spam spam spam spam spam spam spam spam spam spam
spam spam spam spam spam spam spam spam spam spam spam spam
spam spam spam spam spam spam spam spam spam spam spam spam
spam spam spam spam spam spam spam spam spam spam spam spam
spam spam spam spam spam spam.</p>

<p>Spam spam spam spam spam spam spam spam spam spam spam
spam spam spam spam spam spam spam spam spam spam spam spam
spam spam spam spam spam spam spam spam spam spam spam spam
spam spam spam spam spam spam spam spam spam spam spam spam
spam spam spam spam spam spam spam spam spam spam spam spam
spam spam spam spam spam spam spam spam spam spam spam spam
spam spam spam spam spam spam spam spam spam spam spam spam
spam spam spam spam spam spam spam spam spam spam spam spam
spam spam spam.</p>
```

Trong ví dụ này chúng ta dùng từ *spam* để minh họa, tuy nhiên bạn có thể đưa vào nội dung bất kỳ.

4. Đưa vào phần cuối trang HTML.

```
<?php

include_once ('includes/footer.html');

?>
```

5. Lưu tập tin với tên index.php, tải lên máy chủ Web và chạy thử trong trình duyệt (xem hình 11-4). Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 11.3.



Hình 11-4: *Trang chủ của ứng dụng.*

Đoạn mã 11.3: *Trang chủ của ứng dụng.*

```
<?php # Doan ma 11.3 - index.php
$page_title = 'Site quan ly noi dung';
include_once ('includes/header.html');
?>
<p>Spam spam spam spam spam spam spam spam spam spam spam
spam spam spam spam spam spam spam spam spam spam spam
spam spam spam spam spam spam spam spam spam spam spam
spam spam spam spam spam spam spam spam spam spam spam
spam spam spam spam spam spam spam spam spam spam spam
```

```
spam spam spam spam spam spam spam.</p>
<p>Spam spam spam spam spam spam spam spam spam spam spam
spam spam spam spam spam spam spam spam spam spam spam spam
spam spam spam spam spam spam spam spam spam spam spam spam
spam spam spam spam spam spam spam spam spam spam spam spam
spam spam spam spam spam spam spam spam spam spam spam spam
spam spam spam spam spam spam spam spam spam spam spam spam
spam spam spam spam spam spam spam spam spam spam spam spam
spam spam spam spam spam spam spam spam spam spam spam spam
spam spam spam spam spam spam spam spam spam spam spam spam
spam spam spam.</p>
<?php
include_once ('includes/footer.html');
?>
```

Thực hành tạo tập tin **mysql_connect.php** theo các bước sau:

1. Tạo đoạn mã PHP mới trong trình soạn thảo văn bản.

```
<?php # Doan ma 11.4 - mysql_connect.php
```

Đoạn mã này giống như các phiên bản tương tự của nó từ những chương trước (trừ tên đăng nhập, mật khẩu và các thiết lập cơ sở dữ liệu cụ thể). Bạn có thể tạo nó từ đầu hoặc sử dụng phiên bản hiện có.

2. Thiết lập thông tin cơ sở dữ liệu dưới dạng các hằng.

```
define ('DB_USER', 'username');

define ('DB_PASSWORD', 'password');

define ('DB_HOST', 'localhost');

define ('DB_NAME', 'content');
```

Để an toàn, tài khoản MySQL dùng trong ứng dụng này chỉ nên có các quyền (privilege) **INSERT**, **SELECT**, **DELETE** và chỉ dùng cho cơ sở dữ liệu *content*.

3. Kết nối với MySQL và chọn cơ sở dữ liệu.

```
$dbc = @mysql_connect (DB_HOST, DB_USER, DB_PASSWORD)
→ OR die ('Could not connect to MySQL: ' . mysql_error());

mysql_select_db (DB_NAME) OR die ('Could not select the
→ database: ' . mysql_error());

?>
```

Ứng dụng này sử dụng phần quản lý lỗi cơ bản với hàm **mysql_error()** (được mô tả trong Chương 6 "Sử dụng PHP và MySQL").

4. Hoàn tất trang PHP.

```
?>
```

5. Lưu tập tin với tên **mysql_connect.php** và tải lên máy chủ (đặt ngoài thư mục gốc của Web). Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 11.4.

Đoạn mã 11.4: *Đoạn mã kết nối MySQL và chọn cơ sở dữ liệu* content.

```
<?php # Doan ma 11.4 – mysql_connect.php
define ('DB_USER', 'username');
define ('DB_PASSWORD', 'password');
define ('DB_HOST', 'localhost');
define ('DB_NAME', 'content');
$dbc = @mysql_connect (DB_HOST, DB_USER, DB_PASSWORD) OR
→ die ('Could not connect to MySQL: ' . mysql_error() );
mysql_select_db (DB_NAME) OR die ('Could not select the
→ database: ' . mysql_error() );
?>
```

Ví dụ trong chương tới sẽ dùng các kỹ thuật quản lý lỗi cao cấp hơn và phiên bản chỉnh sửa của đoạn mã này.

Xem Phụ lục A "Cài đặt", để biết thêm thông tin về các quyền trong MySQL.

Quản lý các URL

Chúng ta sẽ xây dựng tiếp hai trang để cho phép người dùng nhập thêm URL và xem những URL đã được bổ sung trước đó. Các trang này sử dụng ba bảng trong cơ sở dữ liệu: *urls*, *url_types* và *url_titles*. Xem phần *"Lược đồ cơ sở dữ liệu"* để biết thêm chi tiết.

Bổ sung các URL

Đoạn mã **add_url.php** có lẽ là đoạn mã phức tạp nhất trong toàn bộ ứng dụng Web (vì cấu trúc cơ sở dữ liệu được chuẩn hóa). Biểu mẫu của đoạn mã sẽ nhận vào một URL, một tiêu đề hoặc tên URL, một mô tả và tối đa ba nhóm loại. Khi xử lý biểu mẫu, các giá trị URL, tiêu đề URL và mô tả sẽ được đưa vào bảng *url_titles*. Sau đó, khóa chính của bảng *(title_id)* được dùng cùng với các giá trị *type_id* để thêm từ một đến ba mẩu tin vào bảng *urls*, tùy theo số nhóm loại mà người dùng chọn.

Để xác định giá trị trường *title_id* của mẩu tin mới bổ sung, chúng ta dùng hàm **mysql_insert_id()**. Mỗi khi thực hiện câu truy vấn trên bảng chứa một trường

tăng tự động (thường là khóa chính), MySQL sẽ dùng giá trị lôgic kế tiếp cho trường đó. Hàm `mysql_insert_id()` sẽ trả lại giá trị này.

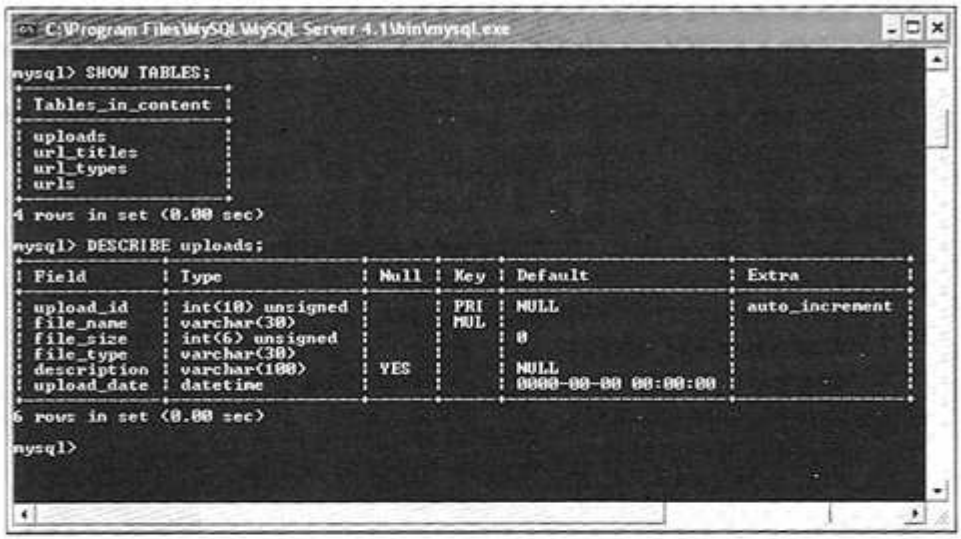
Lược đồ cơ sở dữ liệu

Cơ sở dữ liệu dùng trong chương này có tên content và được tạo trong Chương 5 "SQL và MySQL nâng cao". Khi đó, ba bảng chuẩn hóa được định nghĩa để xử lý các URL (cấu trúc phức tạp cho phép một URL được tập hợp dưới nhiều kiểu). Trong chương này, chúng ta bổ sung bảng uploads *để quản lý các tập tin. Cấu trúc hoàn chỉnh cho cơ sở dữ liệu có thể được tái tạo với các lệnh SQL sau:*

```
CREATE TABLE uploads (
upload_id int(10) UNSIGNED NOT NULL AUTO_INCREMENT,
file_name VARCHAR(30) NOT NULL,
file_size INT(6) UNSIGNED NOT NULL,
file_type VARCHAR(30) NOT NULL,
description VARCHAR(100) DEFAULT NULL,
upload_date DATETIME NOT NULL,
PRIMARY KEY (upload_id),
KEY file_name (file_name)
);
CREATE TABLE url_titles (
title_id SMALLINT(4) UNSIGNED NOT NULL AUTO_INCREMENT,
url VARCHAR(60) NOT NULL,
title VARCHAR(60) NOT NULL,
description TINYTEXT NOT NULL,
PRIMARY KEY (title_id),
UNIQUE KEY url (url)
);
CREATE TABLE url_types (
type_id TINYINT(3) UNSIGNED NOT NULL AUTO_INCREMENT,
type VARCHAR(20) NOT NULL,
PRIMARY KEY (type_id),
UNIQUE KEY type (type)
);
CREATE TABLE urls (
url_id SMALLINT(4) UNSIGNED NOT NULL AUTO_INCREMENT,
title_id SMALLINT(4) UNSIGNED NOT NULL,
type_id tinyint(3) UNSIGNED NOT NULL,
approved CHAR(1) DEFAULT 'N',
date_submitted TIMESTAMP(14) NOT NULL,
PRIMARY KEY (url_id),
KEY title_id (title_id),
KEY type_id (type_id),
KEY date_submitted (date_submitted)
);
```

Bạn có thể kiểm tra cấu trúc cơ sở dữ liệu hoặc bảng bằng các lệnh sau (xem hình 11-5):

```
SHOW TABLES;
DESCRIBE tablename;
```



Hình 11-5: *Kiểm tra cấu trúc cơ sở dữ liệu.*

Thực hành tạo tập tin **add_url.php** theo các bước sau:

1. Tạo hồ sơ PHP mới trong trình soạn thảo văn bản.

```
<?php # Doan ma 11.5 – add_url.php
$page_title = 'Them URL';
include ('includes/header.html');
require_once ('../mysql_connect.php');
```

2. Kiểm tra biểu mẫu có được gửi không và tạo hàm **escape_data()**.

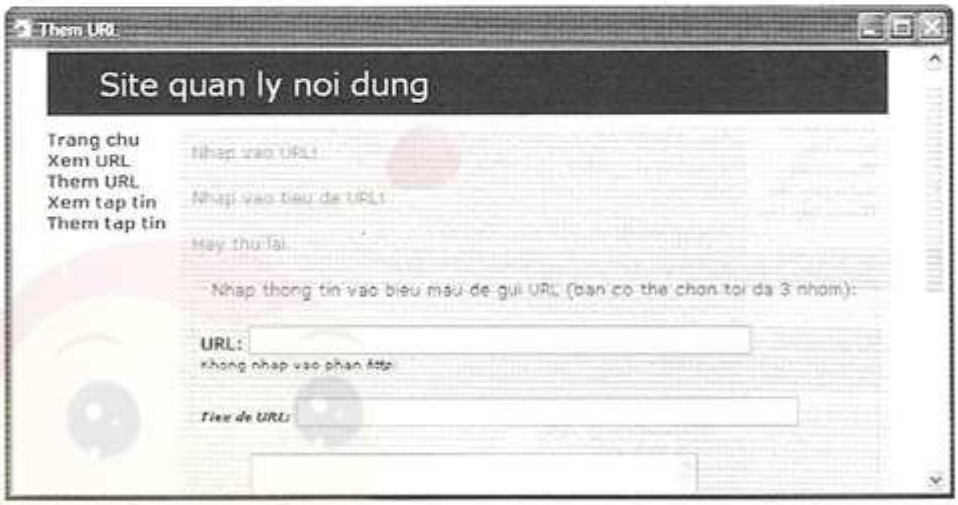
```
If (isset($_POST['submit'])) {
   function escape_data ($data) {
      global $dbc;
      if (ini_get('magic_quotes_gpc')) {
         $data = stripslashes($data);
      }
      return mysql_real_escape_string (trim ($data), $dbc);
   }
```

Như trong những chương trước, hàm **escape_data()** xử lý các giá trị đưa lên để đảm bảo an toàn cho cơ sở dữ liệu, cho dù đặc điểm Magic Quotes có được kích hoạt hay không.

3. Kiểm tra URL và tiêu đề.

```
if (!empty($_POST['url'])) {
   $u = escape_data($_POST['url']);
} else {
   $u = FALSE;
   echo '<p><font color="red">Nhap vao URL!</font></p>';
}
if (!empty($_POST['title'])) {
   $t = escape_data($_POST['title']);
} else {
   $t = FALSE;
   echo '<p><font color="red">Nhap vao tieu de URL!
   →</font></p>';
}
```

Trong mã lệnh này, chúng ta kiểm tra xem các giá trị có được nhập vào không. Bạn có thể dùng các biểu thức qui tắc để tăng độ chính xác (Chương 8 "An toàn", có đề cập đến mã lệnh để kiểm tra URL). Nếu có một trường không được nhập, thông báo lỗi sẽ được hiển thị (xem hình 11-6).



Hình 11-6: *Nếu người dùng không nhập URL hoặc tiêu đề, thông báo lỗi sẽ được hiển thị.*

4. Kiểm tra giá trị của phần mô tả.

```
if (!empty($_POST['description'])) {
   $d = escape_data($_POST['description']);
} else {
```

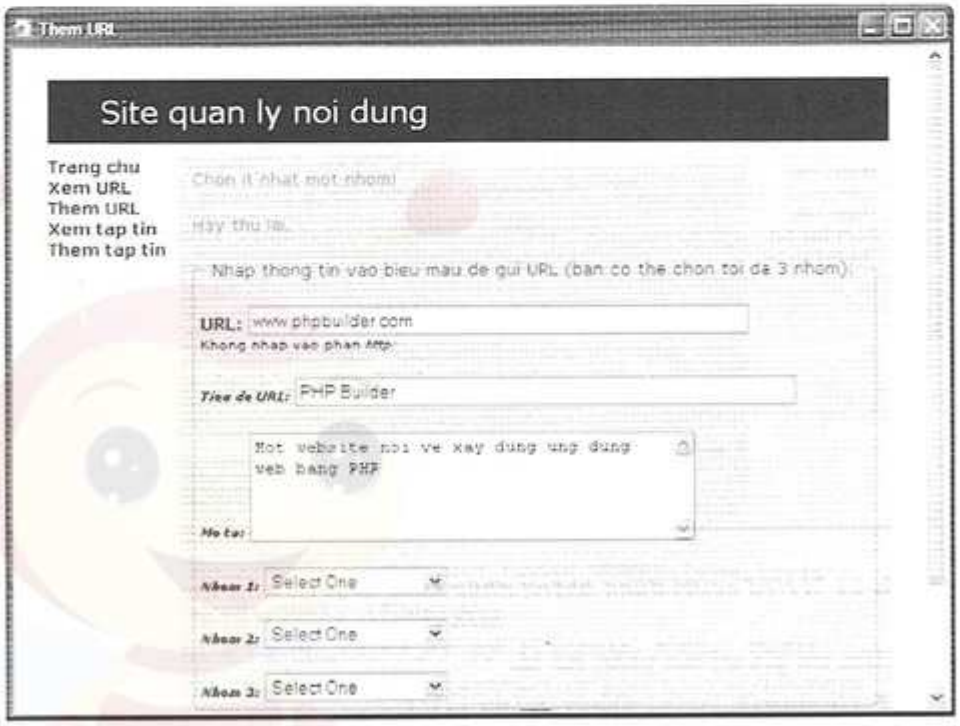
```
    $d = '';
}
```

Vì trường *description* trong bảng *url_titles* là không bắt buộc, chúng ta sẽ xử lý dữ liệu này nếu nó được nhập. Chúng ta sẽ thiết lập biến `$d` (dùng để chứa giá trị của phần mô tả) là một chuỗi rỗng nếu phần mô tả không được nhập.

5. Kiểm tra xem nhóm có được chọn không.

```
if (($_POST['type1'] > 0) OR ($_POST['type2'] > 0) OR
→ ($_POST['type3']) > 0) {
    $type = TRUE;
} else {
    $type = FALSE;
    echo '<p><font color="red">Chon it nhat mot nhom!
    →</font></p>';
}
```

Người dùng có quyền đưa một URL vào ba nhóm có sẵn (được thiết lập trong bảng *url_types*). Phần điều kiện này sẽ kiểm tra để đảm bảo ít nhất một trong ba menu thả xuống được dùng và in thông báo lỗi nếu ngược lại (xem hình 11-7).



Hình 11-7: *Người dùng phải chọn ít nhất một nhóm để phân loại URL.*

6. Đưa URL vào bảng *url_titles*.

```
if ($u && $t && $type) {

   $query = "INSERT INTO url_titles (url, title,
   →description) VALUES ('$u', '$t', '$d')";

   $result = @mysql_query ($query);

   $tid = @mysql_insert_id();
```

Do cấu trúc cơ sở dữ liệu, URL phải được đưa vào bảng *url_titles* trước khi bổ sung dữ liệu vào bảng *urls*. Câu truy vấn này bổ sung URL rồi trả về giá trị của trường *title_id* (khóa chính tăng tự động của bảng), bằng cách dùng hàm **mysql_insert_id()**. Giá trị này sẽ được dùng trong câu truy vấn thứ hai (bước 7).

7. Tạo dựng câu truy vấn chính.

```
if ($tid > 0) {

   $query = 'INSERT INTO urls (title_id, type_id, approved,
   → date_submitted) VALUES ';

   if ($_POST['type1'] > 0) {

      $query .= "($tid, {$_POST['type1']},'Y',NOW()),";

   }

   if ($_POST['type2'] > 0) {

      $query .= "($tid, {$_POST['type2']}'Y',NOW()),";

   }

   if ($_POST['type3'] > 0) {

      $query .= "($tid, {$_POST['type3']}'Y',NOW()),";

   }

   $query = substr ($query, 0, -2);
```

Nếu giá trị **$tid** hợp lệ (lớn hơn 0), chúng ta sẽ tiếp tục với mệnh đề chính (đưa mẩu tin vào bảng *urls*). Để thực hiện, chúng ta bắt đầu định nghĩa và gán phần đầu câu truy vấn cho biến **$query**. Sau đó, ta kiểm tra xem nhóm nào được chọn (từ menu thả xuống của biểu mẫu) và thêm một mẩu tin vào câu truy vấn cho nhóm đó. Cuối cùng, chúng ta dùng hàm **substr()** để cắt bớt hai ký tự cuối (dấu phẩy và khoảng trống) ra khỏi câu truy vấn. Nếu gặp trục trặc với cú pháp của câu truy vấn này, bạn nên in giá trị biến **$query** để xem mã lệnh SQL.

8. Chạy câu truy vấn, thông báo kết quả và hoàn tất các phần điều kiện.

```
   $result = @mysql_query ($query);
```

```
        if ($result) {
            echo '<p><b>Cam on ban da gui URL!</b></p>';
            $_POST = array();
        } else {
            echo '<p><font color="red">Thong tin cua ban khong
            → the xu ly vi co loi he thong. Chung toi xin loi
            → vi su co nay.</font></p>';
        }
    } else {
        echo '<p><font color="red">Thong tin cua ban khong the
        → xu ly vi co loi he thong. Chung toi xin loi vi su
        → co nay.</font></p>';
    }
} else {
    echo '<p><font color="red"> Hay thu lai.</font></p>';
    }
}
```

9. Tạo mã lệnh HTML để xây dựng menu thả xuống cho trường loại URL.

```
$query = "SELECT * FROM url_types ORDER BY type ASC";
$result = @mysql_query ($query);
$pulldown = '<option>Select One</option>';
while ($row = mysql_fetch_array ($result, MYSQL_ASSOC)) {
    $pulldown .= "<option value=\"{$row['type_id']}\">
    →{$row['type']}</option>\n";
}
?>
```

Biểu mẫu chứa ba menu thả xuống, mỗi menu hiển thị danh sách các loại URL được lấy từ cơ sở dữ liệu. Vì chúng ta hiển thị thông tin này ba lần, nên việc gán kết quả cho một biến sẽ hiệu quả hơn so với việc truy vấn cơ sở dữ liệu nhiều lần. Xem mã nguồn của menu thả xuống trong hình 11-8.

```
<p><b>Nhom 1:</b> <select name="type1">
<option>Select One</option><option value="3">Code Libraries</option>
<option value="6">General Database</option>
<option value="5">General MySQL</option>
<option value="1">General PHP</option>
<option value="4">Programming</option>
<option value="2">Web Development</option>
</select></p>
<p><b>Nhom 2:</b> <select name="type2">
<option>Select One</option><option value="3">Code Libraries</option>
<option value="6">General Database</option>
<option value="5">General MySQL</option>
<option value="1">General PHP</option>
<option value="4">Programming</option>
<option value="2">Web Development</option>
</select></p>
<p><b>Nhom 3:</b> <select name="type3">
<option>Select One</option><option value="3">Code Libraries</option>
<option value="6">General Database</option>
<option value="5">General MySQL</option>
<option value="1">General PHP</option>
<option value="4">Programming</option>
<option value="2">Web Development</option>
</select></p>
```

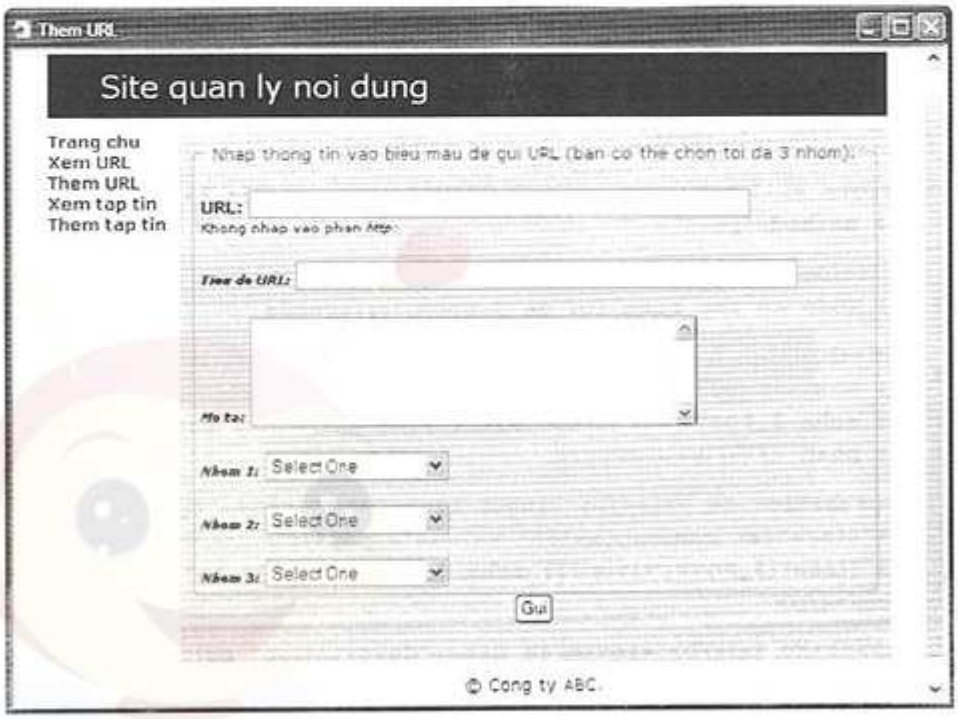
Hình 11-8: *Mã nguồn HTML để tạo động menu thả xuống.*

10. Tạo biểu mẫu HTML.

```
<form action="<?php echo $_SERVER['PHP_SELF']; ?>"
→ method="post">

<fieldset><legend>Nhap thong tin vao bieu mau de gui URL
→ (ban co the chon toi da 3 nhom):</legend>

<p><b>URL:</b> <input type="text" name="url" size="60"
→ maxlength="60" value="<?php if (isset($_POST['url']))
→ echo $_POST['url']; ?>" /><br /><small>Khong nhap vao
→ phan <i>http:

<p><b>Tieu de URL:</b> <input type="text" name="title"
→ size="60" maxlength="60" value="<?php if
→ (isset($_POST['title'])) echo $_POST['title']; ?>"
→ /></p>

<p><b>Mo ta:</b> <textarea name="description" cols="40"
→ rows="5"><?php if (isset($_POST['description']))
→ echo $_POST['description']; ?></textarea></p>

<p><b>Nhom 1:</b> <select name="type1">

<?php echo $pulldown; ?>
```

```
</select></p>
<p><b>Nhom 2:</b> <select name="type2">
<?php echo $pulldown; ?>
</select></p>
<p><b>Nhom 3:</b> <select name="type3">
<?php echo $pulldown; ?>
</select></p>
</fieldset>
<div align="center"><input type="submit" name="submit"
→ value="Gui" /></div>
</form>
```

Biểu mẫu (xem hình 11-9) có một số trường nhập và ghi nhớ các giá trị được nhập trong các hộp văn bản.

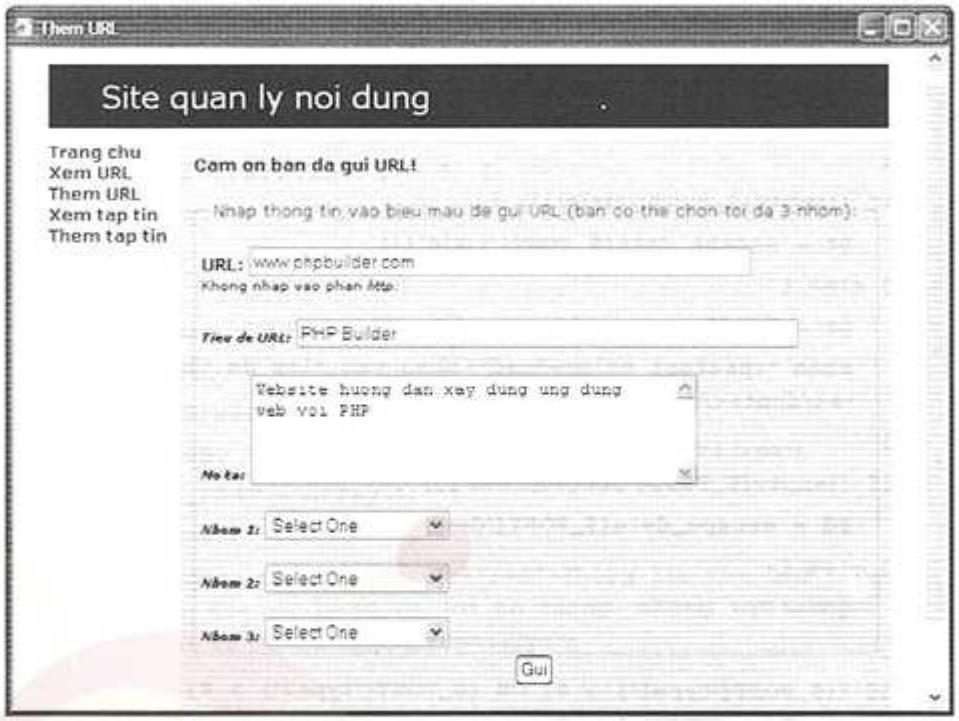


Hình 11-9: *Biểu mẫu để nhập URL.*

11. Hoàn tất trang PHP.

```
<?php

mysql_close();

include ('includes/footer.html');

?>
```

12. Lưu tập tin với tên add_url.php, tải lên máy chủ Web và chạy thử trong trình duyệt (xem hình 11-10). Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 11.5.



Hình 11-10: *Nếu quá trình đưa lên thành công, một thông báo hiển thị và biểu mẫu được thể hiện lại với các giá trị định trước.*

Đoạn mã 11.5: *Đoạn mã này cho phép người dùng nhập các URL vào cơ sở dữ liệu.*

```
<?php # Doan ma 11.5 - add_url.php
$page_title = 'Them URL';
include ('includes/header.html');
require_once ('../mysql_connect.php');
if (isset($_POST['submit'])) {
   function escape_data ($data) {
```

```
    global $dbc;
    if (ini_get('magic_quotes_gpc')) {
       $data = stripslashes($data);
    }
    return mysql_real_escape_string (trim ($data), $dbc);
}
if (!empty($_POST['url'])) {
    $u = escape_data($_POST['url']);
} else {
    $u = FALSE;
    echo '<p><font color="red">Nhap vao URL!</font></p>';
}
if (!empty($_POST['title'])) {
    $t = escape_data($_POST['title']);
} else {
    $t = FALSE;
    echo '<p><font color="red">Nhap vao tieu de URL!
    →</font></p>';
}
if (!empty($_POST['description'])) {
    $d = escape_data($_POST['description']);
} else {
    $d = '';
}
if (($_POST['type1'] > 0) OR ($_POST['type2'] > 0) OR
→ ($_POST['type3']) > 0) {
    $type = TRUE;
} else {
    $type = FALSE;
    echo '<p><font color="red">Chon it nhat mot nhom!
    →</font></p>';
}
if ($u && $t && $type) {
    $query = "INSERT INTO url_titles (url, title,
    → description) VALUES ('$u', '$t', '$d')";
    $result = @mysql_query ($query);
```

```
      $tid = @mysql_insert_id();
      if ($tid > 0) {
         $query = 'INSERT INTO urls (title_id, type_id,
         → approved, date_submitted) VALUES ';
         if ($_POST['type1'] > 0) {
            $query .= "($tid,{$_POST['type1']},'Y',NOW()), ";

         }
         if ($_POST['type2'] > 0) {
            $query .= "($tid,{$_POST['type2']},'Y',NOW()), ";

         }
         if ($_POST['type3'] > 0) {
            $query .= "($tid,{$_POST['type3']},'Y',NOW()), ";

         }
         $query = substr ($query, 0, -2);
         $result = @mysql_query ($query);
         if ($result) {
            echo '<p><b>Cam on ban da gui URL!</b></p>';
            $_POST = array();
         } else {
            echo '<p><font color="red">Thong tin cua ban khong
            → the xu ly vi co loi he thong. Chung toi xin loi
            → vi su co nay.</font></p>';
         }
      } else {
         echo '<p><font color="red">Thong tin cua ban khong the
         → xu ly vi co loi he thong. Chung toi xin loi vi su
         → co nay.</font></p>';
      }
   } else {
      echo '<p><font color="red">Hay thu lai.</font></p>';
   }
}
$query = "SELECT * FROM url_types ORDER BY type ASC";
$result = @mysql_query ($query);
```

```
$pulldown = '<option>Select One</option>';
while ($row = mysql_fetch_array ($result, MYSQL_ASSOC)) {
   $pulldown .= "<option value=\"{$row['type_id']}\">
   →{$row['type']}</option>\n";
}
?>
<form action="<?php echo $_SERVER['PHP_SELF']; ?>"
→ method="post">
<fieldset><legend>Nhap thong tin vao bieu mau de gui URL
→ (ban co the chon toi da 3 nhom):</legend>
<p><b>URL:</b> <input type="text" name="url" size="60"
→ maxlength="60" value="<?php if (isset($_POST['url']))
→ echo $_POST['url']; ?>" /><br /><small>Khong nhap vao phan
→ <i>http:
→<p><b>Tieu de URL:</b> <input type="text" name="title"
→ size="60" maxlength="60" value="<?php if
→ (isset($_POST['title'])) echo $_POST['title']; ?>" /></p>
<p><b>Mo ta:</b> <textarea name="description" cols="40"
→ rows="5"><?php if (isset($_POST['description']))
→ echo $_POST['description']; ?></textarea></p>
<p><b>Nhom 1:</b> <select name="type1">
<?php echo $pulldown; ?>
</select></p>
<p><b>Nhom 2:</b> <select name="type2">
<?php echo $pulldown; ?>
</select></p>
<p><b>Nhom 3:</b> <select name="type3">
<?php echo $pulldown; ?>
</select></p>
</fieldset>
<div align="center"><input type="submit" name="submit"
→ value="Gui" /></div>
</form>
<?php
mysql_close();
include ('includes/footer.html');
?>
```

Nếu không muốn dùng HTML trong các giá trị đưa lên, bạn cần dùng hàm `strip_tags` *để loại bỏ tất cả các thẻ HTML và PHP.*

Nếu muốn hiển thị các thẻ HTML (như khi chúng hiện ra trong mã nguồn), bạn nên dùng các hàm `htmlspecialchars()` *và* `htmlentitis()`.

Hàm `mysql_insert_id()` *là riêng cho mỗi session (phiên truy cập). Do đó, bạn không phải lo lắng việc giá trị trả về bị sai, ngay cả khi nhiều người cùng chạy đoạn mã này.*

Với PHP và HTML, bạn có thể tạo ra biểu mẫu ghi nhớ các tùy chọn đã chọn trong menu thả xuống, nhưng điều này đòi hỏi lập trình nhiều hơn và sẽ không hoạt động dễ dàng với biến `$pulldown` *ở đây.*

Một tùy chọn khác trong ví dụ này là tạo các hộp kiểm để chọn nhóm.

Nếu Magic Quotes được kích hoạt, giá trị trong biểu mẫu cần được chạy qua `stripslashes()` *trước khi chuyển tới trình duyệt.*

Trong ví dụ này, nếu quá trình hoạt động, biểu mẫu sẽ tái hiện với các giá trị đã dùng được thể hiện trong nó. Nếu muốn thay đổi điều này, bạn không nên hiển thị biểu mẫu hoặc thiết lập lại mảng `$_POST` *sau khi truy vấn cơ sở dữ liệu thành công.*

Hàm `mysql_insert_id()` *trong PHP tương đương hàm* `LAST_INSERT_ID` *của MySQL.*

Xem các URL đưa lên

Đoạn mã để xem các URL có hai phần: phần trên hiển thị menu thả xuống của các kiểu có sẵn và phần dưới hiển thị toàn bộ các liên kết cho một kiểu cụ thể. Lần đầu người dùng truy cập trang, không có URL nào hiển thị. Khi người dùng chọn kiểu và gửi biểu mẫu, trang tái hiện, liệt kê các URL cho kiểu đó.

Các chức năng và kỹ thuật mới

Đoạn mã `view_urls.php` *giới thiệu hai ý tưởng mới. Thứ nhất là ép kiểu (type casting): buộc một biến là một kiểu nhất định (số nguyên, chuỗi...). Để ép kiểu một biến, bạn đặt trước nó với tên của kiểu mong muốn trong dấu ngoặc. Ví dụ:*

```
$var = "10"; // string
$var = (int) $var; // integer
```

Những kiểu có sẵn là (trong vài trường hợp, nhiều thuật ngữ dùng cho kiểu giống nhau): int và integer; bool và boolean; float, double và real; string, array và object. PHP sẽ đổi kiểu biến và gán một giá trị lôgic cho kiểu đó, dựa trên giá trị ban đầu của nó (xem tài liệu PHP để biết chi tiết). Bạn cũng có thể đổi một biến thành số nguyên bằng hàm `intval()` *(như chúng ta làm trong đoạn mã này) hoặc chuyển một biến thành chuỗi bằng cách đặt giá trị của nó trong dấu nháy.*

Ý tưởng thứ hai là hàm **list()** *thể hiện trong đoạn mã dưới đây. Hàm này nhận các giá trị của một mảng và gán chúng cho các biến riêng lẻ. Ví dụ:*

```
$var = array ("Larry", "Ullman");
list ($first, $last) = $var;
```

Bây giờ biến **$first** *và* **$last** *sẽ có giá trị tương ứng của Larry và Ullman. Tính năng này cũng có thể được thực hiện bằng hàm* **extract()**.

Thực hành tạo tập tin **view_urls.php** theo các bước sau:

1. Tạo hồ sơ PHP mới trong trình soạn thảo văn bản.

```
<?php # Doan ma 11.6 - view_urls.php
$page_title = 'Xem URL';
include_once ('includes/header.html');
require_once ('../mysql_connect.php');
```

2. Bắt đầu biểu mẫu HTML.

```
echo '<div align="center">
<form method="get" action="view_urls.php">
<select name="type">
<option value="NULL">Chon mot nhom:</option>';
```

Khi trang được truy xuất lần đầu, chúng ta muốn hiển thị biểu mẫu (xem hình 11-11) chỉ gồm một menu thả xuống và một nút gửi biểu mẫu. Mã lệnh trên bắt đầu mã HTML cho biểu mẫu.

Hình 11-11: *Menu thả xuống.*

3. Lấy ra toàn bộ các kiểu URL có sẵn và đưa vào menu thả xuống.

```
$query = 'SELECT * FROM url_types ORDER BY type ASC';
$result = mysql_query ($query);
```

```
while ($row = mysql_fetch_array ($result, MYSQL_NUM)) {

   echo '<option value="', $row[0], '">',
   → stripslashes($row[1]), '</option>';

}
```

Mã lệnh này lấy ra tất cả kiểu URL từ bảng *url_types* và dùng các mẩu tin trả về để tạo mã HTML cho menu thả xuống (xem hình 11-12).

```
<form method="get" action="view_urls.php">
<select name="type">
<option value="NULL">Chon mot nhom:</option>
<option value="3">Code Libraries</option>
<option value="6">General Database</option>
<option value="5">General MySQL</option>
<option value="1">General PHP</option>
<option value="4">Programming</option>
<option value="2">Web Development</option>
</select>
<input type="submit" name="submit" value="Thuc hien!">
</form>
```

Hình 11-12: *Tạo động mã nguồn HTML của biểu mẫu.*

4. Hoàn tất biểu mẫu HTML.

```
echo '</select>

<input type="submit" name="submit" value="Thuc hien!">

</form>

</div>';
```

5. Kiểm tra xem có kiểu nào được chọn không và lấy ra thông tin cho kiểu đó nếu có.

```
if (isset($_GET['type'])) {

   $t = intval($_GET['type']);

   $query = "SELECT type FROM url_types WHERE type_id=$t";

   $result = mysql_query ($query);

   list ($type) = mysql_fetch_array ($result, MYSQL_NUM);

   echo "<hr /><div align=\"center\"><b>$type Links</b><br/>
   → <small>(Tat ca cac lien ket se duoc mo trong cua so
   → rieng. Cac lien ket moi duoc the hien truoc.)</small>
   → </div>\n";
```

Nếu kiểu đã được chọn (trong trường hợp đó, nó sẽ được gắn vào URL và có mặt trong mảng **$_GET**), các URL cho kiểu đó sẽ được lấy ra.

Bước đầu tiên là dùng hàm **intval()** để đảm bảo giá trị của trường *type* là một số nguyên trước khi dùng nó trong câu truy vấn. Sau đó, chúng ta lấy ra tên kiểu bằng hàm **list()** và dùng nó làm tiêu đề trang. Xem phần phụ chú *"Các chức năng và kỹ thuật mới"* hoặc trong tài liệu PHP để biết thêm về hàm **intval()** và **list()**.

6. Bắt đầu biến **$first** và truy vấn cơ sở dữ liệu.

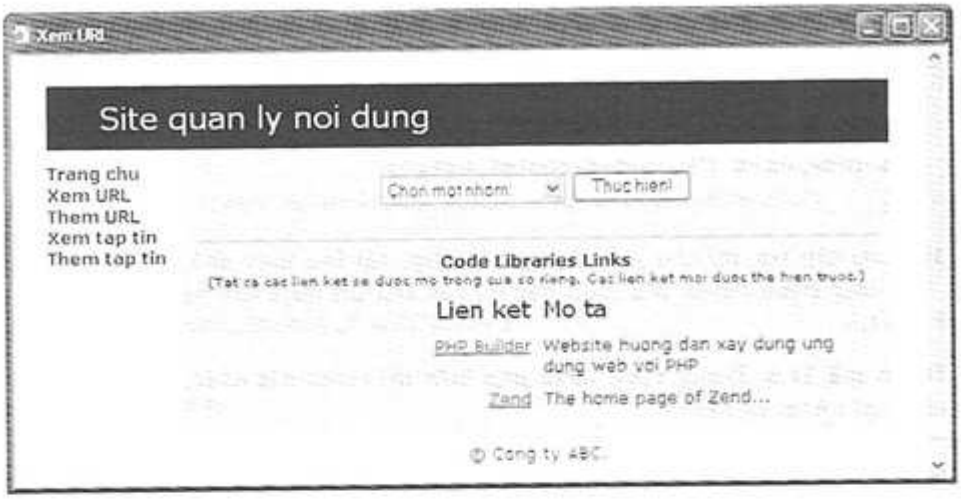
```
$first = TRUE;

$query = "SELECT url, title, description FROM urls AS u,
→ url_titles AS ut WHERE ut.title_id = u.title_id AND
→ u.type_id=$t AND u.approved = 'Y' ORDER BY
→ date_submitted DESC";

$result = mysql_query ($query);
```

Biến **$first** được dùng với hai mục đích. Thứ nhất, nó khởi tạo bảng HTML trước khi hiển thị mẩu tin đầu tiên. Thứ hai, nó được dùng để kiểm tra xem có URL nào được câu truy vấn trả về không.

7. In ra toàn bộ các mẩu tin trả về.

```
while ($row = mysql_fetch_array ($result, MYSQL_ASSOC)) {

   if ($first) {

      echo '<table border="0" width="100%" cellspacing="3"
      → cellpadding="3" align="center">
      → <tr><td align="right" width="50%"><font
      → size="+1">Lien ket</font></td><td align="left"
      → width="50%"><font size="+1">Mo ta</font></td>
      → </tr>';

   }

   echo "<tr><td align=\"right\"><a href=\"http://
   → {$row['url']}\" target=\"_new\">" .
   → stripslashes($row['title']) . "</a></td>
   → <td align=\"left\">{$row['description']}</td></tr>\n";

   $first = FALSE;

}
```

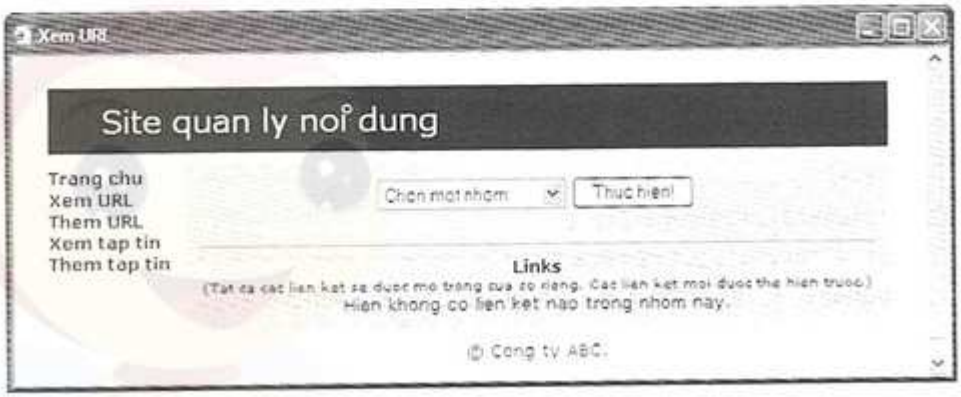
Vòng lặp **while** trả lại từng mẩu tin do câu truy vấn lấy ra. Bảng và phần đầu bảng được gửi đến trình duyệt trước khi mẩu tin đầu tiên được hiển thị (xem hình 11-13).



Hình 11-13: *Danh mục các URL cho một kiểu cụ thể.*

8. In thông báo nếu không có URL nào trả về và hoàn tất phần điều kiện chính.

```
    if ($first) {
        echo '<div align="center">Hien khong co lien ket nao
        → trong nhom nay.</div>';
    } else {
        echo '</table>';
    }
}
```



Hình 11-14: *Trang kết quả nếu kiểu không có các URL được liệt kê.*

Biến `$first` được thiết lập `FALSE` trong vòng lặp `while` nếu có bất kỳ mẩu tin nào được trả về. Do đó, nếu `$first` vẫn `TRUE` tại điểm này, nghĩa là không có

mẩu tin nào được lấy ra và thông báo sẽ được hiển thị (xem hình 11-14). Nếu ngược lại, bảng sẽ được hoàn tất.

9. Hoàn tất trang HTML.

```
mysql_close();
include_once ('includes/footer.html');
?>
```

10. Lưu tập tin với tên gọi **view_urls.php**, tải lên máy chủ Web và chạy thử trong trình duyệt. Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 11.6.

Đoạn mã 11.6: *Trang **view_urls.php** hiển thị menu các nhóm URL và các URL cho một nhóm cụ thể.*

```
<?php # Doan ma 11.6 - view_urls.php
$page_title = 'Xem URL';
include_once ('includes/header.html');
require_once ('../mysql_connect.php');
echo '<div align="center">
<form method="get" action="view_urls.php">
<select name="type">
<option value="NULL">Chon mot nhom:</option>';
$query = 'SELECT * FROM url_types ORDER BY type ASC';
$result = mysql_query ($query);
while ($row = mysql_fetch_array ($result, MYSQL_NUM)) {
   echo '<option value="', $row[0], '">',
   → stripslashes($row[1]), '</option>';
}
echo '</select>
<input type="submit" name="submit" value="Thuc hien!">
</form>
</div>';
if (isset($_GET['type'])) {
   $t = intval($_GET['type']);
   $query = "SELECT type FROM url_types WHERE type_id=$t";
   $result = mysql_query ($query);
   list ($type) = mysql_fetch_array ($result, MYSQL_NUM);
   echo "<hr /><div align=\"center\"><b>$type Links</b><br/>
   → <small>(Tat ca cac lien ket se duoc mo trong cua so
   → rieng. Cac lien ket moi duoc the hien truoc.)</small>
   → </div>\n";
```

```
   $first = TRUE;
   $query = "SELECT url, title, description FROM urls AS u,
   -> url_titles AS ut WHERE ut.title_id = u.title_id AND
   -> u.type_id=$t AND u.approved = 'Y' ORDER BY
   -> date_submitted desc";
   $result = mysql_query ($query);
   while ($row = mysql_fetch_array ($result, MYSQL_ASSOC)) {
      if ($first) {
         echo '<table border="0" width="100%" cellspacing="3"
         -> cellpadding="3" align="center"><tr><td
         -> align="right" width="50%"><font size="+1">Lien ket
         -> </font></td><td align="left" width="50%"><font
         -> size="+1">Mo ta</font></td></tr>';
      }
      echo "<tr><td align=\"right\"><a href=\"http://
      -> {$row['url']}\" target=\"_new\">" .
      -> stripslashes($row['title']) . "</a></td>
      -> <td align=\"left\">{$row['description']}</td>
      -> </tr>\n";
      $first = FALSE;
   }
   if ($first) {
      echo '<div align="center">Hien khong co lien ket nao
      -> trong nhom nay.</div>';
   } else {
      echo '</table>';
   }
}
mysql_close();
include_once ('includes/footer.html');
?>
```

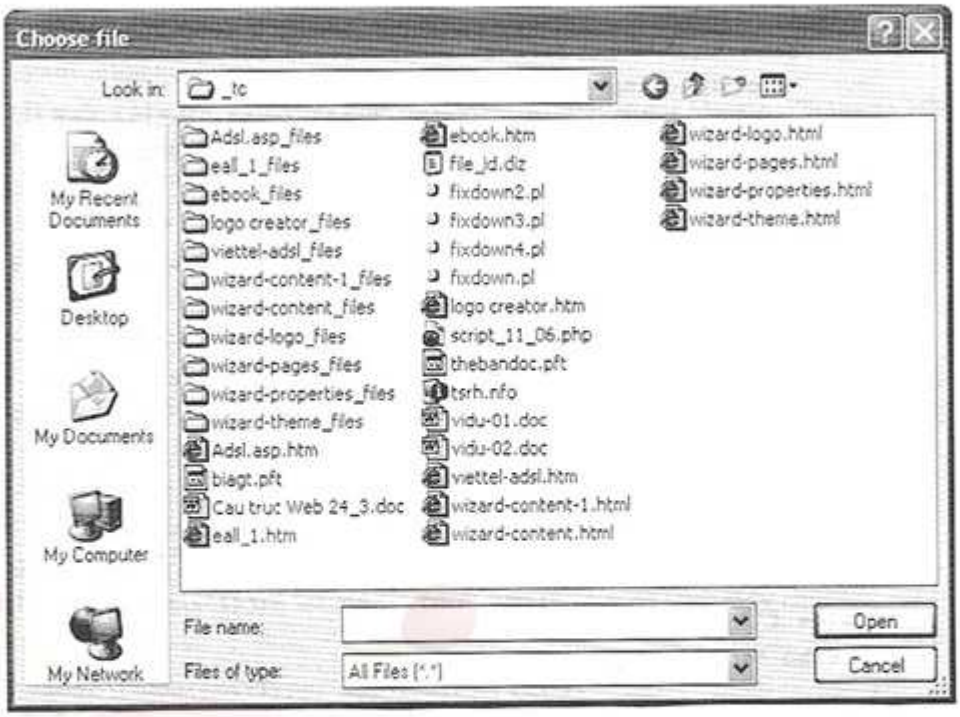
Chương 12 "Ví dụ - Đăng ký người dùng", có đoạn mã thể hiện cách hiển thị các mẩu tin trên một loạt trang (tham khảo đoạn mã view_users.php).

Ví dụ trong chương tiếp theo cũng đề cập việc đăng ký và kiểm tra xác thực người dùng. Bạn có thể sử dụng những kỹ thuật đó với ứng dụng này nếu muốn bảo vệ thông tin.

*Bạn có thể dùng hàm nl2br() (viết tắt của newline to break) cho trường description. Hàm này sẽ chuyển từng ký tự xuống hàng (được tạo bằng cách nhấn phím Return hoặc Enter) sang thẻ (X)HTML
.*

Quản trị các tập tin

Phần cuối của ứng dụng Web này sẽ đề cập đến quản trị các tập tin. Các đoạn mã trong phần này cho phép người dùng chọn tập tin muốn chép thông qua hộp nhắc trong trình duyệt Web (xem hình 11-15). Sau đó, tập tin này được tải lên và được lưu trên máy chủ. Một mẩu tin sẽ được đưa vào cơ sở dữ liệu để phản ánh kết quả tải lên.



Hình 11-15: *Người dùng chọn tập tin muốn tải lên máy chủ.*

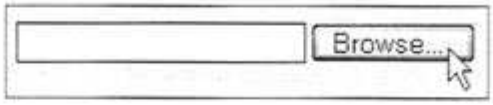
Tải lên các tập tin

Tương tự việc xử lý biểu mẫu HTML bất kỳ, quá trình tải lên tập tin có hai giai đoạn. Đầu tiên, biểu mẫu HTML phải được hiển thị với mã thích hợp để cho phép tải tập tin lên. Sau đó, dựa trên kết quả được gửi lên, đoạn mã PHP phải sao chép tập tin vào vị trí thích hợp.

Cú pháp yêu cầu cho biểu mẫu để xử lý tải lên tập tin có ba phần:

```
<form enctype="mutipart/form-data" action="script.php"
→ method="post">

<input type="hiden" name="MAX_FILE_SIZE" value="30000">

Tap tin <input type="file" name="upload" />
```

Phần **enctype** của thẻ <form> cho biết biểu mẫu có khả năng xử lý nhiều kiểu dữ liệu, bao gồm cả tập tin. Chú ý, biểu mẫu phải dùng phương pháp POST. Trường ẩn **MAX_FILE_SIZE** sẽ giới hạn kích thước tập tin được chọn (tính bằng byte) và phải đi trước trường nhập tập tin. Cuối cùng, trường nhập kiểu **file** tạo nút nhấn thích hợp trong biểu mẫu (xem hình 11-16).



Hình 11-16: *Phần nhập tạo một nút duyệt tìm tập tin trong biểu mẫu HTML.*

Với PHP 4.1, tập tin tải lên được truy xuất thông qua biến siêu toàn cục **$_FILES**. Những phiên bản PHP trước dùng mảng **$HTTP_POST_FILES** hoặc biến **$upload** (tương ứng tên trường nhập **file**), nếu **register_global** được kích hoạt.

Biến tập tin là một mảng giá trị (xem bảng 11.1).

Bảng 11.1: *Dữ liệu cho tập tin tải lên được truy xuất thông qua các thành phần mảng này.*

Chỉ số	Ý nghĩa
name	Tên gốc của tập tin trên máy tính người dùng.
type	Kiểu MIME của tập tin, được cung cấp bởi trình duyệt.
size	Kích thước tập tin (tính theo byte).
tmp_name	Tên tạm của tập tin khi nó được tải lên máy chủ.

Khi biểu mẫu được gửi lên cho mã kịch bản PHP, hàm **move_uploaded_ file()** sẽ di chuyển tập tin từ thư mục tạm thời vào vị trí thường trú.

```
move_upload_file ("tap_tin_tam", "tap_tin_dich");
```

Nếu hàm thành công, phiên bản tạm thời của tập tin sẽ được loại bỏ khỏi máy chủ. Tuy nhiên, để hàm này hoạt động, máy chủ Web phải có quyền ghi trên thư mục mà tập tin sẽ được lưu.

Chú ý, chúng ta cần phải điều chỉnh các thiết lập PHP trong tập tin **php.ini** để cho phép tải lên tập tin.

Thực hành chuẩn bị máy chủ theo các bước sau:

1. Mở tập tin **php.ini** trong trình soạn thảo văn bản.
2. Dưới vùng File Uploads (khoảng dòng 494), điều chỉnh các dòng dưới đây nếu cần (xem hình 11-17):

```
file_uploads = On
upload_tmp_dir = /tmp
upload_max_filesize = 2M
```

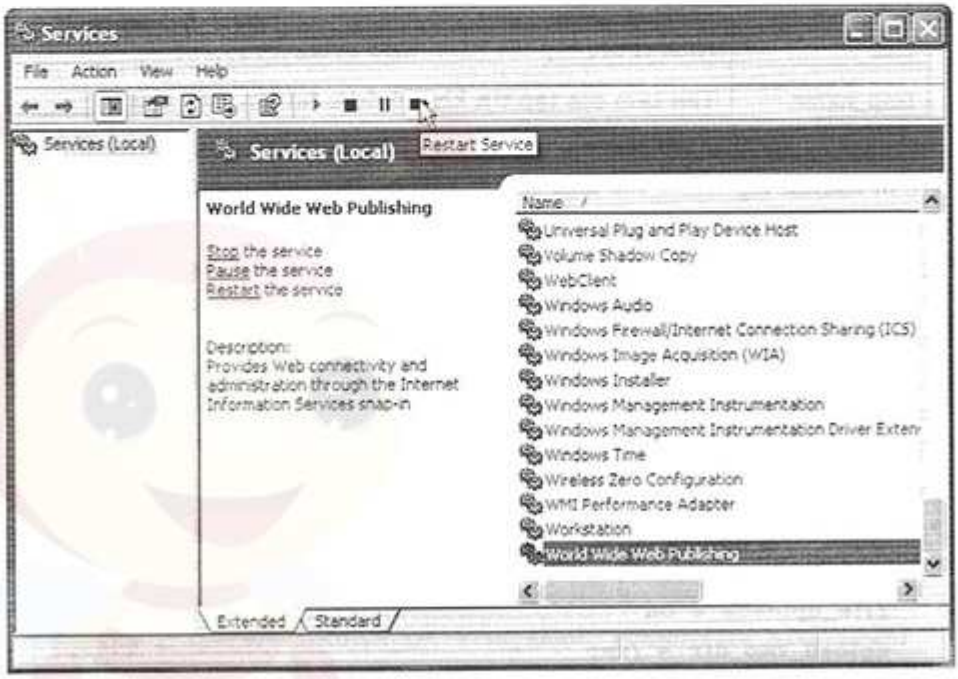
```
489 ;;;;;;;;;;;;;;;;;;
490 ; File Uploads ;
491 ;;;;;;;;;;;;;;;;;;
492
493 ; Whether to allow HTTP file uploads.
494 file_uploads = On
495
496 ; Temporary directory for HTTP uploaded files (will use system default if not
497 ; specified).
498 upload_tmp_dir = C:\Program Files\Php\uploadtemp  ; temporary directory for HTTP
499
500 ; Maximum allowed size for uploaded files.
501 upload_max_filesize = 2M
502
```

Hình 11-17: *Vùng File Uploads của tập tin* **`php.ini`** *kiểm soát cách PHP xử lý các tập tin tải lên.*

Dòng thứ nhất cho biết việc tải lên có được cho phép hay không. Dòng thứ hai xác định nơi các tập tin tải lên được lưu tạm thời. Trên đa số các hệ điều hành, thiết lập này được chú thích bằng một dấu chấm phẩy đứng trước (có hay không cũng được). Người dùng Mac OS và Unix thường thiết lập giá trị này là **`/tmp`**, trong khi người dùng Windows tạo và sử dụng thư mục **`C:\tmp`**.

Dòng cuối cùng thiết lập kích cỡ tối đa tập tin tải lên, tính bằng megabyte.

3. Lưu tập tin và khởi động lại máy chủ Web (xem hình 11-18).



Hình 11-18: *Phải khởi động lại máy chủ Web để các thay đổi PHP có tác dụng.*

```
apachectl graceful
```

Dòng lệnh này khởi động lại máy chủ Apache trên các hệ điều hành Unix. Người dùng Mac OS X khởi động lại máy chủ thông qua bảng điều khiển Sharing và người dùng Windows dùng bảng kiểm soát Services hoặc các công cụ khác được cài đặt trên máy chủ Web.

4. Tạo một thư mục với tên **uploads** và đặt ngoài thư mục Web. Thư mục *uploads* là nơi chứa các tập tin được tải lên. Chúng ta đặt nó ngoài thư mục Web vì lý do an toàn.
5. Thiết lập các quyền trên thư mục *uploads* để máy chủ có thể ghi chép vào đó.

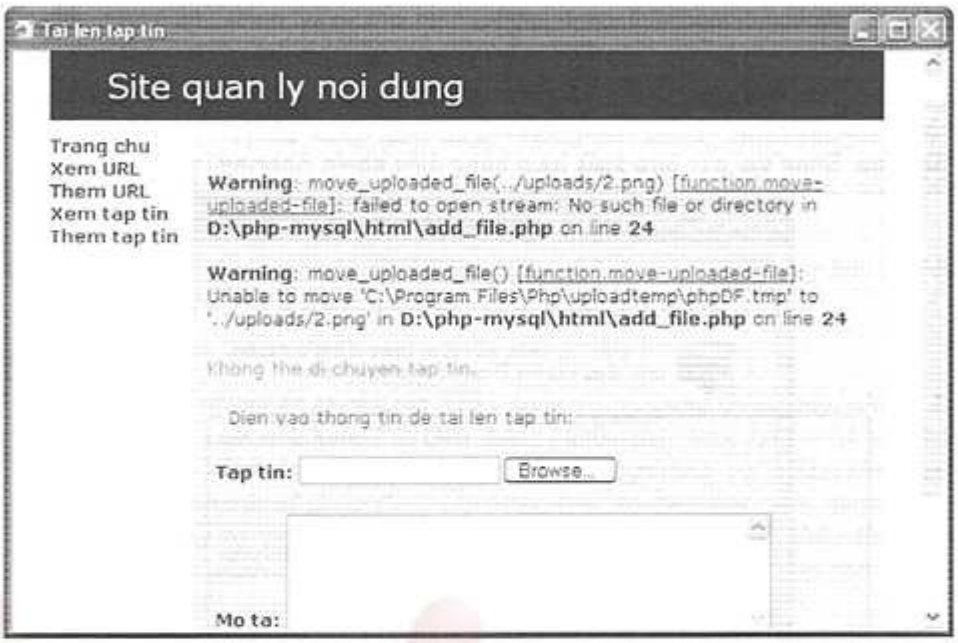
Một tùy chọn đơn giản nhưng kém an toàn là chạy **chmod 0777 uploads** trên máy chủ Unix và Mac OS X. Trên Windows, nhấp chuột phải vào thư mục rồi chọn Sharing. Động tác này làm xuất hiện bảng điều khiển Sharing (xem hình 11-19).

Hình 11-19: *Bảng điều khiển Sharing trên Windows 2000 cho phép thiết lập quyền truy cập cho thư mục.*

Tùy vào hệ điều hành, bạn có khả năng tải lên các tập tin mà không cần qua bước này. Bạn nên thử đoạn mã dưới đây (chỉ để kiểm tra) trước khi thiết lập quyền. Nếu thấy thông báo như trong hình 11-20, thì bạn cần phải điều chỉnh.

MAX_FILE_SIZE là qui định hạn chế về kích thước tập tin trong trình duyệt. Tập tin cấu hình PHP có qui định riêng. Bạn cũng có thể kiểm tra kích thước tập tin tải lên trong đoạn mã PHP đang xử lý.

Thiết lập post_max_size trong tập tin php.ini cho biết tổng kích thước dữ liệu (tính bằng megabyte) được tải lên bằng một mã lệnh đơn. Nó được thiết lập mặc định 8MB.



Hình 11-20: *Nếu thiết lập quyền không thích hợp, bạn sẽ thấy thông báo lỗi như trong hình.*

Điều mới trong PHP 4.2 là chỉ mục error của mảng $_FILES. Nó lưu bất kỳ thông tin lỗi nào liên quan đến tập tin tải lên.

Lưu các tập tin tải lên bên ngoài thư mục Web được dùng vì lý do an toàn. Làm như vậy sẽ tránh được việc người dùng truy cập trực tiếp các tập tin.

Vì việc tải lên một tập tin lớn thường chiếm nhiều thời gian, nên bạn cần thay đổi giá trị max_execution_time trong tập tin php.ini hoặc tạm thời bỏ qua nó bằng cách dùng hàm set_time_limit() trong đoạn mã.

Thực hành viết tập tin **add_file.php** theo các bước sau:

1. Tạo hồ sơ PHP mới trong trình soạn thảo văn bản (xem đoạn mã 11.7).

```
<?php # Doan ma 11.7 - add_file.php
```

```
$page_title = 'Tai len tap tin';
include ('includes/header.html');
```

2. Kiểm tra xem biểu mẫu được đưa lên không và kiểm tra phần mô tả.

```
if (isset($_POST['submit'])) {
   require_once ('../mysql_connect.php');
   function escape_data ($data) {
      global $dbc;
      if (ini_get('magic_quotes_gpc')) {
         $data = stripslashes($data);
      }
      return mysql_real_escape_string (trim ($data), $dbc);
   }
   if (!empty($_POST['description'])) {
      $d = escape_data($_POST['description']);
   } else {
      $d = '';
   }
```

Như trong ví dụ **add_url.php**, chúng ta dùng phép kiểm tra tối thiểu trên phần nhập *description*. Vì biểu mẫu này không có các trường khác cần kiểm tra, nên đây sẽ là phần điều kiện duy nhất.

Bạn cũng có thể kiểm tra kích thước tập tin tải lên để xem nó có nằm trong phạm vi chấp nhận được hay không.

3. Chèn mẩu tin vào cơ sở dữ liệu cho phần tải lên này.

```
$query = "INSERT INTO uploads (file_name, file_size,
→ file_type, description, upload_date) VALUES
→ ('{$_FILES['upload']['name']}',
→ {$_FILES['upload']['size']},
→ '{$_FILES['upload']['type']}', '$d', NOW())";
$result = @mysql_query ($query);
```

Tập tin tải lên được chèn vào trong cơ sở dữ liệu bằng câu truy vấn này. Nó dùng mảng đa chiều **$_FILES** để chèn tên tập tin gốc, kích thước và kiểu MIME. Tất cả thông tin này được cung cấp từ trình duyệt Web. Phần mô tả và thời gian hiện tại cũng sẽ được lưu.

4. Tạo tên tập tin mới.

```
if ($result) {
   $extension = explode ('.', $_FILES['upload']['name']);
   $uid = mysql_insert_id();
```

```
$filename = $uid . '.' . $extension[1];
```

Tập tin được lưu trên máy chủ với tên mới. Điều này an toàn hơn so với sử dụng tên gốc do người dùng đặt. Tên của tập tin sẽ là giá trị của trường *upload_id* ứng với nó từ cơ sở dữ liệu (được lấy ra bằng hàm **mysql_insert_id()**), theo sau là dấu chấm rồi đến phần mở rộng tập tin (được xác định bằng cách tách tên tập tin gốc). Ví dụ, hồ sơ ***chapter.doc*** có thể trở thành ***231.doc*** và ***image.jpg*** được lưu với tên ***49.jpg***.

5. Sao chép tập tin vào vị trí mới trên máy chủ.

```
if (move_uploaded_file($_FILES['upload']['tmp_name'],
→ "../uploads/$filename")) {
   echo '<p>Tap tin da duoc tai len!</p>';
} else {
   echo '<p><font color="red">Khong the di chuyen tap
   → tin.</font></p>';
   $query = "DELETE FROM uploads WHERE upload_id = $uid";
   $result = @mysql_query ($query);
}
```

Chúng ta dùng hàm **move_uploaded_file()** để di chuyển tập tin tạm thời đến vị trí thường trú của nó (trong thư mục *uploads)* với tên mới. Nếu không thể di chuyển tập tin, chúng ta xóa mẩu tin từ cơ sở dữ liệu và in thông báo lỗi (xem lại hình 11-20).

6. Hoàn tất các phần điều kiện và vùng PHP.

```
   } else {
      echo '<p><font color="red">Thong tin cua ban khong the
      → xu ly vi co loi he thong. Chung toi xin loi vi su
      → co nay.</font></p>';
   }
   mysql_close();
}
?>
```

7. Tạo biểu mẫu HTML.

```
<form enctype="multipart/form-data" action="<?php echo
→ $_SERVER['PHP_SELF']; ?>" method="post">
<input type="hidden" name="MAX_FILE_SIZE" value="524288">
<fieldset><legend>Dien vao thong tin de tai len tap tin:
→ </legend>
```

```
$page_title = 'Tai len tap tin';
include ('includes/header.html');
```

2. Kiểm tra xem biểu mẫu được đưa lên không và kiểm tra phần mô tả.

```
if (isset($_POST['submit'])) {
   require_once ('../mysql_connect.php');
   function escape_data ($data) {
      global $dbc;
      if (ini_get('magic_quotes_gpc')) {
         $data = stripslashes($data);
      }
      return mysql_real_escape_string (trim ($data), $dbc);
   }
   if (!empty($_POST['description'])) {
      $d = escape_data($_POST['description']);
   } else {
      $d = '';
   }
```

Như trong ví dụ **add_url.php**, chúng ta dùng phép kiểm tra tối thiểu trên phần nhập *description*. Vì biểu mẫu này không có các trường khác cần kiểm tra, nên đây sẽ là phần điều kiện duy nhất.

Bạn cũng có thể kiểm tra kích thước tập tin tải lên để xem nó có nằm trong phạm vi chấp nhận được hay không.

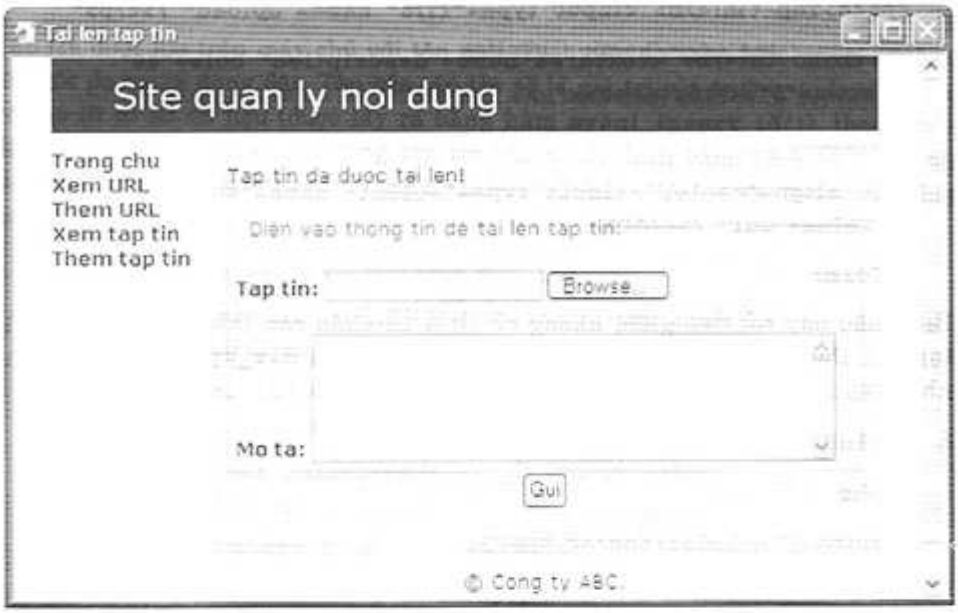
3. Chèn mẩu tin vào cơ sở dữ liệu cho phần tải lên này.

```
$query = "INSERT INTO uploads (file_name, file_size,
→ file_type, description, upload_date) VALUES
→ ('{$_FILES['upload']['name']}',
→ {$_FILES['upload']['size']},
→ '{$_FILES['upload']['type']}', '$d', NOW())";
$result = @mysql_query ($query);
```

Tập tin tải lên được chèn vào trong cơ sở dữ liệu bằng câu truy vấn này. Nó dùng mảng đa chiều **$_FILES** để chèn tên tập tin gốc, kích thước và kiểu MIME. Tất cả thông tin này được cung cấp từ trình duyệt Web. Phần mô tả và thời gian hiện tại cũng sẽ được lưu.

4. Tạo tên tập tin mới.

```
if ($result) {
   $extension = explode ('.', $_FILES['upload']['name']);
   $uid = mysql_insert_id();
```



Hình 11-22: *Kết quả nếu việc tải lên tập tin thành công.*

Đoạn mã 11.7: *Đoạn mã này cho phép tải lên tập tin và lưu nó trên máy chủ.*

```
<?php # Doan ma 11.7 - add_file.php
$page_title = 'Tai len tap tin';
include ('includes/header.html');
if (isset($_POST['submit'])) {
   require_once ('../mysql_connect.php');
   function escape_data ($data) {
      global $dbc;
      if (ini_get('magic_quotes_gpc')) {
         $data = stripslashes($data);
      }
      return mysql_real_escape_string (trim ($data), $dbc);
   }
   if (!empty($_POST['description'])) {
      $d = escape_data($_POST['description']);
   } else {
      $d = '';
   }
   $query = "INSERT INTO uploads (file_name, file_size,
```

```
   →file_type, description, upload_date) VALUES
   → ('{$_FILES['upload']['name']}',
   →{$_FILES['upload']['size']},
   →'{$_FILES['upload']['type']}', '$d', NOW())";
   $result = @mysql_query ($query);
   if ($result) {
      $extension = explode ('.', $_FILES['upload']['name']);
      $uid = mysql_insert_id();
      $filename = $uid . '.' . $extension[1];
      if (move_uploaded_file($_FILES['upload']['tmp_name'],
      →"../uploads/$filename")) {
         echo '<p>Tap tin da duoc tai len!</p>';
      } else {
         echo '<p><font color="red">Khong the di chuyen tap
         →tin.</font></p>';
         $query = "DELETE FROM uploads WHERE upload_id=$uid";
         $result = @mysql_query ($query);
      }
   } else {
      echo '<p><font color="red">Thong tin cua ban khong the
      → xu ly vi co loi he thong. Chung toi xin loi vi su co
      → nay.</font></p>';
   }
   mysql_close();
}
?>
<form enctype="multipart/form-data" action="<?php echo
→ $_SERVER['PHP_SELF']; ?>" method="post">
<input type="hidden" name="MAX_FILE_SIZE" value="524288">
<fieldset><legend>Dien vao thong tin de tai len tap tin:
→</legend>
<p><b>Tap tin:</b> <input type="file" name="upload" /></p>
<p><b>Mo ta:</b> <textarea name="description" cols="40"
→ rows="5"></textarea></p>
</fieldset>
<div align="center"><input type="submit" name="submit"
→ value="Gui" /></div>
</form>
```

```
<?php
include ('includes/footer.html');
?>
```

Sự tồn tại của tập tin tải lên có thể được kiểm tra bằng hàm ***is_upload_file()****.*

Người dùng Windows phải sử dụng dấu chéo thuận hay hai dấu chéo ngược khi tham chiếu thư mục (ví dụ, dùng C:/ hoặc C:\\ thay vì C:\) vì dấu chéo ngược là ký tự escape trong PHP.

Nếu dùng những phiên bản PHP trước đây, bạn không thể dùng mảng ***$_FILES*** *và phải dùng hàm* ***copy()*** *thay cho* ***move_uploaded_file()****.*

Hàm ***move_uploaded_file()*** *sẽ ghi đè lên tập tin hiện có mà không thông báo khi các tập tin trùng tên.*

Khi đưa đoạn mã vào hoạt động, bạn nên làm ẩn các thông báo lỗi bằng cách đặt ký tự ***@*** *trước hàm* ***move_uploaded_file()****.*

Quản trị site

Mặc dù chúng ta không tạo phần quản trị cho ứng dụng này, nhưng làm điều đó cũng không khó. Đầu tiên, bổ sung trường approved vào bảng uploads. Sau đó, chỉnh sửa tập tin ***add_url.php*** *và* ***add_file.php*** *để trường approved được thiết lập mặc định là N. Trên phía quản trị, tạo trang lấy ra tất cả nội dung chưa được chấp thuận. Để chấp thuận một mẩu tin, chạy câu truy vấn UPDATE trên bảng thích hợp cho mẩu tin đó, chuyển approved thành Y.*

Một trong những ưu điểm của ứng dụng này là dễ dàng liên kết các tập tin trên máy chủ với những tập tin trong cơ sở dữ liệu (vì ***upload_id*** *được dùng như tên tập tin). Nếu viết một đoạn mã quản trị để loại các tập tin khỏi máy chủ, bạn có thể xóa mẩu tin ra khỏi bảng và tập tin trong thư mục uploads.*

Xem và tải xuống các tập tin

Hai đoạn mã cuối trong ứng dụng này cho phép người dùng xem các tập tin đã tải lên và sau đó tải chúng xuống (từng tập tin một). Phần xem thì tương đối rõ, nhưng đoạn mã cho việc tải xuống đòi hỏi sử dụng nhiều hàm **header()** của PHP.

Thực hành viết tập tin **view_files.php** theo các bước sau:

1. Tạo hồ sơ PHP mới trong trình soạn thảo văn bản.

```
<?php # Doan ma 11.8 - view_files.php
$page_title = 'Xem tap tin';
include_once ('includes/header.html');
```

```
require_once ('../mysql_connect.php');
$first = TRUE;
```

Biến `$first` được dùng như trong đoạn mã `view_urls.php` để tạo phần đầu trang và cho biết có thể xem được tập tin nào không.

2. Lấy ra toàn bộ các tập tin từ cơ sở dữ liệu.

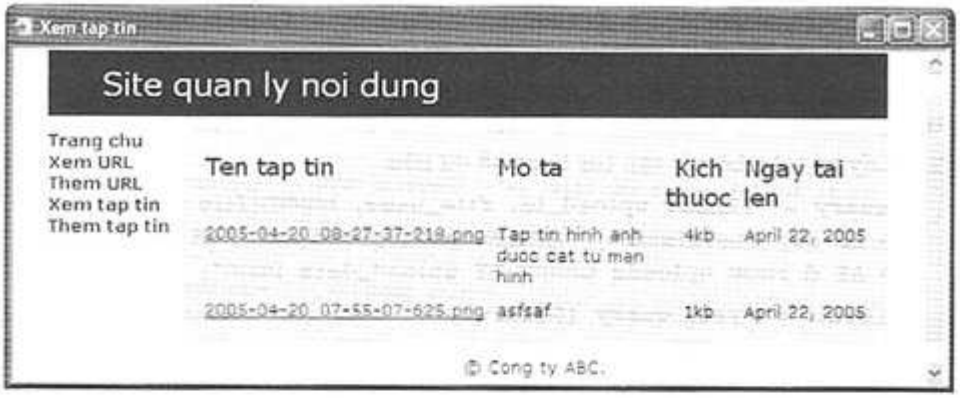
```
$query = "SELECT upload_id, file_name, ROUND(file_size/1024)
→ AS fs, description, DATE_FORMAT(upload_date,'%M %e, %Y')
→ AS d FROM uploads ORDER BY upload_date DESC";
$result = mysql_query ($query);
```

Câu truy vấn này lấy ra trường `upload_id`, `file_name`, `description` và ngày được định dạng cho mỗi tập tin tải lên. Tập tin mới được tải lên được liệt kê đầu tiên. Đồng thời, câu truy vấn trả lại kích thước tập tin tính theo kilobyte, bằng cách chia kích thước tập tin lưu trữ cho 1.024 và sau đó làm tròn số.

3. Hiển thị từng mẩu tin.

```
while ($row = mysql_fetch_array($result, MYSQL_ASSOC)) {
   if ($first) {
      echo '<table border="0" width="100%" cellspacing="3"
      → cellpadding="3" align="center">
      →<tr>
      →<td align="left" width="20%"><font size="+1">Ten
      → tap tin</font></td>
      →<td align="left" width="40%"><font size="+1">Mo ta
      →</font></td>
      →<td align="center" width="20%"><font size="+1">
      →Kich thuoc</font></td>
      →<td align="left" width="20%"><font size="+1">Ngay
      → tai len</font></td></tr>';
   }
   echo "<tr>
   →<td align=\"left\"><a href=\"download_file.php?
   →uid={$row['upload_id']}\">{$row['file_name']}</a></td>
   →<td align=\"left\">" . tripslashes($row['description'])
   → . "</td><td align=\"center\">{$row['fs']}kb</td>
   →<td align=\"left\">{$row['d']}</td></tr>\n";
   $first = FALSE;
}
```

Mã lệnh này rất giống với mã lệnh tương ứng của nó trong `view_urls.php`. Biến `$first` được dùng để tạo phần đầu trang (xem hình 11-23) và sau đó các mẩu tin được in ra. Chúng ta dùng hằng `MYSQL_ASSOC` với hàm `mysql_fetch_array` và do vậy các trường được tham chiếu với cú pháp `$row['ten_truong']`.



Hình 11-23: *Trang* ***view_files.php***

Mỗi tên tập tin là một liên kết với đoạn mã **download_file.php**, với giá trị **upload_id** được gắn vào URL. Giá trị này được đoạn mã tải xuống sử dụng để biết tập tin nào cần gửi cho trình duyệt Web.

4. Hoàn tất bảng hoặc hiển thị thông báo nếu không có tập tin.

```
if ($first) {
    echo '<div align="center">Hien khong co tap tin nao de
    → xem.</div>';
} else {
    echo '</table>';
}
```

5. Hoàn tất trang PHP.

```
mysql_close(); include_once ('includes/footer.html');
?>
```

6. Lưu tập tin với tên **view_files.php**, tải lên máy chủ Web và chạy thử trong trình duyệt. Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 11.8.

Đoạn mã 11.8: *Đoạn mã* ***view_files.php*** *thể hiện các tập tin tải lên, kèm theo mô tả, kích thước và ngày được tải lên.*

```
<?php # Doan ma 11.8 - view_files.php
$page_title = 'Xem tap tin';
include_once ('includes/header.html');
require_once ('../mysql_connect.php');
$first = TRUE;
$query = "SELECT upload_id, file_name, ROUND(file_size/1024)
→ AS fs, description, DATE_FORMAT(upload_date, '%M %e, %Y')
→ AS d FROM uploads ORDER BY upload_date DESC";
```

```
$result = mysql_query ($query);
while ($row = mysql_fetch_array ($result, MYSQL_ASSOC)) {
   if ($first) {
      echo '<table border="0" width="100%" cellspacing="3"
      → cellpadding="3" align="center">
      <tr>
      <td align="left" width="20%"><font size="+1">Ten tap
      → tin</font></td>
      <td align="left" width="40%"><font size="+1">Mo ta
      → </font></td>
      → <td align="center" width="20%"><font size="+1">Kich
      → thuoc</font></td>
      → <td align="left" width="20%"><font size="+1">Ngay tai
      → len</font></td></tr>';
   }
   echo "<tr>
   → <td align=\"left\"><a href=\"download_file.php?uid=
   → {$row['upload_id']}\">{$row['file_name']}</a></td>
   → <td align=\"left\">" . stripslashes($row['description'])
   → . "</td>
   → <td align=\"center\">{$row['fs']}kb</td>
   → <td align=\"left\">{$row['d']}</td>
   → </tr>\n";
   $first = FALSE;
}
if ($first) {
   echo '<div align="center">Hien khong co tap tin nao de
   → xem.</div>';
} else {
   echo '</table>';
}
mysql_close();
include_once ('includes/footer.html');
?>
```

Thực hành viết tập tin **download_file.php** theo các bước sau:

1. Tạo hồ sơ PHP mới trong trình soạn thảo văn bản.

```
<?php # Doan ma 11.9 - download_file.php
```

2. Kiểm tra một trường *upload_id*.

```
if (is_numeric ($_GET['uid'])) {
```

Trước khi tiếp tục, chúng ta cần đảm bảo đoạn mã nhận được một giá trị *upload_id* hợp lệ.

3. Lấy ra thông tin cho tập tin ứng với giá trị *upload_id*.

```
require_once ('../mysql_connect.php');
$query = "SELECT file_name, file_type, file_size FROM
→ uploads WHERE upload_id = {$_GET['uid']}";
$result = mysql_query ($query);
list($fn,$ft,$fs) = mysql_fetch_array ($result,MYSQL_NUM);
mysql_close();
```

Để tải xuống tập tin, cần phải biết tên, kiểu và kích thước tập tin. Chúng ta lấy toàn bộ thông tin từ cơ sở dữ liệu bằng cách dùng `$_GET['uid']` trong câu truy vấn và sau đó dùng hàm `list()`.

4. Xác định tên tập tin.

```
$extension = explode ('.', $fn);
$the_file = '../uploads/' . $_GET['uid'] . '.' .
→ $extension[1];
```

Quá trình này tương tự phương pháp đặt tên dùng trong tập tin **add_file.php**, ngoại trừ việc bổ sung thư mục *../uploads/*. Việc thiết lập tên tập tin dưới dạng một biến sẽ giúp dễ dàng tham chiếu nó trong đoạn mã sau này.

5. Kiểm tra tồn tại của tập tin trên máy chủ.

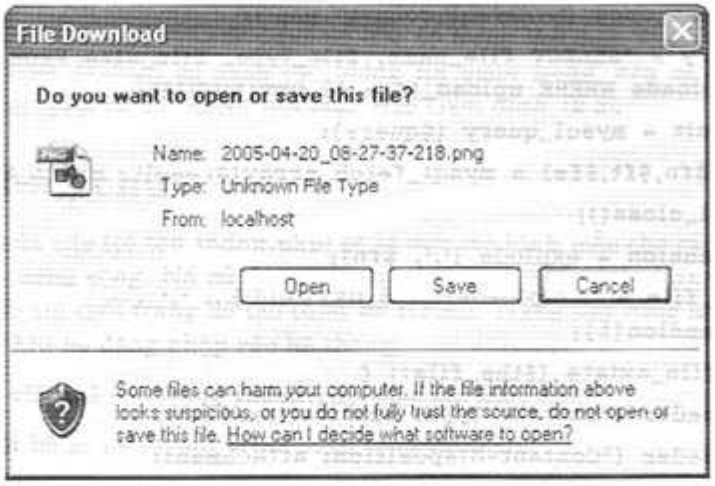
```
if (file_exists ($the_file)) {
```

Trước khi gửi tập tin đến trình duyệt Web, chúng ta cần kiểm tra sự tồn tại của nó. Hàm **file_exist()** trả lại giá trị **TRUE** nếu tập tin có trên máy chủ.

6. Gửi tập tin.

```
header ("Content-Type: application/$ft");
header ("Content-disposition: attachment; filename=$fn");
header ("Content-Length: $fs");
readfile ($the_file);
$message = '<p>Tap tin da duoc gui di.</p>';
```

Mã lệnh trên sẽ gửi dữ liệu tập tin đến trình duyệt Web, tạo ra một hộp nhắc tải xuống (xem hình 11-24). Dòng lệnh thứ nhất chuẩn bị cho trình duyệt tiếp nhận tập tin, dựa trên kiểu MIME (lưu trong cơ sở dữ liệu khi tập tin được tải lên). Dòng thứ hai thiết lập tên tập tin sẽ được tải xuống, sử dụng tên tập tin gốc như trên máy tính của người dùng. Lời gọi hàm **header()** cuối cùng cho biết kích thước dữ liệu sẽ gửi (được xác định khi tập tin được tải lên). Bản thân dữ liệu tập tin được gửi bằng hàm **readfile()**. Hàm này đọc vào tập tin và gửi nội dung của nó đến trình duyệt Web.



Hình 11-24: *Nhấp vào liên kết trong trang* ***view_files.php*** *sẽ dẫn đến hộp nhắc này.*

7. Hoàn tất các phần điều kiện.

```
} else {
   $message = '<p><font color="red">Hay chon mot tap tin de
   → tai xuong.</font></p>';
}
```

8. Tạo nội dung trang Web.

```
$page_title = 'Tai xuong tap tin';
include_once ('includes/header.html');
echo $message;
include_once ('includes/footer.html');
?>
```

Hầu hết các trình duyệt Web sẽ không bao giờ thấy được thông tin này. Điều mà phần lớn người dùng sẽ thấy là hộp nhắc tải xuống sau khi nhấp vào liên kết **view_files.php**. Tuy nhiên, việc đưa vào mã này ở đây sẽ an toàn vì nó được dùng nếu gặp sự cố.

9. Lưu tập tin với tên **download_file.php**, tải lên máy chủ Web và chạy thử trong trình duyệt. Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 11.9.

Đoạn mã 11.9: *Đoạn mã này buộc tải xuống một tập tin bằng cách gửi thông tin thích hợp đến trình duyệt Web.*

```
<?php # Doan ma 11.9 - download_file.php
if (is_numeric ($_GET['uid'])) {
```

```
   require_once ('../mysql_connect.php');
   $query = "SELECT file_name, file_type, file_size FROM
   → uploads WHERE upload_id = {$_GET['uid']}";
   $result = mysql_query ($query);
   list($fn,$ft,$fs) = mysql_fetch_array($result, MYSQL_NUM);
   mysql_close();
   $extension = explode ('.', $fn);
   $the_file = '../uploads/' . $_GET['uid'] . '.' .
   $extension[1];
   if (file_exists ($the_file)) {
      header ("Content-Type: application/$ft");
      header ("Content-disposition: attachment;
      → filename=$fn");
      header ("Content-Length: $fs");
      readfile ($the_file);
      $message = '<p>Tap tin da duoc gui di.</p>';
   } else {
      $message = '<p><font color="red">Khong tim thay tap tin
      →tren may chu. Chung toi xin loi vi dieu nay.</font>
      →</p>';
   }
} else {
   $message = '<p><font color="red">Hay chon mot tap tin de
   → tai xuong.</font></p>';
}
$page_title = 'Tai xuong tap tin';
include_once ('includes/header.html');
echo $message;
include_once ('includes/footer.html');
?>
```

Hàm `mime_content_type()` bắt đầu có từ phiên bản PHP 4.3. Hàm trả lại kiểu MIME cho tập tin trên máy chủ.

Các trình duyệt Web xử lý việc tải xuống tập tin theo các cách khác nhau. Bạn nên kiểm tra các đoạn mã trên nhiều trình duyệt và môi trường khác nhau để đảm bảo kết quả đáng tin cậy.

Chương 12:

VÍ DỤ - ĐĂNG KÝ NGƯỜI DÙNG

Ví dụ thứ hai trong cuốn sách này là một trong những ứng dụng phổ biến của PHP và MySQL: hệ thống đăng ký người dùng. Chương này sẽ phát triển và sử dụng phần lớn mã kịch bản đã được đề cập trong các chương trước và đặt chúng vào trong cùng một ngữ cảnh bằng một lý thuyết lập trình nhất quán.

Ứng dụng có hai phần: phần công cộng (public side) và phần quản trị (administrative side). Người dùng phần công cộng có thể đăng ký, đăng nhập, thoát ra, thay đổi và thiết lập lại mật khẩu (khi họ bị quên). Phần quản trị sẽ chỉ hiển thị danh sách người dùng. Phần công cộng sử dụng session, cookie, trong khi các trang quản trị sẽ được bảo vệ với mã kịch bản kiểm tra xác thực HTTP.

Tạo khuôn mẫu

Hình 12-1: *Thiết kế cơ bản của ứng dụng Web trong chương này.*

Ứng dụng trong chương này sẽ sử dụng một thiết kế khuôn mẫu mới chứ không sử dụng lại các khuôn mẫu trong các chương trước. Khuôn mẫu này sử dụng chủ yếu CSS (Cascading Style Sheet - bảng kiểu), tạo ra dạng thể hiện mới mà không cần nhiều hình ảnh. Nó cũng đã được thử nghiệm trên nhiều trình duyệt hiện tại


```
    require_once ('../mysql_connect.php');
    $query = "SELECT file_name, file_type, file_size FROM
    → uploads WHERE upload_id = {$_GET['uid']}";
    $result = mysql_query ($query);
    list($fn,$ft,$fs) = mysql_fetch_array($result, MYSQL_NUM);
    mysql_close();
    $extension = explode ('.', $fn);
    $the_file = '../uploads/' . $_GET['uid'] . '.' .
    $extension[1];
    if (file_exists ($the_file)) {
        header ("Content-Type: application/$ft");
        header ("Content-disposition: attachment;
        → filename=$fn");
        header ("Content-Length: $fs");
        readfile ($the_file);
        $message = '<p>Tap tin da duoc gui di.</p>';
    } else {
        $message = '<p><font color="red">Khong tim thay tap tin
        →tren may chu. Chung toi xin loi vi dieu nay.</font>
        →</p>';
    }
} else {
    $message = '<p><font color="red">Hay chon mot tap tin de
    → tai xuong.</font></p>';
}
$page_title = 'Tai xuong tap tin';
include_once ('includes/header.html');
echo $message;
include_once ('includes/footer.html');
?>
```

Hàm `mime_content_type()` bắt đầu có từ phiên bản PHP 4.3. Hàm trả lại kiểu MIME cho tập tin trên máy chủ.

Các trình duyệt Web xử lý việc tải xuống tập tin theo các cách khác nhau. Bạn nên kiểm tra các đoạn mã trên nhiều trình duyệt và môi trường khác nhau để đảm bảo kết quả đáng tin cậy.

```
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">
<head>
<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />
<title><?php echo $page_title; ?></title>
<style type="text/css" media="screen">
@import "includes/layout.css";
</style>
</head>
```

Biến **$page_title** được in ra trong phạm vi của thẻ **title**. Sau đó, hồ sơ CSS được đưa vào. Tập tin CSS có tên là **layout.css** và được đặt trong thư mục *includes*.

5. Bắt đầu phần thân HTML.

```
<body>
<div id="Header">Xay dung Website dong voi PHP va
→ MySQL</div>
<div id="Content">
```

Phần thân sẽ tạo ra một biểu ngữ ngang qua phần đầu của trang và sau đó bắt đầu phần nội dung của trang Web (cho đến phần "Tieu de trang" như trong hình 12-1).

6. Lưu tập tin với tên **header.html**. Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 12.1.

Đoạn mã 12.1: *Tập tin đầu trang bắt đầu phần HTML, khởi tạo session và chế độ xuất ra bộ đệm.*

```
<?php # Doan ma 12.1 - header.html
ob_start();
session_start();
if (!isset($page_title)) {
   $page_title = 'Xay dung Website dong voi PHP va MySQL';
}
?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
→ Transitional//EN" "http://www.w3.org/TR/2000/REC-xhtml1-
→20000126/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">
<head>
```

```
<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />
<title><?php echo $page_title; ?></title>
<style type="text/css" media="screen">
@import "includes/layout.css";
</style>
</head>
<body>
<div id="Header">Xay dung Website dong voi PHP va MySQL</div>
<div id="Content">
```

Thực hành tạo tập tin **footer.html** theo các bước sau:

1. Tạo một hồ sơ mới trong trình soạn thảo văn bản.

```
</div>
<div id="Menu">
<a href="index.php">Trang chu</a><br/>
```

2. Hiển thị các liên kết thích hợp.

```
<?php # Doan ma 12.2 - footer.html
if (isset($_SESSION['user_id']) AND
→ (substr($_SERVER['PHP_SELF'], -10) != 'logout.php')) {
   echo '<a href="logout.php">Thoat ra</a><br/><a
   → href="change_password.php">Thay doi mat khau</a>
   →<br/>';
} else {
   echo '<a href="register.php">Dang ky</a><br/>
   → <a href="login.php">Dang nhap</a><br/><a
   → href="forgot_password.php">Quen mat khau</a><br/>';
}
?>
```

Nếu người dùng đã đăng nhập (nghĩa là biến **$_SESSION['user_id']** đã có giá trị) và đây không phải là trang thoát ra, người dùng sẽ thấy liên kết đến trang thoát ra và trang thay đổi mật khẩu. Nếu không, người dùng sẽ thấy liên kết đến trang đăng nhập và trang thiết lập lại mật khẩu bị quên.

3. Hoàn tất phần HTML.

```
<a href="#">Trang khac 1</a><br/>
```

```
    <a href="#">Trang khac 2</a><br/>
</div>
</body>
</html>
```

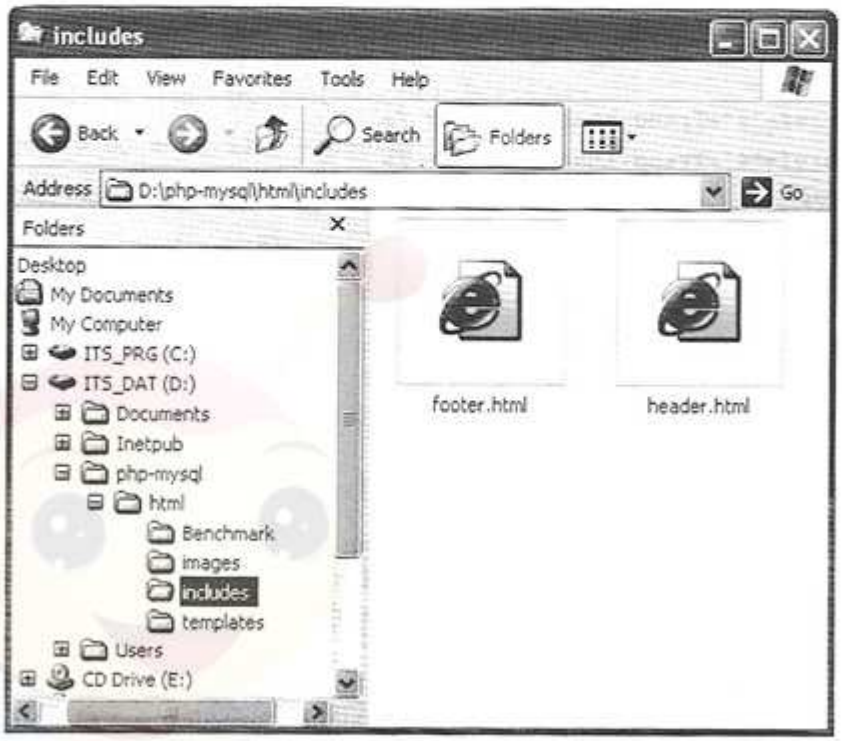
Chúng ta đưa vào hai liên kết giả ở đây để minh họa cho các liên kết đến các trang khác.

4. Gửi nội dung của bộ đệm xuống trình duyệt Web.

```
<?php
ob_flush();
?>
```

Tập tin cuối trang sẽ gửi dữ liệu được tập hợp trong bộ đệm xuống trình duyệt Web, kết thúc chế độ xuất ra bộ đệm đã được khởi tạo trong mã kịch bản đầu trang.

5. Lưu tập tin với tên gọi **footer.html** và tải lên máy chủ cùng với tập tin **header.html**. Đặt các tập tin này trong thư mục *includes* (xem hình 12-2). Mã lệnh đầy đủ của tập tin **footer.html** được thể hiện trong đoạn mã 12.2.



Hình 12-2: *Cấu trúc thư mục của site trên máy chủ Web, với thư mục gốc của Website là html.*

Đoạn mã 12.2: *Tập tin cuối trang kết thúc phần HTML, hiển thị các liên kết dựa trên trạng thái của người dùng (đăng nhập hay chưa đăng nhập) và đẩy toàn bộ dữ liệu trong bộ đệm xuống trình duyệt Web.*

```
</div>
<div id="Menu">
<a href="index.php">Trang chu</a><br />
<?php # Doan ma 12.2 - footer.html
if (isset($_SESSION['user_id']) AND
→ (substr($_SERVER['PHP_SELF'], -10) != 'logout.php')) {
   echo '<a href="logout.php">Thoat ra</a><br/>
   → <a href="change_password.php">Thay doi mat khau</a>
   →<br/>';
} else {
   echo '  <a href="register.php">Dang ky</a><br/>
   → <a href="login.php">Dang nhap</a><br/>
   → <a href="forgot_password.php">Quen mat khau</a><br/>';
}
?>
<a href="#">Trang khac 1</a><br />
<a href="#">Trang khac 2</a><br />
</div>
</body>
</html>
<?php
ob_flush();
?>
```

Viết mã kịch bản cấu hình

Website này sử dụng hai mã kịch bản dạng cấu hình. Mã kịch bản thứ nhất (`config.inc`) dùng để quản lý lỗi. Mã kịch bản thứ hai (`mysql_connect.php`) sẽ chứa tất cả thông tin cần thiết để kết nối cơ sở dữ liệu.

Tạo một tập tin cấu hình

Mục đích duy nhất của tập tin cấu hình là thiết lập chính sách quản lý lỗi trong site. Ở giai đoạn phát triển, chúng ta muốn các lỗi sẽ được thông báo và khi chúng xảy ra, một cảnh báo dạng in đậm sẽ được in ra trình duyệt (xem hình 12-3). Khi site được đưa lên mạng, chúng ta muốn xử lý lỗi theo một cách khác: tắt hoàn toàn chế độ thông báo lỗi và chuyển chúng đến một địa chỉ thư điện tử khi nó xảy ra.

Hình 12-3: *Mã kịch bản cấu hình này được dùng để thiết lập chính sách quản lý lỗi cho ứng dụng.*

Thực hành viết mã cấu hình **config.inc** theo các bước sau:

1. Tạo một hồ sơ mới trong trình soạn thảo văn bản.

```
<?php # Doan ma 12.3 - config.inc
```

2. Thiết lập chế độ thông báo lỗi.

```
//error_reporting (0);

error_reporting (E_ALL);
```

Thông báo lỗi có hai giá trị: **0** - dùng khi Website được đưa lên mạng, và **E_ALL** - dùng trong giai đoạn phát triển. Trong tập tin cấu hình, chúng ta viết cả hai dòng và sau đó chỉ việc ghi chú một dòng, để lại một dòng tùy vào việc ứng dụng đang ở giai đoạn nào.

3. Tạo một hàm xử lý lỗi.

```
function my_error_handler ($e_number, $e_message,
→ $e_file, $e_line) {

   $message = 'Co mot loi xay ra trong ma kich ban ' .
   → $e_file . ' tai dong ' . $e_line . ": $e_message";

   //error_log ($message, 1, 'error@domain.com');

   echo '<font color="red" size="+1">', $message,
   → '</font>';

}
```

```
set_error_handler('my_error_handler');
```

Như được đề cập trong Chương 9 "Phát triển ứng dụng Web", chúng ta có thể viết hàm xử lý lỗi của riêng mình. Khi đó, ta thiết lập PHP để nó sử dụng hàm này bằng cách gửi tên của hàm dưới dạng một tham số cho hàm **set_error_handler()**.

Khi một lỗi xảy ra, hàm **set_error_handler()** sẽ nhận hai tham số: số hiệu lỗi và thông báo lỗi. Chúng ta sử dụng thông tin này, cùng với các hằng **__FILE__** và **__LINE__**, để tạo một thông báo chi tiết. Trong giai đoạn phát triển, nó sẽ được in ở định dạng in đậm (xem hình 12-3). Khi site được đưa lên mạng, bạn sẽ ghi chú dòng lệnh **echo()** và bỏ ghi chú dòng **error_log()**. Với thiết lập này, các thông báo lỗi sẽ được gửi đến một địa chỉ thư điện tử.

4. Đóng phần PHP.

```
?>
```

5. Lưu tập tin với tên gọi **config.inc**, tải lên máy chủ Web và đặt trong thư mục *includes*. Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 12.3.

Đoạn mã 12.3: *Mã kịch bản cấu hình này được sử dụng để thiết lập chính sách xử lý lỗi cho ứng dụng.*

```
<?php # Doan ma 12.3 - config.inc

error_reporting (E_ALL);

function my_error_handler ($e_number, $e_message, $e_file,
→ $e_line) {

   $message = 'Co mot loi xay ra trong ma kich ban ' .
   → $e_file . ' tai dong ' . $e_line . ": $e_message";

   echo '<font color="red" size="+1">', $message, '</font>';

}

set_error_handler('my_error_handler');

?>
```

Nếu Website có các hàm hoặc hằng cần được sử dụng trong nhiều trang, hãy đặt chúng trong tập tin ***config.inc****.*

Một dạng khác thay cho việc sử dụng ghi chú để thiết lập việc xử lý lỗi là sử dụng biến ***$stage****. Biến này sẽ có hai giá trị live và develop. Khi đó, hàm* ***error_reporting()*** *và* ***my_error_handler()*** *sẽ xử lý lỗi dựa trên giá trị của biến* ***$stage****.*

Các hằng ***__FILE__*** *và* ***__LINE__*** *có phạm vi toàn cục, nên chúng có thể được truy xuất phía trong hàm* ***my_error_handler()****.*

Tạo mã kịch bản cơ sở dữ liệu

Mã kịch bản cấu hình thứ hai là **mysql_connect.php**. Đây là một biến thể của tập tin kết nối cơ sở dữ liệu đã được sử dụng nhiều lần trong cuốn sách này. Mục đích chính của nó là kết nối MySQL và chọn cơ sở dữ liệu. Trong chương này, chúng ta sẽ dùng công cụ xử lý lỗi được thiết lập trong tập tin **config.inc**, thay vì sử dụng hàm **die(mysql_error())**. Mã kịch bản này cũng định nghĩa hàm **escape_data()** để xử lý tất cả các dữ liệu biểu mẫu trước khi chúng được sử dụng trong câu truy vấn.

Thực hành viết mã kịch bản **mysql_connect.php** theo các bước sau:

1. Tạo một hồ sơ mới trong trình soạn thảo văn bản.

```
<?php # Doan ma 12.4 - mysql_connect.php
```

2. Thiết lập thông tin truy xuất cơ sở dữ liệu.

```
DEFINE ('DB_USER', 'username');

DEFINE ('DB_PASSWORD', 'password');

DEFINE ('DB_HOST', 'localhost');

DEFINE ('DB_NAME', 'sitename');
```

3. Kết nối MySQL và chọn cơ sở dữ liệu.

```
if ($dbc = @mysql_connect (DB_HOST, DB_USER, DB_PASSWORD)){

   if (!mysql_select_db (DB_NAME)) {
```

Thay vì ngừng thực hiện mã lệnh, chúng ta sẽ dùng hàm xử lý lỗi của mình và viết lại các bước dưới dạng các điều kiện.

Quyền truy xuất cơ sở dữ liệu

Để đảm bảo an toàn, bạn nên thiết lập một tài khoản và mật khẩu riêng để truy xuất cơ sở dữ liệu của ứng dụng này thay vì dùng chung một tài khoản hoặc sử dụng tài khoản root (rất không nên).

Tài khoản MySQL này cần có quyền chèn (INSERT), cập nhật (UPDATE) và chọn (SELECT) mẩu tin. Với một tài khoản mới và quyền truy xuất cụ thể như vậy, bạn sẽ tránh được các rủi ro nếu như thông tin truy xuất cơ sở dữ liệu bị lộ.

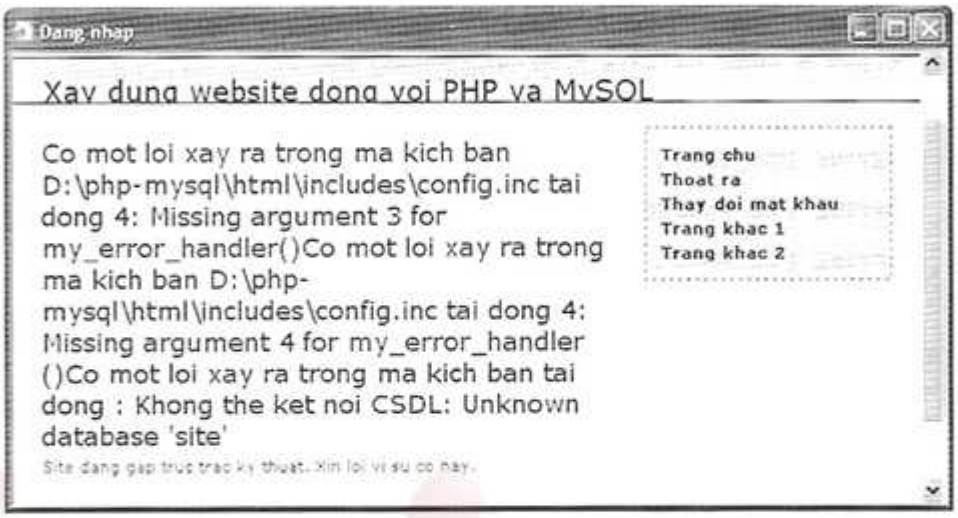
4. Xử lý lỗi nếu cơ sở dữ liệu không được chọn.

```
my_error_handler(mysql_errno(),'Khong the ket noi CSDL: '
→ . mysql_error());
echo '<p><font color="red">Site dang gap truc trac ky
→ thuat. Xin loi vi su co nay.</font></p>';
include_once ('includes/footer.html');
```

```
        exit();
    }
```

Nếu mã kịch bản không thể chọn được cơ sở dữ liệu, thông báo lỗi sẽ được gửi đến cho hàm **my_error_handler()**. Bằng cách này, chúng ta đảm bảo lỗi sẽ được xử lý tùy theo thiết lập xử lý lỗi hiện tại (live hoặc develope).

Nếu không thể chọn được cơ sở dữ liệu, mã kịch bản sẽ không thể tiếp tục xử lý. Trong trường hợp này, chúng ta in ra thông báo cho người dùng biết (khi site được đưa lên mạng), kèm theo phần cuối trang và kết thúc mã kịch bản. Hình 12-4 thể hiện kết quả này trong giai đoạn phát triển và hình 12-5 phản ánh cũng kết quả đó khi site được đưa lên mạng.



Hình 12-4: *Một lỗi cơ sở dữ liệu xảy ra trong quá trình phát triển.*

Hình 12-5: *Kết quả của cùng một lỗi (xem hình 12-4) khi site được đưa lên mạng.*

5. Lặp lại quá trình nếu kết nối không thể được thiết lập.

```
} else {

   my_error_handler(mysql_errno(),'Khong the ket noi CSDL:
   → . mysql_error());

   echo '<p><font color="red">Site dang gap truc trac ky
   → thuat. Xin loi vi su co nay.</font></p>';

   include_once ('includes/footer.html');

   exit();

}
```

Các dòng lệnh trên cũng tương tự mã lệnh trong bước 4 và sẽ được thực hiện nếu mã kịch bản không thể kết nối với MySQL. Trong mã lệnh trên, các hàm `mysql_errno()` và `mysql_error()` được dùng để gửi số hiệu và thông báo lỗi cho hàm `my_error_handler()`.

6. Tạo hàm `escape_data()`.

```
function escape_data ($data) {

   global $dbc;

   if (ini_get('magic_quotes_gpc')) {

      $data = stripslashes($data);

   }

   return mysql_real_escape_string (trim ($data), $dbc);

}
```

Hàm `escape_data()` sẽ được sử dụng để xử lý tất cả dữ liệu nhận được từ biểu mẫu HTML. Vì các trang đăng nhập, đăng ký, thay đổi mật khẩu và thiết lập lại mật khẩu đều có sử dụng kết nối cơ sở dữ liệu, nên chúng ta sẽ đặt hàm này ở đây thay vì đặt trong từng trang.

Chú ý, hàm `escape_data()` sẽ cắt bỏ các khoảng trống dư trong giá trị và sau đó mã hóa escape nó bằng hàm `mysql_real_escape_string()`. Nếu đặc điểm Magic Quotes được kích hoạt, dữ liệu sẽ được giải mã escape trước để loại bỏ các dấu chéo có mặt.

7. Hoàn tất mã lệnh PHP.

```
?>
```

8. Lưu tập tin với tên gọi **mysql_connect.php**, tải nó lên máy chủ và đặt vào một thư mục phía ngoài thư mục gốc của Web (vì lý do an toàn). Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 12.4.

Đoạn mã 12.4: *Mã kịch bản kết nối cơ sở dữ liệu sẽ tạo hàm để mã hóa escape dữ liệu và quản lý các lỗi liên quan đến MySQL.*

```
<?php # Doan ma 12.4 - mysql_connect.php
DEFINE ('DB_USER', 'username');
DEFINE ('DB_PASSWORD', 'password');
DEFINE ('DB_HOST', 'localhost');
DEFINE ('DB_NAME', 'sitename');
if ($dbc = @mysql_connect (DB_HOST, DB_USER, DB_PASSWORD)) {
   if (!mysql_select_db (DB_NAME)) {
      my_error_handler(mysql_errno(),'Khong the ket noi
      → CSDL: ' . mysql_error());
      echo '<p><font color="red">Site dang gap truc trac ky
      → thuat. Xin loi vi su co nay.</font></p>';
      include_once ('includes/footer.html');
      exit();
   }
} else {
   my_error_handler (mysql_errno(), 'Khong the ket noi
   → CSDL: ' . mysql_error());
   echo '<p><font color="red">Site dang gap truc trac ky
   → thuat. Xin loi vi su co nay.</font></p>';
   include_once ('includes/footer.html');
   exit();
}
function escape_data ($data) {
   global $dbc;
   if (ini_get('magic_quotes_gpc')) {
      $data = stripslashes($data);
   }
   return mysql_real_escape_string (trim ($data), $dbc);
}
?>
```

Bạn có thể đặt nội dung của cả hai tập tin cấu hình trong cùng một tập tin để tiện tham chiếu. Tuy nhiên, cách này làm tăng một lượng xử lý không cần thiết (kết nối cơ sở dữ liệu) đối với các mã kịch bản không cần kết nối cơ sở dữ liệu (như ***index.php****).*

Vì tập tin cấu hình không chứa thông tin nhạy cảm, nên chúng ta sử dụng phần mở rộng là **.inc** *và đặt nó trong thư mục Web. Mã kịch bản kết nối MySQL có chứa các thông tin quan trọng, nên sẽ có phần mở rộng* **.php** *và đặt ngoài thư mục Web (xem hình 12-2).*

Tạo trang chủ

Trang chủ của site (có tên **index.php**) sẽ là một mô hình mẫu cho các trang khác trong phần công cộng. Nó sử dụng tập tin cấu hình (để xử lý lỗi), tập tin đầu trang và tập tin cuối trang để tạo thiết kế HTML. Trang này cũng sẽ chào mừng người dùng khi họ đăng nhập vào hệ thống.

Thực hành viết mã cho trang **index.php** theo các bước sau:

1. Tạo một hồ sơ mới trong trình soạn thảo văn bản.

```
<?php # Doan ma 12.5 - index.php
```

2. Đưa vào tập tin cấu hình, thiết lập tiêu đề trang và đưa vào tập tin đầu trang.

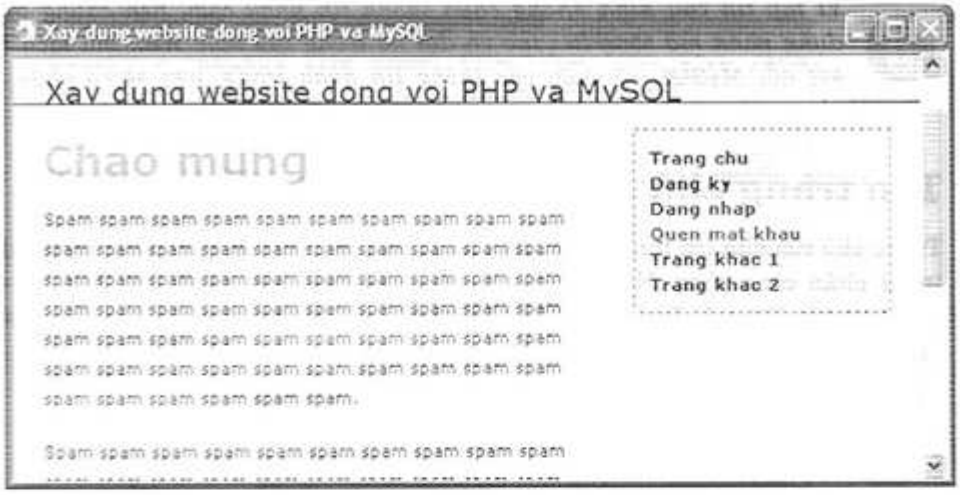
```
require_once ('includes/config.inc');

$page_title = 'Xay dung Website dong voi PHP va MySQL';

include_once ('includes/header.html');
```

Mã kịch bản đưa vào tập tin cấu hình để những gì xảy ra sau đó sẽ được xử lý bằng quá trình quản lý lỗi được thiết lập trong đó. Tiếp đến, tập tin **header.html** được đưa vào để khởi tạo chế độ xuất ra bộ đệm, bắt đầu session và khởi tạo phần đầu của bố cục HTML.

3. Chào mừng người dùng và hoàn tất mã lệnh PHP.

```
echo '<h1>Chao mung';
if (isset($_SESSION['first_name'])) {
   echo ", {$_SESSION['first_name']}!";
}
echo '</h1>';
?>
```

Thông điệp *Chao mung* sẽ được in cho tất cả mọi người. Nếu biến **$_SESSION['first_name']** được thiết lập, tên của người dùng cũng sẽ được in ra. Vì thế kết quả của khối lệnh này có thể là *Chao mung* (xem hình 12-6) hoặc *Chao mung, <ten nguoi dung>!* (xem hình 12-7).



Hình 12-6: *Nếu người dùng chưa đăng nhập, họ sẽ thấy trang này.*

Hình 12-7: *Nếu người dùng đã đăng nhập, họ sẽ thấy trang này.*

4. Tạo nội dung cho trang.

```
<p>Spam spam spam spam spam spam spam spam spam spam spam
spam spam spam spam spam spam spam spam spam spam spam
spam spam spam spam spam spam spam spam spam spam spam
spam spam spam spam spam spam spam spam spam spam spam
spam spam spam spam spam spam spam spam spam spam spam
spam spam spam spam spam spam spam.</p>
```

```
<p>Spam spam spam spam spam spam spam spam spam spam spam
spam spam spam spam spam spam spam spam spam spam spam spam
spam spam spam spam spam spam spam spam spam spam spam spam
spam spam spam spam spam spam spam spam spam spam spam spam
spam spam spam spam spam spam spam spam spam spam spam spam
spam spam spam spam spam spam spam spam spam spam spam spam
spam spam spam spam spam spam spam spam spam spam spam spam
spam spam spam spam spam spam spam spam spam spam spam spam
spam spam spam spam spam spam spam spam spam spam spam spam
spam spam spam.</p>
```

5. Đưa vào tập tin cuối trang.

```
<?php
include_once ('includes/footer.html');
?>
```

Tập tin cuối trang sẽ hoàn tất bố cục HTML (chủ yếu là thanh menu ở phía phải của trang) và kết thúc chế độ xuất ra bộ đệm.

6. Lưu tập tin với tên gọi **index.php**, tải nó lên máy chủ và chạy thử trong trình duyệt. Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 12.5.

Đoạn mã 12.5: *Mã kịch bản cho trang chủ của site.*

```
<?php # Doan ma 12.5 - index.php
require_once ('includes/config.inc');
$page_title = 'Xay dung Website dong voi PHP va MySQL';
include_once ('includes/header.html');
echo '<h1>Chao mung';
if (isset($_SESSION['first_name'])) {
   echo ", {$_SESSION['first_name']}!";
}
echo '</h1>';
?>
<p>Spam spam spam spam spam spam spam spam spam spam spam
spam spam spam spam spam spam spam spam spam spam spam spam
spam spam spam spam spam spam spam spam spam spam spam spam
spam spam spam spam spam spam spam spam spam spam spam spam
spam spam spam spam spam spam spam spam spam spam spam spam
spam spam spam spam spam spam spam.</p>
<p>Spam spam spam spam spam spam spam spam spam spam spam
spam spam spam spam spam spam spam spam spam spam spam spam
spam spam spam spam spam spam spam spam spam spam spam spam
```

```
spam spam spam spam spam spam spam spam spam spam spam spam
spam spam spam spam spam spam spam spam spam spam spam spam
spam spam spam spam spam spam spam spam spam spam spam spam
spam spam spam spam spam spam spam spam spam spam spam spam
spam spam spam spam spam spam spam spam spam spam spam spam
spam spam spam spam spam spam spam spam spam spam spam spam
spam spam spam.</p>
<?php
include_once ('includes/footer.html');
?>
```

Nếu so sánh menu chứa các liên kết trong hình 12-6 và 12-7, bạn sẽ thấy điều kiện trong tập tin cuối trang **`footer.html`** *làm thay đổi các tùy chọn như thế nào.*

Đăng ký

Mã kịch bản đăng ký được viết lần đầu trong Chương 6 "Sử dụng PHP và MySQL". Nó cũng đã được cải tiến theo nhiều cách khác nhau. Phiên bản này của tập tin **`register.php`** sẽ thực hiện các nhiệm vụ sau:

- Hiển thị và xử lý biểu mẫu.
- Kiểm tra tất cả dữ liệu được gửi lên bằng cách sử dụng các biểu thức quy tắc.
- Hiển thị lại biểu mẫu với các giá trị được ghi nhớ nếu có một vấn đề nào đó xảy ra.
- Xử lý dữ liệu được gửi lên bằng hàm **`escape_data()`** có trong mã kịch bản **`mysql_connect.php`**.
- Đảm bảo tên đăng nhập là duy nhất.

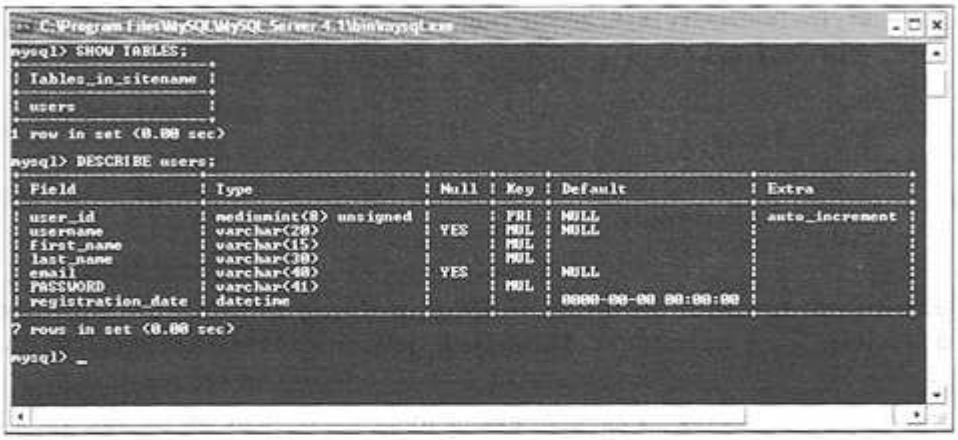
Lược đồ cơ sở dữ liệu

Cơ sở dữ liệu được sử dụng trong ứng dụng này được tạo ra trong Chương 4 "Giới thiệu SQL và MySQL" và đã được chỉnh sửa một số lần. Nó chỉ gồm một bảng users mà bạn có thể tạo lại bằng câu lệnh SQL sau:

```
CREATE TABLE users (
user_id MEDIUMINT UNSIGNED NOT NULL AUTO_INCREMENT,
first_name VARCHAR(15) NOT NULL,
last_name VARCHAR(30) NOT NULL,
email VARCHAR(40),
password CHAR(16) NOT NULL,
registration_date DATETIME NOT NULL,
PRIMARY KEY (user_id),
UNIQUE KEY username (username),
```

```
KEY first_name (first_name),
KEY last_name (last_name),
KEY password (password),
);
```

Bạn có thể kiểm tra cấu trúc của một cơ sở dữ liệu bằng lệnh **SHOW TABLES**, *cấu trúc bảng với lệnh* **DESCRIBE <tên bảng>** *(xem hình 12-8).*



Hình 12-8: *Trước khi viết mã kịch bản PHP tương tác với cơ sở dữ liệu, bạn nên ghi lại cấu trúc của cơ sở dữ liệu.*

Thực hành viết mã kịch bản **register.php** theo các bước sau:

1. Tạo một hồ sơ mới trong trình soạn thảo văn bản.

```
<?php # Doan ma 12.6 - register.php
```

2. Đưa vào tập tin cấu hình và tập tin đầu trang HTML.

```
require_once ('includes/config.inc');
$page_title = 'Register';
include ('includes/header.html');
```

3. Viết điều kiện kiểm tra xem biểu mẫu có được gửi hay không và đưa vào mã kịch bản kết nối cơ sở dữ liệu.

```
if (isset($_POST['submit'])) {
   require_once ('../mysql_connect.php');
```

Trong khi các phiên bản khác của mã kịch bản này định nghĩa hàm **escape_data()** tại đây, mã kịch bản **register.php** lại lấy nó từ mã kịch bản **mysql_connect.php** cùng với kết nối cơ sở dữ liệu.

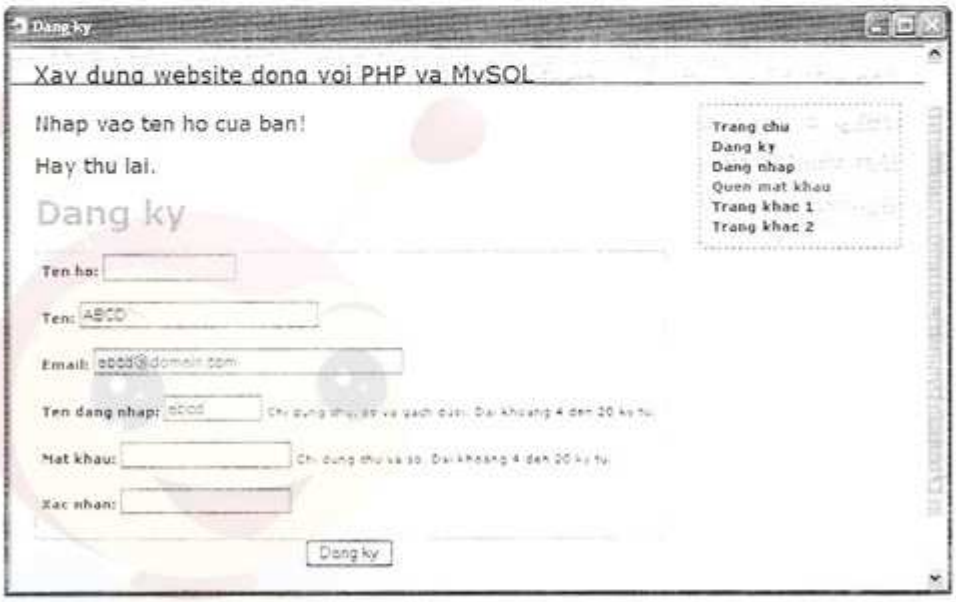
4. Kiểm tra tên và tên họ.

```
if (eregi ("^[[:alpha:].' -]{2,15}$",
→ stripslashes(trim($_POST['first_name'])))) {
   $fn = escape_data($_POST['first_name']);
```

```
} else {
    $fn = FALSE;
    echo '<p><font color="red" size="+1">Nhap vao ten ho
    → cua ban!</font></p>';
}
if (eregi ("^[[:alpha:].' -]{2,30}$",
→ stripslashes(trim($_POST['last_name'])))) {
    $ln = escape_data($_POST['last_name']);
} else {
    $ln = FALSE;
    echo '<p><font color="red" size="+1">Nhap vao ten cua
    → ban!</font></p>';
}
```

Biểu mẫu sẽ được kiểm tra bằng biểu thức quy tắc (đã được đề cập trong Chương 8 "An toàn"). Giá trị của tên chỉ được phép chứa các chữ cái, dấu chấm, dấu nháy và dấu chéo. Ngoài ra, chiều dài của nó phải từ 2 đến 15 ký tự.

Nếu điều kiện được thỏa mãn, biến **$fn** sẽ nhận giá trị được gửi lên sau khi được mã hóa bằng hàm **escape_data()**. Nếu không, **$fn** sẽ có giá trị false và một thông báo sẽ được in ra (xem hình 12-9).



Hình 12-9: *Nếu giá trị của tên không đáp ứng được điều kiện, một thông báo lỗi sẽ được in ra.*

5. Kiểm tra địa chỉ thư điện tử và tên đăng nhập (xem hình 12-10).

```
if (eregi ("^[[:alnum:]][a-z0-9_.-]*@[a-z0-9.-]+\.
→ [a-z]{2,4}$", stripslashes(trim($_POST['email'])))) {
   $e = escape_data($_POST['email']);
} else {
   $e = FALSE;
   echo '<p><font color="red" size="+1">Nhap vao dia chi
   → thu dien tu cua ban!</font></p>';
}
if (eregi ("^[[:alnum:]_]{4,20}$",
→ stripslashes(trim($_POST['username'])))) {
   $u = escape_data($_POST['username']);
} else {
   $u = FALSE;
   echo '<p><font color="red" size="+1">Nhap vao ten dang
   → nhap!</font></p>';
}
```

Hình 12-10: *Địa chỉ thư điện tử và tên đăng nhập phải có định dạng chính xác.*

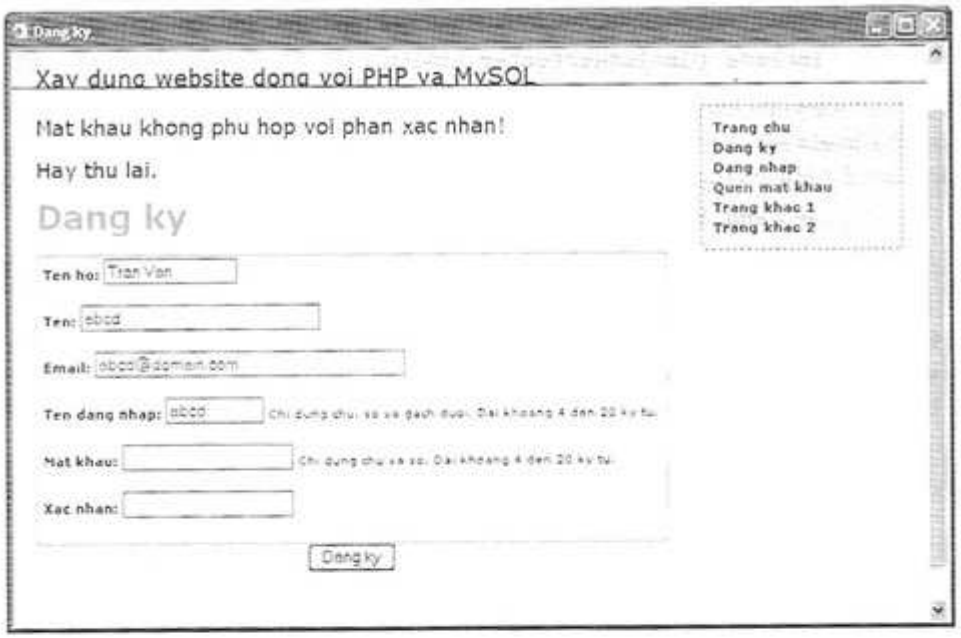
Mẫu kiểm tra của địa chỉ thư điện tử đã được mô tả trong Chương 8. Biểu thức quy tắc kiểm tra tên đăng nhập sẽ giới hạn giá trị trong phạm vi các chữ cái, chữ số, gạch dưới và có chiều dài khoảng từ 4 đến 20 ký tự. Trong mã lệnh trên, chúng ta

chỉ kiểm tra giá trị của biến đã được cắt xén các khoảng trắng ở đầu và cuối, các dấu chéo đã được loại bỏ để tránh vấn đề với Magic Quotes (làm cho giá trị không đáp ứng được biểu thức quy tắc). Các giá trị được xử lý thông qua hàm **escape_data()** và **eregi()**.

6. Kiểm tra mật khẩu (xem hình 12-11 và 12-12).

```
if (eregi ("^[[:alnum:]]{4,20}$",
→ stripslashes(trim($_POST['password1'])))) {
   if ($_POST['password1'] == $_POST['password2']) {
      $p = escape_data($_POST['password1']);
   } else {
      $p = FALSE;
      echo '<p><font color="red" size="+1">Mat khau khong
      → phu hop voi phan xac nhan!</font></p>';
   }
} else {
   $p = FALSE;
   echo '<p><font color="red" size="+1">Nhap vao mat khau
   → cua ban!</font></p>';
}
```

Hình 12-11: *Mật khẩu được kiểm tra để đảm bảo đúng định dạng, chiều dài.*



Hình 12-12: *Giá trị mật khẩu và giá trị kiểm tra phải phù hợp nhau.*

Mật khẩu phải có chiều dài từ 4 đến 20 ký tự và chỉ chứa chữ cái và chữ số. Hơn thế nữa, mật khẩu đầu (password1) phải trùng với mật khẩu kiểm tra (password2).

7. Nếu tất cả các bước kiểm tra đều thành công, chúng ta tiếp tục kiểm tra tính duy nhất của tên đăng nhập.

```
if ($fn && $ln && $e && $u && $p) {

    $query = "SELECT user_id FROM users WHERE
    -> username='$u'";

    $result = @mysql_query ($query);
```

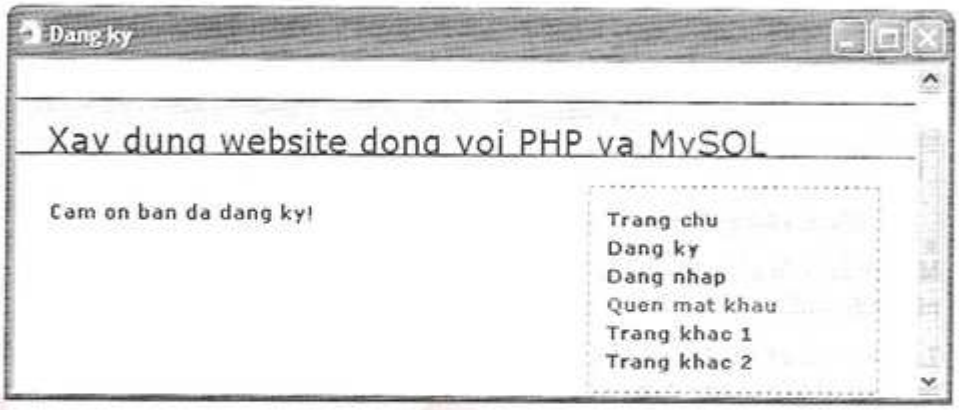
Nếu biểu mẫu vượt qua được tất cả các bước kiểm tra, điều kiện sẽ là TRUE. Khi đó, chúng ta cần kiểm tra xem tên đăng nhập được gửi lên đã có chưa để đảm bảo tính duy nhất của cột tương ứng trong cơ sở dữ liệu.

8. Nếu tên đăng nhập đáp ứng mọi điều kiện, người dùng sẽ được đăng ký.

```
if (mysql_num_rows($result) == 0) {
    $query = "INSERT INTO users (username, first_name,
    -> last_name, email, password, registration_date)
    -> VALUES ('$u', '$fn', '$ln', '$e', PASSWORD('$p'),
    -> NOW() )";
    $result = @mysql_query ($query);
    if ($result) {
```

```
        echo '<p><b>Cam on ban da dang ky!</b></p>';
        include ('includes/footer.html');
        exit();
    } else {
        echo '<p><font color="red" size="+1">Ban khong the
        → dang ky vi co loi he thong. Xin loi vi su co
        → nay.</font></p>';
    }
```

Phần đầu của mã lệnh trên thực hiện việc đăng ký người dùng: chèn một mẩu tin vào trong cơ sở dữ liệu. Thông báo cảm ơn sẽ được in ra khi thành công (xem hình 12-13). Nếu cần, một thư điện tử với thông tin xác nhận cũng sẽ được gửi.



Hình 12-13: *Trang kết quả sau khi người dùng đăng ký thành công.*

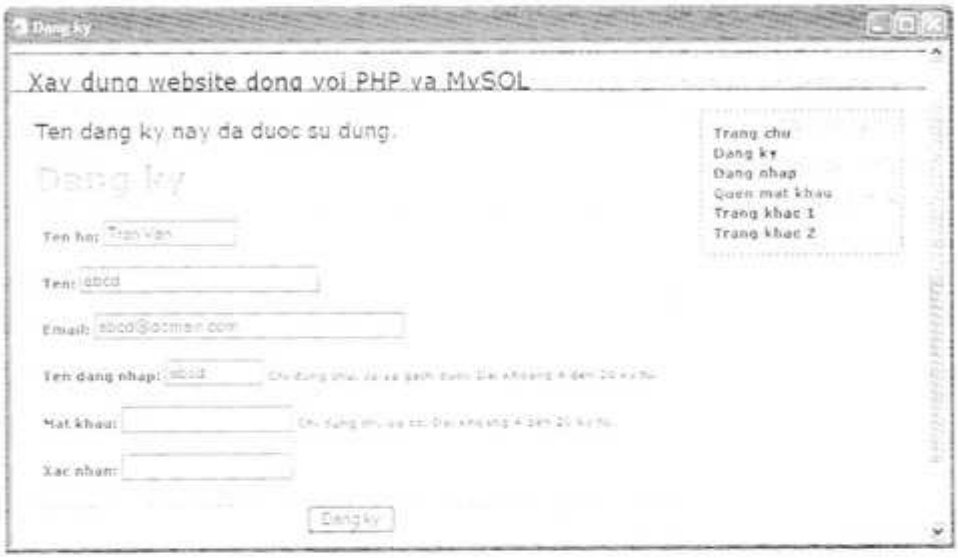
Nếu câu truy vấn không thực hiện được vì một lý do nào đó, thông báo lỗi sẽ được in ra trình duyệt. Về mặt lôgic, chúng ta có thể gọi hàm `my_error_handler()` tại thời điểm này (tuy nhiên, phiên bản hoàn tất của ứng dụng sẽ không bao giờ xảy ra một vấn đề gì tại điểm này).

9. Hoàn tất điều kiện và mã lệnh PHP.

```
    } else {
        echo '<p><font color="red" size="+1">Ten dang ky
        → nay da duoc su dung.</font></p>';
    }
    mysql_close();
} else {
    echo '<p><font color="red" size="+1">Hay thu lai.
    →</font></p>';
}
```

```
}
?>
```

Mệnh đề **else** đầu tiên sẽ được thực hiện nếu người dùng thử đăng ký với một tên đăng nhập đã có trong cơ sở dữ liệu (xem hình 12-14). Mệnh đề **else** thứ hai được áp dụng khi dữ liệu gửi lên không đáp ứng tất cả các điều kiện kiểm tra (xem lại từ hình 12-9 đến 12-12).

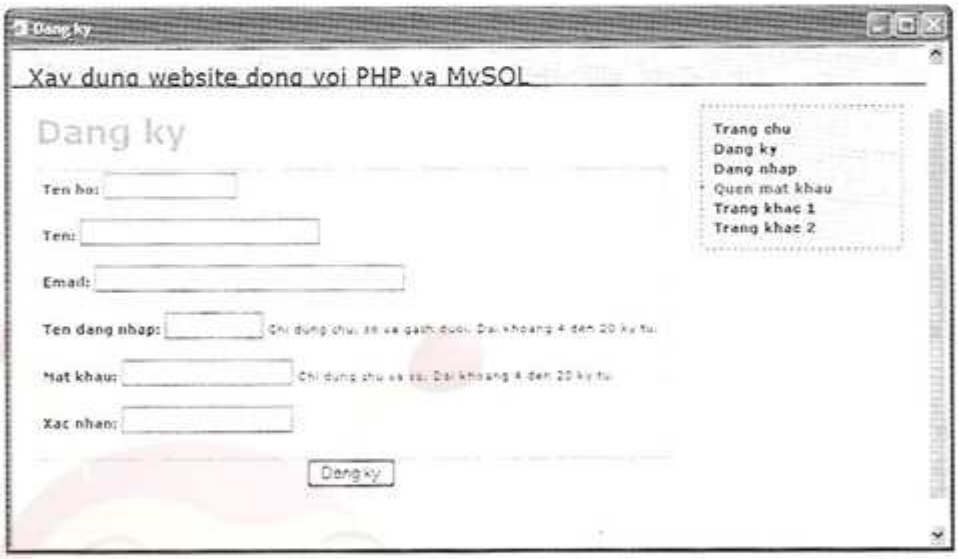


Hình 12-14: *Nếu tên đăng nhập đã được sử dụng, người dùng sẽ được thông báo để đăng ký lại với tên khác.*

10. Hiển thị biểu mẫu HTML (xem hình 12-15).

```
<h1>Dang ky</h1>
<form action="<?php echo $_SERVER['PHP_SELF']; ?>"
→ method="post">
<fieldset>
<p><b>Ten ho:</b> <input type="text" name="first_name"
→ size="15" maxlength="15" value="<?php if
→ (isset($_POST['first_name'])) echo
→ $_POST['first_name'];?>" /></p>
<p><b>Ten:</b> <input type="text" name="last_name"
→ size="30" maxlength="30" value="<?php if
→ (isset($_POST['last_name'])) echo $_POST['last_name'];
→ ?>" /></p>
<p><b>Email:</b> <input type="text" name="email"
→ size="40" maxlength="40" value="<?php if
→ (isset($_POST['email'])) echo $_POST['email']; ?>" /> </p>
```

```
<p><b>Ten dang nhap:</b> <input type="text"
→ name="username" size="10" maxlength="20" value="<?php
→ if (isset($_POST['username'])) echo $_POST['username'];
→ ?>" /> <small>Chi dung chu, so va gach duoi. Dai
→ khoang 4 den 20 ky tu.</small></p>

<p><b>Mat khau:</b> <input type="password"
→ name="password1" size="20" maxlength="20" /> <small>Chi
→ dung chu va so. Dai khoang 4 den 20 ky tu.</small></p>

<p><b>Xac nhan:</b> <input type="password"
→ name="password2" size="20" maxlength="20" /></p>

</fieldset>

<div align="center"><input type="submit" name="submit"
→ value="Dang ky" /></div>

</form>
```



Hình 12-15: *Biểu mẫu đăng ký khi người dùng truy cập lần đầu.*

Biểu mẫu HTML này sử dụng mã lệnh PHP nhằm thiết lập thuộc tính action (để nó tham chiếu đến chính mã kịch bản này) và thiết lập trước các giá trị nhập nếu như chúng tồn tại. Nó cũng đưa vào một mô tả ngắn gọn về cú pháp đúng cho phần tên đăng ký và mật khẩu.

11. Đưa vào phần cuối HTML.

```
<?php

include ('includes/footer.html');

?>
```

12. Lưu tập tin với tên gọi **register.php**, tải nó lên máy chủ Web và chạy thử trong trình duyệt. Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 12.6.

Đoạn mã 12.6: *Mã kịch bản đăng ký sử dụng các biểu thức quy tắc kiểm tra và tạo biểu mẫu thuận tiện cho người dùng.*

```
<?php # Doan ma 12.6 - register.php
require_once ('includes/config.inc');
$page_title = 'Dang ky';
include ('includes/header.html');
if (isset($_POST['submit'])) {
   require_once ('../mysql_connect.php');
   if (eregi ("^[[:alpha:].' -]{2,15}$",
   → stripslashes(trim($_POST['first_name'])))) {
      $fn = escape_data($_POST['first_name']);
   } else {
      $fn = FALSE;
      echo '<p><font color="red" size="+1">Nhap vao ten ho
      → cua ban!</font></p>';
   }
   if (eregi ("^[[:alpha:].' -]{2,30}$",
   → stripslashes(trim($_POST['last_name'])))) {
      $ln = escape_data($_POST['last_name']);
   } else {
      $ln = FALSE;
      echo '<p><font color="red" size="+1">Nhap vao ten cua
      → ban!</font></p>';
   }
   if (eregi ("^[[:alnum:]][a-z0-9_.-]*@[a-z0-9.-]+\.
   → [a-z]{2,4}$", stripslashes(trim($_POST['email'])))) {
      $e = escape_data($_POST['email']);
   } else {
      $e = FALSE;
      echo '<p><font color="red" size="+1">Nhap vao dia chi
      → thu dien tu cua ban!</font></p>';
   }
   if (eregi ("^[[:alnum:]_]{4,20}$",
   → stripslashes(trim($_POST['username'])))) {
```

```
    $u = escape_data($_POST['username']);
} else {
    $u = FALSE;
    echo '<p><font color="red" size="+1">Nhap vao ten dang
    → nhap!</font></p>';
}
if (eregi ("^[[:alnum:]]{4,20}$",
→ stripslashes(trim($_POST['password1'])))) {
    if ($_POST['password1'] == $_POST['password2']) {
        $p = escape_data($_POST['password1']);
    } else {
        $p = FALSE;
        echo '<p><font color="red" size="+1">Mat khau khong
        → phu hop voi phan xac nhan!</font></p>';
    }
} else {
    $p = FALSE;
    echo '<p><font color="red" size="+1">Nhap vao mat khau
    → cua ban!</font></p>';
}
if ($fn && $ln && $e && $u && $p) {
    $query = "SELECT user_id FROM users WHERE
    → username='$u'";
    $result = @mysql_query ($query);
    if (mysql_num_rows($result) == 0) {
        $query = "INSERT INTO users (username, first_name,
        → last_name, email, password, registration_date)
        → VALUES ('$u', '$fn', '$ln', '$e', PASSWORD('$p'),
        → NOW() )";
        $result = @mysql_query ($query);
        if ($result) {
            echo '<p><b>Cam on ban da dang ky!</b></p>';
            include ('includes/footer.html');
            exit();
        } else {
            echo '<p><font color="red" size="+1">Ban khong the
            → dang ky vi co loi he thong. Xin loi vi su co
            → nay.</font></p>';
```

```
        }
      } else {
        echo '<p><font color="red" size="+1">Ten dang ky nay
        → da duoc su dung.</font></p>';
      }
      mysql_close();
    } else {
      echo '<p><font color="red" size="+1">Hay thu lai.
      →</font></p>';
    }
}
?>
<h1>Dang ky</h1>
<form action="<?php echo $_SERVER['PHP_SELF']; ?>"
→ method="post">
<fieldset>
<p><b>Ten ho:</b> <input type="text" name="first_name"
→ size="15" maxlength="15" value="<?php if
→ (isset($_POST['first_name'])) echo $_POST['first_name'];
→ ?>" /></p>
<p><b>Ten:</b> <input type="text" name="last_name" size="30"
→ maxlength="30" value="<?php if (isset($_POST['last_name']))
→ echo $_POST['last_name']; ?>" /></p>
<p><b>Email:</b> <input type="text" name="email" size="40"
→ maxlength="40" value="<?php if (isset($_POST['email']))
→ echo $_POST['email']; ?>" /> </p>
<p><b>Ten dang nhap:</b> <input type="text" name="username"
→ size="10" maxlength="20" value="<?php if
→ (isset($_POST['username'])) echo $_POST['username']; ?>"/>
→ <small>Chi dung chu, so va gach duoi. Dai khoang 4 den 20
→ ky tu.</small></p>
<p><b>Mat khau:</b> <input type="password" name="password1"
→ size="20" maxlength="20" /> <small>Chi dung chu va so. Dai
→ khoang 4 den 20 ky tu.</small></p>
<p><b>Xac nhan:</b> <input type="password" name="password2"
→ size="20" maxlength="20" /></p>
</fieldset>
<div align="center"><input type="submit" name="submit"
→ value="Dang ky" /></div>
```

```
</form>
<?php
include ('includes/footer.html');
?>
```

Vì các cột trong bảng users không được phép có giá trị NULL, nên các giá trị đều phải được nhập. Nếu bảng có một trường tùy chọn, chúng ta vẫn cần phải đảm bảo chúng có đúng định dạng nếu được nhập, nhưng không bắt buộc phải có giá trị. Một ví dụ cho điều này là trường mô tả khi bổ sung URL trong chương trước.

Ngoài các trường được mã hóa (như mật khẩu), chiều dài tối đa của các trường nhập trên biểu mẫu và biểu thức quy tắc phải tương ứng với chiều dài tối đa của cột trong bảng.

Nếu Magic Quotes có hiệu lực, thì nội dung ghi nhớ của biểu mẫu phải được loại bỏ các dấu chéo trước khi in lại. Ví dụ:

```
<?php if(isset($_POST['username']))
→ echo stripslashes($_POST['username']);?>
```

Đăng nhập và thoát ra

Chương 7 "Cookie và Session" đã viết nhiều phiên bản của mã kịch bản **login.php** và **logout.php** với cookie và session. Chương này sẽ phát triển phiên bản chuẩn hóa của cả hai bằng cách gắn chúng với ứng dụng.

Thực hành viết mã kịch bản **login.php** theo các bước sau:

1. Tạo một hồ sơ mới trong trình soạn thảo văn bản.

```
<?php # Doan ma 12.7 - login.php
```

2. Đưa vào tập tin cấu hình và phần đầu HTML.

```
require_once ('includes/config.inc');
$page_title = 'Login';
include ('includes/header.html');
```

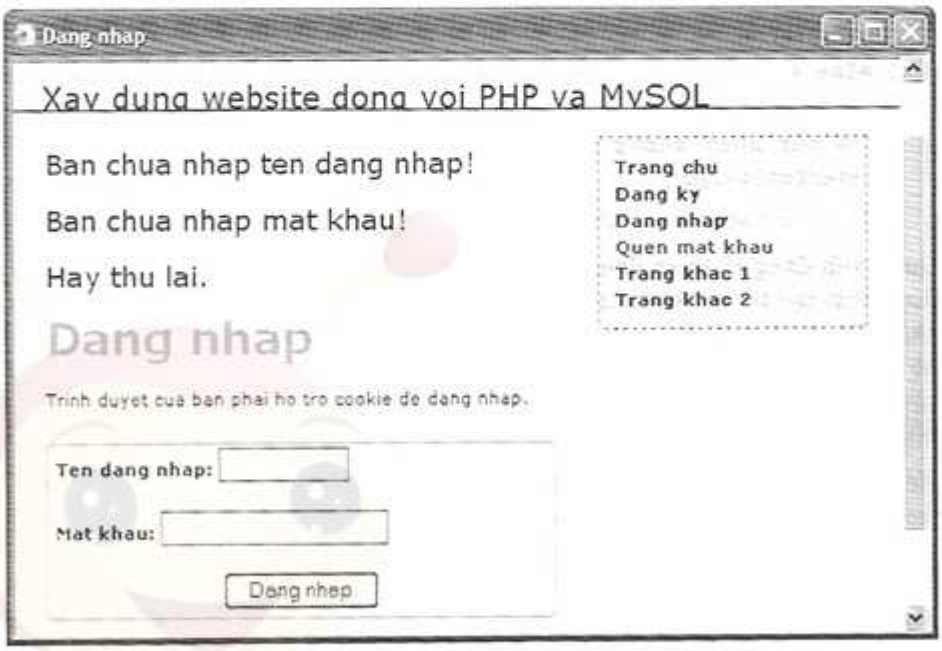
3. Kiểm tra xem biểu mẫu có được gửi không, kết nối cơ sở dữ liệu và kiểm tra dữ liệu gửi.

```
if (isset($_POST['submit'])) {
  require_once ('../mysql_connect.php');
  if (empty($_POST['username'])) {
    $u = FALSE;
    echo '<p><font color="red" size="+1">Ban chua nhap
    → ten dang nhap!</font></p>';
  } else {
```

```
        $u = escape_data($_POST['username']);
    }
    if (empty($_POST['password'])) {
        $p = FALSE;
        echo '<p><font color="red" size="+1">Ban chua nhap
        → mat khau!</font></p>';
    } else {
        $p = escape_data($_POST['password']);
    }
```

Có thể kiểm tra dữ liệu bằng cách dùng biểu thức quy tắc. Tuy nhiên, mức kiểm tra này không đảm bảo và làm chậm quá trình xử lý. Vì tên đăng nhập và mật khẩu phải đáp ứng chỉ tiêu nhất định để quá trình đăng nhập có thể hoạt động, nên các giá trị gửi lên phải phù hợp với các chỉ tiêu đó. Do đó, chúng ta không cần áp dụng biểu thức quy tắc ở đây.

Nếu người dùng không nhập bất kỳ giá trị nào trong biểu mẫu, một thông báo lỗi sẽ được in ra (xem hình 12-16).



Hình 12-16: *Biểu mẫu đăng nhập chỉ kiểm tra xem các giá trị có được nhập hay không mà không sử dụng biểu thức quy tắc.*

4. Nếu cả hai thủ tục kiểm tra đều thỏa mãn, chúng ta lấy ra thông tin người dùng.

```
if ($u && $p) {
    $query = "SELECT user_id, first_name FROM users WHERE
    → username='$u' AND password=PASSWORD('$p')";
    $result = @mysql_query ($query);
    $row = mysql_fetch_array ($result, MYSQL_NUM);
```

Câu truy vấn (giống như trong các ví dụ của các chương trước) lấy ra trường **first_name** và **user_id** của mẩu tin phù hợp với tên đăng nhập và mật khẩu đã được gửi. Kết quả của nó sẽ được gán cho mảng **$row**.

5. Nếu tìm thấy một mẩu tin trong cơ sở dữ liệu, cho phép người dùng đăng nhập.

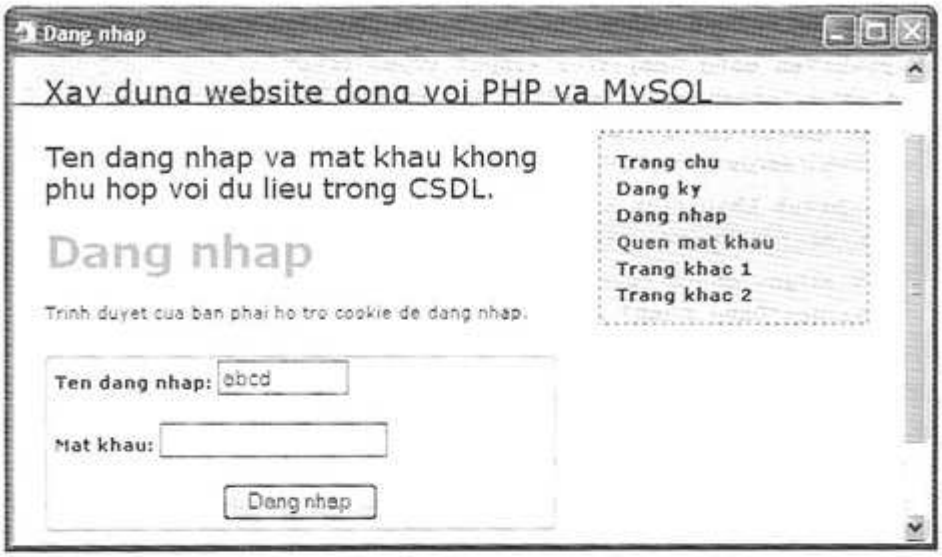
```
if ($row) {
    $_SESSION['first_name'] = $row[1];
    $_SESSION['user_id'] = $row[0];
    ob_end_clean();
    header ("Location:  http://" . $_SERVER['HTTP_HOST'] .
    → dirname($_SERVER['PHP_SELF']) . "/index.php");
    exit();
} else {
    echo '<p><font color="red" size="+1">Ten dang nhap va
    → mat khau khong phu hop voi du lieu trong CSDL.
    →</font></p>';
}
```

Quá trình đăng nhập sẽ đăng ký các giá trị lấy ra từ cơ sở dữ liệu với session (đã được khởi tạo trong tập tin **header.html**) và sau đó chuyển người dùng đến trang chủ. Hàm **ob_end_clean()** sẽ xóa dữ liệu hiện có trong bộ đệm (chế độ xuất ra bộ đệm đã được khởi tạo trong **header.html**) vì nó không được sử dụng. Mã lệnh chuyển hướng tương tự như phương pháp được sử dụng trong Chương 7 (xem phần *"Sử dụng URL tuyệt đối với header()"*).

Nếu tên đăng nhập và mật khẩu được gửi lên không phù hợp với mẩu tin nào trong cơ sở dữ liệu, thông báo lỗi sẽ được hiển thị (xem hình 12-17).

6. Đóng kết nối cơ sở dữ liệu và hoàn tất các điều kiện.

```
        mysql_close();
    } else {
        echo '<p><font color="red" size="+1">Hay thu lai.
        → </font></p>';
    }
}
?>
```



Hình 12-17: *Thông báo lỗi được hiển thị nếu tên đăng nhập và mật khẩu được gửi lên không phù hợp với giá trị có trong cơ sở dữ liệu.*

7. Hiển thị biểu mẫu đăng nhập (xem hình 12-18).

Hình 12-18: *Biểu mẫu đăng nhập.*

```
<h1>Dang nhap</h1>
<p>Trinh duyet cua ban phai ho tro cookie de dang
→ nhap.</p>
<form action="<?php echo $_SERVER['PHP_SELF']; ?>"
→ method="post">
```

```
<fieldset>
<p><b>Ten dang nhap:</b> <input type="text"
→ name="username" size="10" maxlength="20" value="<?php
→ if (isset($_POST['username'])) echo $_POST['username'];
→ ?>" /></p>
<p><b>Mat khau:</b> <input type="password"
→ name="password" size="20" maxlength="20" /></p>
<div align="center"><input type="submit" name="submit"
→ value="Dang nhap" /></div>
</fieldset></form>
```

Biểu mẫu đăng nhập (cũng giống như biểu mẫu đăng ký) sẽ gửi dữ liệu cho chính nó. Nếu quá trình đăng nhập có vấn đề, biểu mẫu này sẽ được hiển thị lại và tên đăng nhập trước đó sẽ được đưa vào hộp nhập **username**.

Chú ý, chúng ta đã đưa vào chỉ dẫn để thông báo cho người dùng biết là họ phải kích hoạt cookie để có thể sử dụng (nếu không, họ sẽ không bao giờ đăng nhập vào được).

8. Đưa vào phần cuối trang HTML.

```
<?php
include ('includes/footer.html');
?>
```

9. Lưu tập tin với tên **login.php**, tải nó lên máy chủ và chạy thử trong trình duyệt (xem hình 12-19). Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 12.7.

Hình 12-19: *Nếu quá trình đăng nhập thành công, người dùng sẽ được đưa trở lại trang chủ, tại đó họ sẽ nhận được lời chào mừng.*

Đoạn mã 12.7: *Biểu mẫu đăng nhập sẽ chuyển người dùng đến trang chủ sau khi đăng ký tên và giá trị ID với session.*

```
<?php # Script 12.7 - login.php
require_once ('includes/config.inc');
$page_title = 'Dang nhap';
include ('includes/header.html');
if (isset($_POST['submit'])) {
   require_once ('../mysql_connect.php');
   if (empty($_POST['username'])) {
      $u = FALSE;
      echo '<p><font color="red" size="+1">Ban chua nhap ten
      → dang nhap!</font></p>';
   } else {
      $u = escape_data($_POST['username']);
   }
   if (empty($_POST['password'])) {
      $p = FALSE;
      echo '<p><font color="red" size="+1">Ban chua nhap mat
      → khau!</font></p>';
   } else {
      $p = escape_data($_POST['password']);
   }
   if ($u && $p) {
      $query = "SELECT user_id, first_name FROM users WHERE
      → username='$u' AND password=PASSWORD('$p')";
      $result = @mysql_query ($query);
      $row = mysql_fetch_array ($result, MYSQL_NUM);
      if ($row) {
         $_SESSION['first_name'] = $row[1];
         $_SESSION['user_id'] = $row[0];
         ob_end_clean();
         header ("Location:  http://" . $_SERVER['HTTP_HOST']
         → . dirname($_SERVER['PHP_SELF']) . "/index.php");
         exit();
      } else {
         echo '<p><font color="red" size="+1">Ten dang nhap va
         → mat khau khong phu hop voi du lieu trong CSDL.
```

```
        →</font></p>';
      }
      mysqli_close();
   } else {
      echo '<p><font color="red" size="+1">Hay thu lai.
      → </font></p>';
   }
}
?>
<h1>Dang nhap</h1>
<p>Trinh duyet cua ban phai ho tro cookie de dang nhap.</p>
<form action="<?php echo $_SERVER['PHP_SELF']; ?>"
→ method="post">
<fieldset>
<p><b>Ten dang nhap:</b> <input type="text" name="username"
→ size="10" maxlength="20" value="<?php if
→ (isset($_POST['username'])) echo $_POST['username']; ?>"
→ /></p>
<p><b>Mat khau:</b> <input type="password" name="password"
→ size="20" maxlength="20" /></p>
<div align="center"><input type="submit" name="submit"
→ value="Dang nhap" /></div>
</fieldset></form>
<?php
include ('includes/footer.html');
?>
```

Thực hành viết mã kịch bản **logout.php** theo các bước sau:

1. Tạo một hồ sơ mới trong trình soạn thảo văn bản và đưa vào các tập tin cần thiết.

```
<?php # Doan ma 12.8 - logout.php
require_once ('includes/config.inc');
$page_title = 'Thoat ra';
include_once ('includes/header.html');
```

2. Kiểm tra xem người dùng hiện đã đăng nhập chưa.

```
if (!isset($_SESSION['first_name'])) {
```

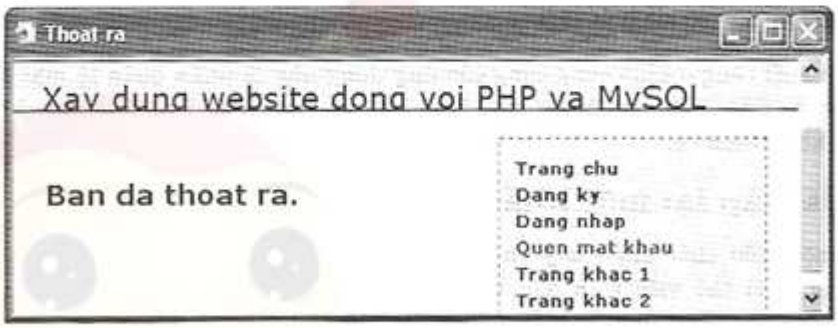
```
    header ("Location:  http://" . $_SERVER['HTTP_HOST'] .
    → dirname($_SERVER['PHP_SELF']) . "/index.php");

    ob_end_clean();

    exit();

} else {

    $_SESSION = array();

    session_destroy();

    setcookie (session_name(), '', time()-300, '/', '', 0);

}
```

Nếu người dùng chưa đăng nhập (bằng cách kiểm tra biến `$_SESSION['first_name']`), họ sẽ được chuyển đến trang chủ. Nếu họ đã đăng nhập, chúng ta xóa giá trị của session và cookie gắn với session.

3. In thông báo thoát ra và hoàn tất trang HTML.

```
echo "<h3>Ban da thoat ra.</h3>";

include ('includes/footer.html');

?>
```

4. Lưu tập tin với tên **logout.php**, tải nó lên máy chủ và chạy thử trong trình duyệt (xem hình 12-20). Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 12.8.



Hình 12-20: *Kết quả thoát ra thành công.*

Đoạn mã 12.8: *Mã kịch bản thoát ra sẽ xóa toàn bộ thông tin session, kể cả cookie.*

```
<?php # Doan ma 12.8 - logout.php
require_once ('includes/config.inc');
$page_title = 'Thoat ra';
```

```
include_once ('includes/header.html');
if (!isset($_SESSION['first_name'])) {
   header ("Location:  http://" . $_SERVER['HTTP_HOST']
   → dirname($_SERVER['PHP_SELF']) . "/index.php");
   ob_end_clean();
   exit();
} else {
   $_SESSION = array();
   session_destroy();
   setcookie (session_name(), '', time()-300, '/', '', 0);
}
echo "<h3>Ban da thoat ra.</h3>";
include ('includes/footer.html');
?>
```

Nếu bổ sung trường **loggedin** *vào bảng users và thiết lập giá trị của nó là 1 khi người dùng đăng nhập, chúng ta sẽ đếm được số lần họ đã truy cập vào Website.*

Nếu bổ sung trường **last_login** *vào bảng users và cập nhật nó khi người dùng thoát ra, chúng ta sẽ biết được thời điểm người dùng truy xuất lần sau cùng.*

Quản lý mật khẩu

Phần cuối cùng ở phía công cộng của ứng dụng này là phần quản lý mật khẩu. Có hai quá trình chúng ta cần quan tâm: thiết lập lại mật khẩu bị quên và thay đổi mật khẩu hiện có.

Thiết lập lại mật khẩu

Có một điều chắc chắn xảy ra là người dùng đôi khi quên mật khẩu truy cập Website, vì thế việc tính đến khả năng này khi xây dựng Website là điều rất quan trọng. Một tùy chọn để người dùng gửi thư điện tử đến người quản trị khi điều này xảy ra. Tuy nhiên, do việc quản trị site rất phức tạp ngay cả khi chưa tính đến điều này, nên ở đây chúng ta sẽ chỉ tạo mã kịch bản với mục đích thiết lập lại mật khẩu.

Vì mật khẩu được chứa trong cơ sở dữ liệu ở dạng mã hóa bằng hàm **PASSWORD()** của MySQL nên không thể lấy lại phiên bản gốc của nó. Do đó, chúng ta sẽ tạo một giá trị mới có tính ngẫu nhiên và thay đổi mật khẩu hiện có với giá trị này. Thay vì hiển thị trong trình duyệt của người dùng (do không an toàn), mật khẩu mới sẽ được gửi đến địa chỉ thư điện tử mà họ đã đăng ký.

Thực hành viết mã kịch bản **forgot_password.php** theo các bước sau:

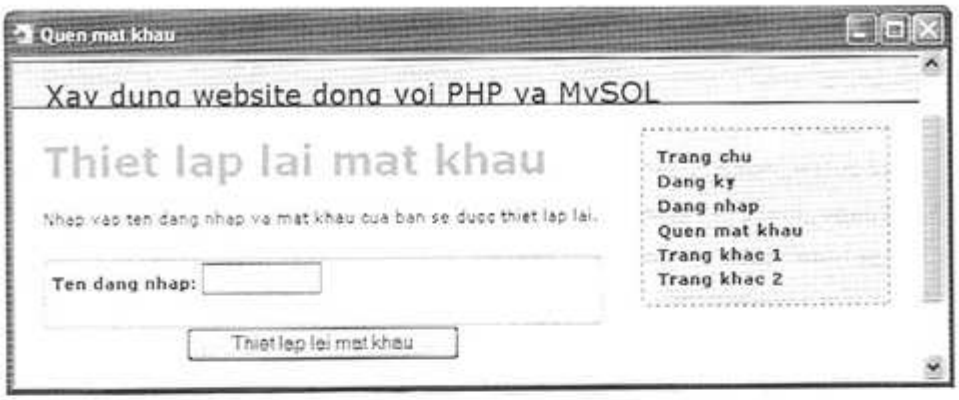
1. Tạo một hồ sơ mới trong trình soạn thảo văn bản và đưa vào các tập tin cần thiết.

```
<?php # Doan ma 12.9 - forgot_password.php
require_once ('includes/config.inc');
$page_title = 'Quen mat khau';
include_once ('includes/header.html');
```

2. Kiểm tra xem biểu mẫu có được gửi lên không và sau đó kiểm tra tên đăng nhập.

```
if (isset($_POST['submit'])) {
   require_once ('../mysql_connect.php');
   if (empty($_POST['username'])) {
      $u = FALSE;
      echo '<p><font color="red" size="+1">Ban chua nhap
      → ten dang nhap!</font></p>';
   } else {
      $u = escape_data($_POST['username']);
      $query = "SELECT user_id, email FROM users WHERE
      → username='$u'";
      $result = @mysql_query ($query);
      $row = mysql_fetch_array ($result, MYSQL_NUM);
      if ($row) {
         $uid = $row[0];
         $email = $row[1];
      } else {
         echo '<p><font color="red" size="+1">Khong tim
         → thay ten dang nhap trong CSDL!</font></p>';
         $u = FALSE;
      }
   }
```

Mã kịch bản này nhận vào tên đăng nhập và cập nhật mẩu tin. Bước đầu tiên là kiểm tra xem tên đăng nhập có được nhập hay không (để tránh phải sử dụng biểu thức quy tắc vốn tiêu tốn nhiều tài nguyên của máy). Tiếp đến, lấy ra ID và địa chỉ thư điện tử của người dùng từ cơ sở dữ liệu, dựa trên tên đăng nhập được cung cấp. Nếu không có mẩu tin nào đáp ứng yêu cầu, một thông báo lỗi được hiển thị (xem hình 12-21).



Hình 12-21: *Nếu người dùng cung cấp tên đăng nhập không có trong cơ sở dữ liệu, một thông báo lỗi sẽ được hiển thị.*

3. Tạo ra mật khẩu mới có tính ngẫu nhiên.

```
if ($u) {

   $p = substr ( md5(uniqid(rand(),1)), 3, 10);
```

Để tạo một mật khẩu mới có tính ngẫu nhiên, chúng ta sẽ sử dụng bốn hàm của PHP. Hàm đầu tiên là **uniquid()**. Hàm này trả lại một định danh duy nhất. Nó nhận vào các tham số là hàm **rand()** và 1 để trả lại một chuỗi có tính ngẫu nhiên hơn. Giá trị trả về này sau đó được gửi cho hàm **md5()** để tính toán giá trị băm (hash) MD5 của một chuỗi. Tại giai đoạn này, một giá trị ID duy nhất sẽ được trả lại - đây là một chuỗi dài 32 ký tự.

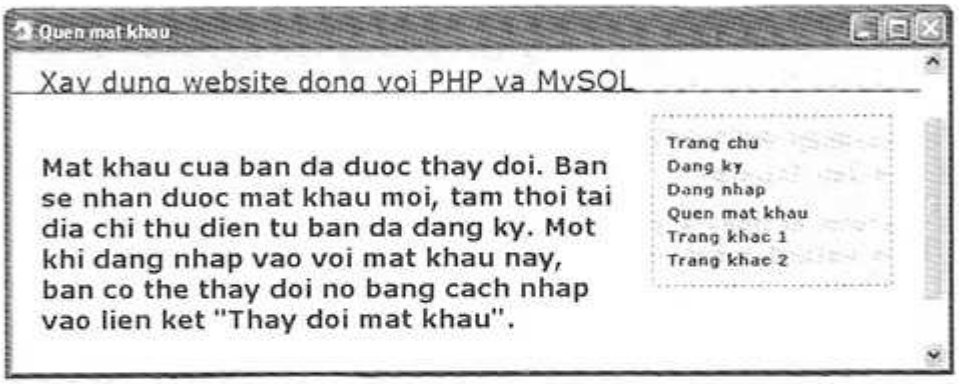
Từ chuỗi này, mật khẩu được xác định bằng cách lấy ra 10 ký tự, bắt đầu từ vị trí thứ ba với việc sử dụng hàm **substr()**. Mã lệnh này sẽ trả lại một giá trị rất ngẫu nhiên với chiều dài 10 ký tự, bao gồm chữ số và chữ cái, có thể được dùng làm mật khẩu tạm thời.

4. Cập nhật mật khẩu trong cơ sở dữ liệu.

```
$query = "UPDATE users SET password=PASSWORD('$p') WHERE
→ user_id=$uid";

$result = @mysql_query ($query);

if (mysql_affected_rows() == 1) {

   $body = "Mat khau de dang nhap vao SITENAME da duoc tam
   → thoi chuyen thanh '$p'. Hay dang nhap bang cach dung
   → mat khau nay va ten dang nhap cua ban. Khi do ban co
   → the thay doi lai mat khau cua minh.";

   mail ($email, 'Mat khau tam thoi.', $body,
   → 'From: admin@sitename.com');
```

```
        echo '<h3>Mat khau cua ban da duoc thay doi. Ban se
        → nhan duoc mat khau moi, tam thoi tai dia chi thu
        → dien tu ban da dang ky. Mot khi dang nhap vao voi
        → mat khau nay, ban co the thay doi no bang cach nhap
        → vao lien ket "Thay doi mat khau".</h3>';

        include ('includes/footer.html');

        exit();
```

Mật khẩu của người dùng ứng với giá trị ID được lấy ra trong bước 2 sẽ được cập nhật với giá trị của hàm `PASSWORD($p)`. Sau đó, một thư điện tử được gửi đến người dùng (sử dụng địa chỉ thư điện tử đã được lấy ra trước đây) và một thông báo cũng được in ra trình duyệt (xem hình 12-22).



Hình 12-22: *Trang kết quả sau khi thiết lập lại mật khẩu thành công.*

5. Hoàn tất phần điều kiện và mã lệnh PHP.

```
        } else {
            $message = '<p><font color="red" size="+1">Khong the
            → thay doi mat khau do loi he thong. Xin loi vi su
            → co nay.</font></p>';
        }
        mysql_close();
    } else {
        echo '<p><font color="red" size="+1">Hay thu lai.
        → </font></p>';
    }
}
?>
```

6. Tạo biểu mẫu HTML (xem hình 12-23).

Hình 12-23: *Biểu mẫu đơn giản để thiết lập lại mật khẩu.*

```
<h1>Thiet lap lai mat khau</h1>

<p>Nhap vao ten dang nhap va mat khau cua ban se duoc thiet
→ lap lai.</p>

<form action="<?php echo $_SERVER['PHP_SELF']; ?>"
→ method="post">

<fieldset>

<p><b>Ten dang nhap:</b> <input type="text"
→ name="username" size="10" maxlength="20" value="<?php
→ if (isset($_POST['username'])) echo $_POST['username'];
→ ?>" /></p>

</fieldset>

<div align="center"><input type="submit" name="submit"
→ value="Thiet lap lai mat khau" /></div>

</form>
```

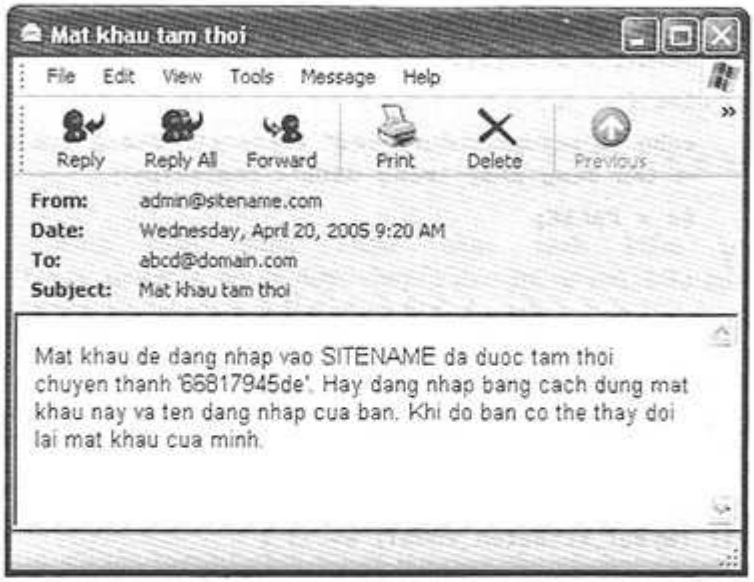
Biểu mẫu chỉ chứa một trường nhập để nhận vào tên đăng nhập. Nếu có một vấn đề nào đó khi gửi biểu mẫu (xem lại hình 12-21), tên đăng nhập sẽ được hiển thị trở lại.

7. Đưa vào phần cuối HTML.

```
<?php

include ('includes/footer.html');

?>
```

8. Lưu tập tin với tên **forgot_password.php**, tải nó lên máy chủ và chạy thử trong trình duyệt. Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 12.9.

9. Kiểm tra hộp thư để xem thông báo kết quả sau khi thiết lập lại mật khẩu thành công (xem hình 12-24).



Hình 12-24: *Thư điện tử thông báo nhận được sau khi thiết lập lại mật khẩu.*

Đoạn mã 12.9: *Mã kịch bản* ***forgot_password.php*** *cho phép người dùng thiết lập lại mật khẩu mà không cần trợ giúp của phần quản trị.*

```
<?php # Doan ma 12.9 - forgot_password.php
require_once ('includes/config.inc');
$page_title = 'Quen mat khau';
include_once ('includes/header.html');
if (isset($_POST['submit'])) {
   require_once ('../mysql_connect.php');
   if (empty($_POST['username'])) {
      $u = FALSE;
      echo '<p><font color="red" size="+1">Ban chua nhap ten
      → dang nhap!</font></p>';
   } else {
      $u = escape_data($_POST['username']);
```

```
    $query = "SELECT user_id, email FROM users WHERE
    → username='$u'";
    $result = @mysql_query ($query);
    $row = mysql_fetch_array ($result, MYSQL_NUM);
    if ($row) {
       $uid = $row[0];
       $email = $row[1];
    } else {
       echo '<p><font color="red" size="+1">Khong tim thay
       → ten dang nhap trong CSDL!</font></p>';
       $u = FALSE;
    }
}
if ($u) {
    $p = substr ( md5(uniqid(rand(),1)), 3, 10);
    $query = "UPDATE users SET password=PASSWORD('$p')
    → WHERE user_id=$uid";
    $result = @mysql_query ($query);
    if (mysql_affected_rows() == 1) {
       $body = "Mat khau de dang nhap vao SITENAME da duoc
       → tam thoi chuyen thanh '$p'. Hay dang nhap bang
       → cach dung mat khau nay va ten dang nhap cua ban.
       → Khi do ban co the thay doi lai mat khau cua minh.";
       mail ($email, 'Mat khau tam thoi.', $body,
       → 'From: admin@sitename.com');
       echo '<h3>Mat khau cua ban da duoc thay doi. Ban se
       → nhan duoc mat khau moi, tam thoi tai dia chi thu
       → dien tu ban da dang ky. Mot khi dang nhap vao voi
       → mat khau nay, ban co the thay doi no bang cach
       → nhap vao lien ket "Thay doi mat khau".</h3>';
       include ('includes/footer.html');
       exit();
    } else {
       $message = '<p><font color="red" size="+1">Khong the
       → thay doi mat khau do loi he thong. Xin loi vi su
       → co nay.</font></p>';
    }
    mysql_close();
```

```
    } else {
        echo '<p><font color="red" size="+1">Hay thu lai.
        → </font></p>';
    }
}
?>
<h1>Thiet lap lai mat khau</h1>
<p>Nhap vao ten dang nhap va mat khau cua ban se duoc thiet
→ lap lai.</p>
<form action="<?php echo $_SERVER['PHP_SELF']; ?>"
→ method="post">
<fieldset>
<p><b>Ten dang nhap:</b> <input type="text" name="username"
→ size="10" maxlength="20" value="<?php if
→ (isset($_POST['username'])) echo $_POST['username']; ?>"
→ /></p>
</fieldset>
<div align="center"><input type="submit" name="submit"
→ value="Thiet lap lai mat khau" /></div>
</form>
<?php
include ('includes/footer.html');
?>
```

Một tùy chọn khác để người dùng thiết lập lại mật khẩu là để họ nhập vào địa chỉ thư điện tử đã đăng ký. Khi sử dụng phương pháp này, chúng ta cần đảm bảo một địa chỉ thư điện tử chỉ được sử dụng bởi một người (rất có thể một người đăng ký nhiều lần với cùng một địa chỉ thư điện tử).

*Nếu quan tâm đến vấn đề vận hành, bạn nên thay đổi giá trị của thuộc tính action trong thẻ form thành giá trị tĩnh (**forgot_password.php**) thay vì để PHP thiết lập mỗi lần chạy mã kịch bản (sử dụng **echo $_SERVER['PHP_SELF']**).*

Thay đổi mật khẩu

Mã kịch bản **change_password.php** đã được viết lần đầu trong Chương 6 dưới dạng một ví dụ cho câu truy vấn UPDATE. Mã kịch bản được phát triển ở đây cũng tương tự như vậy. Chỉ có một ràng buộc là người dùng phải đăng nhập vào để truy xuất chức năng này. Do đó, chúng ta chỉ cần nhận vào mật khẩu mới và giá trị xác nhận của nó (mật khẩu hiện tại và tên đăng nhập của người dùng đã được kiểm tra lúc họ đăng nhập).

Thực hành viết mã kịch bản **change_password.php** theo các bước sau:

1. Tạo một hồ sơ mới trong trình soạn thảo văn bản và đưa vào các tập tin cần thiết.

```
<?php # Doan ma 12.10 - change_password.php
require_once ('includes/config.inc');
$page_title = 'Thay doi mat khau';
include_once ('includes/header.html');
```

2. Kiểm tra xem người dùng đã đăng nhập chưa.

```
if (!isset($_SESSION['first_name'])) {
    header ("Location:  http://" . $_SERVER['HTTP_HOST'] .
    → dirname($_SERVER['PHP_SELF']) . "/index.php");
    ob_end_clean();
    exit();
```

Chúng ta muốn trang này chỉ có thể được truy cập bởi những ai đã đăng nhập. Để làm điều này, mã kịch bản sẽ kiểm tra sự hiện diện của **$_SESSION['first_name']**. Nếu nó chưa được thiết lập, người dùng sẽ được chuyển về trang chủ.

3. Kiểm tra xem biểu mẫu có được gửi lên không và đưa vào kết nối MySQL.

```
} else {
    if (isset($_POST['submit'])) {
        require_once ('../mysql_connect.php');
```

Vì người dùng sẽ được chuyển đến trang chủ nếu chưa đăng nhập, mệnh đề **else** của điều kiện chính này sẽ là tùy chọn, nhưng chúng ta đưa vào để duy trì lôgic của mã kịch bản. Điều cần nhớ là mã kịch bản này có bốn giai đoạn khác nhau: chưa đăng nhập (sẽ được chuyển hướng), đã đăng nhập, đang xem biểu mẫu (đã đăng nhập) và đang xử lý biểu mẫu (đã đăng nhập).

4. Kiểm tra mật khẩu được gửi lên.

```
if (eregi ("^[[:alnum:]]{4,20}$",
→ stripslashes(trim($_POST['password1'])))) {
    if ($_POST['password1'] == $_POST['password2']) {
        $p = escape_data($_POST['password1']);
    } else {
        $p = FALSE;
        echo '<p><font color="red" size="+1">Mat khau khong
        → khop voi phan xac nhan!</font></p>';
    }
} else {
    $p = FALSE;
```

```
        echo '<p><font color="red" size="+1">Nhap vao mat khau
        -> hop le!</font></p>';
    }
```

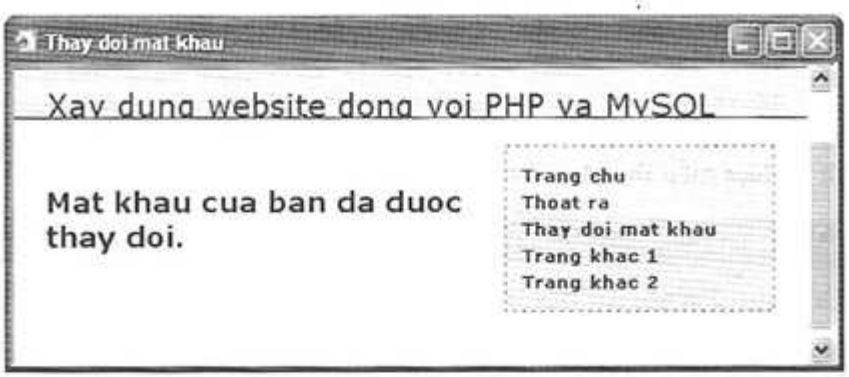
Mật khẩu mới sẽ được kiểm tra tương tự như trong quá trình đăng ký. Các thông báo lỗi sẽ được hiển thị nếu có một vấn đề nào đó xảy ra (xem hình 12-25).

Hình 12-25: *Mật khẩu mới được kiểm tra tương tự như trong quá trình đăng ký.*

5. Cập nhật mật khẩu.

```
if ($p) {
    $query = "UPDATE users SET password=PASSWORD('$p')
    -> WHERE user_id={$_SESSION['user_id']}";
    $result = @mysql_query ($query);
    if (mysql_affected_rows() == 1) {
        echo '<h3>Mat khau cua ban da duoc thay doi.</h3>';
        include ('includes/footer.html');
        exit();
    } else {
        $message = '<p><font color="red" size="+1">Mat khau
        -> cua ban khong thay doi duoc do loi he thong. Xin
        -> loi vi su co.</font></p>';
    }
```

Trường **password** trong bảng **users** được cập nhật bằng cách sử dụng ID của người dùng (được chứa trong session khi họ đăng nhập). Nếu cập nhật thành công, một thông báo sẽ được in ra trong trình duyệt (xem hình 12-26).

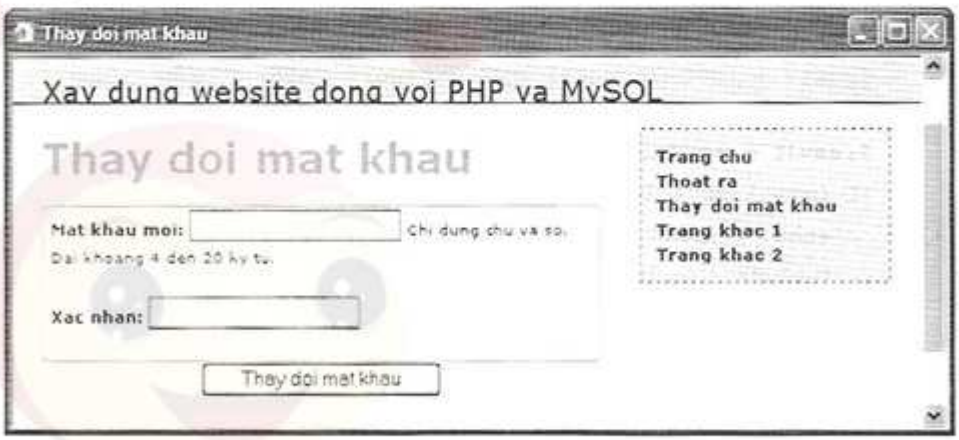


Hình 12-26: *Mã kịch bản đã thay đổi thành công mật khẩu người dùng.*

6. Đóng kết nối cơ sở dữ liệu, hoàn tất phần điều kiện và kết thúc mã lệnh PHP.

```
        mysql_close();
    } else {
        echo '<p><font color="red" size="+1">Hay thu
        → lai.</font></p>';
    }
}
?>
```

7. Tạo biểu mẫu HTML (xem hình 12-27).



Hình 12-27: *Biểu mẫu thay đổi mật khẩu.*

```
<h1>Thay doi mat khau</h1>
<form action="<?php echo $_SERVER['PHP_SELF']; ?>"
→ method="post">
<fieldset>
```

```
<p><b>Mat khau moi:</b> <input type="password"
→ name="password1" size="20" maxlength="20" /> <small>Chi
→ dung chu va so. Dai khoang 4 den 20 ky tu.</small></p>
<p><b>Xac nhan:</b> <input type="password"
→ name="password2" size="20" maxlength="20" /></p>
</fieldset>
<div align="center"><input type="submit" name="submit"
→ value="Thay doi mat khau" /></div>
</form>
```

Biểu mẫu này nhận vào hai giá trị: mật khẩu mới và giá trị xác nhận. Nó cũng có một chỉ dẫn ngắn gọn để người dùng biết định dạng của mật khẩu.

Vì không thể thiết lập trước giá trị cho các trường nhập mật khẩu trong biểu mẫu HTML, nên không cần phải có mã lệnh PHP để thực hiện điều này.

8. Hoàn tất trang HTML.

```
<?php
}
include ('includes/footer.html');
?>
```

9. Lưu tập tin với tên gọi **change_password.php**, tải lên máy chủ và chạy thử trong trình duyệt. Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 12.10.

Đoạn mã 12.10: *Với trang này, người dùng có thể thay đổi mật khẩu (sau khi họ đã đăng nhập).*

```
<?php # Doan ma 12.10 - change_password.php
require_once ('includes/config.inc');
$page_title = 'Thay doi mat khau';
include_once ('includes/header.html');
if (!isset($_SESSION['first_name'])) {
   header ("Location:  http://" . $_SERVER['HTTP_HOST'] .
   → dirname($_SERVER['PHP_SELF']) . "/index.php");
   ob_end_clean();
   exit();
} else {
   if (isset($_POST['submit'])) {
      require_once ('../mysql_connect.php');
      if (eregi ("^[[:alnum:]]{4,20}$",
      → stripslashes(trim($_POST['password1'])))) {
```

```
        if ($_POST['password1'] == $_POST['password2']) {
            $p = escape_data($_POST['password1']);
        } else {
            $p = FALSE;
            echo '<p><font color="red" size="+1">Mat khau
            → khong khop voi phan xac nhan!</font></p>';
        }
    } else {
        $p = FALSE;
        echo '<p><font color="red" size="+1">Nhap vao mat
        → khau hop le!</font></p>';
    }
    if ($p) {
        $query = "UPDATE users SET password=PASSWORD('$p')
        → WHERE user_id={$_SESSION['user_id']}";
        $result = @mysql_query ($query);
        if (mysql_affected_rows() == 1) {
            echo '<h3>Mat khau cua ban da duoc thay doi.</h3>';
            include ('includes/footer.html');
            exit();
        } else {
            $message = '<p><font color="red" size="+1">Mat
            → khau cua ban khong thay doi duoc do loi he
            → thong. Xin loi vi su co.</font></p>';
        }
        mysql_close();
    } else {
        echo '<p><font color="red" size="+1">Hay thu lai.
        → </font></p>';
    }
}
?>
<h1>Thay doi mat khau</h1>
<form action="<?php echo $_SERVER['PHP_SELF']; ?>"
→ method="post">
<fieldset>
<p><b>Mat khau moi:</b> <input type="password"
```

```
    → name="password1" size="20" maxlength="20" /> <small>Chi
    → dung chu va so. Dai khoang 4 den 20 ky tu.</small></p>
    <p><b>Xac nhan:</b> <input type="password"
    → name="password2" size="20" maxlength="20" /></p>
    </fieldset>
    <div align="center"><input type="submit" name="submit"
    → value="Thay doi mat khau" /></div>
    </form>
<?php
}
include ('includes/footer.html');
?>
```

Sau khi mã kịch bản này hoàn tất, người dùng có thể thiết lập lại mật khẩu với mã kịch bản ***forgot_password.php*** *(xem lại phần "Thiết lập lại mật khẩu"), rồi đăng nhập vào bằng mật khẩu tạm. Lúc đăng nhập thành công, họ sẽ thay đổi mật khẩu bằng mã kịch bản này.*

Hàm ***mysql_affected_rows()*** *sẽ trả lại giá trị 0 nếu câu truy vấn* ***UPDATE*** *không làm thay đổi cột nào của một mẩu tin. Vì lý do này, nếu người dùng thử thay đổi mật khẩu của họ nhưng sử dụng giá trị mật khẩu hiện tại làm giá trị cập nhật thì sẽ không có điều gì xảy ra. Rất may, đây không phải là vấn đề cần phải quan tâm.*

Vì việc kiểm tra xác thực của một site không phụ thuộc vào mật khẩu của người dùng trên cơ sở từng trang (nói cách khác, không cần phải kiểm tra mật khẩu cho từng trang sau khi họ đăng nhập thành công), nên việc thay đổi mật khẩu sẽ không buộc người dùng phải đăng nhập lại.

Quản trị site

Phần thứ hai của ứng dụng là phần quản trị. Với mục đích minh họa, chúng ta sẽ tạo ra một trang quản trị rất cơ bản chỉ để hiển thị danh sách người dùng đã đăng ký. Để truy xuất mã kịch bản này, ta cần tạo ra một trang chủ và một tập tin kiểm tra xác thực để bảo vệ. Bạn cũng cần điều chỉnh các khuôn mẫu hiện có để phục vụ cho phần quản trị. Phần lớn các trang quản trị (không tính các khuôn mẫu) sẽ được chứa trong thư mục với tên gọi *4dm1n*. Vì mục đích an toàn, việc đặt tên cho thư mục quản trị một tên gọi như vậy sẽ giúp giảm thiểu cơ hội một ai đó có ý định săn tìm nó.

Khuôn mẫu

Để làm cho site dễ quản trị và chỉnh sửa hơn, các khuôn mẫu sẽ được đặt trong cùng thư mục *includes* với các khuôn mẫu phía công cộng. Do đó, chúng ta cần thay đổi tham chiếu đến tập tin CSS để nó có thể hoạt động. Đồng thời, tập tin

cuối trang cũng cần được thay đổi để hiển thị các liên kết quản trị thay vì các liên kết như trong phần công cộng.

Thực hành sửa đổi tập tin **header.html** theo các bước sau:

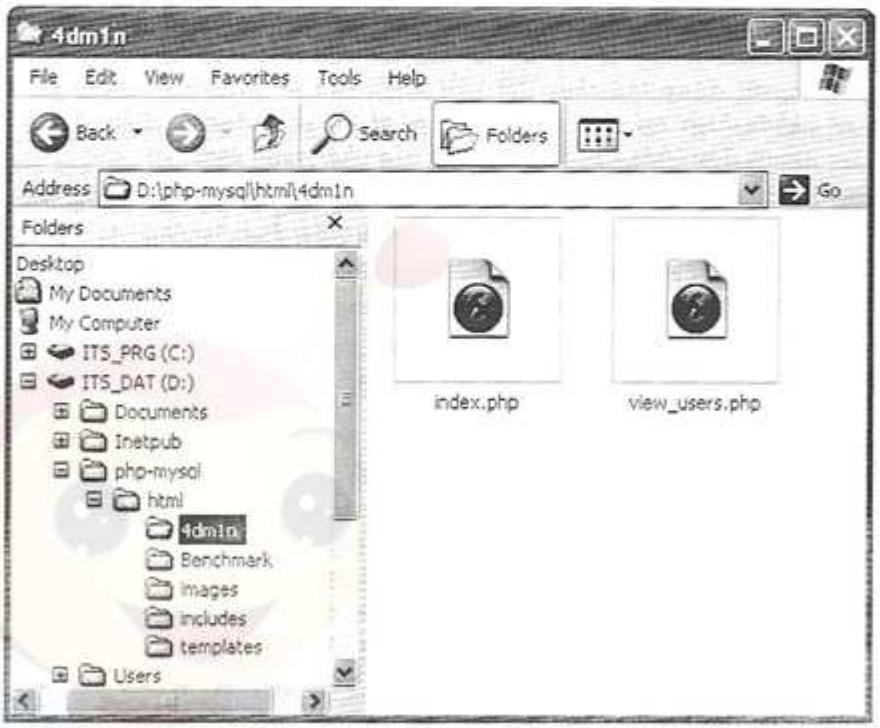
1. Mở tập tin **header.html** (tham khảo lại đoạn mã 12.1) trong trình soạn thảo văn bản.
2. Xóa dòng thiết lập xuất ra bộ đệm và session.

Phía quản trị của ứng dụng sẽ không sử dụng chế độ xuất ra bộ đệm cũng như session, vì thế hai dòng lệnh này sẽ được loại bỏ.

3. Thay đổi tham chiếu đến tập tin **layout.css**.

```
<style type="text/css" media="screen">
@import "../includes/layout.css";
</style>
```

Tập tin **header.html** sẽ được đưa vào bởi các tập tin trong thư mục quản trị. Nghĩa là, để truy cập tập tin **layout.css** (được chứa trong thư mục *includes*) mã kịch bản cần tham chiếu đến nó với đường dẫn **../includes/layout.css**. Xem hình 12-28 và xem lại hình 12-2 để biết thêm cấu trúc thư mục của Website.



Hình 12-28: *Thư mục quản trị.*

4. Lưu tập tin với tên gọi **admin_header.html** và tải nó lên máy chủ (vào trong thư mục *includes*). Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 12.11.

Đoạn mã 12.11: *Tập tin khuôn mẫu đầu trang có một đường dẫn khác đến tập tin* ***layout.css*** *và không sử dụng chế độ xuất ra bộ đệm, cũng như session.*

```
<?php # Doan ma 12.11 - admin_header.html
if (!isset($page_title)) {
   $page_title = 'Xay dung Website dong voi PHP va MySQL';
}
?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
→ "http://www.w3.org/TR/2000/REC-xhtml1-20000126/DTD/xhtml1-
→transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">
<head>
<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />
<title><?php echo $page_title; ?></title>
<style type="text/css" media="screen">
@import "../includes/layout.css";
</style>
</head>
<body>
<div id="Header">Xay dung Website dong voi PHP va MySQL</div>
<div id="Content">
```

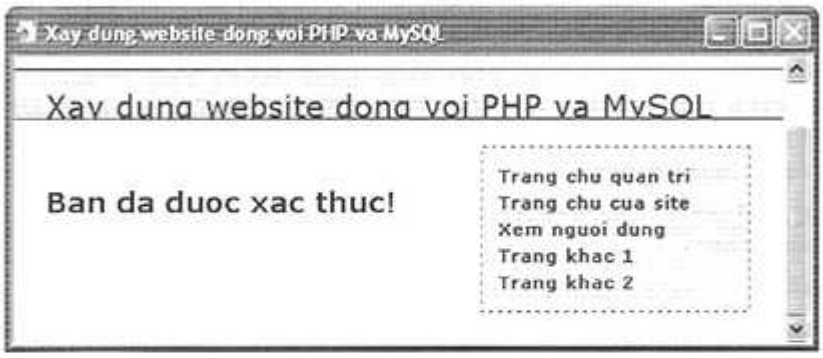
Thực hành tạo tập tin cuối trang theo các bước sau:

1. Tạo một hồ sơ HTML mới trong trình soạn thảo văn bản.

```
</div>
<div id="Menu">
<a href="index.php">Trang chu quan tri</a><br />
<a href="../index.php">Trang chu cua site</a><br />
<a href="view_users.php">Xem nguoi dung</a><br />
<a href="#">Trang khac 1</a><br />
<a href="#">Trang khac 2</a><br />
```

```
</div>
<!-- # Doan ma 12.12 - admin_footer.html -->
</body>
</html>
```

Tập tin cuối trang chứa các liên kết đến trang chủ quản trị, trang xem người dùng và trang chính của phần công cộng. Chúng ta cũng tạo ra một số liên kết để giành cho các trang khác với mục đích minh họa (xem hình 12-29).



Hình 12-29: *Khuôn mẫu quản trị sử dụng cùng tập tin CSS nhưng với các liên kết khác.*

2. Lưu tập tin với tên gọi **admin_footer.html** và tải nó lên máy chủ (trong thư mục *includes*). Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 12.12.

Đoạn mã 12.12: *Phiên bản quản trị của tập tin cuối trang chứa các liên kết khác và không sử dụng chế độ xuất ra bộ đệm.*

```
</div>
<div id="Menu">
<a href="index.php">Trang chu quan tri</a><br />
<a href="../index.php">Trang chu cua site</a><br />
<a href="view_users.php">Xem nguoi dung</a><br />
<a href="#">Trang khac 1</a><br />
<a href="#">Trang khac 2</a><br />
</div>
<!-- # Doan ma 12.12 - admin_footer.html -->
</body>
</html>
```

Thực hiện kiểm tra xác thực

Để bảo vệ mã kịch bản phía quản trị của ứng dụng Web, chúng ta sẽ sử dụng kiểm tra xác thực HTTP. Mã kịch bản được sử dụng ở đây là phiên bản đã được thực hiện trong Chương 8 "An toàn". Trong trường hợp này, tên đăng nhập và mật khẩu sẽ được mã hóa cố định trong mã kịch bản thay vì được lấy từ cơ sở dữ liệu.

Thực hành viết mã kịch bản tập tin **authentication.php** theo các bước sau:

1. Tạo một hồ sơ PHP mới trong trình soạn thảo văn bản.

```
<?php # Doan ma 12.13 - authentication.php
```

Tập tin này sẽ được đưa vào tất cả các trang trong phần quản trị. Do đó, nó không cần sử dụng tập tin đầu và cuối trang HTML.

2. Khởi động biến kiểm tra.

```
$authorized = FALSE;
```

Biến này được sử dụng để cho biết người dùng đã được kiểm tra hay chưa. Mặc định là chưa.

3. Kiểm tra các giá trị được gửi lên so với các giá trị đã được thiết lập.

```
if ( (isset($_SERVER['PHP_AUTH_USER']) AND
→ isset($_SERVER['PHP_AUTH_PW'])) ) {

   if ( ($_SERVER['PHP_AUTH_USER'] == 'name') AND
   → ($_SERVER['PHP_AUTH_PW'] == 'password') ) {

      $authorized = TRUE;

   }

}
```

Không giống như ví dụ trong Chương 8 (trong đó các giá trị được gửi lên sẽ được so sánh với các giá trị trong cơ sở dữ liệu), mã kịch bản này sẽ so sánh chúng với các giá trị cố định được ghi sẵn trong nó. Bạn có thể thay đổi tên đăng nhập và mật khẩu cho phù hợp với mình.

4. Tạo cửa sổ kiểm tra xác thực HTTP.

```
if (!$authorized) {
   header('WWW-Authenticate: Basic realm="Administration"');
   header('HTTP/1.0 401 Unauthorized');
}
```

Nếu người dùng chưa được kiểm tra (do họ chưa thấy cửa sổ kiểm tra xác thực hoặc nhập sai giá trị), mã lệnh trên sẽ tạo cửa sổ kiểm tra xác thực HTTP (xem hình 12-30).

Hình 12-30: *Cửa sổ kiểm tra xác thực HTTP.*

5. Hoàn tất trang PHP.

```
?>
```

6. Lưu tập tin với tên gọi **authentication.php**, tải lên máy chủ và đặt phía ngoài thư mục gốc của Web. Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 12.13.

Mã kịch bản kiểm tra xác thực cần phải được đặt vào nơi an toàn, trong cùng thư mục với mã kịch bản **mysql_connect.php**. Xem lại hình 12-28 để biết thêm cấu trúc thư mục của ứng dụng này.

Đoạn mã 12.13: *Mã kịch bản kiểm tra xác thực sẽ được sử dụng trong tất cả các trang quản trị để kiểm soát truy cập.*

```
<?php # Doan ma 12.13 - authentication.php
$authorized = FALSE;
if ( (isset($_SERVER['PHP_AUTH_USER']) AND
→ isset($_SERVER['PHP_AUTH_PW'])) ) {
   if ( ($_SERVER['PHP_AUTH_USER'] == 'name') AND
   → ($_SERVER['PHP_AUTH_PW'] == 'password') ) {
      $authorized = TRUE;
   }
}
if (!$authorized) {
   header('WWW-Authenticate: Basic realm="Administration"');
   header('HTTP/1.0 401 Unauthorized');
}
?>
```

Trang chủ phía quản trị

Với ví dụ này, trang chủ quản trị không làm gì ngoài việc sử dụng một khuôn mẫu thích hợp và tập tin kiểm tra xác thực. Tất cả các tập tin trong phần quản trị đều sử dụng các tập tin này.

Thực hành viết tập tin **index.php** theo các bước sau:

1. Tạo một hồ sơ mới trong trình soạn thảo văn bản và đưa vào các tập tin cần thiết.

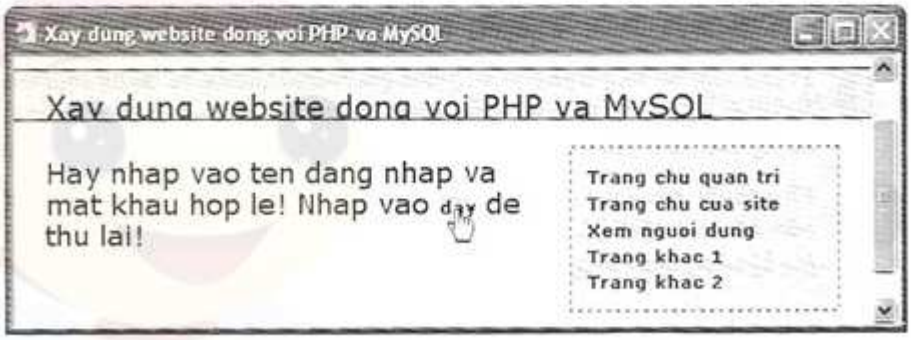
```
<?php # Doan ma 12.14 - index.php (trang chu quan tri)
require_once ('../includes/config.inc');
require_once ('../../authentication.php');
$page_title = 'Xay dung Website dong voi PHP va MySQL';
include_once ('../includes/admin_header.html');
```

Bước đầu tiên là đưa vào tập tin cấu hình để sử dụng chung phần quản lý lỗi với phần công cộng của site. Tiếp đến, chúng ta đưa vào mã kịch bản kiểm tra xác thực (được chứa trên thư mục hiện tại hai cấp) để bảo vệ tập tin này. Cuối cùng, khuôn mẫu HTML được đưa vào.

2. Kiểm tra xem người dùng có được phép truy cập không.

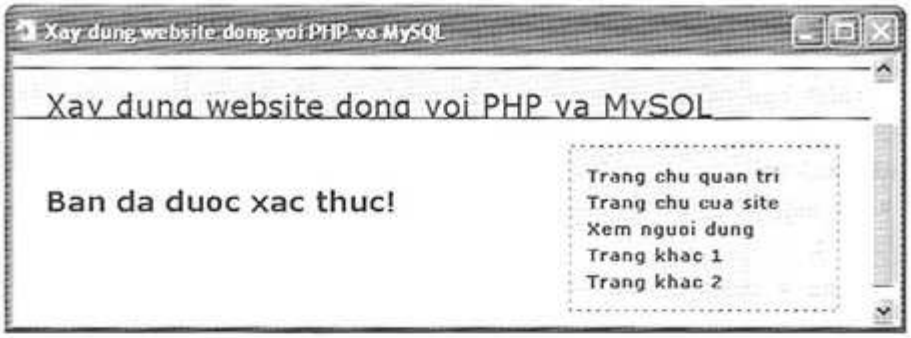
```
if (!$authorized) {
   echo '<p><font color="red" size="+1">Hay nhap vao ten
   → dang nhap va mat khau hop le! Nhap vao <a
   → href="index.php">day</a> de thu lai!</font></p>';
} else {
   echo '<h3>Ban da duoc xac thuc!</h3>';
}
```

Phần này của trang cũng giống như trong mã kịch bản gốc trong Chương 8. Thông báo lỗi sẽ được thể hiện nếu người dùng nhấp vào nút Cancel hoặc nhập sai thông tin, tùy theo trình duyệt của họ (xem hình 12-31).



Hình 12-31: *Truy xuất đến mã kịch bản bị từ chối.*

3. Lưu tập tin với tên **index.php**, tải nó lên máy chủ (vào trong thư mục quản trị) và chạy thử trong trình duyệt (xem hình 12-32). Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 12.14.



Hình 12-32: *Trang thể hiện sau khi kiểm tra xác thực thành công.*

Đoạn mã 12.14: *Mã kịch bản cho trang chủ phía quản trị của ứng dụng Web.*

```
<?php # Doan ma 12.14 - index.php (admin home page)
require_once ('../includes/config.inc');
require_once ('../../authentication.php');
$page_title = 'Xay dung Website dong voi PHP va MySQL';
include_once ('../includes/admin_header.html');
if (!$authorized) {
   echo '<p><font color="red" size="+1">Hay nhap vao ten dang
   → nhap va mat khau hop le! Nhap vao <a href="index.php">
   → day</a> de thu lai!</font></p>';
} else {
   echo '<h3>Ban da duoc xac thuc!</h3>';
}
include_once ('../includes/admin_footer.html');
?>
```

Trong một số ứng dụng Web, phía quản trị thường sử dụng một mã kịch bản cấu hình khác với phần công cộng vì nó chứa các chức năng và thông tin liên quan đến phần quản trị của site.

Phụ thuộc vào mục đích của site, một số ứng dụng Web sử dụng mã kịch bản kết nối MySQL riêng. Mã kịch bản này sẽ kết nối cơ sở dữ liệu với tài khoản, mật khẩu và quyền truy cập khác với phần công cộng.

Xem người dùng

Mã kịch bản cuối cùng trong ví dụ này cho phép người quản trị xem danh sách người dùng đã đăng ký. Tiền thân của mã kịch bản này đã được viết trong Chương 6. Tuy nhiên, phiên bản này sẽ kiểm tra xác thực và sử dụng một khuôn mẫu khác. Quan trọng hơn, nó sử dụng hai kỹ thuật mới và rất hữu dụng.

Đầu tiên, mã kịch bản sẽ hiển thị một số mẩu tin trên mỗi trang và tạo ra các liên kết để xem các trang khác. Đây được gọi là phân trang và là một phần của một động cơ tìm kiếm bất kỳ trên Internet. Ưu điểm của việc phân trang là mỗi lần chỉ cần hiển thị khoảng 10 hoặc 25 mẩu tin thay vì phải hiển thị tất cả. Kỹ thuật thứ hai mang tính trang trí nhiều hơn: hiển thị mỗi hàng trong bảng với một màu nền được thay đổi luân phiên (xem hình 12-33). Hiệu ứng này được thực hiện dễ dàng với sự trợ giúp của toán tử ba ngôi (xem phần *"Toán tử ba ngôi"* để biết thêm chi tiết).

Hình 12-33: *Việc sử dụng màu nền thay đổi luân phiên cho mỗi dòng sẽ giúp bảng dễ dùng hơn.*

Toán tử ba ngôi

Trong ví dụ này, chúng ta sử dụng một toán tử chưa được giới thiệu trước đây: toán tử ba ngôi. Cấu trúc của nó như sau:

```
(điều kiện) ? giá trị T : giá trị F;
```

Điều kiện trong dấu ngoặc sẽ được định giá trị và nếu nó có giá trị **TRUE**, *giá trị thứ nhất sẽ được trả lại. Nếu điều kiện có giá trị* **FALSE**, *giá trị thứ hai sẽ được trả lại.*

Thông thường, biểu thức sử dụng toán tử này được dùng cùng với hàm để giá trị trả về của nó sẽ được cung cấp cho hàm. Ví dụ:

```
echo (isset($var)) ? 'SET' : 'NOT SET';
```

Dòng lệnh trên sẽ in ra SET hoặc NOT SET tùy theo trạng thái của biến ***$var****.*

Trong ví dụ sau, toán tử ba ngôi này sẽ được sử dụng để xác định màu nền của mỗi dòng trong bảng. Cũng còn nhiều cách khác để thiết lập giá trị này, tuy nhiên toán tử ba ngôi là thích hợp nhất.

Thực hành tạo mã kịch bản **view_users.php** theo các bước sau:

1. Tạo một hồ sơ PHP mới trong trình soạn thảo văn bản, đưa vào các tập tin cần thiết.

```
<?php # Doan ma 12.15 - view_users.php
require_once ('../includes/config.inc');
require_once ('../../authentication.php');
$page_title = 'Xem nguoi dung hien tai';
include_once ('../includes/admin_header.html');
```

2. Kiểm tra xác thực.

```
if (!$authorized) {
   echo '<p><font color="red" size="+1">Hay nhap vao ten
   → dang nhap va mat khau hop le! Nhap vao <a
   → href="index.php">day</a> de thu lai!</font></p>';
} else {
```

3. Đưa vào mã kịch bản kết nối MySQL và kiểm tra số trang yêu cầu.

```
require_once ('../../mysql_connect.php');
$display = 10;
if (isset($_GET['np'])) {
   $num_pages = $_GET['np'];
} else {
```

Để mã kịch bản này hiển thị người dùng trên một số trang, nó cần xác định tổng số trang kết quả cần hiển thị. Lần đầu mã kịch bản này được gọi, con số này sẽ được tính. Từ đó trở đi, con số này sẽ được truyền đến mã kịch bản dưới dạng một tham số của URL (biến *np*). Nếu biến này được thiết lập, giá trị của nó sẽ được gán cho biến **$num_pages**. Nếu không, chúng ta cần tính toán số trang.

4. Định ra số trang cần hiển thị.

```
   $query = "SELECT CONCAT(last_name, ', ', first_name) AS
   → name, DATE_FORMAT(registration_date, '%M %d, %Y') AS
   → dr FROM users ORDER BY registration_date ASC";
   $query_result = mysql_query ($query) or
   → die (mysql_error());
   $num_records = @mysql_num_rows ($query_result);
   if ($num_records > $display) {
      $num_pages = ceil ($num_records/$display);
   } else {
      $num_pages = 1;
   }
}
```

Số trang cần để hiển thị được xác định dựa trên tổng số mẩu tin sẽ được thể hiện và số dòng hiển thị trên mỗi trang (được thiết lập bởi biến **$display**). Để thực hiện tính toán này, chúng ta thực hiện câu truy vấn cơ sở dữ liệu và gọi hàm **mysql_num_rows()**. Nếu có nhiều mẩu tin cần hiển thị hơn số dòng trên một trang, số trang cần thiết sẽ được tính. Nếu không, ta chỉ cần một trang.

5. Xác định điểm bắt đầu trong cơ sở dữ liệu.

```
if (isset($_GET['s'])) {
   $start = $_GET['s'];
} else {
   $start = 0;
}
```

Tham số thứ hai mà mã kịch bản sẽ nhận (trong những lần xem tiếp theo) là mẩu tin bắt đầu. Khi mã kịch bản được gọi lần đầu, 10 mẩu tin đầu tiên sẽ được hiển thị (mẩu tin đầu tiên có chỉ số là 0). Trang thứ hai sẽ thể hiện các mẩu tin từ thứ 10 đến 20; trang thứ ba: từ 20 đến 30... Trang đầu tiên, biến *s* chưa được thiết lập, do đó chúng ta sẽ gán cho biến **$start** giá trị 0. Các trang tiếp theo sẽ nhận được biến *s* từ URL và giá trị đó sẽ được gán cho biến **$start**.

6. Truy vấn cơ sở dữ liệu.

```
$query = "SELECT CONCAT(last_name, ', ', first_name) AS
→ name, DATE_FORMAT(registration_date, '%M %d, %Y') AS dr
→ FROM users ORDER BY registration_date ASC LIMIT $start,
→ $display";
$result = @mysql_query ($query);
$num = mysql_num_rows ($result);
```

Câu truy vấn này được lấy từ mã kịch bản trong Chương 6 và bổ sung thêm mệnh đề **LIMIT**. Mệnh đề này cho biết bắt đầu lấy từ mẩu tin nào (**$start**) và lấy ra bao nhiêu mẩu tin (**$display**). Lần đầu trang này được thực hiện, câu truy vấn sẽ là **SELECT CONCAT... LIMIT 0, 10**. Nhấp vào trang kế sẽ dẫn đến câu truy vấn **SELECT CONCAT... LIMIT 10, 10**.

7. Nếu có mẩu tin cần thể hiện, chúng ta tạo các liên kết đến các trang khác.

```
if ($num > 0) {
   echo "<h3>Nguoi dung da dang ky</h3>";
   if ($num_pages > 1) {
      echo '<p>';
      $current_page = ($start/$display) + 1;
      if ($current_page != 1) {
         echo '<a href="view_users.php?s=' . ($start -
         → $display) . '&np=' . $num_pages . '">
         → Trang truoc</a> ';
      }
```

Hình 12-34: *Liên kết "Trang trước" chỉ xuất hiện nếu trang hiện tại không phải là trang đầu tiên.*

Nếu có nhiều trang, các liên kết thích hợp sẽ được tạo ở đầu trang (xem hình 12-33). Để tạo các liên kết này, trước hết chúng ta sẽ xác định trang hiện tại (được tính dựa trên con số bắt đầu chia cho số dòng trên mỗi trang và cộng thêm một). Ví dụ, với trang thứ hai, `$start` sẽ có giá trị 10, nên 10/10 + 1 = 2. Nếu trang hiện tại không phải là trang đầu tiên, chúng ta sẽ hiển thị liên kết để quay về trang trước (xem hình 12-34).

Mỗi liên kết sẽ gồm tên mã kịch bản, cộng với điểm bắt đầu và số trang. Điểm bắt đầu đối với trang trước sẽ bằng điểm bắt đầu của trang hiện tại trừ đi số dòng trên mỗi trang.

8. Hoàn tất việc tạo các liên kết.

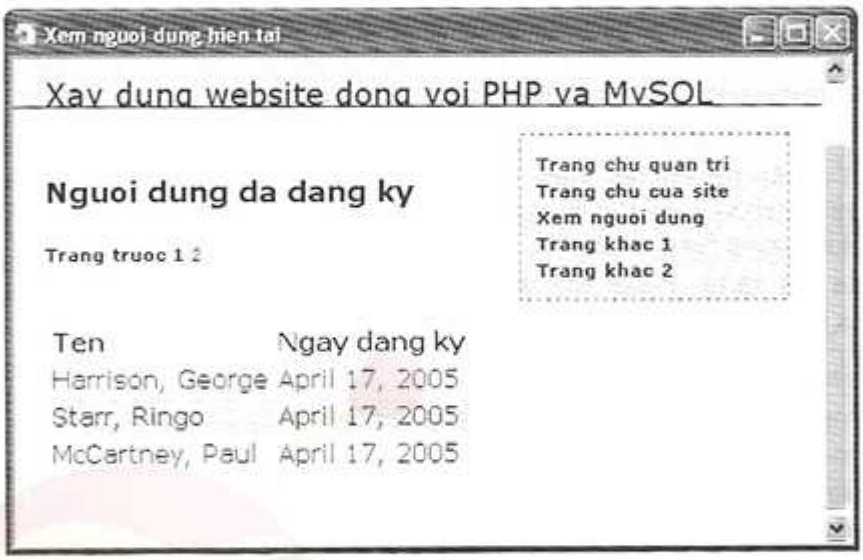
```
for ($i = 1; $i <= $num_pages; $i++) {
   if ($i != $current_page) {
      echo '<a href="view_users.php?s=' . (($display *
      → ($i - 1))) . '&np=' . $num_pages . '">' . $i .
      → '</a> ';
   } else {
      echo $i . ' ';
   }
}
if ($current_page != $num_pages) {
```

```
    echo '<a href="view_users.php?s=' . ($start + $display)
    → . '&np=' . $num_pages . '">Trang ke</a> ';
}
echo '</p><br />';
```

Các liên kết được tạo bằng cách duyệt từ 1 cho đến tổng số trang. Mỗi trang sẽ là một liên kết, ngoại trừ trang hiện tại. Cuối cùng, liên kết đến trang kế tiếp sẽ được hiển thị nếu như đây chưa phải là trang cuối cùng (xem hình 12-35).

9. Tạo phần đầu bảng và khởi tạo biến chứa màu nền.

```
echo '<table align="left" cellspacing="2"
→ cellpadding="2"><tr><td align="left"><b>Ten</b></td>
→ <td align="left"><b>Ngay dang ky</b></td></tr>';

$bg = '#eeeeee';
```



Hình 12-35: *Trang cuối cùng sẽ không hiển thị liên kết "Trang kế".*

Đầu tiên, chúng ta tạo phần tiêu đề bảng và sau đó khởi tạo biến **$bg** (với giá trị **$eeeeee**). Biến này được dùng để làm màu nền cho các hàng trong bảng.

10. In ra các mẩu tin.

```
while ($row = mysql_fetch_array($result, MYSQL_NUM)) {
    $bg = ($bg=='#eeeeee' ? '#ffffff' : '#eeeeee');
    echo '<tr bgcolor="', $bg, '"><td align="left">',
    → stripslashes($row[0]), '</td><td align="left">',
    → $row[1], '</td></tr>';
}
```

Vòng lặp sẽ duyệt qua từng mẩu tin được trả về (tổng số lên đến 10), in ra tên của người dùng (`$row[0]` và ngày đăng ký `$row[1]`). Màu nền gán cho biến `$bg` sẽ được xác định dựa trên giá trị hiện tại. Nếu nó bằng với `#eeeeee`, nó sẽ được thiết lập là `#ffffff` và ngược lại. Đối với dòng đầu tiên, `$bg` sẽ là `#eeeeee` và do đó màu nền sẽ được gán là `#ffffff` (màu trắng). Đối với dòng thứ hai, `$bg` không bằng với `#eeeeee`, vì thế nó sẽ được gán giá trị này và hàng sẽ có màu xám.

11. Hoàn tất bảng HTML và điều kiện.

```
        echo '</table>';
        mysql_free_result ($result);
    } else {
        echo '<p>Hien chua co ai dang ky.</p>';
    }
    mysql_close();
}
```

12. Hoàn tất trang PHP.

```
include_once ('../includes/admin_footer.html');
?>
```

13. Lưu tập tin với tên **view_users.php**, tải nó lên máy chủ (trong thư mục quản trị) và chạy thử trong trình duyệt. Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 12.15.

Đoạn mã 12.15: *Trang này thể hiện danh sách người dùng đã đăng ký, mỗi trang với một số lượng nhất định.*

```
<?php # Doan ma 12.15 - view_users.php
require_once ('../includes/config.inc');
require_once ('../../authentication.php');
$page_title = 'Xem nguoi dung hien tai';
include_once ('../includes/admin_header.html');
if (!$authorized) {
    echo '<p><font color="red" size="+1">Hay nhap vao ten dang
    → nhap va mat khau hop le! Nhap vao <a href="index.php">
    → day</a> de thu lai!</font></p>';
} else {
    require_once ('../../mysql_connect.php');
    $display = 10;
    if (isset($_GET['np'])) {
        $num_pages = $_GET['np'];
    } else {
        $query = "SELECT CONCAT(last_name, ', ', first_name) AS
        → name, DATE_FORMAT(registration_date, '%M %d, %Y') AS
```

```
    → dr FROM users ORDER BY registration_date ASC";
    $query_result = mysql_query ($query) or
    → die (mysql_error());
    $num_records = @mysql_num_rows ($query_result);
    if ($num_records > $display) {
       $num_pages = ceil ($num_records/$display);
    } else {
       $num_pages = 1;
    }
}
if (isset($_GET['s'])) {
   $start = $_GET['s'];
} else {
   $start = 0;
}
$query = "SELECT CONCAT(last_name, ', ', first_name) AS
→ name, DATE_FORMAT(registration_date, '%M %d, %Y') AS dr
→ FROM users ORDER BY registration_date ASC LIMIT $start,
→ $display";
$result = @mysql_query ($query);
$num = mysql_num_rows ($result);
if ($num > 0) {
   echo "<h3>Nguoi dung da dang ky</h3>";
   if ($num_pages > 1) {
      echo '<p>';
      $current_page = ($start/$display) + 1;
      if ($current_page != 1) {
         echo '<a href="view_users.php?s=' . ($start -
         → $display) . '&np=' . $num_pages .
         → '">Trang truoc</a> ';
      }
      for ($i = 1; $i <= $num_pages; $i++) {
         if ($i != $current_page) {
            echo '<a href="view_users.php?s=' .
            → (($display * ($i - 1))) . '&np=' . $num_pages
            → . '">' . $i . '</a> ';
         } else {
            echo $i . ' ';
         }
```

```
        }
        if ($current_page != $num_pages) {
           echo '<a href="view_users.php?s=' . ($start +
           → $display) . '&np=' . $num_pages . '">Trang
           → ke</a> ';
        }
        echo '</p><br/>';
     }
     echo '<table align="left" cellspacing="2"
     → cellpadding="2"><tr><td align="left"><b>Ten</b></td>
     → <td align="left"><b>Ngay dang ky</b></td></tr>';
     $bg = '#eeeeee';
     while ($row = mysql_fetch_array($result, MYSQL_NUM)) {
        $bg = ($bg=='#eeeeee' ? '#ffffff' : '#eeeeee');
        echo '<tr bgcolor="', $bg, '"><td align="left">',
        → stripslashes($row[0]), '</td><td align="left">',
        → $row[1], '</td></tr>';
     }
     echo '</table>';
     mysql_free_result ($result);
  } else {
     echo '<p>Hien chua co ai dang ky.</p>';
  }
  mysql_close();
}
include_once ('../includes/admin_footer.html');
?>
```

Bạn có thể bổ sung các liên kết cho trang này (ngày đăng ký, tên đăng nhập...) để khi được nhấp sẽ làm thay đổi thứ tự thể hiện của các mẫu tin.

Nếu tên của mã kịch bản và phần chính của câu truy vấn đã được thiết lập dưới dạng các biến ở đầu trang, kỹ thuật phân trang này có thể dễ dàng được áp dụng cho các mã kịch bản khác chỉ với một chút sửa đổi.

Tùy vào sự hiểu biết của nhà quản trị site, nhiều mức quản trị liên quan đến MySQL có thể được thực hiện với phpMyAdmin.

Chương 13:

VÍ DỤ – THƯƠNG MẠI ĐIỆN TỬ

Thương mại điện tử ngày càng đóng vai trò quan trọng trên Internet và đây là nơi mà PHP và MySQL thể hiện vai trò của mình.

Trong chương này, chúng ta sẽ phát triển một Website về chủ đề: Website bán các ấn phẩm nghệ thuật. Tuy nhiên, để viết và giải thích toàn bộ ứng dụng như vậy đòi hỏi phải có một quyển sách riêng và mã lệnh cụ thể của nó cũng không thể áp dụng được cho mọi ứng dụng thương mại điện tử. Do đó, trọng tâm của chương này là nói về chức năng cơ bản của ứng dụng, hướng dẫn những nguyên tắc cốt yếu liên quan tới bất cứ site thương mại điện tử nào. Những đặc điểm khác như đăng ký thành viên và quản trị nâng cao sẽ được lược bỏ, nhưng chúng tôi sẽ cung cấp các gợi ý để tạo ra chúng. Ví dụ này cũng đề cập nhiều đến an toàn, vì đây là yếu tố chủ chốt trong mọi ứng dụng thương mại.

Tạo cơ sở dữ liệu

Site thương mại điện tử trong ví dụ này sẽ dùng cơ sở dữ liệu có tên là *ecommerce*, được tạo ra trong Chương 8 "An toàn". Chúng ta sẽ xem xét toàn bộ cấu trúc cơ sở dữ liệu và tạo các bảng cần thiết.

Cơ sở dữ liệu *ecommerce* gồm bốn bảng: hai cho quản lý sản phẩm đang được bán, một cho khách hàng và một cho các đơn đặt hàng. Bảng *artists* (xem bảng 13.1) chứa thông tin về các họa sỹ có tác phẩm được bán. Hiện tại, nó chỉ chứa thông tin tối thiểu, nhưng bạn có thể dễ dàng bổ sung ngày sinh, ngày mất, các dữ liệu về tiểu sử...

Bảng 13.1: *Bảng* artists *dùng để liên kết tên họa sỹ với từng tác phẩm.*

Cột	Loại
artist_id	INT(3)
first_name	VARCHAR(20)
middle_name	VARCHAR(20)
last_name	VARCHAR(20)

Bảng *prints* (xem bảng 13.2) là bảng sản phẩm chính của site. Nó chứa tên tác phẩm, giá cả và các chi tiết liên quan khác (như liên kết đến ID của họa sỹ từ bảng *artists*).

Bảng 13.2: *Bảng* prints *tương đương bảng sản phẩm trong các ứng dụng thương mại điện tử khác.*

Cột	Loại
print_id	INT(4)
artist_id	INT(3)
print_id	VARCHAR(60)
price	DECIMAL(6,2)
size	VARCHAR(60)
description	VARCHAR(255)
image_name	VARCHAR(30)

Bảng *customers* (xem bảng 13.3), cũng được tạo ra trong Chương 8, sẽ giữ lại cấu trúc ban đầu của nó trong ví dụ này. Hiện tại, nó chỉ chứa ID khách hàng, tên truy cập, mật khẩu và số thẻ tín dụng. Tuy nhiên, bạn có thể bổ sung tên khách hàng, địa chỉ...

Bảng 13.3: *Bảng* customers *được sử dụng ở dạng hiện có, mặc dù nó có thể chứa nhiều trường hơn.*

Cột	Loại
customer_id	INT(5)
username	VARCHAR(16)
password	VARCHAR(16)
card_number	TINYBLOB

Bảng cuối cùng của cơ sở dữ liệu (xem bảng 13.4) dùng để lưu trữ thông tin đặt hàng. Nó bao gồm các cột: *order_id, customer_id* (liên kết với bảng *customers*), *total* (tổng giá trị đặt hàng), *order_date, approved* (*Y* hoặc *N*) và *cart* (lưu nội dung lệnh đặt hàng).

Bảng 13.4: *Bảng* orders *lưu trữ lịch sử mua sắm.*

Cột	Loại
order_id	INT(10)
customer_id	INT(5)
total	DECIMAL(10, 2)
order_date	DATETIME
approved	CHAR(1)
cart	VARCHAR(75)

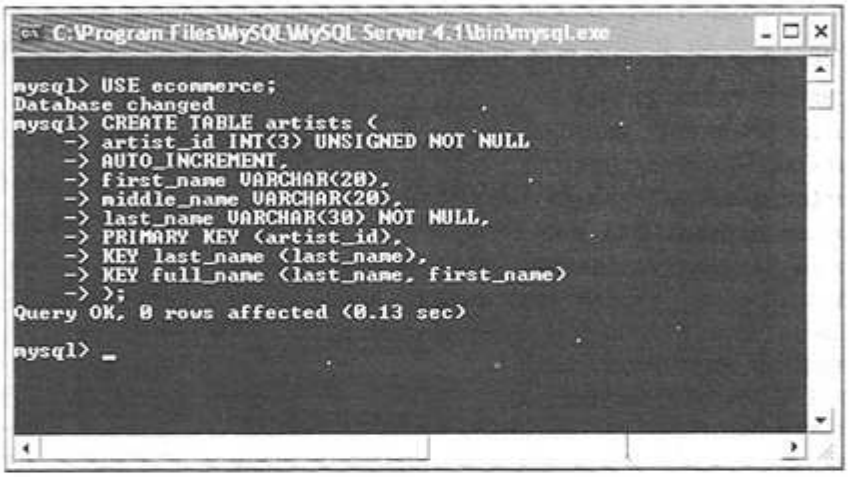
Thực hành tạo cơ sở dữ liệu theo các bước sau:

1. Đăng nhập trình quản lý mysql và chọn cơ sở dữ liệu *ecommerce*.

```
mysql –u root –p
USE  ecommerce;
```

Bạn có thể dùng trình quản lý mysql hoặc dùng công cụ khác (như phpMyAdmin) cho những bước này.

2. Tạo bảng *artists* (xem hình 13-1).



Hình 13-1: *Tạo bảng đầu tiên.*

```
CREATE TABLE artists (
artist_id INT(3) UNSIGNED NOT NULL AUTO_INCREMENT,
first_name VARCHAR(20),
middle_name VARCHAR(20),
last_name VARCHAR(30) NOT NULL,
PRIMARY KEY (artist_id),
KEY last_name (last_name),
KEY full_name (last_name, first_name)
);
```

Chúng ta bổ sung định nghĩa cho các chỉ mục (hoặc khóa) vào phần tạo bảng và thiết lập các trường buộc phải có giá trị.

An toàn

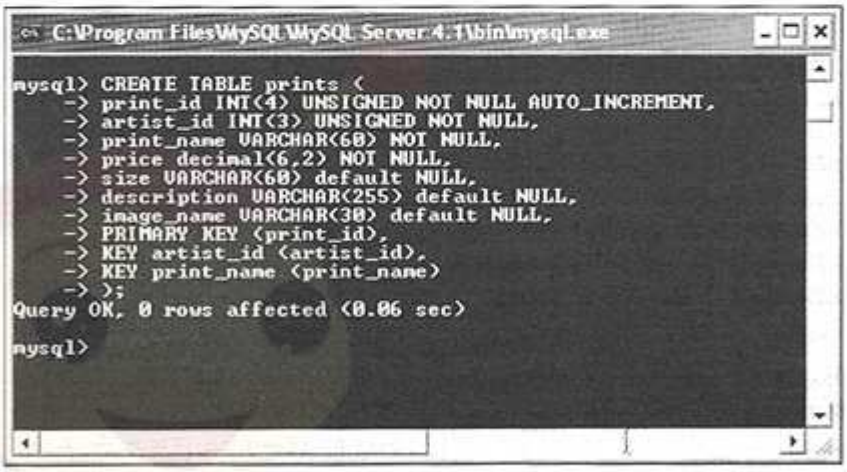
Trong một site thương mại điện tử, có ba mối quan tâm chính về an toàn. Điều đầu tiên là cách lưu dữ liệu trên máy chủ. Bạn cần bảo vệ cơ sở dữ liệu MySQL (bằng cách thiết lập các quyền truy cập thích hợp) và thư mục chứa thông tin của các phiên truy cập (xem Chương 7 "Cookies và Sessions" để biết thêm thông tin).

Mối quan tâm về an toàn thứ hai là bảo vệ sự truy nhập những thông tin nhạy cảm. Vì phần quản trị của site có khả năng xem được các đơn đặt hàng và hồ sơ khách hàng, nên nó phải được bảo vệ ở mức cao nhất (sẽ đề cập chi tiết trong phần sau của chương này).

Yếu tố thứ ba là bảo vệ dữ liệu trong chuyển tải. Tại thời điểm khách hàng thanh toán (lúc họ nhập thẻ tín dụng và thông tin chuyển hàng), giao dịch an toàn phải được sử dụng. Để làm điều đó, hãy thiết lập Secure Sockets Layer (SSL) trên máy chủ với chứng thực hợp lệ rồi chuyển qua https://URL. Cũng nên cảnh giác với thông tin được gửi qua e-mail, vì những tin nhắn đó thường không được chuyển qua những đường an toàn.

3. Tạo bảng *prints* (xem hình 13-2).

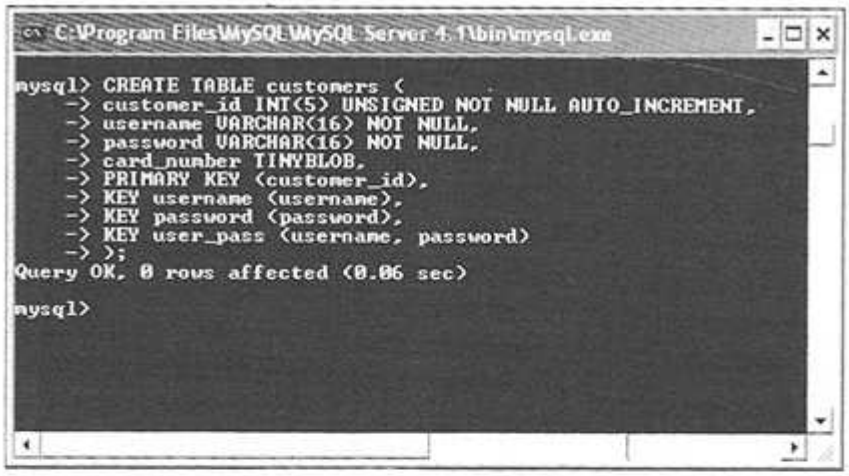
```
CREATE TABLE prints (
print_id INT(4) UNSIGNED NOT NULL AUTO_INCREMENT,
artist_id INT(3) UNSIGNED NOT NULL,
print_name VARCHAR(60) NOT NULL,
price decimal(6,2) NOT NULL,
size VARCHAR(60) default NULL,
description VARCHAR(255) default NULL,
image_name VARCHAR(30) default NULL,
PRIMARY KEY (print_id),
KEY artist_id (artist_id),
KEY print_name (print_name)
);
```



Hình 13-2: *Tạo bảng thứ hai.*

Hầu hết các cột trong bảng *prints* đòi hỏi phải có giá trị, ngoại trừ *size*, *description* và *image_name*. Chúng ta cũng thiết lập chỉ mục trên các trường *artist_id* và *print_name*, vì đây là hai cột thường được tìm nhất.

4. Tạo bảng *customers* (xem hình 13-3).



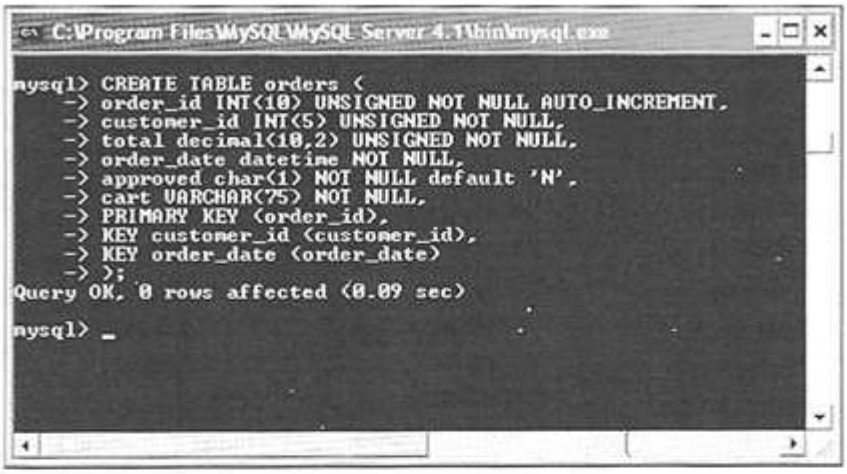
Hình 13-3: *Dùng SQL để tạo bảng* customers *nếu cần.*

```
CREATE TABLE customers (
customer_id INT(5) UNSIGNED NOT NULL AUTO_INCREMENT,
username VARCHAR(16) NOT NULL,
password VARCHAR(16) NOT NULL,
card_number TINYBLOB,
PRIMARY KEY (customer_id),
KEY username (username),
KEY password (password),
KEY user_pass (username, password)
);
```

Đây là mã lệnh dùng để tạo bảng *customers*. Nếu cơ sở dữ liệu của bạn chưa có bảng này, hãy bổ sung nó. Bạn có thể thêm vào các trường khác (như tên khách hàng, địa chỉ, số điện thoại...). Nếu bảng đã có sẵn trong cơ sở dữ liệu, bạn có thể dùng câu truy vấn ALTER để thêm vào bất cứ trường nào khác.

5. Tạo bảng *orders* (xem hình 13-4).

```
CREATE TABLE orders (
order_id INT(10) UNSIGNED NOT NULL AUTO_INCREMENT,
customer_id INT(5) UNSIGNED NOT NULL,
total decimal(10,2) UNSIGNED NOT NULL,
order_date datetime NOT NULL,
approved char(1) NOT NULL default 'N',
cart VARCHAR(75) NOT NULL,
PRIMARY KEY (order_id),
KEY customer_id (customer_id),
KEY order_date (order_date)
);
```



Hình 13-4: *Tạo bảng cuối cùng.*

Tất cả các trường của bảng *orders* bắt buộc phải có giá trị và hai chỉ mục cũng được tạo.

Tùy theo site bán gì mà bạn có thể có những bảng khác thay cho các bảng artists và prints. Thuộc tính quan trọng nhất của cơ sở dữ liệu là có một bảng sản phẩm liệt kê từng mặt hàng đang bán, kèm theo ID cho mỗi sản phẩm.

Phần quản trị

Đầu tiên, chúng ta sẽ viết mã lệnh để bổ sung ấn phẩm vào trong cơ sở dữ liệu. Trang này sẽ cho phép nhà quản trị chọn họa sỹ theo tên hoặc nhập vào một tên mới, tải lên một hình và nhập chi tiết cho ấn phẩm. Hình ảnh sẽ được lưu trữ trên máy chủ và mẩu tin ấn phẩm được chèn vào cơ sở dữ liệu. Nhưng vì điều này đòi hỏi một mã kịch bản kết nối với cơ sở dữ liệu MySQL nên chúng ta sẽ viết mã kịch bản đó trước.

Thực hành tạo mã kịch bản **mysql_connect.php** theo các bước sau:

1. Tạo hồ sơ PHP mới trong trình soạn thảo văn bản.

```
<?php # Doan ma 13.1 – mysql_connect.php
```

2. Định nghĩa các hằng kết nối cơ sở dữ liệu.

```
define ('DB_USER', 'username');
define ('DB_PASSWORD', 'password');
define ('DB_HOST', 'localhost');
define (DB_NAME, 'ecommerce');
```

3. Kết nối với MySQL và chọn cơ sở dữ liệu.

```
$dbc = @mysql_connect (DB_HOST, DB_USER, DB_PASSWORD) OR
→ die ('Khong the ket noi  MySQL:' . mysql_error());

mysql_select_db (DB_NAME) OR die ('Khong the chon CSDL: '
→ . mysql_error());
```

Mặc dù có thể dùng những kỹ thuật quản lý lỗi tiên tiến hơn (như các kỹ thuật được đề cập trong Chương 12 "Ví dụ - Đăng ký người dùng"), nhưng để ngắn gọn, chúng ta sử dụng hàm **mysql_error()** chuẩn.

4. Định nghĩa hàm **escape_data()** và hoàn thành trang PHP.

```
function escape_data ($data) {
   global $dbc;
   if (ini_get('magic_quotes_gpc')) {
      $data = stripslashes($data);
   }
   return mysql_real__escape_string(trim ($data), $dbc);
}
?>
```

5. Lưu tập tin với tên **mysql_connect.php** và tải lên máy chủ Web (ngoài thư mục gốc của Web). Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 13.1.

Cấu trúc của site này giống với hai ví dụ trước và bố cục của nó được thể hiện như hình 13-5.

Hình 13-5: *Cấu trúc site cho ứng dụng Web.*

Đoạn mã 13.1: *Mã kịch bản* ***mysql_connect.php*** *xử lý tất cả các tính năng chính liên quan đến cơ sở dữ liệu.*

```
<?php # Doan ma 13.1 - mysql_connect.php

define ('DB_USER', 'username');

define ('DB_PASSWORD', 'password');

define ('DB_HOST', 'localhost');

define ('DB_NAME', 'ecommerce');

$dbc = @mysql_connect (DB_HOST, DB_USER, DB_PASSWORD) OR
→ die ('Khong the ket noi MySQL: ' . mysql_error() );

mysql_select_db (DB_NAME) OR die ('Khong the chon duoc
→ CSDL: ' . mysql_error());

function escape_data ($data) {

   global $dbc;

   if (ini_get('magic_quotes_gpc')) {

      $data = stripslashes($data);

   }

   return mysql_real_escape_string (trim ($data), $dbc);

}

?>
```

Thực hành tạo mã kịch bản **add_print.php** theo các bước sau:

1. Tạo một hồ sơ PHP mới, bắt đầu với phần đầu HTML.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
→ Transitional//EN" "http://www.w3.org/TR/2000/REC-xhtml1-
→ 20000126/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">

<head>

<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />

<title>Bo sung mot an pham</title>

</head>

<body>

<?php # Doan ma 13.2 - add_print.php
```

Chúng ta thường tạo một hệ thống mẫu cho phần quản trị. Tuy nhiên, ví dụ này chỉ viết một mã kịch bản, nên ta bỏ qua bước đó.

2. Đưa vào mã kịch bản kết nối cơ sở dữ liệu và kiểm tra biểu mẫu có được gửi không.

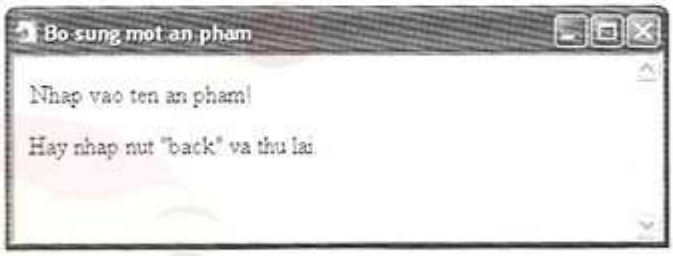
```
require_once ('../../mysql_connect.php');

if (isset($_POST['submit']))
```

Thư mục quản trị được đặt trong thư mục chính (html) nên mã kịch bản kết nối sẽ nằm trên nó hai cấp. Bạn cần nhớ cấu trúc thư mục khi đưa vào các tập tin.

3. Kiểm tra tên ấn phẩm.

```
if (!empty($_POST['print_name'])) {
   $pn = escape_data($_POST['print_name']);
} else {
   $pn = FALSE;
   echo '<p><font color="red">Nhap vao ten an
   → pham!</font></p>';
}
```

Tên ấn phẩm (trường print_name) bắt buộc phải có trong bảng *prints* nên cần được kiểm tra giá trị. Ở đây, bạn có thể sử dụng biểu thức qui tắc. Tuy nhiên, để đơn giản chúng ta sử dụng mã lệnh kiểm tra như trên. Nếu không có giá trị đưa vào, một thông báo lỗi sẽ được gửi đi (xem hình 13-6).



Hình 13-6: *Cơ sở dữ liệu sẽ không được bổ sung hồ sơ nếu tên ấn phẩm không được đưa vào.*

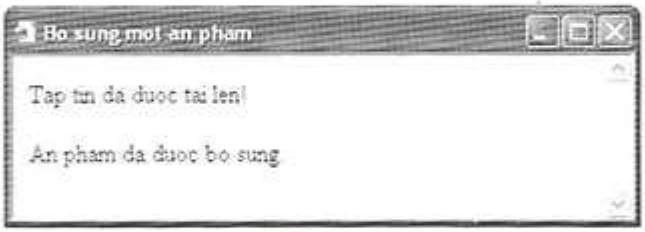
4. Xử lý tập tin ảnh nếu nó được chọn.

```
if (is_uploaded_file ($_FILES['image']['tmp_name'])) {
   if (move_uploaded_file($_FILES['image']['tmp_name'],
   → "../../uploads/{$_FILES['image']['name']}")) {
      echo '<p>Tap tin da duoc tai len!</p>';
   } else {
```

```
        echo '<p><font color="red">Khong the di chuyen tap
        → tin.</font></p>';
        $i = '';
    }
    $i = $_FILES['image']['name'];
} else {
    $i = '';
}
```

Khi giới thiệu các kỹ thuật xử lý tải lên tập tin với PHP (trong Chương 11 "Ví dụ - Quản lý nội dung"), chúng ta đã đề cập hàm `is_uploaded_file()`. Hàm này trả lại giá trị `TRUE` nếu tập tin được tải lên và `FALSE` khi ngược lại. Nếu một tập tin được tải lên, mã lệnh sẽ di chuyển nó vào thư mục *uploads*. Thông báo xác nhận thao tác thành công sẽ được in ra trang Web (xem hình 13-7).

Cuối cùng, biến `$i` được thiết lập với tên tập tin (tải thành công) hay một chuỗi rỗng (không thể tải tập tin).



Hình 13-7: *Kết quả tạo ra nếu tập tin được chọn cho hình ảnh ấn phẩm và tải lên thành công.*

5. Kiểm tra kích thước, giá cả và mô tả.

```
if (!empty($_POST['size'])) {
    $s = escape_data($_POST['size']);
} else {
    $s = '<i>Thong tin kich thuoc khong co hieu luc.</i>';
}
if (is_numeric($_POST['price'])) {
    $p = $_POST['price'];
} else {
    $p = FALSE;
    echo '<p><font color="red">Nhap vao gia an
    → pham!</font></p>';
}
```

```
y($_POST['description'])) {
escape_data($_POST['description']);

<i>Phan mo ta khong co hieu luc.</i>';
```

các giá trị về kích cỡ và mô tả là tùy chọn. Với một phép kiểm tra ; ta đảm bảo giá được gửi lên là một số (nên là số thập phân) bằng m `is_numeric()`. Một thông báo lỗi sẽ hiện ra nếu không có giá ợp lệ (xem hình 13-8).

13-8: *Giá đồ phẩm được kiểm tra về giá trị và kiểu thích hợp.*

ời dùng không nhập kích cỡ và mô tả, chúng ta thiết lập các biến các thông báo mặc định.

xem có nhập vào họa sỹ mới không.

```
T['artist'] == 'new') {

r = 'INSERT INTO artists (artist_id, first_name,
dle_name, last_name) VALUES (NULL, ';
```

sỹ cho ấn phẩm, nhà quản trị có hai lựa chọn (xem hình 13-9): chọn iện có (từ mẩu tin trong bảng *artists*) bằng menu thả xuống hoặc sỹ mới. Nếu nhập vào họa sỹ mới, mẩu tin phải được chèn vào ước khi ấn phẩm được bổ sung vào bảng *prints*. Ở đây, chúng ta uy vấn để bổ sung họa sỹ.

Hình 13-9: *Phần lựa chọn họa sỹ của biểu mẫu.*

việc tập hợp câu truy vấn.

```
y($_POST['first_name'])) {
r .= "'" . escape_data($_POST['first_name']) .
";
```

```
.= 'NULL, ';

($_POST['middle_name'])) {
 .= "'" . escape_data($_POST['middle_name']) .
";

.= 'NULL, ';

($_POST['last_name'])) {
.= "'" . escape_data($_POST['last_name']) .

 = @mysql_query ($query);
ysql_insert_id();

LSE:
p><font color="red">Nhap vao ten cua ban
/font></p>';
```

n là trường tùy chọn, trong khi tên họ thì không (vì có những họa
hiếu chỉ bằng một tên). Tên và tên đệm sẽ được đưa vào câu truy
LL, tùy vào việc chúng có giá trị hay không. Quá trình này tạo ra
dạng như: *INSERT...VALUES (NULL, NULL, NULL, 'Sargeant')*
VALUES (NULL, 'John', 'Singer', 'Sargeant'). Vì tên họ là trường
hông báo lỗi sẽ hiện ra nếu nó bị sót trong mẫu tin họa sỹ mới
0).

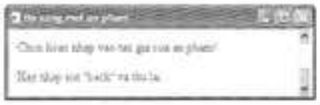
i được bổ sung vào cơ sở dữ liệu, ID của họa sỹ sẽ được khôi phục
câu truy vấn ấn phẩm) bằng cách dùng hàm mysql_insert_id().

13-10: *Nếu nhập họa sỹ mới, họ tên phải được nhập vào.*

h phần điều kiện kiểm tra họa sỹ.

```
( ($_POST['artist'] == 'existing') &&
T['existing'] > 0)) {
_POST['existing'];

'ALSE;
<p><font color="red">Chon hoac nhap vao tac gia cua
pham!</font></p>';
```

i trị chọn một họa sỹ hiện có, chúng ta sẽ kiểm tra xem giá trị đó menu thả xuống không. Nếu điều kiện này sai, một thông báo lỗi m hình 13-11).



Nếu không có họa sỹ được nhập hay chọn, thông báo lỗi sẽ hiện ra.

ào vào cơ sở dữ liệu.

```
a $p && $al {
r = "INSERT INTO prints (artist_id, print_name,
ce, size, description, image_name) VALUES ('$a,
', $p, '$s', '$d', '$i')";
esult = @mysql_query ($query)) {
o '<p>An pham da duoc bo sung.</p>';
 {
o '<p><font color="red">Thong tin cua ban khong
the duoc xu ly do loi he thong.</font></p>';
```

g yêu cầu phải có (print_name, price và artist_id) được điền vào, ấn đưa vào cơ sở dữ liệu. Thông báo sẽ được gửi tới trình duyệt Web, truy vấn được thực hiện thành công (xem hình 13-12).

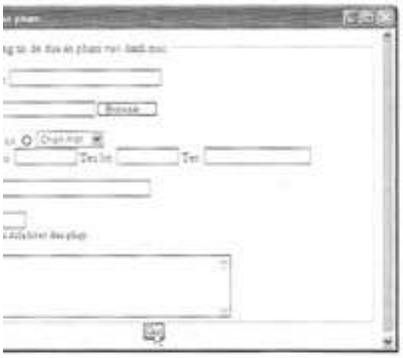
Kết quả một ấn phẩm được bổ sung vào danh mục sản phẩm.

phần các điều kiện.

```
{
' '<p><font color="red">Hay nhap nut "back" va
u lai.</font></p>';
```

thứ nhất được áp dụng khi biểu mẫu không vượt qua được một kiểm tra. Lệnh **else** thứ hai được dùng để thể hiện biểu mẫu nếu

ẫu HTML (xem hình 13-13).



Hình 13-13: *Biểu mẫu HTML khi hoàn tất.*

```
type="multipart/form-data" action="<?php echo
ER['PHP_SELF']; ?>" method="post">
e="hidden" name="MAX_FILE_SIZE" value="524288">
<legend>Dien vao thong tin de dua an pham vao danh
egend>
```

```
n sĩ:</b> <input type="text" name="print_name"
30" maxlength="60" /></p>

nh ảnh:</b> <input type="file" name="image" /></p>
```

này cho phép người dùng tải lên một tập tin, chúng ta phải đưa
ctype vào thẻ <form> và trường ẩn MAX_FILE_SIZE.

thả xuống cho phần họa sỹ.

```
a sy:</b>
<input type="radio" name="artist" value="existing"/>
ame="existing"><option>Chon mot</option>


 "SELECT artist_id, CONCAT(last_name, ', ',
name) AS name FROM artists ORDER BY last_name

= @mysql_query ($query);
ow = mysql_fetch_array ($result, MYSQL_ASSOC)) {
'<option value=\"{$row['artist_id']}\">
w['name']}</option>\n";

ose();
```


ng trong phần họa sỹ sẽ được tạo động (xem hình 13-14) từ các mẩu
rong bảng *artists* bằng mã lệnh PHP này. Nó bắt đầu bằng một nút
nhà quản trị có thể chọn một họa sỹ hiện có hoặc nhập vào một
ơn bước 13).

```
<br/>
put type="radio" name="artist" value="existing" />
e="existing"><option>Chon mot</option>
ue="6">4, Nguyen</option>
ue="7">5, Tran</option>
ue="8">2, Nguyen</option>
ue="11">8, Cao</option>
ue="9">7, Le</option>
t />
type="radio" name="artist" value="new" />
put type="text" name="first_name" size="15" maxlength="20" />
type="text" name="middle_name" size="15" maxlength="20" />
 type="text" name="last_name" size="20" maxlength="30" />
```

Hình 13-14: *Mã nguồn cho phần họa sỹ của biểu mẫu.*

phần nhập để bổ sung họa sỹ mới.

```
 type="radio" name="artist" value="new" />
nput type="text" name="first_name" size="15"
th="30" />
input type="text" name="middle_name" size="15"
th="30" />
t type="text" name="last_name" size="20"
th="30" />
```

t biểu mẫu riêng để bổ sung họa sỹ vào cơ sở dữ liệu, chúng ta sẽ
ng trang này. Mã PHP xử lý biểu mẫu sẽ có mã lệnh để thực hiện

iểu mẫu HTML

```
 thuoc:</b> <input type="text" name="size"
0" maxlength="60" /></p>
:</b> <input type="text" name="price" size="10"
th="10" /><br /><small>Khong nhap dau dola hoac
y.</small></p>
ta:</b> <textarea name="description" cols="40"
"></textarea></p>
>
="center"><input type="submit" name="submit"
Gui" /></div>
```

u mục cần thiết trên máy chủ.

này yêu cầu hai thư mục mới. Thư mục thứ nhất là *ddm1n* (xem
13-16) chứa các tập tin quản trị. Thư mục thứ hai, *uploads*, nên
thư mục Web và được thiết lập quyền truy cập để PHP có thể chép
nó. Xem Chương 11 để biết thêm thông tin.

phần điều kiện PHP và trang HTML.

g này của mã lệnh PHP hoàn tất phần **else** của điều kiện (thể
nếu chưa được gửi).

tin **add_print.php**, tải lên máy chủ (đặt vào thư mục quản trị) và
trong trình duyệt (xem hình 13-15 và 13-16). Mã lệnh đầy đủ của
ực thể hiện trong đoạn mã 13.2.

13-15: *Chúng ta dùng họa sỹ hiện có để bổ sung ấn phẩm.*

-16: *Trong ví dụ này, chúng ta bổ sung ấn phẩm cho họa sỹ mới.*

12: *Trong quản trị này sẽ bổ sung ấn phẩm vào cơ sở dữ liệu.*

```
 html PUBLIC "-//W3C//DTD XHTML 1.0
ional//EN" "http://www.w3.org/TR/1998/REC-xhtml1-
t6/DTD/xhtml1-transitional.dtd">
s="http://www.w3.org/1999/xhtml" xml:lang="en"
n">

:-equiv="content-type" content="text/html;
:iso-8859-1" />
sung mot an pham</title>


oan ma 13.2 - add_print.php
```

```
e ('../../mysql_connect.php');
POST['submit'])) {
y($_POST['print_name'])) {
escape_data($_POST['print_name']);

FALSE;
<p><font color="red">Nhap vao ten an pham!</font>
>';

loaded_file ($_FILES['image']['tmp_name'])) {
ve_uploaded_file($_FILES['image']['tmp_name'],
'../uploads/($_FILES['image']['name'])")) {
o '<p>Tep tin da duoc tai len!</p>';
 {
o '<p><font color="red">Khong the di chuyen tep
tin.</font></p>';
= '';

_FILES['image']['name'];

';

y($_POST['size'])) {
escape_data($_POST['size']);

<i>Thong tin kich thuoc khong co hieu luc.</i>';

umeric($_POST['price'])) {
$_POST['price'];

FALSE;
'<p><font color="red">Nhap vao gia an pham!</font>
>';

y($_POST['description'])) {
escape_data($_POST['description']);
```

```
}
'<li>Phan mo ta khong co hieu luc.</li>';

)ST['artist'] == 'new') {
ry = 'INSERT INTO artists (artist_id, first_name,
ddle_name, last_name) VALUES (NULL, ';
empty($_POST['first_name'])) {
query .= "'" . escape_data($_POST['first_name']) .
"', ";
ie {
query .= 'NULL, ';

empty($_POST['middle_name'])) {
query .= "'" . escape_data($_POST['middle_name']) .
"', ";
ie {
query .= 'NULL, ';

empty($_POST['last_name'])) {
query .= "'" . escape_data($_POST['last_name']) .
"')";
result = @mysqli_query ($query);
= @mysqli_insert_id();
ie {
= FALSE;
ho '<p><font color="red">Nhap vao ten cua hoa sy(
</font></p>';

( ($_POST['artist'] == 'existing') &&
ST['existing'] > 0)) {
$_POST['existing'];
{
FALSE;
'<p><font color="red">Chon hoac nhap vao tac gia
a an pham!</font></p>';

&& $p && $a) {
```

```
' = "INSERT INTO prints (artist_id, print_name,
pe, size, description, image_name) VALUES ($a,
', $p, '$s', '$d', '$i')";
esult = @mysql_query ($query)) {
o '<p>An pham da duoc bo sung.</p>';
 {
o '<p><font color="red">Thong tin cua ban khong
he duoc xu ly do loi he thong.</font></p>';


<p><font color="red">Hay nhap nut "back" va thu
</font></p>';



pe="multipart/form-data" action="<?php echo
['PHP_SELF']; ?>" method="post">
<"hidden" name="MAX_FILE_SIZE" value="524288">
egend>Dien vao thong tin de dua an pham vao danh
end>
n pham:</b> <input type="text" name="print_name"
 maxlength="60" /></p>
anh:</b> <input type="file" name="image" /></p>
y:</b>
ut type="radio" name="artist" value="existing"/>
e="existing"><option>Chon mot</option>

ELECT artist_id, CONCAT(last_name, ', ',
e) AS name FROM artists ORDER BY last_name ASC";
ysql_query ($query);
 = mysql_fetch_array ($result, MYSQL_ASSOC)) {
ption value=\"{$row['artist_id']}\">
'name']}</option>\n";

();

r />
```

```
Moi <input type="radio" name="artist" value="new" />
Ten ho: <input type="text" name="first_name" size="10"
→ maxlength="30" />
Ten lot: <input type="text" name="middle_name" size="10"
→ maxlength="30" />
Ten: <input type="text" name="last_name" size="20"
→ maxlength="30" />
</p>
<p><b>Kich thuoc:</b> <input type="text" name="size"
→ size="30" maxlength="60" /></p>
<p><b>Gia:</b> <input type="text" name="price" size="10"
→ maxlength="10" /><br /><small>Khong nhap dau dola hoac dau
→ phay.</small></p>
<p><b>Mo ta:</b> <textarea name="description" cols="40"
→ rows="5"></textarea></p>
</fieldset>
<div align="center"><input type="submit" name="submit"
→ value="Gui" /></div>
</form>
<?php
}
?>
</body>
</html>
```

Để đơn giản, trong ví dụ này chúng ta sử dụng chung tài khoản truy cập MySQL cho cả phần quản trị và phần công cộng. Tuy nhiên, bạn nên tạo một tài khoản cho phần công cộng và một cho phần quản trị. Tài khoản phía quản trị cần các quyền ***SELECT***, ***INSERT***, ***UPDATE*** *và* ***DELETE***, *còn tài khoản phía công cộng chỉ cần quyền* ***SELECT*** *và* ***INSERT****.*

Các trang quản trị phải được bảo vệ ở mức an toàn cao nhất bằng các kỹ thuật như kèm theo phần xác thực HTTP, hệ thống đăng nhập hoặc đặt chúng trên máy chủ khác (có thể không nối mạng).

Tạo khuôn mẫu cho phần công cộng

Trước khi đi vào trọng tâm của phần công cộng, chúng ta cần tạo các tập tin đầu và cuối trang cần thiết. Chúng ta sẽ lướt nhanh qua phần này, vì các kỹ thuật liên quan đều đã được đề cập trong các chương trước.

Thực hành tạo tập tin **header.html** theo các bước sau:

1. Tạo một hồ sơ PHP mới trong trình biên soạn văn bản.

```
<?php # Doan ma 13.3 - header.htm
```

2. Bắt đầu session.

```
session_start();
?>
```

Do session của người dùng được duy trì qua từng trang, nên chúng ta sẽ bắt đầu session trong tập tin đầu trang. Nếu session không được giữ nhất quán, một session mới sẽ được tạo trong những trang tiếp theo và quá trình mua sắm của khách hàng (như giỏ mua hàng) sẽ không thể thực hiện được.

3. Tạo phần đầu HTML.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
→ Transitional//EN" "http://www.w3.org/TR/2000/REC-
→ xhtml1-20000126/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">

<head>

<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />

<title><?php echo $page_title; ?></title>

</head>

<body>
```

Cũng như với các phiên bản khác của mã kịch bản này, tiêu đề trang sẽ được thiết lập cho một biến PHP và được in ra trong các thẻ tiêu đề.

4. Tạo hàng trên cùng của bảng.

```
<table cellspacing="0" cellpadding="0" border="0"
→ align="center" width="600">

<tr>

<td align="center" colspan="3"><img src="images/title.jpg"
→ width="600" height="61" border="0" alt="title" /></td>

</tr>

<tr>

<td><a href="index.php"><img src="images/home.jpg"
→ width="200" height="39" border="0" alt="home page"
→ /></a></td>
```

```
<td><a href="browse_prints.php"><img src="images/prints.jpg"
→ width="200" height="39" border="0" alt="view the prints"
→ /></a></td>

<td><a href="view_cart.php"><img src="images/cart.jpg"
→ width="200" height="39" border="0" alt="view your cart"
→ /></a></td>

</tr>
```

Bố cục này dùng hình ảnh để tạo kết nối cho các trang công cộng (xem hình 13-17).

Hình 13-17: *Biểu ngữ do tập tin đầu trang tạo ra.*

5. Bắt đầu hàng giữa.

```
<tr>

<td align="left" colspan="3" bgcolor="#ffffcc"><br />
```

6. Lưu tập tin với tên **header.html** và tải lên máy chủ Web (đặt trong thư mục *includes*). Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 13.3.

Đoạn mã 13.3: *Tập tin đầu trang tạo ra mã lệnh phần đầu HTML và khởi tạo session.*

```
<?php # Doan ma 13.3 - header.html
session_start();
?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
→ Transitional//EN" "http://www.w3.org/TR/2000/REC-xhtml1-
→ 20000126/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
→ lang="en">
<head>
<meta http-equiv="content-type" content="text/html;
→ charset=iso-8859-1" />
<title><?php echo $page_title; ?></title>
</head>
```

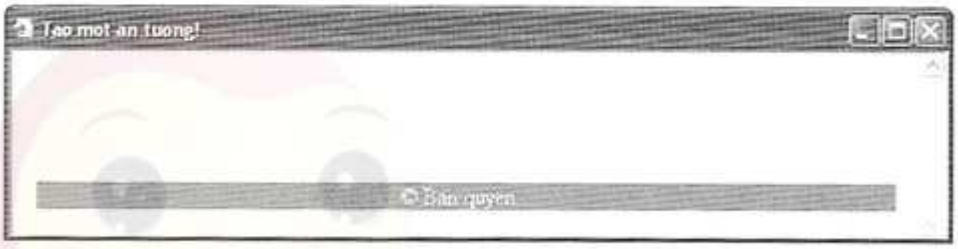
```
<body>
<table cellspacing="0" cellpadding="0" border="0"
→ align="center" width="600">
<tr>
<td align="center" colspan="3"><img src="images/title.jpg"
→ width="600" height="61" border="0" alt="title" /></td>
</tr>
<tr>
<td><a href="index.php"><img src="images/home.jpg" width="200"
→ height="39" border="0" alt="home page" /></a></td>
<td><a href="browse_prints.php"><img src="images/prints.jpg"
→ width="200" height="39" border="0" alt="view the prints"
→ /></a></td>
<td><a href="view_cart.php"><img src="images/cart.jpg"
→ width="200" height="39" border="0" alt="view your cart"
→ /></a></td>
</tr>
<tr>
<td align="left" colspan="3" bgcolor="#ffffcc"><br />
```

Thực hành tạo tập tin **footer.html** theo các bước sau:

1. Tạo một hồ sơ HTML mới trong trình soạn thảo.

```
<!-- Doan ma 13.4 - footer.html -->
```

2. Hoàn tất dòng giữa, tạo dòng cuối và hoàn tất HTML (xem hình 13-18).



Hình 13-18: *Dòng bản quyền do tập tin cuối trang tạo nên.*

```
<br/></td>
</tr>
<tr>
<td align="center" colspan="3" bgcolor="#669966">
→ <font color="#ffffff">&copy; Ban quyen...</font></td>
```

```
</tr>
</table>
</body>
</html>
```

3. Lưu tập tin với tên **footer.html** và tải lên máy chủ (cũng đưa vào thư mục *includes).* Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 13.4.

Đoạn mã 13.4: *Tập tin cuối trang đóng HTML, tạo ra hàng bản quyền trong quá trình.*

```
<!-- Doan ma 13.4 - footer.html -->
<br/></td>
</tr>
<tr>
<td align="center" colspan="3" bgcolor="#669966">
→ <font color="#ffffff">&copy; Ban quyen...</font></td>
</tr>
</table>
</body>
</html>
```

Tạo tập tin **index.php** theo các bước sau:

1. Tạo một hồ sơ PHP mới trong trình soạn thảo văn bản.

```
<?php # Doan ma 13.5 - index.php
$page_title = 'Tao mot an tuong!';
include_once ('includes/header.html');
?>
```

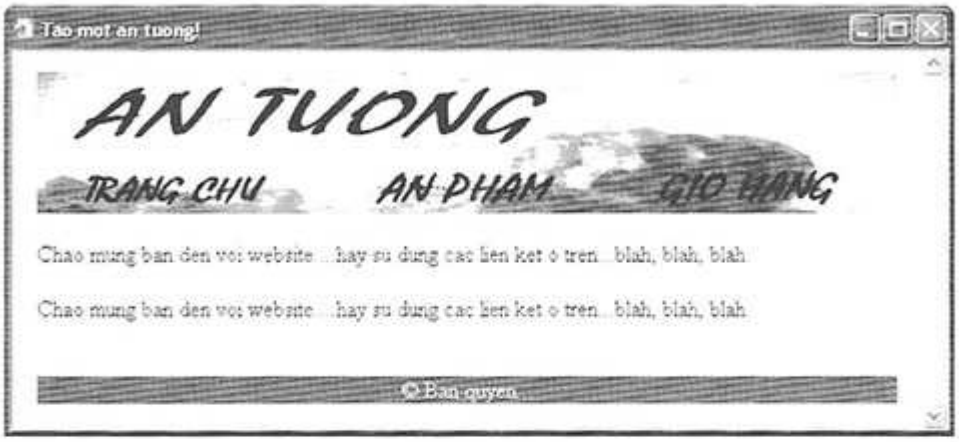
2. Tạo nội dung trang.

```
<p>Chao mung ban den voi Website....hay su dung cac lien ket o
tren...blah, blah, blah.</p>
<p>Chao mung ban den voi Website....hay su dung cac lien ket o
tren...blah, blah, blah.</p>
```

3. Hoàn tất trang HTML.

```
<?php include_once ('includes/footer.html');
?>
```

4. Lưu tập tin với tên **index.php**, tải lên máy chủ Web và chạy thử trong trình duyệt (xem hình 13-19). Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 13.5.



Hình 13-19: *Trang chủ phần công cộng của site thương mại.*

Đoạn mã 13.5: *Mã kịch bản cho trang chủ của site.*

```
<?php # Doan ma 13.5 - index.php

$page_title = 'Tao mot an tuong!';

include_once ('includes/header.html');

?>

<p>Chao mung ban den voi Website....hay su dung cac lien ket o
tren...blah, blah, blah.</p>

<p>Chao mung ban den voi Website....hay su dung cac lien ket o
tren...blah, blah, blah.</p>

<?php

include_once ('includes/footer.html');

?>
```

Vì các session là chìa khóa cho các tính năng của ứng dụng này, bạn nên xem lại thông tin trong Chương 7 hoặc trong tài liệu PHP để hiểu kỹ về session.

Danh mục sản phẩm

Để khách hàng có thể mua ấn phẩm, họ phải xem được hàng trước. Để thực hiện điều này, chúng ta sẽ tạo hai đoạn mã để truy cập danh mục sản phẩm. Đoạn mã

thứ nhất, **browse_prints.php**, sẽ hiển thị danh sách ấn phẩm có sẵn. Nếu họa sỹ nào đó được chọn, chỉ có tác phẩm của họa sỹ đó được chỉ ra. Trong trường hợp còn lại, tất cả ấn phẩm sẽ được liệt kê.

Đoạn mã thứ hai, **view_print.php**, được dùng để hiển thị thông tin từng ấn phẩm riêng lẻ. Trên trang này khách hàng sẽ thấy một kết nối *Dua vao gio hang* để đưa ấn phẩm vào giỏ hàng.

Thực hành tạo tập tin **browse_prints.php** theo các bước sau:

1. Tạo hồ sơ PHP mới trong trình soạn thảo văn bản.

```
<?php # Doan ma 13.6 - browse_prints.php

$page_title = 'Duyet tim an pham';

include_once ('includes/header.html');

require_once ('../mysql_connect.php');
```

2. Lập câu truy vấn.

```
if (isset($_GET['aid'])) {

   $query = "SELECT * FROM artists, prints WHERE
   -> artists.artist_id = prints.artist_id AND
   -> prints.artist_id = {$_GET['aid']} ORDER BY
   -> prints.print_name";

} else {

   $query = "SELECT * FROM artists, prints WHERE
   -> artists.artist_id = prints.artist_id ORDER BY
   -> artists.last_name ASC, prints.print_name ASC";

}
```

Câu truy vấn là liên kết chuẩn giữa các bảng *artists* và *prints* (để lấy ra tên họa sỹ của mỗi ấn phẩm). Lần đầu trang được xem, tất cả ấn phẩm của các họa sỹ sẽ được lấy ra. Nếu người dùng nhấp chuột vào tên một họa sỹ, họ sẽ được đưa trở lại trang này với URL mới, ví dụ: *browse_prints.php?aid=529*. Khi đó, mã lệnh `AND prints.artist_id = {$_GET['aid']}` sẽ được đưa vào câu truy vấn và mệnh đề `ORDER BY` được điều chỉnh đôi chút. Điều này sẽ làm cho trang chỉ hiển thị những ấn phẩm của họa sỹ được chọn.

Như vậy, hai vai trò khác nhau của đoạn mã được quyết định bởi điều kiện này, trong khi phần còn lại có chức năng như nhau trong cả hai trường hợp.

3. Tạo phần đầu bảng.

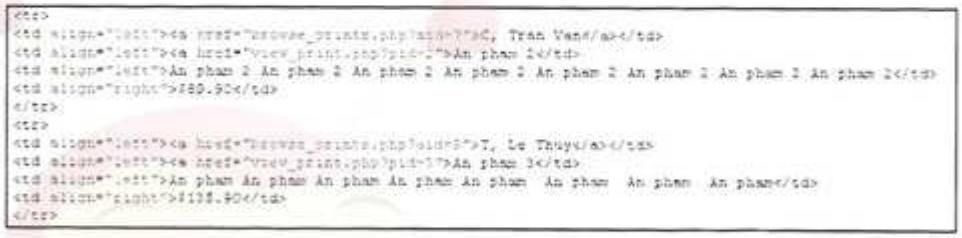
```
echo '<table border="0" width="90%" cellspacing="3"
-> cellpadding="3" align="center">
-> <tr>
```

```
→ <td align="left" width="20%"><b>Hoa sy</b></td>
→ <td align="left" width="20%"><b>Ten an pham</b></td>
→ <td align="left" width="40%"><b>Mo ta</b></td>
→ <td align="right" width="20%"><b>Gia</b></td>
→ </tr>';
```

4. Hiển thị từng mẩu tin trả về.

```
$result = mysql_query ($query);

while ($row = mysql_fetch_array ($result, MYSQL_ASSOC)) {

   echo "  <tr>
   → <td align=\"left\"><a href=\"browse_prints.php?
   → aid={$row['artist_id']}\">{$row['last_name']},
   → {$row['first_name']} {$row['middle_name']}</a></td>
   → <td align=\"left\"><a href=\"view_print.php?
   → pid={$row['print_id']}\">{$row['print_name']}</td>
   → <td align=\"left\">" .
   → stripslashes($row['description']) . "</td>
   → <td align=\"right\">\${$row['price']}</td></tr>\n";

}
```

Chúng ta muốn trang hiển thị tên họ, tên ấn phẩm, mô tả và giá cho mỗi mẩu tin trả về. Ngoài ra, tên họa sỹ còn được liên kết về trang này (với ID họa sỹ gắn vào URL) và tên ấn phẩm được liên kết với **view_print.php** (với ID ấn phẩm gắn vào URL). Hình 13-20 thể hiện mã lệnh nguồn của trang kết quả.



Hình 13-20: *Mã nguồn của trang cho thấy cách thức gắn ID họa sỹ và ID ấn phẩm vào liên kết.*

5. Đóng bảng, đóng kết nối cơ sở dữ liệu và trang HTML.

```
echo '</table>';

mysql_close();

include_once ('includes/footer.html');

?>
```

Hình 13-21: *Danh sách sản phẩm hiện có do tập tin* `browse_prints.php` *tạo ra.*

Hình 13-22: *Nếu một họa sỹ được chọn (bằng cách nhấp vào tên họa sỹ), trang sẽ chỉ hiển thị tác phẩm của họa sỹ đó.*

6. Lưu tập tin với tên **browse_prints.php**, tải lên trang chủ Web và chạy thử trong trình duyệt (xem hình 13-21 và 13-22). Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 13.6.

Đoạn mã 13.6: *Mã kịch bản* ***browse_prints.php*** *sẽ hiển thị tất cả ấn phẩm có trong danh mục hoặc các ấn phẩm của một họa sỹ cụ thể.*

```
<?php # Doan ma 13.6 - browse_prints.php
$page_title = 'Duyet tim an pham';
include_once ('includes/header.html');
require_once ('../mysql_connect.php');
if (isset($_GET['aid'])) {
   $query = "SELECT * FROM artists, prints WHERE
   -> artists.artist_id = prints.artist_id AND
   -> prints.artist_id = {$_GET['aid']} ORDER BY
   -> prints.print_name";
} else {
   $query = "SELECT * FROM artists, prints WHERE
   -> artists.artist_id = prints.artist_id ORDER BY
   -> artists.last_name ASC, prints.print_name ASC";
}
echo '<table border="0" width="90%" cellspacing="3"
-> cellpadding="3" align="center">
-> <tr>
-> <td align="left" width="20%"><b>Hoa sy</b></td>
-> <td align="left" width="20%"><b>Ten an pham</b></td>
-> <td align="left" width="40%"><b>Mo ta</b></td>
-> <td align="right" width="20%"><b>Gia</b></td>
-> </tr>';
$result = mysql_query ($query);
while ($row = mysql_fetch_array ($result, MYSQL_ASSOC)) {
   echo "  <tr>
   -> <td align=\"left\"><a href=\"browse_prints.php?
   -> aid={$row['artist_id']}\">{$row['last_name']},
   -> {$row['first_name']} {$row['middle_name']}</a></td>
   -> <td align=\"left\"><a href=\"view_print.php?
   -> pid={$row['print_id']}\">{$row['print_name']}</td>
   -> <td align=\"left\">" . stripslashes($row['description'])
   -> . "</td>
   -> <td align=\"right\">\${$row['price']}</td></tr>\n";
}
echo '</table>';
```

```
mysql_close();
include_once ('includes/footer.html');
?>
```

Tìm kiếm danh mục sản phẩm

Cấu trúc của cơ sở dữ liệu này giúp dễ dàng tạo nên tính năng tìm kiếm. Có ba trường lôgic có thể được dùng cho mục đích này: tên ấn phẩm, mô tả và họ của họa sỹ. Câu truy vấn **LIKE** *được thực hiện trên những trường này có cú pháp sau:*

```
SELECT...WHERE prints.description LIKE '%keyword%' OR
→ prints.print_name LIKE '%keyword%' ...
```

Một lựa chọn khác là tạo ra khả năng tìm kiếm nâng cao (giống như nhiều Website tìm kiếm thường có), ở chỗ người dùng có thể lựa chọn tìm kiếm tên họa sỹ hoặc tên ấn phẩm hay không.

Cuối cùng, bạn có thể dùng khả năng tìm kiếm toàn văn của MySQL để trả lại kết quả theo trật tự nhất định. Xem tài liệu MySQL để biết thêm thông tin về các chỉ mục và tìm kiếm toàn văn.

Bạn có thể dễ dàng lấy menu thả xuống được tạo động từ mã kịch bản **add_print.php** *và dùng nó như một công cụ di chuyển ở phía công cộng của site. Thiết lập thuộc tính action của biểu mẫu với* **browse_print.php**, *đổi tên menu thành aid và dùng phương pháp* **get**. *Khi người dùng chọn một họa sỹ và nhấp vào nút submit, họ sẽ được đưa đến trang với liên kết có dạng như browse_print.php?aid=5.*

Bạn có thể phân trang dựa trên kết quả trả về bằng cách sử dụng kỹ thuật được nêu trong Chương 12 (tham khảo mã kịch bản **view_users.php**).

Một đặc điểm khác có thể bổ sung vào trang này là lựa chọn cách hiển thị các ấn phẩm. Bằng cách thêm các kết nối vào đầu cột (ví dụ browse_prints.php?order=price), bạn có thể đổi **ORDER BY** *trong câu truy vấn và do đó làm thay đổi kết quả hiển thị.*

Thực hành tạo tập tin **view_print.php** theo các bước sau:

1. Tạo hồ sơ PHP mới trong trình soạn thảo văn bản.

```
<?php # Doan ma 13.7 - view_print.php
```

2. Kiểm tra xem ấn phẩm được đưa lên chưa.

```
if (is_numeric ($_GET['pid'])) {
```

Đoạn mã này sẽ không hoạt động nếu nó chưa nhận được ID ấn phẩm hợp lệ. Nếu nó nhận được (và hợp lệ với phần điều kiện này), trang sẽ hiển thị thông tin. Nếu không, người dùng sẽ được đưa về lại trang chủ (xem bước 7).

3. Lấy thông tin từ cơ sở dữ liệu.

```
require_once ('../mysql_connect.php');

$query = "SELECT * FROM artists, prints WHERE
→ artists.artist_id = prints.artist_id AND
→ prints.print_id = {$_GET['pid']}";

$result = mysql_query ($query);

$row = mysql_fetch_array ($result, MYSQL_ASSOC);

mysql_close();
```

Câu truy vấn là một liên kết giống như trong mã kịch bản **browse_prints.php**, nhưng nó chỉ lấy thông tin cho một ấn phẩm cụ thể.

4. Thiết lập tiêu đề trang và đưa vào phần đầu HTML.

```
$page_title = $row['print_name'];

include_once ('includes/header.html');
```

Tiêu đề của cửa sổ trình duyệt (xem hình 13-23) là tên của ấn phẩm.

Hình 13-23: *Tiêu đề cửa sổ trình duyệt chính là tên của ấn phẩm đang được thể hiện.*

5. Bắt đầu hiển thị thông tin ấn phẩm.

```
echo "<div align=\"center\">
→ <b>{$row['print_name']}</b> cua {$row['first_name']}
→ {$row['middle_name']} {$row['last_name']}

→ <br />{$row['size']} <br />\${$row['price']}
→ <a href=\"add_cart.php?pid={$row['print_id']}\">Dua vao
→ gio hang</a></div><br />";
```

Phần đầu thông tin ấn phẩm sẽ là tên của nó (in đậm), tiếp đến là tên họa sỹ, kích thước ấn phẩm và giá cả. Cuối cùng, một liên kết được hiển thị để khách hàng chọn và đưa ấn phẩm vào giỏ mua hàng (xem hình 13-24). Liên kết này tham chiếu đến đoạn mã **add_cart.php** và truyền cho nó ID ấn phẩm.

An pham 2 cua Tran Van C
50x30
$89.90 Dua vao gio hang

Hình 13-24: *Thông tin ấn phẩm và một liên kết để đưa nó vào giỏ hàng được hiển thị ở đầu trang.*

6. Hiển thị hình ảnh và mô tả.

```
if ($image = @getimagesize
→ ("../uploads/{$row['image_name']}")) {
   echo "<div align=\"center\"><img
   → src=\"../uploads/{$row['image_name']}\" $image[3]
   → alt=\"{$row['print_name']}\" />";
} else {
   echo "<div align=\"center\">Khong co hinh anh nao.";
}
echo '<br />' . stripslashes($row['description']) .
→ '</div>';
```

Phần này của đoạn mã sẽ lấy ra kích thước của hình ảnh bằng hàm `getimagesize()`. Nếu thao tác thành công, hình ảnh sẽ được hiển thị.

Nếu đoạn mã không thể lấy ra kích thước hình ảnh (vì nó không có trên máy chủ hoặc chưa được tải lên), một thông báo lỗi sẽ hiện ra. Cuối cùng, phần mô tả ấn phẩm sẽ được thêm vào (xem hình 13-25).

An pham 2 An pham 2 An pham 2 An pham 2 An pham 2 An pham 2 An pham 2 An pham 2

Hình 13-25: *Hình ấn phẩm, tiếp theo sau là phần mô tả.*

7. Đưa vào phần cuối trang HTML, hoàn tất phần điều kiện chính và trang PHP.

```
    include_once ('includes/footer.html');

} else {

    header ("Location:  http://" . $_SERVER['HTTP_HOST']
    → dirname($_SERVER['PHP_SELF']) . "/index.php");

    exit();

}

?>
```

8. Lưu tập tin với tên **view_print.php**, tải lên máy chủ và chạy thử trong trình duyệt Web (xem hình 13-26). Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 13.7.

Hình 13-26: *Trang* ***view_print.php*** *thể hiện một ấn phẩm từ danh mục sản phẩm.*

Đoạn mã 13.7: *Mã kịch bản* ***view_print.php*** *hiển thị chi tiết cho một ấn phẩm cụ thể và đưa vào một liên kết để đưa sản phẩm đó vào giỏ mua hàng.*

```
<?php # Doan ma 13.7 - view_print.php
if (is_numeric ($_GET['pid'])) {
   require_once ('../mysql_connect.php');
   $query = "SELECT * FROM artists, prints WHERE
   → artists.artist_id = prints.artist_id AND
   → prints.print_id = {$_GET['pid']}";
   $result = mysql_query ($query);
   $row = mysql_fetch_array ($result, MYSQL_ASSOC);
   mysql_close();
   $page_title = $row['print_name'];
   include_once ('includes/header.html');
   echo "<div align=\"center\">
   → <b>{$row['print_name']}</b> cua
   → {$row['first_name']} {$row['middle_name']}
   → {$row['last_name']}
   → <br />{$row['size']}
   → <br />\${$row['price']}
   → <a href=\"add_cart.php?pid={$row['print_id']}\">Dua vao
   → gio hang</a></div><br />";
   if ($image = @getimagesize
   → ("../uploads/{$row['image_name']}")) {
      echo "<div align=\"center\"><img
      → src=\"../uploads/{$row['image_name']}\" $image[3]
      → alt=\"{$row['print_name']}\" />";
   } else {
      echo "<div align=\"center\">Khong co hinh anh nao.";
   }
   echo '<br />' . stripslashes($row['description']) .
   → '</div>';
   include_once ('includes/footer.html');
} else {
   header ("Location:  http://" . $_SERVER['HTTP_HOST'] .
   → dirname($_SERVER['PHP_SELF']) . "/index.php");
   exit();
}
?>
```

Nhiều site thương mại điện tử dùng một hình ảnh thay cho liên kết "Dua vao gio hang". Nếu muốn thực hiện điều này, bạn cần thay văn bản "Dua vao gio hang" (trong thẻ liên kết <a>) bằng mã lệnh của hình ảnh được sử dụng.

Nếu muốn thể hiện hiệu lực của sản phẩm, bạn nên bổ sung trường quantity (số lượng) hoặc availability (có hiệu lực) vào bảng prints (hoặc các sản phẩm).

Tùy thuộc thiết lập Magic Quotes của cài đặt PHP, bạn có thể dùng hoặc không dùng hàm `stripslashes()` *khi hiển thị thông tin ấn phẩm trong đoạn mã này và đoạn mã trước.*

Giỏ mua hàng

Sau khi tạo danh mục ấn phẩm, việc tạo giỏ mua hàng rất đơn giản. Phương pháp chúng ta dùng trong ví dụ này là ghi ID và số lượng sản phẩm vào trong session. Với hai thông tin này, các mã kịch bản sẽ tính được tổng số tiền và các yêu cầu khác.

Quá trình thanh toán

Quá trình thanh toán gồm bốn bước:

1. Xác nhận lệnh đặt hàng.

2. Xác nhận, gửi thông tin thanh toán và chuyển hàng.

3. Xử lý lệnh đặt hàng.

4. Đưa lệnh đặt hàng vào cơ sở dữ liệu.

Cho đến giờ, bước 1 và 2 khá dễ dàng để các nhà lập trình trung bình tự hoàn thành. Trong nhiều trường hợp, phần lớn dữ liệu trong bước 2 sẽ được lấy từ bảng customers, sau khi người dùng đăng ký và đăng nhập.

Bước 3 là bước phức tạp nhất và chưa được đề cập đầy đủ trong bất cứ cuốn sách nào. Chi tiết của bước này phụ thuộc nhiều vào cách thức và người xử lý thanh toán, các luật qui định khác nhau dựa trên sản phẩm đang bán được vận chuyển đến sau hay được giao ngay (như truy cập một Website hoặc một tập tin có khả năng tải xuống)...

Hầu hết các site thương mại điện tử từ cỡ nhỏ đến trung bình dùng tổ chức thứ ba (third-party organization) để xử lý giao dịch tài chính. Điều này thường liên quan đến việc gửi thông tin thanh toán, lệnh đặt hàng và mã cửa hàng (tham chiếu bản thân site thương mại điện tử) đến một Website khác. Site này xử lý quá trình thanh toán, ghi nợ khách hàng và vào sổ. Sau đó, một mã lệnh kết quả được gửi trở lại site thương mại điện tử để có xử lý thích hợp tiếp theo. Trong những trường hợp như vậy,

tổ chức thứ ba xử lý thanh toán sẽ cung cấp cho nhà phát triển mã lệnh và những hướng dẫn thích hợp để giao tiếp với các hệ thống của họ.

*Hai ví dụ tiếp theo sẽ cung cấp toàn bộ chức năng cần thiết cho giỏ mua hàng. Đoạn mã thứ nhất (**add_cart.php**) sẽ đưa các ấn phẩm vào giỏ mua hàng. Đoạn mã thứ hai (**view_cart.php**) sẽ hiển thị nội dung giỏ hàng và cho phép khách hàng cập nhật nó.*

PHP có khả năng làm việc trực tiếp với một số hệ xử lý thẻ tín dụng thông thường (ví dụ CCVS). Xem các tài liệu liên quan để biết thêm thông tin.

Một tìm kiếm nhanh trên Internet sẽ giúp bạn có được những ví dụ và bài học đa dạng để tạo các ứng dụng thương mại điện tử với PHP.

Bổ sung mặt hàng

Đoạn mã **add_cart.php** nhận vào một đối số (ID ấn phẩm được chọn) và dùng nó để cập nhật giỏ hàng. Bản thân giỏ hàng là một biến session, nghĩa là nó được truy cập thông qua **$_SESSION['cart']** (hoặc **$HTTP_SESSION_VARS['cart']** trong những phiên bản cũ của PHP). Giỏ hàng là một mảng có khóa là ID của ấn phẩm và giá trị là số lượng.

Thực hành tạo tập tin **add_cart.php** theo các bước sau:

1. Tạo hồ sơ PHP mới trong trình soạn thảo văn bản.

```
<?php # Doan ma 13.8 - add_cart.php
```

2. Kiểm tra ấn phẩm có được chọn không và đưa vào phần đầu trang.

```
if (is_numeric ($_GET['pid'])) {
   $pid = $_GET['pid'];
   $page_title = 'Dua vao gio hang';
   include_once ('includes/header.html');
```

Cũng như đoạn mã **view_print.php**, mã kịch bản này không thể tiếp tục nếu không nhận được ID của ấn phẩm.

3. Kiểm tra xem ấn phẩm này đã được chọn chưa.

```
if (isset ($_SESSION['cart'][$pid])) {
   $qty = $_SESSION['cart'][$pid] + 1;
} else {
   $qty = 1;
}
```

Trước khi đưa ấn phẩm vào giỏ mua hàng (bằng cách thiết lập số lượng của nó là 1), chúng ta cần kiểm tra xem nó đã có trong giỏ chưa. Ví dụ, nếu khách hàng đã chọn ấn phẩm #519 và sau đó quyết định đặt mua thêm ấn phẩm này, thì giỏ hàng chứa hai ấn phẩm #519. Do đó, đầu tiên chúng ta cần kiểm tra xem giỏ

hàng có chứa ấn phẩm được chọn không. Nếu có, số lượng của nó sẽ được tăng lên 1. Nếu không, số lượng được dùng là 1.

4. Cập nhật giỏ hàng, hiển thị một thông báo và đưa vào cuối trang HTML.

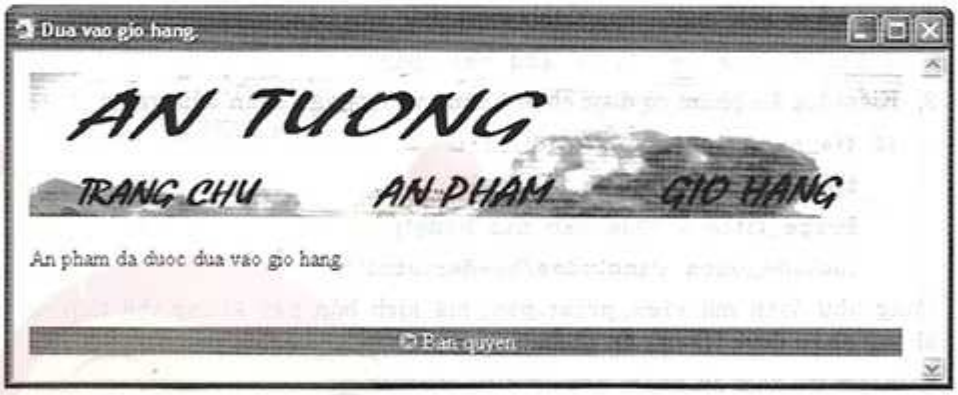
```
$_SESSION['cart'][$pid] = $qty;
echo '<p>An pham da duoc dua vao gio hang.</p>';
include_once ('includes/footer.html');
```

Để cập nhật giỏ hàng, chúng ta thiết lập một giá trị với **$qty** cho ID của ấn phẩm hiện có.

5. Hoàn tất phần điều kiện chính và trang PHP.

```
} else {
   header ("Location:  http://" . $_SERVER['HTTP_HOST'] .
   → dirname($_SERVER['PHP_SELF']) . "/index.php");
   exit();
}
?>
```

Lưu tập tin với tên **add_cart.php**, tải lên máy chủ và chạy thử trong trình duyệt Web (xem hình 13-27). Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 13.8.



Hình 13-27: *Kết quả sau khi bổ sung ấn phẩm vào giỏ mua hàng.*

Đoạn mã 13.8: *Đoạn mã* ***add_cart.php*** *đưa ấn phẩm vào giỏ mua hàng bằng cách tham chiếu ID của nó.*

```
<?php # Doan ma 13.8 - add_cart.php
if (is_numeric ($_GET['pid'])) {
   $pid = $_GET['pid'];
   $page_title = 'Dua vao gio hang.';
   include_once ('includes/header.html');
```

```
    if (isset ($_SESSION['cart'][$pid])) {
        $qty = $_SESSION['cart'][$pid] + 1;
    } else {
        $qty = 1;
    }
    $_SESSION['cart'][$pid] = $qty;
    echo '<p>An pham da duoc dua vao gio hang.</p>';
    include_once ('includes/footer.html');
} else {
    header ("Location:  http://" . $_SERVER['HTTP_HOST'] .
    → dirname($_SERVER['PHP_SELF']) . "/index.php");
    exit();
}
?>
```

Nếu muốn hiển thị nội dung giỏ hàng sau khi được bổ sung, bạn có thể kết hợp mã kịch bản trên với đoạn mã ***view_cart.php*** *(xem đoạn mã 13.9).*

Tương tự, bạn có thể dễ dàng sao chép kỹ thuật dùng trong đoạn mã ***view_print.php*** *vào mã kịch bản trên để hiển thị chi tiết sản phẩm mới bổ sung.*

Giá sản phẩm sẽ thay đổi theo thời gian, nên có thể được bổ sung vào giỏ hàng bằng cách dùng ***$_SESSION['cart'] = array ('Qty' => $qty, 'Price' => $price);***

Xem giỏ mua hàng

Đoạn mã **view_cart.php** phức tạp hơn **add_cart.php** vì nó phục vụ hai mục đích. Thứ nhất, nó hiển thị nội dung chi tiết giỏ hàng. Thứ hai, nó cho phép khách hàng cập nhật bằng cách thay đổi số lượng ấn phẩm có trong giỏ hàng (hoặc xóa một mặt hàng bằng cách tạo số lượng của nó là **0**). Để hoàn thành cả hai vai trò, chúng ta hiển thị nội dung giỏ hàng dưới dạng một biểu mẫu và thiết lập thuộc tính action để khi gửi sẽ quay lại chính nó. Trang này cũng liên kết với mã kịch bản **checkout.php** để chuyển sang quá trình thanh toán.

Thực hành tạo tập tin **view_cart.php** theo các bước sau:

1. Tạo hồ sơ PHP mới trong trình soạn thảo văn bản.

```
<?php # Doan ma 13.9 - view_cart.php

$page_title = 'Xem gio hang';

include_once ('includes/header.html');
```

2. Cập nhật giỏ hàng nếu biểu mẫu đã được gửi.

```
if (isset ($_POST['submit'])) {
   foreach ($_POST['qty'] as $key => $value) {
      if (($value == 0) AND (is_numeric ($value)) ) {
         unset ($_SESSION['cart'][$key]);
      } elseif ( is_numeric ($value) AND ($value > 0) ) {
         $_SESSION['cart'][$key] = $value;
      }
   }
}
```

Nếu biểu mẫu được gửi, mã lệnh cần cập nhật giỏ hàng để phản ánh số lượng nhập. Các giá trị này sẽ được đưa vào mảng **$qty** (có thể truy cập thông qua siêu toàn cục **$_POST**) với chỉ mục là ID của ấn phẩm và giá trị là số lượng mới (xem hình 13-28 để biết mã nguồn HTML của biểu mẫu). Nếu số lượng mới là **0**, mặt hàng đó được loại bỏ bằng cách xóa thiết lập. Nếu số lượng mới lớn hơn **0**, giỏ hàng sẽ được cập nhật để phản ánh điều này.

```
<form action="view_cart.php" method="post"> <tr>
    <td align="left">Nguyen Van A</td>
    <td align="left">An pham 1</td>
    <td align="right">$119.50</td>
    <td align="center"><input type="text" size="3" name="qty[1]" value="1" /></td>
    <td align="right">$119.50</td>
    </tr>
<tr>
    <td align="left">Le Thuy T</td>
    <td align="left">An pham 5</td>
    <td align="right">$150.00</td>
    <td align="center"><input type="text" size="3" name="qty[4]" value="1" /></td>
    <td align="right">$150.00</td>
    </tr>
<tr>
<td colspan="4" align="right"><b>Tong cong:<b></td>
<td align="right">$269.50</td>
</tr>
</table><div align="center"><input type="submit" name="submit" value="Cap nhat gio hang" /></form>
```

Hình 13-28: *Mã nguồn HTML của biểu mẫu.*

Nếu số lượng (được truy cập trong vòng lặp với tên **$value**) là số nhỏ hơn **0**, thì không có gì xảy ra với giỏ hàng. Điều này ngăn ngừa người dùng nhập số âm, tạo ra cán cân thanh toán âm và phải hoàn trả tiền cho họ.

3. Xác định giỏ hàng có trống hay không.

```
$empty = TRUE;
if (isset ($_SESSION['cart'])) {
   foreach ($_SESSION['cart'] as $key => $value) {
```

```
        if (isset($value)) {
            $empty = FALSE;
        }
    }
}
```

Vì nội dung giỏ hàng có thể bị thay đổi (nếu biểu mẫu đã được gửi), chúng ta cần kiểm tra nó có dữ liệu không trước khi hiển thị. Để làm điều này, ta kiểm tra xem biến `$_SESSION['cart']` đã được thiết lập chưa. Nếu nó đã được thiết lập, thì giỏ hàng mới được thêm vào hoặc loại bỏ ít nhất một ấn phẩm. Khi đó, mã lệnh sẽ duyệt qua giỏ hàng, kiểm tra số lượng từng mặt hàng. Nếu có ít nhất một mặt hàng có số lượng lớn hơn `0` thì giỏ hàng đã có hàng và biến `$empty` được thiết lập là FALSE.

4. Nếu có ấn phẩm trong giỏ hàng, chúng ta tạo câu truy vấn để hiển thị nội dung của nó.

```
if (!$empty) {
    require_once ('../mysql_connect.php');
    $query = 'SELECT * FROM artists, prints WHERE
    -> artists.artist_id = prints.artist_id AND
    -> prints.print_id IN (';
    foreach ($_SESSION['cart'] as $key => $value) {
        $query .= $key . ',';
    }
    $query = substr ($query, 0, -1) . ')
    -> ORDER BY artists.last_name ASC';
    $result = mysql_query ($query);
```

Câu truy vấn là một dạng liên kết đã được dùng nhiều lần trong chương này. Chỉ có một khác biệt nhỏ là sự bổ sung mệnh đề `IN`. Thay vì chỉ truy tìm thông tin cho một ấn phẩm (như trong mã kịch bản `view_print.php`), chúng ta truy tìm thông tin cho tất cả ấn phẩm trong giỏ hàng. Để thực hiện điều này, ta có thể dùng câu truy vấn `SELECT * ... IN (519, 42, 427)...` hoặc `SELECT * ... print_id=519 OR print_id=42 OR print_id=427...`

5. Tạo bảng và bắt đầu biểu mẫu HTML.

```
echo '<table border="0" width="90%" cellspacing="3"
-> cellpadding="3" align="center">
-> <tr>
-> <td align="left" width="30%"><b>Hoa sy</b></td>
```

```
→ <td align="left" width="30%"><b>Ten an pham</b></td>
→ <td align="right" width="10%"><b>Gia</b></td>
→ <td align="center" width="10%"><b>So luong</b></td>
→ <td align="right" width="10%"><b>Tong tien</b></td>
→ </tr>
→ <form action="view_cart.php" method="post">';
```

6. In ra các mẫu tin trả về (xem hình 13-29).

Hình 13-29: *Giỏ hàng hiển thị dưới dạng một biểu mẫu, trong đó số lượng ấn phẩm có thể được thay đổi.*

```
$total = 0;

while ($row = mysql_fetch_array ($result, MYSQL_ASSOC)) {

   $subtotal = $_SESSION['cart'][$row['print_id']] *
   → $row['price'];

   $total += $subtotal;

   echo "<tr>
   → <td align=\"left\">{$row['first_name']}
   → {$row['middle_name']} {$row['last_name']}</td>
   → <td align=\"left\">{$row['print_name']}</td>
   → <td align=\"right\">\${$row['price']}</td>
```

```
    → <td align=\"center\"><input type=\"text\" size=\"3\"
    → name=\"qty[{$row['print_id']}]\"
    → value=\"{$_SESSION['cart'][$row['print_id']]}\" /></td>
    → <td align=\"right\">$" . number_format ($subtotal, 2) .
    → "</td></tr>\n";
}
```

Khi hiển thị giỏ hàng, chúng ta cũng tính tổng giá trị đơn hàng. Biến **$total** được dùng cho mục đích này. Trong vòng lặp, với mỗi ấn phẩm trả về, ta nhân giá của nó với số lượng trong giỏ hàng để xác định số tiền và cộng dồn giá trị này vào biến **$total**. Đồng thời, mỗi mẩu tin cũng được in ra như một hàng trong bảng, với số lượng hiện tại được hiển thị trong ô nhập văn bản.

7. Hoàn tất bảng và biểu mẫu.

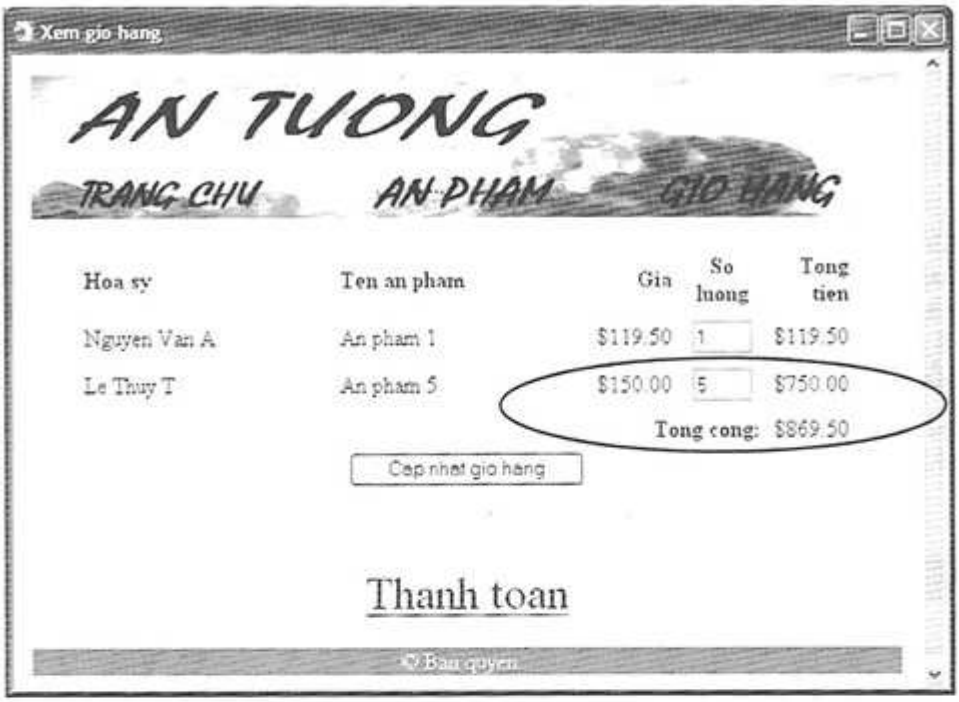
```
echo '  <tr>
→ <td colspan="4" align="right"><b>Tong cong:<b></td>
→ <td align="right">$' . number_format ($total, 2) .
→ '</td></tr>
→ </table><div align="center"><input type="submit"
→ name="submit" value="Cap nhat gio hang" /></form><br/>
→ <br/><a href="checkout.php"><font size="+3">Thanh
→ toan</font></a></div>';

mysql_close();
```

Tổng giá trị đơn hàng được hiển thị trong hàng cuối của bảng sau khi được định dạng bằng hàm **number_format()**.

8. Kết thúc phần điều kiện chính và trang PHP.

```
} else {
   echo '<p>Gio hang cua ban hien dang trong.</p>';
}
include_once ('includes/footer.html');
?>
```

9. Lưu tập tin với tên **view_cart.php**, tải lên máy chủ Web và chạy thử trong trình duyệt (xem hình 13-30 và 13-31). Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã 13.9.



Hình 13-30: *Nếu thay đổi số lượng của một ấn phẩm và nhấp vào liên kết "Cap nhat gio hang", giỏ hàng và tổng giá trị đơn hàng sẽ được cập nhật (so sánh với hình 13-29).*

Hình 13-31: *Loại bỏ tất cả những gì có trong giỏ hàng bằng cách thiết lập số lượng là 0.*

Đoạn mã 13.9: *Mã kịch bản* ***view_cart.php*** *hiển thị nội dung của giỏ hàng và cho phép người dùng cập nhật nó.*

```
<?php # Doan ma 13.9 - view_cart.php
$page_title = 'Xem gio hang';
include_once ('includes/header.html');
if (isset ($_POST['submit'])) {
   foreach ($_POST['qty'] as $key => $value) {
      if ( ($value == 0) AND (is_numeric ($value)) ) {
         unset ($_SESSION['cart'][$key]);
      } elseif ( is_numeric ($value) AND ($value > 0) ) {
         $_SESSION['cart'][$key] = $value;
      }
   }
}
$empty = TRUE;
if (isset ($_SESSION['cart'])) {
   foreach ($_SESSION['cart'] as $key => $value) {
      if (isset($value)) {
         $empty = FALSE;
      }
   }
}
if (!$empty) {
   require_once ('../mysql_connect.php');
   $query = 'SELECT * FROM artists, prints WHERE
   -> artists.artist_id = prints.artist_id AND
   -> prints.print_id IN (';
   foreach ($_SESSION['cart'] as $key => $value) {
      $query .= $key . ',';
   }
   $query = substr ($query, 0, -1) . ') ORDER BY
   -> artists.last_name ASC';
   $result = mysql_query ($query);
   echo '<table border="0" width="90%" cellspacing="3"
   -> cellpadding="3" align="center">
   -> <tr>
   -> <td align="left" width="30%"><b>Hoa sy</b></td>
   -> <td align="left" width="30%"><b>Ten an pham</b></td>
   -> <td align="right" width="10%"><b>Gia</b></td>
   -> <td align="center" width="10%"><b>So luong</b></td>
   -> <td align="right" width="10%"><b>Tong tien</b></td>
   -> </tr>
```

```
   <form action="view_cart.php" method="post">';
   $total = 0;
   while ($row = mysql_fetch_array ($result, MYSQL_ASSOC)) {
      $subtotal = $_SESSION['cart'][$row['print_id']] *
      → $row['price'];
      $total += $subtotal;
      echo "  <tr>
      → <td align=\"left\">{$row['first_name']}
      → {$row['middle_name']} {$row['last_name']}</td>
      → <td align=\"left\">{$row['print_name']}</td>
      → <td align=\"right\">\${$row['price']}</td>
      → <td align=\"center\"><input type=\"text\" size=\"3\"
      → name=\"qty[{$row['print_id']}]\"
      → value=\"{$_SESSION['cart'][$row['print_id']]}\"
      → /></td>
      → <td align=\"right\">$" . number_format ($subtotal, 2)
      → . "</td></tr>\n";
   }
   echo '  <tr>
   → <td colspan="4" align="right"><b>Tong cong:<b></td>
   → <td align="right">$' . number_format ($total, 2) .
   → '</td></tr>
   → </table><div align="center"><input type="submit"
   → name="submit" value="Cap nhat gio hang" /></form><br/>
   → <br/><a href="checkout.php"><font size="+3">Thanh toan
   → </font></a></div>';
   mysql_close();
} else {
   echo '<p>Gio hang cua ban hien dang trong.</p>';
}
include_once ('includes/footer.html');
?>
```

Trong những ứng dụng thương mại điện tử phức tạp hơn, mọi người có khuynh hướng viết một hàm riêng để hiển thị nội dung giỏ hàng.

Nếu dùng phiên bản PHP cũ, bạn cần theo những hướng dẫn đã được đề cập trong cuốn sách để dùng ***$HTTP_SESSION_VARS*** *thay cho* ***$_SESSION*** *và* ***$HTTP_POST_VARS*** *thay cho* ***$_POST****.*

Điều chủ yếu về ứng dụng thương mại điện tử an toàn là phải xem dữ liệu được gửi và sử dụng như thế nào. Ví dụ, sẽ kém an toàn khi đặt giá sản phẩm trong URL vì đây là nơi dễ bị thay đổi.

Phụ lục A:
CÀI ĐẶT

Nếu phát triển ứng dụng Web trên máy chủ của mình, bạn cần cài đặt các phần mềm cần thiết. Nếu sử dụng dịch vụ ký gửi Web (Web hosting), mọi chuyện đã sẵn sàng để làm việc.

Như phần giới thiệu của cuốn sách đã đề cập, để phát triển ứng dụng, chúng ta cần ba yêu cầu kỹ thuật: MySQL (ứng dụng cơ sở dữ liệu), PHP (ngôn ngữ kịch bản) và máy chủ Web (thực hiện mã lệnh PHP). Phụ lục này sẽ đề cập việc cài đặt các công cụ này trên hệ điều hành Windows - môi trường mà hiện nay phần lớn mọi người sử dụng.

Cài đặt trên Windows

Người dùng Windows có nhiều lựa chọn để cài đặt các ứng dụng yêu cầu. Hệ điều hành cũng có máy chủ Web của riêng mình: IIS (Internet Information Services) và PWS (Personal Web Server). Bạn cũng có thể dùng Apache, Xitami và nhiều loại máy chủ Web khác nữa.

MySQL chạy trên phần lớn hệ điều hành Windows (đặc biệt là các hệ thống 32 bit, như Windows 95, 98, ME, NT, 2000 và XP). Trong phụ lục này, chúng ta sẽ cài đặt phiên bản mới nhất của MySQL, cùng với IIS và PHP, trên Windows XP.

Cài đặt MySQL trên Windows theo các bước sau:

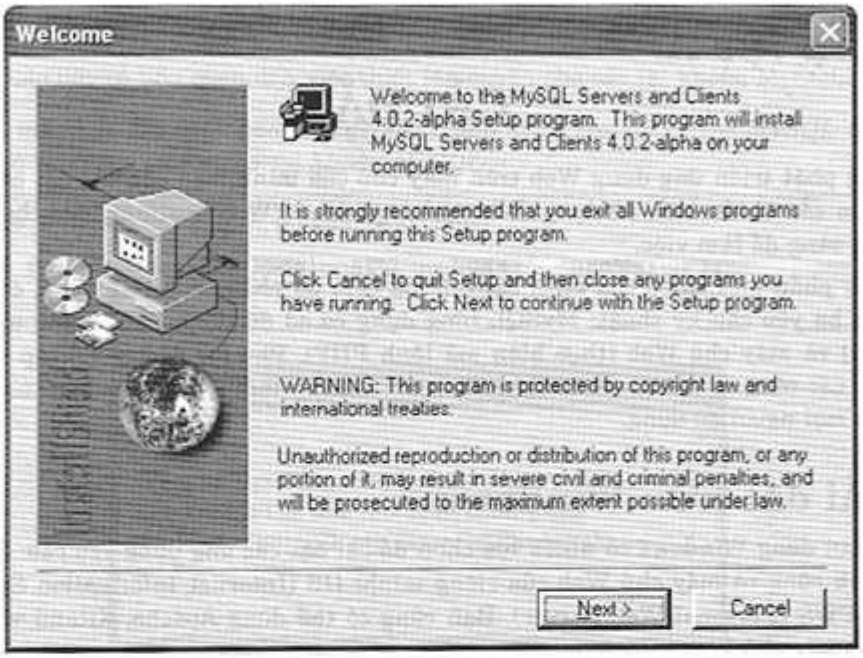
1. Tải xuống tập tin **zip** từ Website MySQL (xem hình A-1).

Hình A-1: *Khi làm việc với phiên bản 4.0 của MySQL, phần mềm có thể có ở hai dạng: dạng nén và dạng đóng gói.*

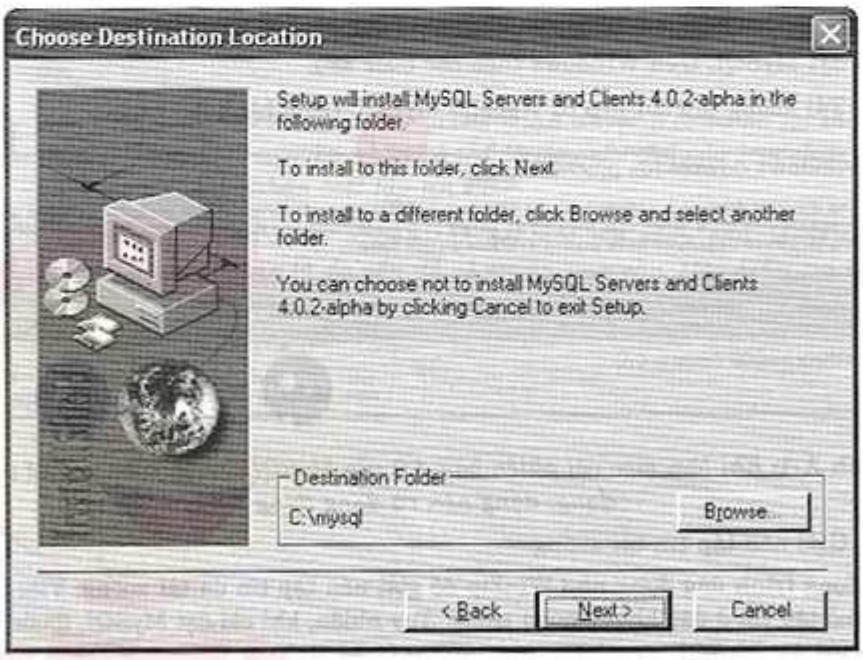
2. Giải nén tập tin tải xuống.

Sử dụng trình ứng dụng như WinZip để giải nén tập tin đã tải xuống. Việc chọn vị trí mở tập tin ở đâu không quan trọng. Tuy nhiên, khi cài đặt MySQL (bước 3), bạn nên cài đặt MySQL vào thư mục **C:\mysql** (mặc định).

3. Chạy ứng dụng *setup.exe* bằng cách nhấp đúp vào tập tin (xem hình A-2).



Hình A-2: *Bắt đầu quá trình cài đặt.*



Hình A-3: *Chọn thư mục để cài đặt MySQL.*

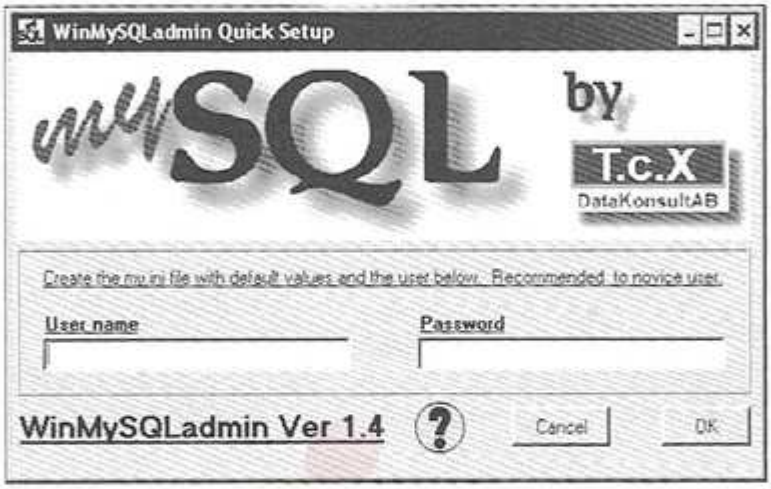
Ứng dụng sẽ hướng dẫn từng bước cài đặt. Bạn cần chọn thư mục (xem hình A-3) và dạng cài đặt (Typical, Compact hoặc Custom) khi được yêu cầu. Trong phần minh họa này chúng ta chọn dạng cài đặt Typical vào thư mục `C:\mysql`.

Nếu đang dùng phiên bản Windows NT, bạn phải đăng nhập với tư cách người quản trị với quyền cho phép cài đặt phần mềm.

4. Mở thư mục mới tạo `C:\mysql\bin` trong Windows Explorer.

Giả sử bạn chọn thư mục `C:\mysql` làm đích cài đặt cho phần mềm, khi đó tất cả các ứng dụng đi kèm MySQL sẽ được lưu trong thư mục con `bin`.

5. Chạy tiện ích WinMySQLadmin (xem hình A-4) bằng cách nhấp đúp vào tập tin.

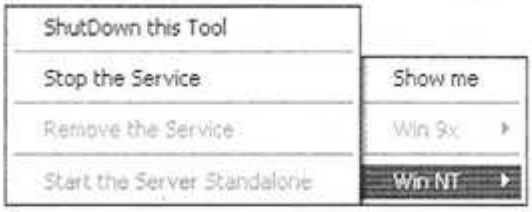


Hình A-4: *Trình ứng dụng WinMySQLadmin rất thích hợp để quản trị máy chủ MySQL trên Windows.*

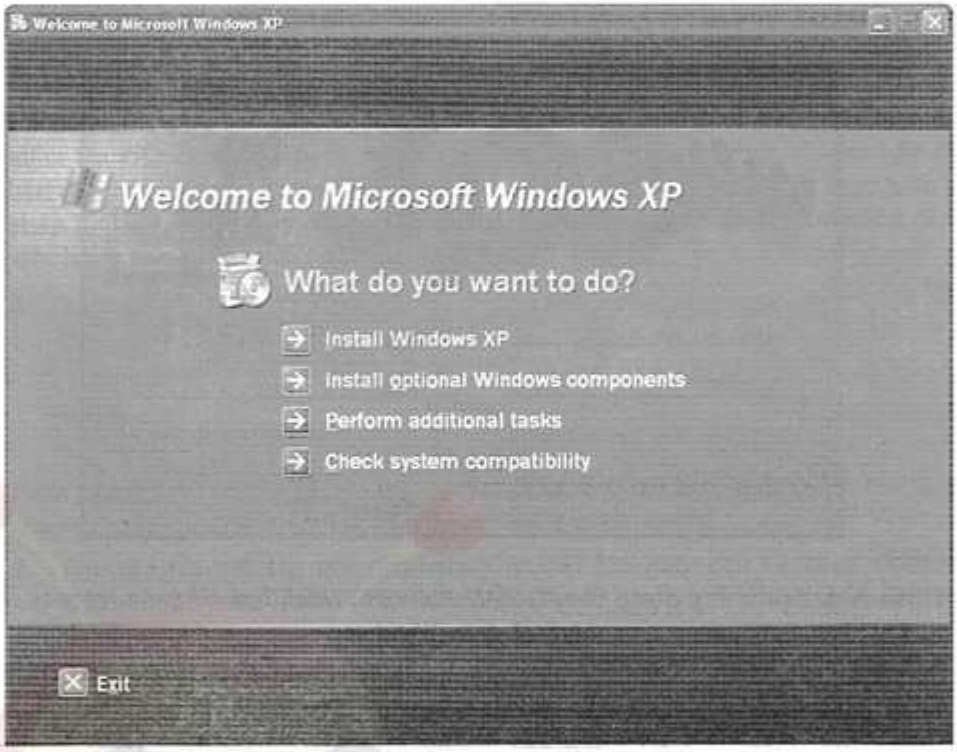
Ứng dụng WinMySQLadmin là giao diện quản lý chính của máy chủ MySQL cho người dùng Windows. Lần đầu sử dụng, bạn sẽ được nhắc để nhập tên và mật khẩu đăng nhập. Giá trị nhập vào sẽ đại diện cho người dùng quản lý trong MySQL: nó phải được ghi nhớ và bảo vệ kỹ.

Bước này cũng khởi động cho máy chủ MySQL và đặt biểu tượng đèn tín hiệu trên khay hệ thống để chỉ trạng thái của MySQL (đối với Windows 2000 và XP).

Công cụ WinMySQLadmin dùng để kiểm soát máy chủ MySQL. Nó được truy xuất bằng cách nhấp vào biểu tượng đèn tín hiệu trên khay hệ thống (xem hình A-5).



Hình A-5: *Biểu tượng đèn tín hiệu cung cấp lối tắt (shortcut) để kiểm soát MySQL và truy cập WinMySQLadmin.*



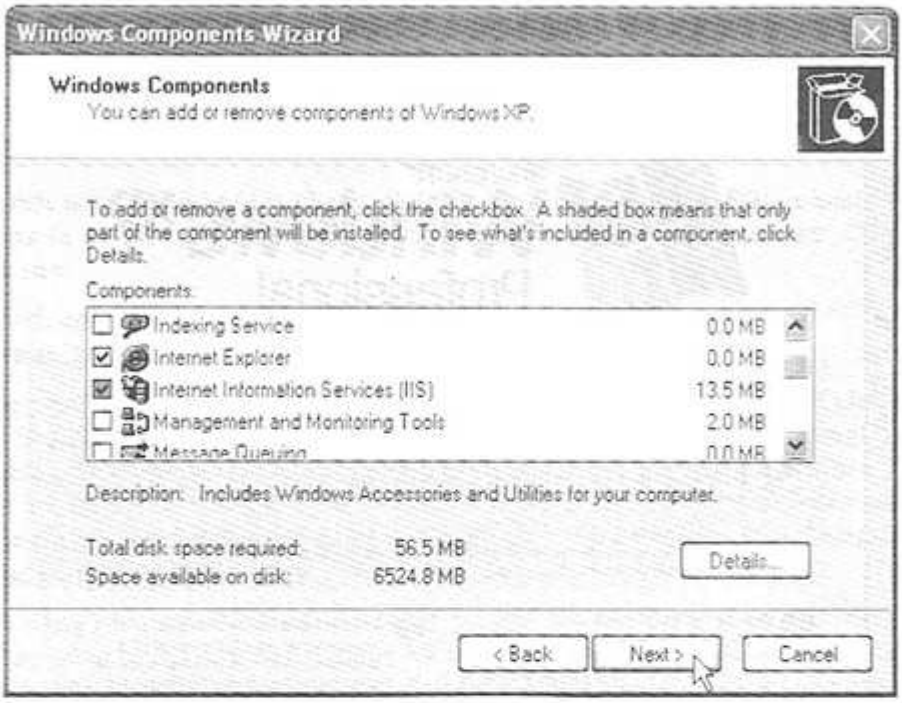
Hình A-6: *Menu chính của CD-ROM Windows XP.*

Tuy không bắt buộc, nhưng bạn nên cài đặt MySQL vào thư mục `C:\mysql`. *Nếu vì một nguyên nhân nào đó cần thay đổi, bạn vẫn có thể cài đặt phần mềm vào một vị trí khác bằng cách xác định một thư mục đó khi chương trình setup.exe yêu cầu.*

Cài đặt PHP và IIS theo các bước sau:

1. Đặt đĩa CD hệ điều hành vào ổ đĩa CD-ROM và nhấp vào mục *Install Optional Windows Components* (xem hình A-6). IIS có thể được cài đặt bằng cách sử dụng đĩa CD-ROM đi kèm với hệ điều hành.

2. Chọn Internet Information Services (IIS) để cài đặt và nhấp vào Next (xem hình A-7).



Hình A-7: *Windows Components Wizard cho phép chọn các thành phần bổ sung để cài đặt.*

3. Sau khi hoàn tất việc cài đặt, kiểm tra tình trạng làm việc của IIS bằng cách truy cập vào địa chỉ *http://localhost* trong trình duyệt (xem hình A-8).

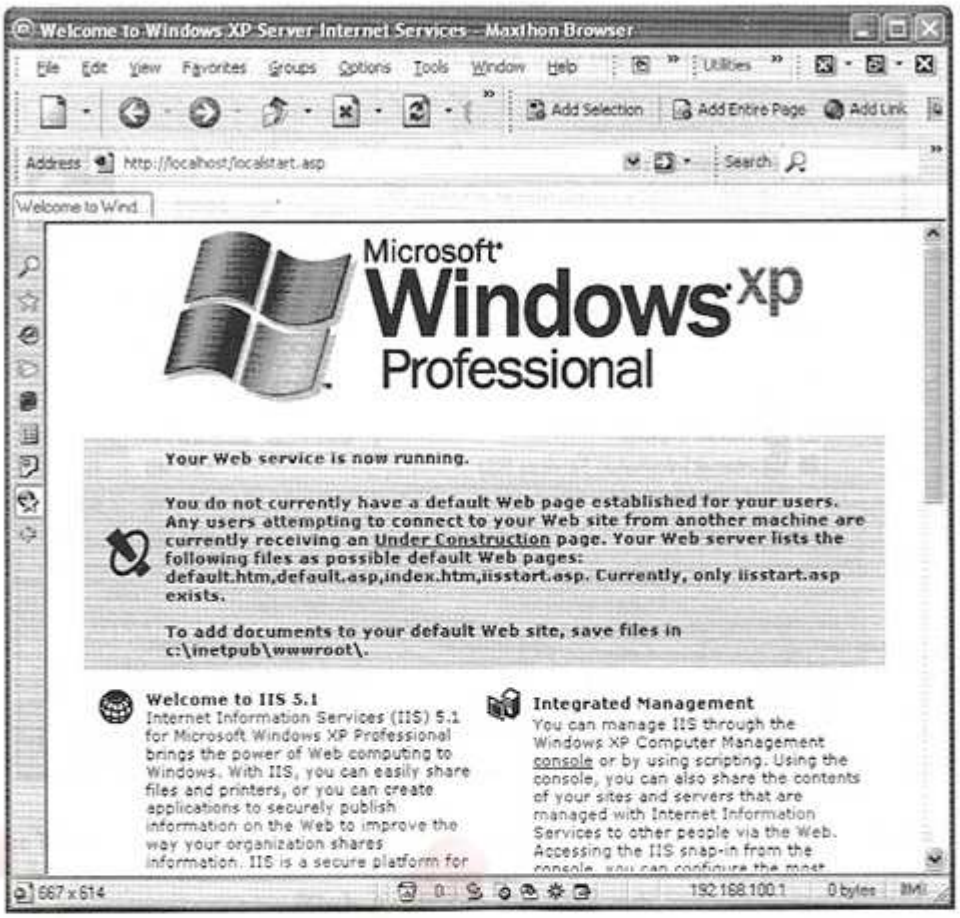
4. Dừng IIS (xem hình A-9).

Trước khi cài đặt PHP, bạn cần phải dừng máy chủ Web. Để thực hiện, hãy truy cập vào Start > Settings > Control Panel > Administrative Tools > Internet Services Manager. Bảng điều khiển Internet Information Services xuất hiện như hình A-9. Nhập tên máy tính của bạn vào cột bên trái, sau đó nhấp nút chuột phải vào Default Web Site trong cột bên phải. Chọn Stop từ menu hiện ra.

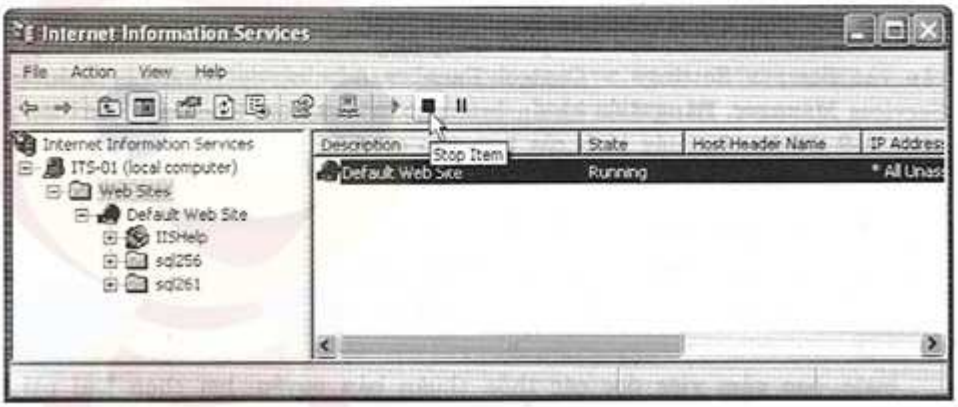
5. Tải xuống phiên bản mới nhất của PHP từ *www.php.net/downloads*.

Đối với hệ điều hành Windows, PHP có hai phiên bản. Bạn nên tải xuống phiên bản cài đặt, không tải xuống phiên bản đóng gói.

6. Giải nén PHP và chạy phiên bản cài đặt. Quá trình cài đặt PHP có nhiều bước, bao gồm việc đọc các thỏa thuận bản quyền, lựa chọn loại cài đặt (chúng tôi gợi ý chọn Standard) và chọn thư mục cài đặt (nên dùng `C:\php`).

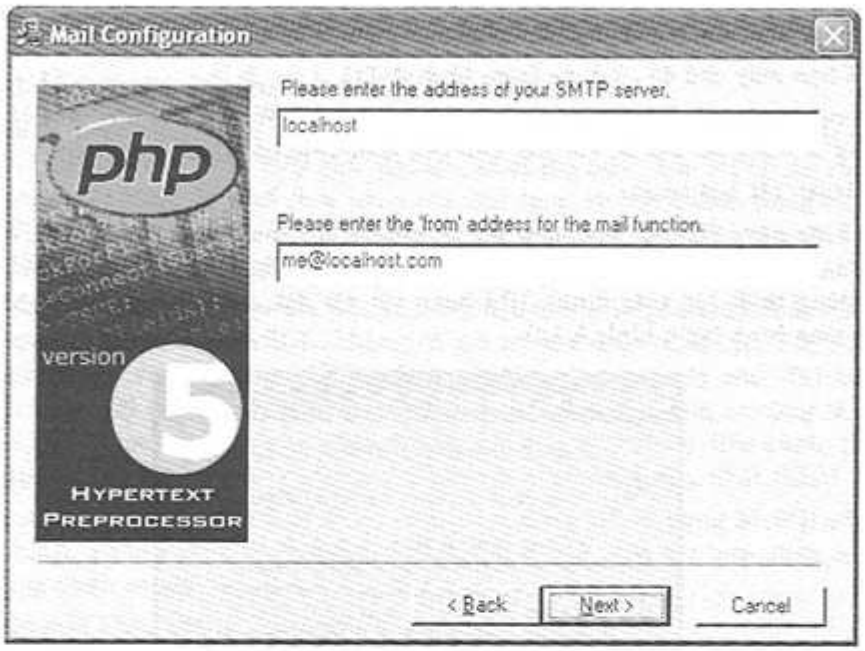


Hình A-8: *Bằng cách truy cập vào URL http://localhost trong trình duyệt Web, bạn có thể biết được tình trạng hoạt động của máy chủ.*

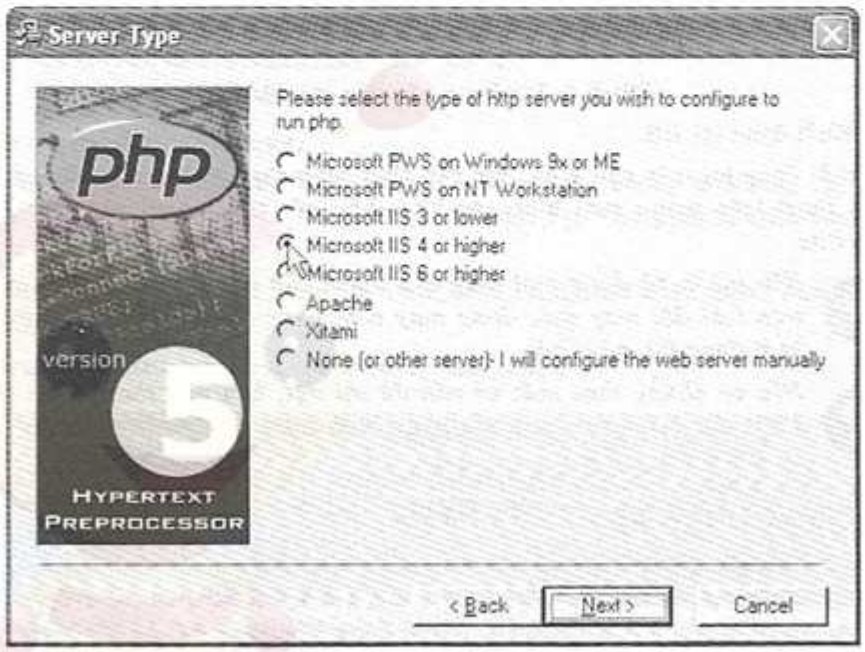


Hình A-9: *Bảng điều khiển Internet Information Services.*

7. Nhập vào thông tin SMTP (xem hình A-10).



Hình A-10: *Giá trị nhập vào ở đây có thể được thay đổi trong tập tin* `php.ini`.



Hình A-11: *Chương trình cài đặt PHP làm việc với tất cả các máy chủ phổ biến.*

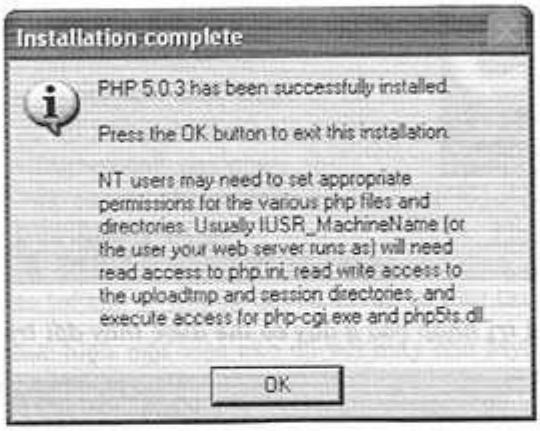
Nếu muốn truy cập vào máy chủ SMTP (Standard Mail Transfer Protocol), bạn cần nhập thông tin này ở đây. Ngược lại, chỉ cần nhập giá trị giả (trong trường hợp này, hàm ***mail()*** không làm việc).

8. Chọn máy chủ để sử dụng (xem hình A-11).

Chương trình cài đặt cho phép lựa chọn máy chủ Web được dùng với PHP. Vì chúng ta đang cài đặt nó với IIS, bạn cần chọn *Microsoft IIS 4 or higher.*

9. Hoàn tất quá trình.

Sau bước này, PHP sẽ sao chép tất cả những tập tin vào nơi cần thiết. Sau đó, chương trình sẽ yêu cầu bạn xác nhận lại việc thiết lập ánh xạ IIS Script (bạn có thể dùng thiết lập mặc định). Khi hoàn tất cài đặt, một thông báo sẽ hiển thị trên màn hình (xem hình A-12).



Hình A-12: *Hoàn tất việc cài đặt PHP.*

10. Khởi động lại IIS.

Để khởi động lại máy chủ Web, bạn cần thực hiện theo hướng dẫn của bước 4 và chọn Start trên menu xuất hiện sau khi nhấp nút chuột phải vào mục Default Web Site.

Nếu muốn sử dụng một máy chủ khác, bạn cần áp dụng theo các bước trên (cài đặt máy chủ, dừng máy chủ, chạy chương trình cài đặt PHP, khởi động lại máy chủ).

Nếu có những thắc mắc về vấn đề cài đặt, bạn nên tham khảo tài liệu PHP, đặc biệt chú trọng đến phần bình luận, góp ý của người dùng trên các diễn đàn.

Quyền truy xuất MySQL

Khi cài đặt xong MySQL, bạn nên thiết lập ngay mật khẩu cho tài khoản root. Chừng nào bạn còn chưa thực hiện điều này, bất kỳ ai cũng có thể truy cập cơ sở dữ liệu và có quyền ở mức quản trị. Người dùng Windows đã cài đặt MySQL theo các bước như hướng dẫn trên đây có thể bỏ qua giai đoạn này vì tài khoản root đã được tạo ra khi chạy ứng dụng WinMySQLadmin lần đầu tiên.

Khi đã thiết lập xong mật khẩu cho tài khoản root, bạn có thể thiết lập các tài khoản dùng để truy cập cơ sở dữ liệu (ví dụ từ mã kịch bản PHP). Việc dùng tài khoản root cho mục đích chung thường không an toàn.

Thiết lập mật khẩu cho tài khoản root

Tiện ích MySQLadmin được dùng để thực hiện các công việc quản lý trên cơ sở dữ liệu. Nó bao gồm việc dừng máy chủ MySQL, thiết lập mật khẩu tài khoản root và nhiều công việc khác. Một số chức năng của ứng dụng MySQLadmin trong hệ điều hành Window cũng có trong WinMySQLadmin. Hơn nữa, những công việc làm được với MySQLadmin thì cũng có thể làm trực tiếp hơn với trình quản lý mysql.

Một trong những ứng dụng đầu tiên của MySQLadmin là gán mật khẩu cho tài khoản root. Khi cài đặt MySQL, không có giá trị nào được thiết lập. Đây là một điều nguy hiểm và cần được giải quyết trước khi sử dụng máy chủ. Tài khoản MySQL khác với các tài khoản của hệ điều hành cho dù chúng có cùng tên. Vì vậy, tài khoản root của MySQL khác với tài khoản root của hệ điều hành. Nó có tính năng khác và mật khẩu cũng có thể khác (tuy nhiên không nhất thiết).

Điều quan trọng là máy chủ MySQL phải hoạt động để sử dụng MySQLadmin. Nếu MySQL không hoạt động, bạn cần khởi động nó theo các bước được hướng dẫn trong phần cài đặt.

Gán một mật khẩu cho tài khoản root theo các bước sau:.

1. Đăng nhập vào hệ thống và chuyển sang giao diện dòng lệnh.
2. Vào thư mục **mysql/bin** hoặc **mysql** (phụ thuộc vào hệ điều hành) bằng một trong hai cách sau:

```
cd /usr/local/mysql(Unix hay Mac OS X)
```

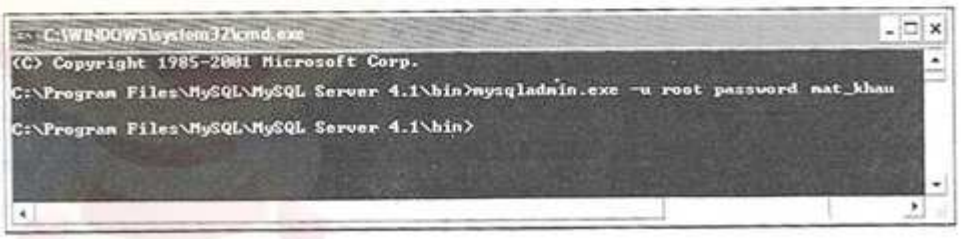
hoặc:

```
cd C:\mysql\bin(Windows)
```

Trên hầu hết các hệ điều hành (không phải Windows), chúng ta không thể truy cập tiện ích MySQLadmin trực tiếp được. Vì vậy, bạn cần phải chuyển vào một thư mục dưới nó.

3. Nhập vào câu lệnh sau, thay thế *mat_khau* bằng mật khẩu muốn dùng (xem hình A-13):

```
bin/mysqladmin -u root password 'mat_khau'
```



Hình A-13: *Thiết lập mật khẩu cho tài khoản root.*

Chú ý, mật khẩu trong MySQL có phân biệt dạng chữ, nên *Kazan* và *kazan* khác nhau. Từ khóa *password* đi trước chuỗi mật khẩu *'mat_khau'* để yêu cầu MySQL mã hóa chuỗi đó.

Một số người dùng Windows không dùng được dấu nháy khi tạo ra mật khẩu cho tài khoản root. Nếu gặp phải vấn đề như vậy, bạn có thể sử dụng lệnh sau:

```
mysqladmin - u root password mat_khau
```

Tạo tài khoản và phân quyền

Sau khi cài đặt, chạy chương trình và thiết lập mật khẩu cho tài khoản root thành công, bạn có thể bổ sung nhiều tài khoản khác. Để tăng tính an toàn cho cơ sở dữ liệu, chúng ta cần tạo những tài khoản mới để truy cập cơ sở dữ liệu thay vì phải tiếp tục dùng tài khoản root.

Hệ thống phân quyền MySQL được thiết kế để bảo đảm quyền cho một lệnh nào đó trên những cơ sở dữ liệu cụ thể. Chính nhờ công nghệ này mà một dịch vụ ký gửi Web (Web hosting) có thể cho phép nhiều người dùng cùng truy cập trên cơ sở dữ liệu mà không có vấn đề gì. Mỗi người dùng MySQL đều có những khả năng nhất định trên những cơ sở dữ liệu cụ thể. Tài khoản root của MySQL (không phải của các hệ thống khác) có toàn quyền và được dùng để tạo ra những tài khoản phụ khác (có thể có những quyền tương tự như tài khoản root).

Khi người dùng thực hiện một việc nào đó với máy chủ MySQL, trước tiên nó sẽ kiểm tra xem họ có được phép kết nối với máy chủ không (dựa vào tên đăng nhập, mật khẩu và các thông tin trong bảng user của cơ sở dữ liệu mysql). Tiếp theo, MySQL sẽ kiểm tra xem người sử dụng có quyền thực hiện câu lệnh SQL trên một cơ sở dữ liệu cụ thể không (như chọn dữ liệu, nhập dữ liệu hoặc tạo bảng mới). Để xác định được điều này, MySQL dùng các bảng *db, host, user, tables_priv* và *column_priv* trong cơ sở dữ liệu mysql. Bảng A.1 liệt kê các quyền khác nhau dùng để thiết lập quyền cho từng người sử dụng.

Bảng A.1: *Danh mục các quyền của người dùng.*

Quyền	Cho phép
SELECT	Đọc các mẩu tin từ các bảng.
INSERT	Thêm mẩu tin mới vào bảng.
UPDATE	Cập nhật dữ liệu hiện có trong bảng.
DELETE	Loại bỏ mẩu tin ra khỏi bảng.
INDEX	Tạo và loại bỏ các chỉ mục trong bảng.
ALTER	Thay đổi cấu trúc của bảng.
CREATE	Tạo bảng hoặc cơ sở dữ liệu mới.
DROP	Loại bỏ bảng hoặc cơ sở dữ liệu đang có.

RELOAD	Tải lại bảng phân quyền (và do đó tác động đến người dùng).
SHUTDOWN	Ngừng máy chủ MySQL.
PROCESS	Quan sát và dừng các tiến trình hiện có của MySQL.
FILE	Nhập dữ liệu vào bảng từ các tập tin văn bản.
GRANT	Tạo người sử dụng mới.
REVOKE	Xóa bỏ quyền của người dùng.

Có một số cách để tạo tài khoản và thiết lập quyền trong MySQL, nhưng ở đây chúng ta sẽ thao tác thủ công với trình quản lý mysql và lệnh **GRANT**. Cú pháp như sau:

```
GRANT <quyen> ON <csdl>.* TO <tai_khoan> IDENTIFIED BY
→ 'mat_khau' ;
```

Trong phần *quyền* của câu lệnh này, bạn có thể liệt kê các quyền theo như bảng A.1 hoặc dùng **ALL** để cho phép sử dụng tất cả các quyền (cẩn thận trọng). Phần ***<csdl>.**** của câu lệnh xác định bảng và cơ sở dữ liệu mà người dùng được phép làm việc. Bạn có thể xác định bảng cụ thể bằng cú pháp ***<csdl>.<tên bảng>*** hoặc cho phép với mọi cơ sở dữ liệu hiện có với ***.*** (cẩn hết sức thận trọng). Cuối cùng, bạn xác định tên đăng nhập và mật khẩu.

Tên đăng nhập có độ dài tối đa 16 ký tự. Khi tạo tên đăng nhập, cần đảm bảo không dùng khoảng trống (sử dụng ký tự gạch dưới để thay thế) và chú ý là tên đăng nhập phân biệt chữ hoa và chữ thường. Mật khẩu có độ dài không giới hạn và cũng phân biệt dạng chữ. Nó sẽ được mã hóa trong cơ sở dữ liệu mysql, nghĩa là không thể khôi phục trở lại dạng văn bản ban đầu. Việc bỏ qua mệnh đề **IDENTIFIED BY 'matkhau'** sẽ dẫn đến kết quả là người sử dụng không buộc phải nhập vào mật khẩu khi truy cập (đây là một điều nên tránh).

Cuối cùng, MySQL còn có một tùy chọn hạn chế người dùng theo tên máy chủ. Đây là tên của máy tính trên đó máy chủ MySQL đang chạy (phổ biến là *localhost*) hoặc là tên của máy tính mà từ đó người dùng truy cập vào máy chủ. Chúng ta có thể dùng địa chỉ IP thay cho tên. Để xác định tên máy chủ, bạn cần chuyển câu lệnh trên thành:

```
GRANT ALL ON <csdl>.* TO <tai_khoan>@<ten_may> IDENTIFIED BY
→ 'mat_khau';
```

Để cho phép truy xuất từ bất kỳ một máy nào, bạn nên sử dụng ký tự đại diện (%):

```
GRANT ALL ON <csdl>.* TO <tai_khoan>@'%' IDENTIFIED BY
→ 'mat_khau';
```

Chúng ta sẽ tạo ra hai tài khoản mới với các quyền cụ thể để minh họa cho những điều nêu trên.

Để tạo ra người sử dụng mới ta làm theo các bước sau:

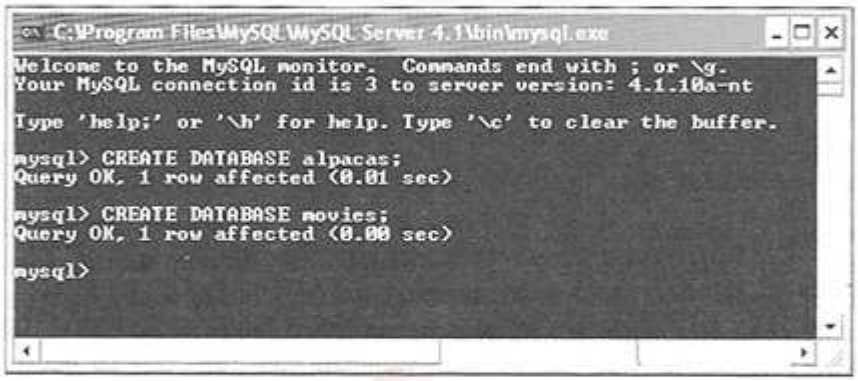
1. Đăng nhập vào hệ thống và chuyển sang giao diện dòng lệnh.
2. Vào thư mục ***mysql/bin*** hoặc ***mysql*** (tùy vào hệ điều hành).
3. Đăng nhập vào trình quản lý mysql:

```
mysql - u root -p
```

Bạn nên đăng nhập với tài khoản có khả năng tạo cơ sở dữ liệu và những tài khoản khác.

4. Tạo hai cơ sở dữ liệu mới (xem hình A-14):

```
CREATE DATABASE alpacas ;

CREATE DATABASE movies ;
```



Hình A-14: *Tạo hai bảng mới.*

Đây chỉ là hai cơ sở dữ liệu ví dụ. Chúng không được sử dụng trong các ví dụ của cuốn sách này.

5. Tạo tài khoản có quyền quản trị trên cơ sở dữ liệu *alpacas* (xem hình A-15):

```
GRANT SELECT,INSERT,UPDATE,DELETE,CREATE,DROP,ALTER,INDEX
→ ON alpacas.* TO llama@localhost IDENTIFIED BY 'camel';
```

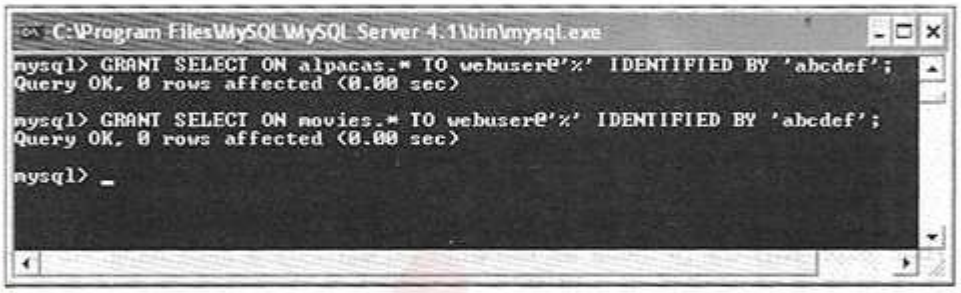
Hình A-15: *Tạo tài khoản có quyền quản trị cơ sở dữ liệu* alpacas.

Người sử dụng *llama* sẽ có khả năng tạo bảng, thay đổi bảng, nhập dữ liệu, cập nhật dữ liệu... trên cơ sở dữ liệu *alpacas*. Về thực chất, lệnh trên thiết lập tất cả quyền quản trị trừ quyền tạo tài khoản mới. Bạn nên thiết lập mật khẩu và xác định tên máy chủ cụ thể.

6. Tạo tài khoản có quyền duyệt cả hai cơ sở dữ liệu (xem hình A-16):

```
GRANT SELECT ON alpacas.* TO Webuser@% IDENTIFIED BY
→ 'BroWs1ng';

GRANT SELECT ON movies.* TO Webuser@% IDENTIFIED BY
→ 'BroWs1ng';
```

Giờ đây, tài khoản *Webuser* có thể duyệt qua các mẩu tin (`SELECT FROM ten_bang`) nhưng không chỉnh sửa được dữ liệu trong đó. Cách đơn giản hơn để tạo tài khoản cho mọi cơ sở dữ liệu là dùng mã lệnh **`GRANT SELECT ON *.* TO Webuser@'%' IDENTIFIED BY 'BroWs1ng'`**. Tuy nhiên, điều này cũng cho phép người dùng chọn từ cơ sở dữ liệu mysql, nên đây không phải là giải pháp tốt. Khi tạo tài khoản và quyền, bạn nên làm từ dưới lên và thiết quyền chặt chẽ dần.



Hình A-16: *Tạo tài khoản chung cho cả hai cơ sở dữ liệu.*

7. Áp dụng các thay đổi (xem hình A-17):

```
FLUSH PRIVILEGES;
```

Hình A-17: *Lệnh* **`Flush Privileges`** *phải được thực hiện để các thay đổi có tác dụng.*

Các thay đổi sẽ không có tác dụng nếu bạn chưa báo cho MySQL thiết lập lại danh mục tài khoản và quyền. Đây là điều mà lệnh trên sẽ thực hiện. Bạn cũng có thể thực hiện điều này với lệnh **`mysqladmin -u root -p reload`**. Lỗi

thường gặp là bỏ quên bước này và như vậy người dùng mới không thể truy cập vào cơ sở dữ liệu.

Mọi cơ sở dữ liệu có tên bắt đầu bằng test_ đều có thể được cập nhật bởi bất kỳ người sử dụng nào được phép kết nối vào MySQL. Vì vậy, cần cẩn thận để không tạo ra những cơ sở dữ liệu như vậy trừ khi nó được dùng cho mục đích thử nghiệm.

Có một cách thủ công để tạo tài khoản mới: thực hiện lệnh ***INSERT*** *với bảng user và các bảng khác của cơ sở dữ liệu mysql. Công việc này chỉ thích hợp với những người sử dụng có kinh nghiệm, hiểu biết về mối liên hệ giữa bảng user, db và các bảng khác của cơ sở dữ liệu mysql.*

Kiểm tra việc cài đặt

Để hoàn tất phụ lục này, chúng ta sẽ tạo hai mã kịch bản PHP với mục đích kiểm tra việc cài đặt. Nếu có một lỗi nào xảy ra trong quá trình, thì lúc này chắc bạn đã phải biết rồi. Tuy nhiên, các bước này sẽ cho phép chúng ta thực hiện các phép kiểm tra trên máy của mình (hoặc trên một máy chủ khác) trước khi đi vào việc lập trình PHP phức tạp hơn.

Kiểm tra PHP theo các bước sau:

1. Tạo ra hồ sơ PHP sau đây trong trình soạn thảo.

```
<?php
phpinfo()
?>
```

Hàm ***phpinfo()*** trả về thông tin cấu hình cài đặt PHP ở dạng bảng. Nó là một công cụ hoàn hảo để kiểm tra PHP làm việc tốt hay không.

2. Lưu tập tin với tên gọi **phpinfo.php**, tải lên máy chủ và chạy thử trong trình duyệt (xem hình A-18). Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã A.1.

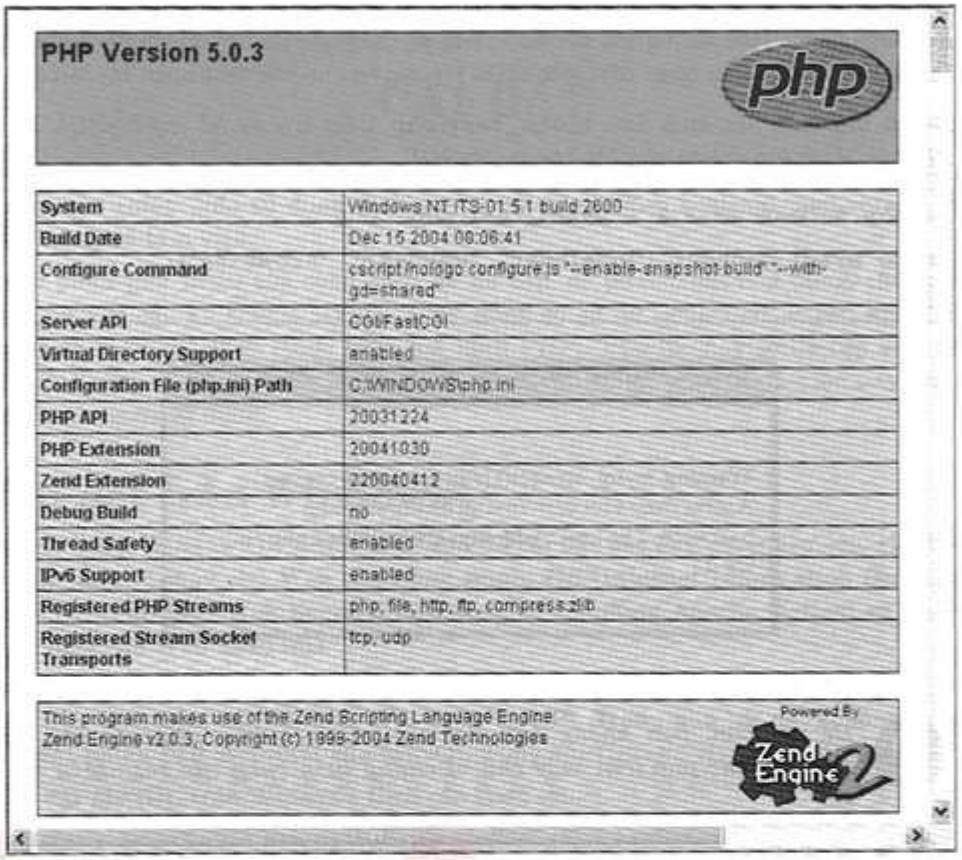
Đoạn mã A.1: *Mã kịch bản* ***phpinfo.php*** *để kiểm tra và thông báo việc cài đặt PHP.*

```
<?php # Doan ma A.1 - phpinfo.php

phpinfo();

?>
```

Kiểm tra PHP và MySQL theo các bước sau:

1. Tạo ra hồ sơ PHP mới trong trình soạn thảo.

```
<?php
echo mysql_connect('localhost','Webuser','BroWs1ng');
?>
```



PHP Version 5.0.3

System	Windows NT ITS-01 5.1 build 2600
Build Date	Dec 15 2004 08:06:41
Configure Command	cscript /nologo configure.js "--enable-snapshot-build" "--with-gd=shared"
Server API	CGI/FastCGI
Virtual Directory Support	enabled
Configuration File (php.ini) Path	C:\WINDOWS\php.ini
PHP API	20031224
PHP Extension	20041030
Zend Extension	220040412
Debug Build	no
Thread Safety	enabled
IPv6 Support	enabled
Registered PHP Streams	php, file, http, ftp, compress.zlib
Registered Stream Socket Transports	tcp, udp

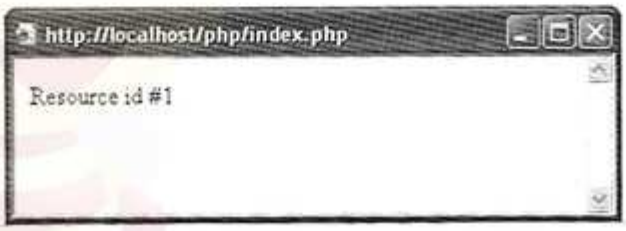
This program makes use of the Zend Scripting Language Engine
Zend Engine v2.0.3, Copyright (c) 1998-2004 Zend Technologies

Powered By Zend Engine 2

Hình A-18: *Thông tin về cấu hình PHP của máy chủ.*

Mã kịch bản này sẽ kết nối với máy chủ MySQL bằng tên đăng nhập và mật khẩu như đã thiết lập trong phần trước của phụ lục. Kết quả của việc kết nối này sẽ được thể hiện trong trình duyệt qua việc sử dụng câu lệnh ***echo()***.

2. Lưu tập tin với tên gọi **mysql_test.php**, tải lên trang chủ và chạy thử trong trình duyệt (xem hình A-19). Mã lệnh đầy đủ của tập tin được thể hiện trong đoạn mã A.2.

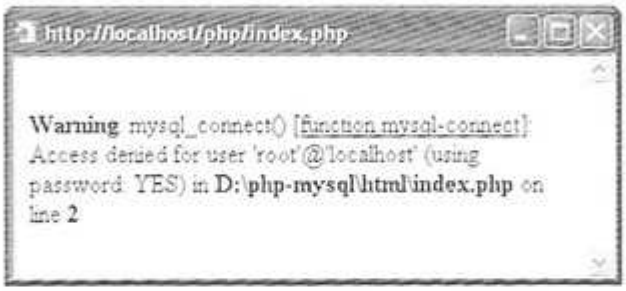


Hình A-19: *Mã kịch bản PHP có thể kết nối với máy chủ MySQL.*

Nếu kết nối được với cơ sở dữ liệu, mã kịch bản sẽ thể hiện thông báo có dạng như *Resource id #2*. Nếu không kết nối được, bạn sẽ thấy báo lỗi như hình A-20. Hầu hết thông báo này có liên quan đến việc thiết lập quyền cho các tài khoản MySQL.

Đoạn mã A.2: *Mã kịch bản* ***mysql_test.php*** *kiểm tra sự hỗ trợ MySQL trong PHP và thiết lập quyền cho tài khoản MySQL.*

```
<?php # Doan ma A.2 - mysql_test.php
echo mysql_connect ('localhost', 'Webuser', 'BroWs1ng');
?>
```



Hình A-20: *Mã kịch bản không thể kết nối vào máy chủ MySQL.*

Để an toàn, bạn không nên đặt các mã kịch bản ***phpinfo.php*** *hoặc* ***mysql_test.php*** *trên máy chủ vì chúng luôn cung cấp rất nhiều thông tin.*

Nếu chạy thử mã kịch bản PHP trong trình duyệt Web và nó lại cố gắng tải tập tin xuống, nguyên nhân có thể là do máy chủ của bạn chưa được thiết lập đúng để nhận biết tập tin PHP.

Nếu mã kịch bản PHP không thể kết nối với máy chủ MySQL, thì nguyên nhân thường là do vấn đề về quyền truy cập. Hãy kiểm tra kỹ lại tên đăng nhập, mật khẩu, tên máy đang dùng và đảm bảo đã thực hiện lệnh ***Flush Privileges****.*

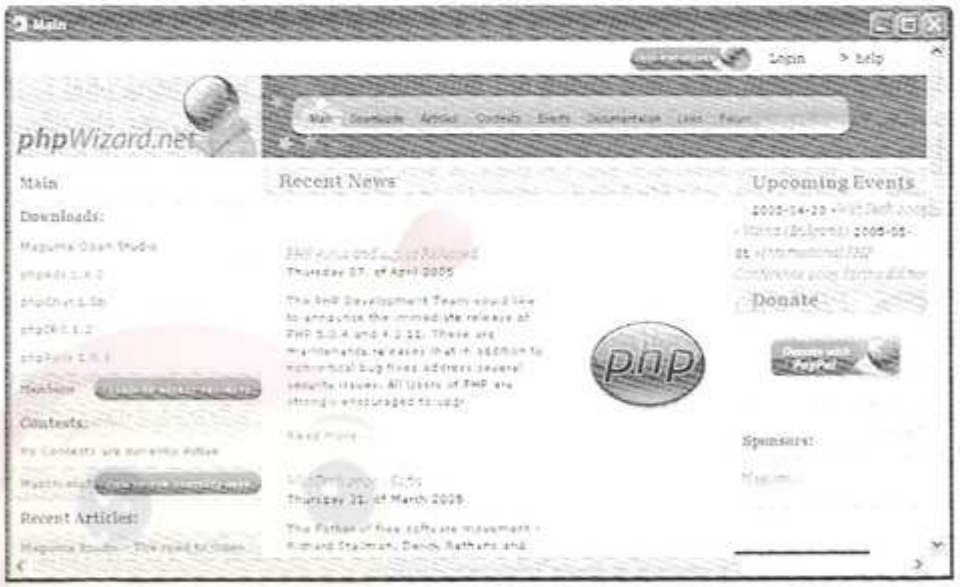
Phụ lục B:

CÁC ỨNG DỤNG NHÓM THỨ BA

Phụ lục này sẽ giới thiệu một số ứng dụng nhóm thứ ba (third-party application) phổ biến nhất cho PHP và MySQL. Chúng ta không làm lại những gì mà người đi trước đã thực hiện, nhưng đôi khi điều này lại cần để tích lũy kinh nghiệm. Trước khi đi vào phát triển những ứng dụng lớn hơn, bạn cần biết những ứng dụng có sẵn đã được viết rất tốt. Chúng có thể có phí hoặc miễn phí.

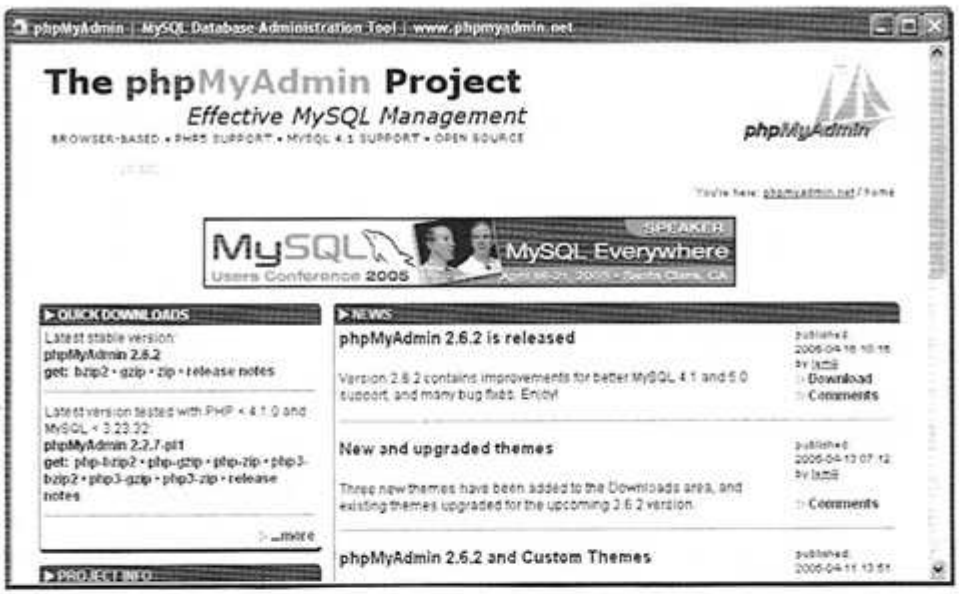
phpMyAdmin

Chúng ta đã nhiều lần đề cập tới phpMyAdmin vì nó là một trong những ứng dụng tốt nhất được viết trong PHP, cho phép tương tác với cơ sở dữ liệu MySQL. Mục đích chủ yếu của tiện ích này là tương tác với MySQL, cung cấp cho người dùng một hệ thống quản trị cơ sở dữ liệu MySQL trên nền Web.



Hình B-1: *Trong nhiều công cụ tạo ra bởi phpWizard, phpMyAdmin là phổ biến nhất.*

Phần mềm được hình thành bởi một số thành viên của phpWizard (địa chỉ *www.phpwizard.net*, xem hình B-1) và sau đó được chuyển sang hệ thống SourceForge (*www.sourceforge.net*). Phiên bản mới nhất có thể được tải xuống miễn phí tại Website *www.phpmyadmin.net* (xem hình B-2).



Hình B-2: *Trang chủ của phpMyAdmin (www.phpmyadmin.net).*

Hướng dẫn cài đặt có trong gói phần mềm và việc cài đặt khá đơn giản. Nếu cài đặt phpMyAdmin trên máy không nối mạng với máy chủ, bạn không phải lo lắng nhiều về vấn đề an toàn. Nếu có nối mạng, bạn cần tuân theo các chỉ dẫn bảo đảm an toàn vì ứng dụng này cung cấp truy cập mức quản trị đến cơ sở dữ liệu.

Nhóm tạo phpMyAdmin cũng là tác giả của tiện ích phpAds (hệ thống quảng cáo), phpPolls (hệ thống đánh giá) và các ứng dụng khác. Bạn có thể tải chúng miễn phí từ Website www.phpwizard.net.

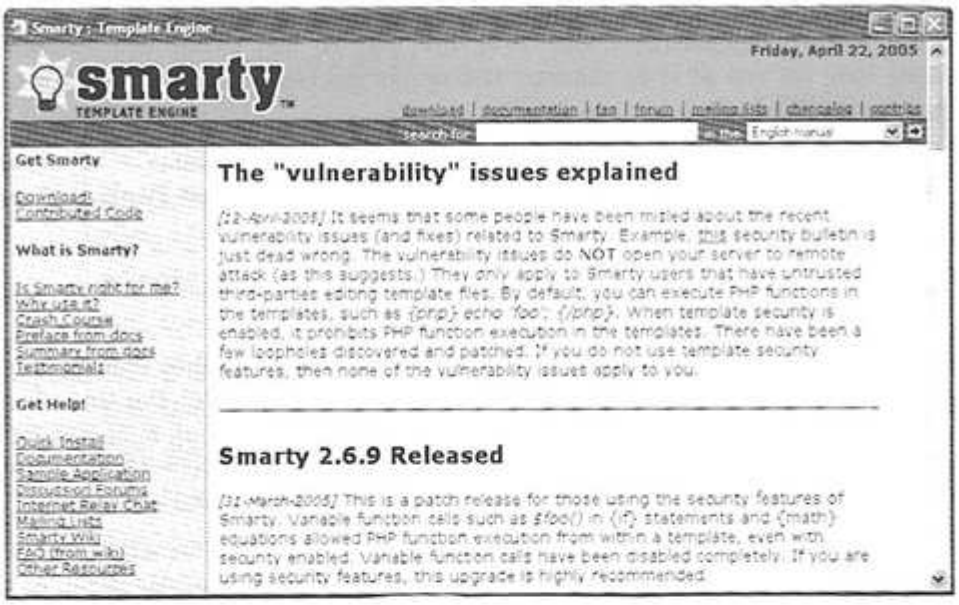
PHPMyEdit (www.phpmyedit.org/home.php) cũng là một công cụ hữu ích để quản trị MySQL.

MySTRI (www.burney.ws/software/mystri) là một phiên bản nâng cao của phpMyAdmin.

Các hệ thống khuôn mẫu

Cho đến nay, các nhà phát triển PHP đã sử dụng nhiều hệ thống khuôn mẫu để tách riêng mã lệnh PHP (phần lôgic của trang) ra khỏi thiết kế HTML (phần thể hiện). Hai phương pháp phổ biến là kèm theo tập tin (đã đề cập ở Chương 3 "Tạo Website động") và hệ thống lập trình hướng đối tượng (ví dụ Fast Template).

Hệ thống khuôn mẫu Smarty gần đây nhất đã có những cải tiến đáng kể. Nó có tùy chọn cho phép tạo hệ thống khuôn mẫu có thể biên dịch thành trang tĩnh. Nói cách khác, trang kết quả HTML chỉ được làm một lần và sau đó có thể được cập nhật dựa trên sự thay đổi của cơ sở dữ liệu. Một ưu điểm khác của việc ứng dụng hệ thống Smarty là nó được thiết kế để phần lôgic của PHP có thể được thay thế ngay trong khuôn mẫu.



Hình B-3: *Smarty là hệ thống khuôn mẫu mạnh đáng để quan tâm.*

Hình B-4: *TemplateTamer đưa vào môi trường phát triển tích hợp (IDE: Intergrated Development Environment) hỗ trợ cho việc phát sinh mã PHP.*

Ngày nay, Smarty là một phần chính thức của PHP. Nó có tại địa chỉ *http://smarty.php.net* (xem hình B-3). Site này có một số tài liệu về cách sử dụng cùng với một số ví dụ căn bản. Hướng dẫn chi tiết hơn có thể được tìm thấy tại địa chỉ Zend.com (*www.zend.com*).

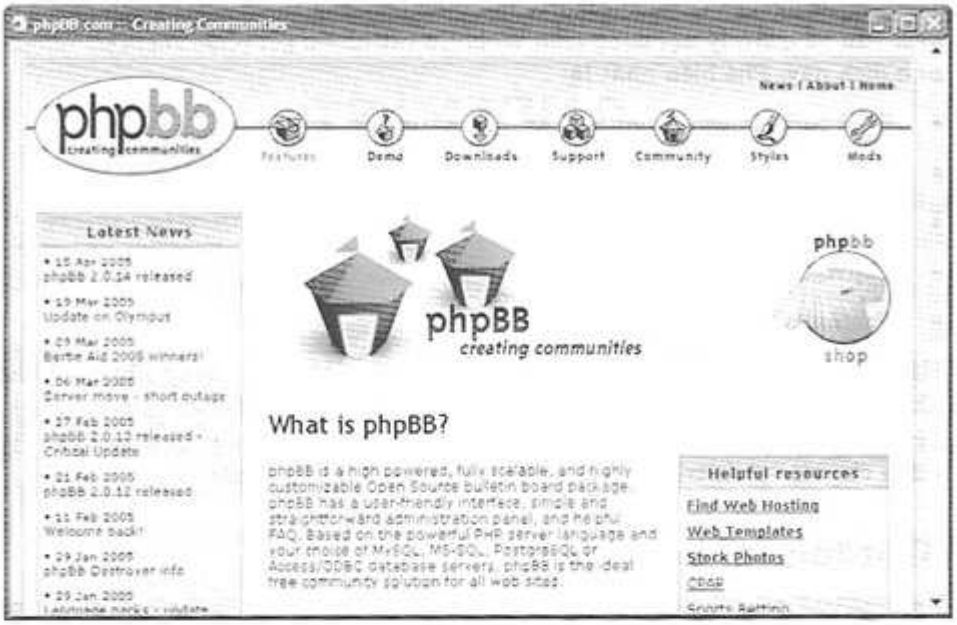
Các hệ thống khuôn mẫu phổ biến khác là EasyTemplate (www.onlinetools.org/tools/easytemplate/index.php) và TemplateTamer (www.templatetamer.com). Xem hình B-4.

Phần mềm diễn đàn

Một ví dụ thường thấy trong các sách của PHP và MySQL là diễn đàn trực tuyến, trong đó người dùng có thể tạo một mảng các chủ đề, trả lời thư tín... Ở đây, chúng ta không dùng các ví dụ này nữa vì đã có những ứng dụng miễn phí rất tốt cho vấn đề này. Hai phần mềm diễn đàn chính được chọn ở đây là Phorum (*http://phorum.org*) - (xem hình B-5) và phpBB (*www.phpbb.com*) - (xem hình B-6).

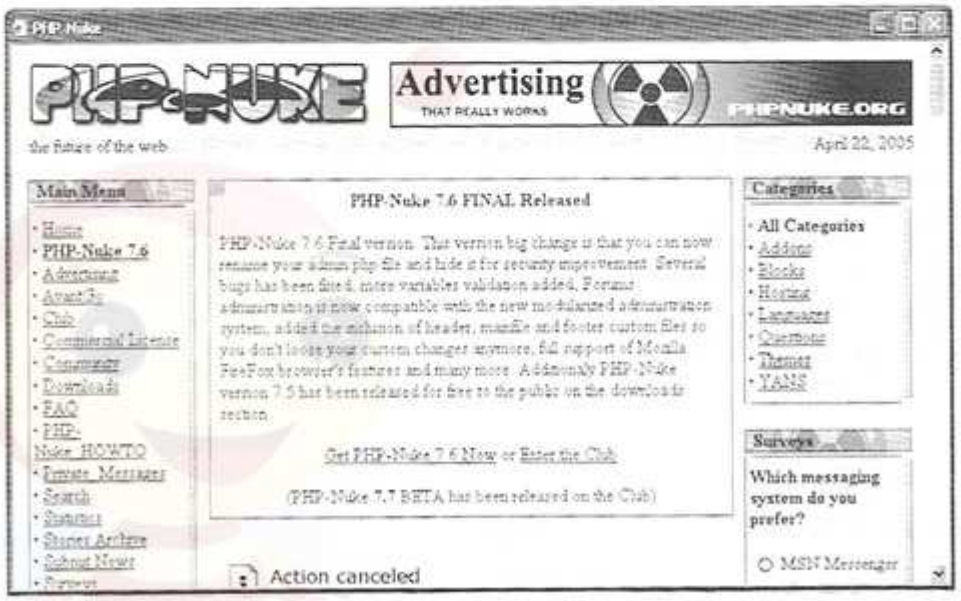
Cả hai phần mềm diễn đàn đều phát triển dựa trên MySQL. Chúng có khả năng tùy biến, mạnh và đáng tin cậy. Chúng khác nhau về một số đặc điểm, do vậy cần phân tích nhu cầu và xem bản minh họa trực tuyến trước khi chọn phần mềm nào để sử dụng.

Hình B-5: *Phần mềm diễn đàn phổ biến Phorum.*



Hình B-6: *Phần mềm diễn đàn phpBB.*

Quản lý nội dung



Hình B-7: *Hệ thống quản lý nội dung PHP-Nuke được phổ biến rộng rãi và có khả năng tùy biến cao.*

Trong Chương 11 "Ví dụ - Quản lý nội dung", chúng ta đã phát triển một số mã kịch bản về quản lý nội dung trên Web. Có rất nhiều gói phần mềm phù hợp cho mục đích này. Phổ biến nhất là:

- PHP-Nuke (*www.phpnuke.org*) - (xem hình B-7).
- myPHPNuke (*www.myphpnuke.com*).
- pMachine (*www.machine.com*).
- phpWebsite (*www.phpwebsite.appstate.edu*).
- Postnuke (*www.postnuke.com*).

Hầu hết các gói phần mềm này đều có hệ thống đăng ký, đăng nhập, hệ thống khuôn mẫu, khả năng tìm kiếm, quản trị và nhiều đặc điểm khác nữa. Đặc biệt, PHP-Nuke là một gói phần mềm được nhiều người sử dụng, có nhiều site dành riêng cho việc phát triển thành phần phụ và các môđun khác.

Thương mại điện tử

Hình B-8: *Nếu muốn có một site thương mại điện tử trực tuyến nhanh, bạn nên tham khảo phpShop.*

Trong Chương 13 "Ví dụ - Thương mại điện tử", chúng ta có để cập đến chức năng cơ bản của một site thương mại điện tử. Khi phát triển site của riêng mình, bạn có thể xây dựng dựa trên thông tin của Chương 13 hoặc dùng các ứng dụng có sẵn dưới đây:

- phpShop(*www.phpshop.org*) - (xem hình B-8): bao gồm phần đăng ký, quản lý người dùng, quản lý danh mục sản phẩm và các đặc điểm khác.
- osCommerce (*www.oscommerce.com*) là một ứng dụng thương mại điện tử mã nguồn mở rất phổ biến. Nó miễn phí, dễ dàng cài đặt, dễ sử dụng và được viết dành riêng cho MySQL.
- FishCart mới phổ biến trong thời gian gần đây. Nó được viết bằng PHP4, làm việc với cơ sở dữ liệu MySQL và PostgreSQL, cho phép dùng nhiều ngôn ngữ khác nhau và thậm chí gắn hệ thống theo dõi UPS trong phiên bản mới nhất. Trang chủ của FishCart là *www.fishcart.org* (xem hình B-9).

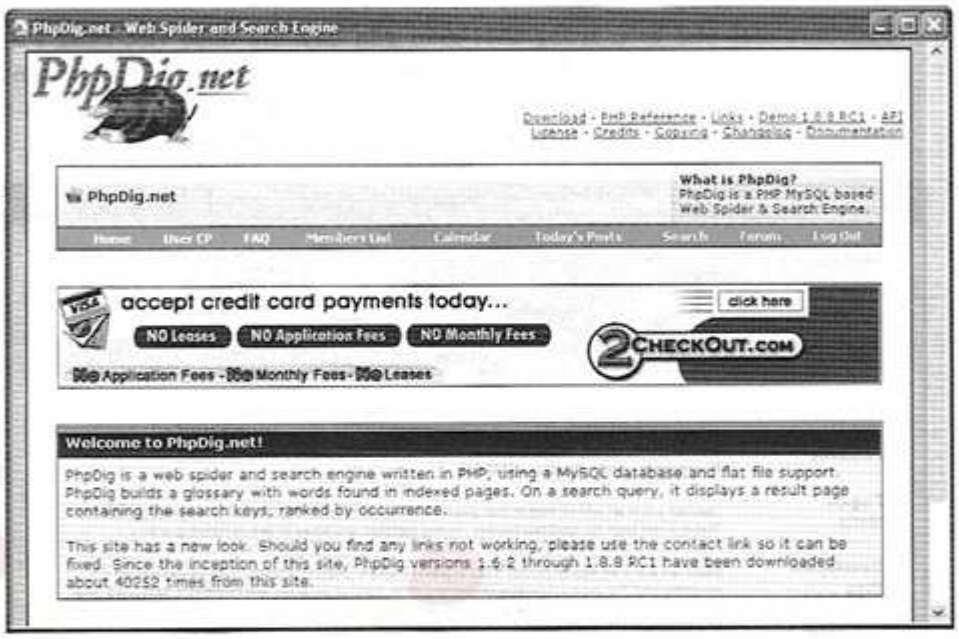
Hình B-9: *FishCart (ra đời năm 1998) đã trở thành một trong số các giải pháp thương mại điện tử mã nguồn mở phổ biến.*

Các công cụ tìm kiếm

Rất dễ để viết một công cụ tìm kiếm căn bản bằng PHP, nhưng để viết một công cụ tìm kiếm có hiệu quả lại là một chuyện khác. Nó đòi hỏi phải có nhiều nỗ lực hơn. phpDig đã khám phá được bí mật của quá trình này bằng cách cung cấp một phương tiện tìm kiếm nhưng sử dụng ít bộ nhớ.

phpDig (tên của phần mềm) làm việc như một HTTP spider (một dạng chương trình tìm kiếm các trang Web trên Internet, cung cấp chúng cho động cơ tìm kiếm), phân tích các hồ sơ của một Website, sau đó kết nối, lưu trữ kết quả trong cơ sở dữ liệu MySQL. Nó có thể tạo chỉ mục cho các hồ sơ Microsoft Word và các tập tin PDF khi đã cài đặt những phần mở rộng tương ứng. Khi Website của mình đã được tạo chỉ mục với phpDig, bạn chỉ cần bổ sung một hộp tìm kiếm để độc giả có thể dễ dàng tìm thấy những thông tin mà họ cần.

phpDig có tại Website *http://phpdig.toiletoine.net* (xem hình B-10).



Hình B-10: *phpDig tuy nhỏ nhưng là một công cụ tìm kiếm mạnh có thể bổ sung vào Website.*

Những phương tiện tìm kiếm khác viết bằng PHP:

- PHP Search Engine (SEARCpHp) có tại: *http://www.hansanderson.com/php/search*.
- phpMysearch có tại *http://phpmysearch.web4.hm*.

Thư viện mã lệnh

Ngày nay không thiếu các thư viện mã lệnh trên mạng. Trên Internet có nhiều site tổ chức nhiều mã lệnh PHP viết sẵn một cách có hệ thống và dễ dàng để tải xuống. Các kho mã trực tuyến phổ biến nhất gồm:

- HotScript.com (*www.hotscript.com*).

- PX: PHP Code Exchange (*http://px.sklar.com*) - (xem hình B-11).

Hình B-11: *PX cung cấp mã PHP trong nhiều hạng mục khác nhau.*

- PHP Resource Index: (*http://php.resourceindex.com/Complete_Scripts/*) - (xem hình B-12).

Hình B-12: *PHP Resource Index có nhiều thông tin hữu ích, bao gồm hàng trăm mã kịch bản PHP.*

- PHP Classes Repository (*www.phpclasses.org*).

Khi muốn tìm kiếm nhanh một vấn đề hoặc muốn xem các nhà phát triển khác làm gì, bạn nên quan tâm đến những site trên.

Bạn cũng có thể tìm ví dụ mã kịch bản PHP tại Website Zend.com và PHPBuilder.com.

Bạn có thể tìm kiếm mã kịch bản PHP bằng công cụ Google hoặc các công cụ tìm kiếm khác.

Phụ lục C:

THAM KHẢO

Phụ lục này sẽ tập hợp các phần tham khảo phổ biến nhất. Nó bao gồm các bảng biểu cơ bản (hầu hết chúng đều đã xuất hiện trong cuốn sách), danh mục các hàm liên quan đến MySQL của PHP và một số thông tin mới khác.

PHP

Tuy phụ lục này không thể thay thế cho tài liệu PHP, nhưng các bảng và các thông tin liệt kê ở đây có thể giúp bạn không cần phải lên mạng để xem các thông tin đó. Ở đây, bạn có thể tìm thấy:

- Toán tử, toán tử so sánh và thứ tự ưu tiên.
- Các hàm và các thông số định dạng thời gian, ngày tháng.
- Ký tự và lớp biểu thức quy tắc.
- Các hàm liên quan đến MySQL.
- Và nhiều thứ khác nữa.

Toán tử và toán tử so sánh

Các toán tử và toán tử so sánh của PHP được liệt kê trong bảng C.1.

Bảng C.1: *Các toán tử của PHP để thực hiện các phép toán.*

Biểu tượng	Ý nghĩa	Loại	Ví dụ
=	Gán giá trị cho	Gán giá trị	`$n=1`
.	Kết hợp	Hỗn hợp	`$var = $x.$y`
==	Bằng	So sánh	`$x == $y`
!=	Không bằng	So sánh	`$x != $y`
<	Nhỏ hơn	So sánh	`$x < $y`
>	Lớn hơn	So sánh	`$x > $y`
<=	Nhỏ hơn hoặc bằng	So sánh	`$x <= $y`
>=	Lớn hơn hoặc bằng	So sánh	`$x >= $y`
!	Khác	Logic	`! $x`
&&	Và	Logic	`$x && $y`
||	Hoặc	Logic	`$x || $y`

+	Cộng	Số học	$x + $y
-	Trừ	Số học	$x - $y
*	Nhân	Số học	$x * $y
/	Chia	Số học	$x / $y
%	Modulo	Số học	$x % $y

Bảng C.2 là bảng C.1 được sắp xếp theo thứ tự ưu tiên, từ cao đến thấp. Bạn cần nhớ danh mục này khi viết các câu lệnh phức tạp hoặc sử dụng dấu ngoặc đơn để đảm bảo trình tự thực hiện các phép toán.

Bảng C.2: *Thứ tự ưu tiên của các toán tử PHP được xếp từ cao tới thấp.*

```
!  ++  -
*  /  %
+  -  .
<  <=  >  >=
==  !=  ===
&&
||
=  +=  -=  *=  /=  %=
and
xor
or
```

Ngày tháng và thời gian

Bảng C.3 liệt kê các thông số định dạng của hàm `date()` theo thứ tự năm, tháng, ngày, giờ, phút, giây.

Bảng C.3: *Các thông số định dạng dùng trong hàm* ***date()****.*

Ký tự	Ý nghĩa	Ví dụ
Y	Năm có 4 chữ số.	2003
y	Năm có 2 chữ số.	03
n	Tháng có 1 hoặc 2 chữ số.	2
m	Tháng có 2 chữ số.	02
F	Tháng.	February
M	Tháng có 3 chữ cái.	Feb

j	Ngày của tháng có 1 hoặc 2 ký tự.	8
d	Ngày của tháng có 2 ký tự.	08
l	Ngày của tuần.	Monday
D	Ngày của tuần có 3 chữ cái.	Mon
w	Ngày của tuần có 1 chữ cái đơn.	0 (Sunday)
z	Ngày của năm: từ 0 đến 365.	152
t	Số ngày trong tháng.	31
g	Giờ, theo thang 12, định dạng bằng 1 hoặc 2 chữ số.	6
S	Hậu tố bằng tiếng Anh có 2 ký tự.	rd
G	Giờ, theo thang 24, định dạng bằng 1 hoặc 2 chữ số.	18
h	Giờ, theo thang 12, định dạng bằng 2 chữ số.	06
H	Giờ, theo thang 24, định dạng bằng 2 chữ số.	18
i	Phút.	45
a	am hoặc pm.	am
A	AM hoặc PM.	PM
s	Giây.	18
U	Giây tính từ mốc thời gian 1/1/1970.	1048623008

Cú pháp của hàm **date()** như sau:

```
string date(string format[,int timestamp])
```

Hàm này nhận vào thông số dạng chuỗi và một tham số tùy chọn **timestamp** (như là số nguyên). Hàm **date()** trả về giá trị chuỗi, ví dụ:

```
echo date ("F j , Y ");
```

Hàm **getdate()**, đã đề cập trong Chương 3 "Tạo Website động", trả về một mảng các thông tin cho một ngày tháng cụ thể. Khóa và giá trị lưu trong mảng được trình bày trong bảng C.4.

```
array getdate([int timestamp])
```

```
$date_array = getdate();
```

Hàm này cũng có tham số tùy chọn **timestamp**.

Bảng C.4: *Mảng trả về của hàm* ***getdate()****.*

Chỉ số	Ý nghĩa	Ví dụ
Year	Năm	2003
Mon	Tháng	12
Month	Tên tháng	December

Mday	Ngày của tháng	25
weekday	Ngày của tuần	Tuesday
hours	Giờ	11
minutes	Phút	56
seconds	Giây	47

Các hàm về MySQL của PHP

Các mã kịch bản chủ yếu của PHP giao tiếp với cơ sở dữ liệu MySQL sẽ sử dụng một số hàm chuẩn sau:

```
mysql_connect()
mysql_select_db()
mysql_query()
mysql_fetch_array()
mysql_error()
mysql_close()
mysql_insert_id()
mysql_num_rows()
mysql_affected_rows()
```

Bảng C.5 liệt kê danh mục đầy đủ các hàm và mục đích sử dụng của chúng. Bảng này liệt kê những hàm hiện có trong phiên bản PHP 4.3 và đã loại bỏ tất cả các hàm lỗi thời (là những hàm vẫn còn có giá trị sử dụng nhưng đã bị thay thế). Nhiều hàm đã được bổ sung vào phiên bản PHP 4 và một số bổ sung vào phiên bản PHP 4.3. Chi tiết về cách dùng và các phiên bản của chúng có thể tìm thấy trong các tài liệu về PHP.

Bảng C.5: *Danh mục các hàm liên quan đến MySQL của PHP.*

Hàm	Mục đích
mysql_affected_rows	Trả về số dòng bị tác động bởi câu truy vấn trước đó.
mysql_client_encoding	Trả về hệ thống ký tự đang dùng.
mysql_close	Đóng một kết nối mở.
mysql_connect	Kết nối với MySQL.
mysql_data_seek	Di chuyển con trỏ đến vị trí mới trong tập hợp kết quả.
mysql_db_name	Trả về tên của cơ sở dữ liệu.

mysql_errno	Trả về số hiệu lỗi gần nhất trong MySQL.
mysql_error	Trả về thông báo dạng chuỗi của lỗi gần nhất.
mysql_escape_string	Mã hóa escape chuỗi dùng trong câu truy vấn.
mysql_fetch_array	Trả về một dòng dưới dạng một mảng.
mysql_fetch_assoc	Trả về dòng đặc biệt dưới dạng một mảng kết hợp.
mysql_fetch_field	Trả về chi tiết việc mô tả cột dưới dạng đối tượng.
mysql_fetch_lengths	Trả về độ dài của dữ liệu đầu ra trong kết quả.
mysql_fetch_object	Trả về một dòng dưới dạng một đối tượng.
mysql_fetch_row	Trả về một dòng riêng biệt dưới dạng một mảng số.
mysql_field_flags	Trả về cờ hiệu kết hợp với một trường cụ thể trong kết quả.
mysql_field_len	Trả về độ dài của một trường cụ thể.
mysql_field_name	Trả về tên một trường cụ thể trong kết quả.
mysql_field_seek	Đặt con trỏ tại vị trí cụ thể trong tập hợp kết quả.
mysql_field_table	Trả một trường cụ thể về tên bảng.
mysql_field_type	Trả về kiểu cột của một trường cụ thể trong kết quả.
mysql_free_result	Giải phóng bộ nhớ đã dùng bằng một tập hợp kết quả.
mysql_get_client_info	Trả về thông tin máy khách hàng MySQL.
mysql_get_host_info	Trả về thông tin địa chỉ máy chủ MySQL.
mysql_get_proto_info	Trả về thông tin giao thức của MySQL.
mysql_get_server_info	Trả về thông tin máy chủ MySQL.
mysql_info	Trả về thông tin của câu truy vấn chạy trước đó.
mysql_insert_id	Trả về ID của cột AUTO_INCREMENTED, phát sinh sau khi thực hiện câu lệnh INSERT.
mysql_list_dbs	Liệt kê các cơ sở dữ liệu hiện có.
mysql_list_fields	Liệt kê các cột trả về.
mysql_list_processes	Liệt kê các quá trình.
mysql_list_tables	Liệt kê các bảng trong cơ sở dữ liệu.
mysql_num_fields	Trả về số hiệu của cột trong tập hợp kết quả.
mysql_num_rows	Trả về số hiệu của dòng trong tập hợp kết quả.

mysql_pconnect	Mở một kết nối thường trực.
mysql_ping	Ping (một thủ tục kiểm tra kết nối) tới máy chủ MySQL.
mysql_query	Chạy một câu truy vấn.
mysql_real_escape_string	Mã hóa escape một ký tự đặc biệt trong chuỗi để dùng trong câu lệnh SQL, có quan tâm đến bộ ký tự hiện hành của kết nối.
mysql_result	Trả về nội dung của cột từ trong mẩu tin.
mysql_select_db	Chọn cơ sở dữ liệu để dùng.
mysql_stat	Trả về trạng thái hệ thống MySQL hiện hành.
mysql_tablename	Trả về tên của bảng cho một cột cụ thể.
mysql_thread_id	Trả về ID của luồng (thread) hiện hành.
mysql_unbuffered_query	Gửi một câu truy vấn tới MySQL mà không lấy ra kết quả hoặc đưa vào bộ đệm.

Các biểu thức quy tắc

Trong Chương 8 "An toàn", chúng ta đã đề cập đến các biểu thức quy tắc như một cách để kiểm tra tính hợp lệ của dữ liệu biểu mẫu gửi lên. Phần này chỉ lặp lại các bảng với một chút thay đổi (xem bảng C.6, C.7 và C.8). Các ký tự này dùng để thiết lập mẫu cho các hàm ***ereg()***, ***eregi()***, ***ereg_replace()*** và ***eregi_replace()***.

Bảng C.6: *Những ký tự có ý nghĩa đặc biệt cho các biểu thức quy tắc.*

Ký tự	Ý nghĩa
^	Bắt đầu một chuỗi.
$	Kết thúc một chuỗi.
.	Một ký tự.
|	Sự thay đổi.
\	Mã hóa escape ký tự tiếp theo.
()	Dùng cho việc tạo nhóm.
[]	Dùng xác định lớp.

Bảng C.7: *Các ký tự định lượng trong biểu thức quy tắc.*

Ký tự	Ý nghĩa
?	Xuất hiện 0 hoặc 1 lần.
*	Xuất hiện từ 0 lần trở lên.
+	Xuất hiện từ 1 lần trở lên.

{x}	Xuất hiện chính xác *x* lần.
{x,y}	Xuất hiện từ *x* đến *y* lần.
{x,}	Xuất hiện ít nhất *x* lần.

Bảng C.8: *Các lớp viết tắt cho các bộ ký tự phổ biến.*

Lớp	Ý nghĩa
[a-z]	Mẫu tự viết thường.
[A-Z]	Mẫu tự viết hoa.
[a-zA-Z]	Một chữ cái.
[0-9]	Một số.
[\f\r\t\n\v]	Một khoảng trắng.
[aeiou]	Một nguyên âm.
[[:alnum:]]	Một mẫu tự hoặc số.
[[:alpha:]]	Một mẫu tự (giống [a-zA-Z].
[[:blank:]]	Một tab hoặc dấu cách.
[[:digit:]]	Một số (giống [0-9]).
[[:lower:]]	Một mẫu tự viết thường.
[[:upper:]]	Một mẫu tự viết hoa.
[[:punct:]]	Dấu ngắt (.,;:-).
[[:space:]]	Một khoảng trắng.

Các tham chiếu khác

Phần tham chiếu cuối cùng này đúc kết lại những ý tưởng đã được đề cập trong cuốn sách. Bảng C.9 liệt kê các ký tự mà khi được mã hóa escape trong dấu nháy kép sẽ mang một ý nghĩa đặc biệt. Dấu nháy đơn (không liệt kê trong bảng này) không cần phải mã hóa escape trong dấu nháy kép, nhưng phải được mã hóa escape khi được đặt trong dấu nháy đơn.

Bảng C.9: *Một số ký tự phải được mã hóa escape trong dấu nháy kép.*

Ký tự	Ý nghĩa
\"	Dấu nháy kép.
\\	Dấu chéo ngược.
\n	Dòng mới.
\r	Trở về đầu dòng.
\t	Tab.
\$	Ký hiệu đô la.

Bảng C.10 liệt kê những cặp *khóa-giá trị* trả về từ hàm ***getimagesize()***. Giá trị thường dùng nhất được xác định tại chỉ số 3. Đây là một chuỗi dùng để tạo mã HTML cho chiều cao và chiều rộng của hình. Phần tử thứ ba trong mảng là biểu diễn bằng số về kiểu của hình, với 1 tương đương với GIF, 2 là JPG, 3 là PNG, 4 là SWF (Shockwave), 5 là PSD (Photoshop), 6 là BMP (Bitmap), 7 và 8 là TIFF (hai loại khác nhau)...

Bảng C.10: *Hàm **getimagesize()** trả về một mảng với khóa và giá trị.*

Chỉ số	Ý nghĩa	Ví dụ
`0`	Chiều rộng tính bằng pixel.	`100`
`1`	Chiều cao tính bằng pixel.	`150`
`2`	Loại hình ảnh.	`2`
`3`	Mã HTML.	`Height="150", Width="100"`

Cuối cùng, bảng C.11 trình bày nội dung mảng **`$_FILES`**, được dùng khi tải các tập tin thông qua trình duyệt Web.

Bảng C.11: *Mảng **`$_FILES`** được dùng để truy cập thông tin về tập tin được tải lên (áp dụng cho phiên bản PHP 4.1 hoặc mới hơn).*

Chỉ mục	Ý nghĩa
`name`	Tên ban đầu của tập tin (trên máy tính người dùng).
`type`	Dạng MIME của tập tin, do trình duyệt cung cấp.
`size`	Kích thước của tập tin tải lên, tính bằng bytes.
`tmp_name`	Tên tạm thời của tập tin tải lên, được lưu trữ trên máy chủ.
`error`	Lỗi có liên quan đến những vấn đề đang xảy ra.

MySQL

Việc chọn kiểu dữ liệu thích hợp cho các cột trong bảng là bí quyết thành công của một cơ sở dữ liệu. Bảng C.12 liệt kê các chuỗi, số và kiểu dữ liệu khác mà bạn có thể dùng, cùng với kích thước mà nó chiếm chỗ trên ổ cứng máy chủ. Khi chọn kiểu dữ liệu cho một cột, bạn nên dùng loại có hiệu quả nhất (tức là có kích thước tiết kiệm nhất) dựa vào giá trị lớn nhất của cột.

Bảng C.12: *Danh mục các kiểu dữ liệu dùng để định nghĩa cột trong bảng MySQL.*

Kiểu	Kích thước	Mô tả
`CHAR[Length]`	Độ dài	Chuỗi có độ dài cố định từ 0 đến 255 ký tự.
`VARCHAR(Length)`	Độ dài chuỗi + 1 bytes	Chuỗi có độ dài biến đổi từ 0 đến 255 ký tự.
`TINYTEXT`	Độ dài chuỗi + 1 bytes	Chuỗi với độ dài tối đa 255 ký tự.

TEXT	Độ dài chuỗi + 2 bytes	Chuỗi với độ dài tối đa 65.535 ký tự.
MEDIUMTEXT	Độ dài chuỗi + 3 bytes	Chuỗi với độ dài tối đa 16.777.215 ký tự.
LONGTEXT	Độ dài chuỗi + 4 bytes	Chuỗi với độ dài tối đa 4.294.967.295 ký tự.
TINYINT[Length]	1 bytes	Số nguyên có giá trị từ -128 đến 127 hoặc số dương từ 0 đến 255.
SMALLINT[Length]	2 bytes	Số nguyên từ -32.768 đến 32.767 hoặc số dương từ 0 đến 65.535.
MEDIUMINT[Length]	3 bytes	Số nguyên từ -8.388.608 đến 8.388.607 hoặc số dương từ 0 đến 16.777.215.
INT[Length]	4 bytes	Số nguyên từ -2.147.483.648 đến 2.147.483.647 hoặc số dương từ 0 đến 4.294.967.295.
BIGINT[Length]	8 bytes	Số nguyên từ -9.223.372.036.854.775.808 đến 9.223.372.036.854.775.807 hoặc số dương từ 0 đến 18.446.744.073.709.551.615.
FLOAT	4 bytes	Số thập phân nhỏ.
DOUBLE [Length,Decimal]	8 bytes	Số thập phân lớn.
DECIMAL [Length, Decimal]	Độ dài + 1 hoặc 2 bytes	Số thập phân DOUBLE được lưu dưới dạng chuỗi.
DATE	3 bytes	Ngày theo định dạng YYYY-MM-DD.
DATETIME	8 bytes	Ngày giờ theo định dạng YYYY-MM-DD HH:MM:SS.
TIMESTAMP	4 bytes	Giờ theo định dạng YYYYMMDDHHMMSS; giá trị cho phép đến năm 2037.
TIME	3 bytes	Giờ theo định dạng HH:MM:SS.
YEAR	1 bytes	Năm theo định dạng YY hoặc YYYY.
ENUM	1 hoặc 2 bytes	Viết tắt của tập hợp. Mỗi cột chỉ được phép có một giá trị trong tập hợp cho trước.
SET	1,2,3,4 hoặc 8 bytes.	Giống ENUM, ngoại trừ mỗi cột có thể có nhiều giá trị từ tập hợp cho trước.
TINYBLOB	Độ dài chuỗi + 1 bytes	Tập tin nhị phân có độ dài tối đa 255 ký tự.

BLOB	Độ dài chuỗi + 2 bytes	Tập tin nhị phân có độ dài tối đa 65.535 ký tự.
MEDIUMBLOB	Độ dài chuỗi + 3 bytes	Tập tin nhị phân có độ dài tối đa 16.777.215 ký tự.
LONGBLOB	Độ dài chuỗi + 4 bytes	Tập tin nhị phân có độ dài tối đa 4.294.967.295 ký tự.

Chú ý, khi định nghĩa cột, nó có thể chấp nhận NULL hoặc NOT NULL, các số nguyên có thể là UNSIGNED và bất kỳ số nào cũng có thể là ZEROFILL. Một cột số nguyên cũng có thể được chỉ định là AUTO_INCREMENT nếu nó được thiết lập làm khóa chính cho bảng.

Bảng C.13 giới thiệu phần lớn các toán tử dùng trong câu truy vấn SQL trên cơ sở dữ liệu MySQL. Hầu hết chúng đều là thành phần của chuẩn SQL và làm việc với bất kỳ ứng dụng cơ sở dữ liệu nào.

Bảng C.14 là danh mục các thuật ngữ cơ bản nhất của SQL. Chúng đã được đề cập nhiều trong Chương 4 "Giới thiệu về SQL và MySQL" và Chương 5 "SQL và MySQL nâng cao".

Bảng C.13: *Các toán tử dùng trong các câu truy vấn với MySQL.*

Toán tử	Ý nghĩa
+	Cộng.
-	Trừ.
*	Nhân.
/	Chia.
%	Modulo.
=	Bằng.
<	Nhỏ hơn.
>	Lớn hơn.
<=	Nhỏ hơn hoặc bằng.
>=	Lớn hơn hoặc bằng.
!=	Không bằng.
IS NOT NULL	Có một giá trị.
IS NULL	Không có giá trị.
BETWEEN	Nằm trong khoảng.
NOT BETWEEN	Nằm ngoài khoảng.
OR (hoặc ||)	Khi một trong hai điều kiện là đúng.
AND (hoặc &&)	Khi cả hai điều kiện đều đúng.
NOT (hoặc !)	Khi điều kiện không đúng.
LIKE	Khi một giá trị phù hợp với một chuỗi.

NOT LIKE	Khi một giá trị không phù hợp với một chuỗi.
%	Ký tự đại diện đa (dùng với LIKE và NOT LIKE).
-	Ký tự đại diện đơn (dùng với LIKE và NOT LIKE).
REGEXP	Khi giá trị phù hợp với một mẫu so khớp.
NOT REGEXP	Khi giá trị không phù hợp với một mẫu so khớp.

Bảng C.14: *SQL có ít thuật ngữ nhưng vẫn thực hiện được các thủ tục phức tạp.*

Thuật ngữ	Sử dụng
ALTER	Thay đổi định dạng của bảng.
CREATE	Tạo ra bảng hoặc cơ sở dữ liệu.
DELETE	Xóa mẩu tin ra khỏi bảng.
DROP	Xóa toàn bộ bảng hoặc cơ sở dữ liệu.
INSERT	Thêm mẩu tin vào bảng.
SELECT	Lấy thông tin từ bảng.
SHOW	Lấy thông tin về cấu trúc của bảng hoặc cơ sở dữ liệu.
UPDATE	Cập nhật dữ liệu vào bảng.

Các hàm

Việc định dạng các kết quả trả về từ một câu truy vấn sẽ làm cho chúng hữu dụng hơn và giúp giảm bớt một lượng lập trình đáng kể. Để làm điều này, chúng ta dùng các hàm có sẵn trong MySQL (đã được giới thiệu trong Chương 5). Bảng C.15 giới thiệu các hàm dùng cho chuỗi. Bảng C.16 giới thiệu một số hàm sử dụng cho số.

Bảng C.15: *Các hàm MySQL được dùng để thao tác giá trị chuỗi.*

Hàm	Cách dùng	Mục đích
CONCAT()	CONCAT(x,y...)	Tạo ra chuỗi mới của dạng *xy...*
LENGTH()	LENGTH(column)	Trả về độ dài của giá trị lưu trong cột.
LEFT()	LEFT(column, x)	Trả về *x* ký tự bên trái từ giá trị của cột.
LOWER()	LOWER(column)	Chuyển chuỗi lưu trữ thành định dạng chữ thường.
RIGHT()	RIGHT(column, x)	Trả về *x* ký tự bên phải từ giá trị của cột.
SUBSTRING()	SUBSTRING(column, start, length)	Trả về *length* ký tự của cột *column* từ vị trí *start* (bắt đầu là 0).
TRIM()	TRIM(column)	Cắt bỏ phần dư đầu và cuối của giá trị lưu trữ.
UPPER()	UPPER(column)	Viết hoa toàn bộ chuỗi lưu trữ.

Bảng C.16: *Các hàm trong MySQL thực hiện các phép tính, định dạng và xử lý số.*

Hàm	Cách dùng	Mục đích
ABS()	ABS(x)	Trả về giá trị tuyệt đối của *x*.
CEILING()	CEILING(x)	Trả về số nguyên cao gần nhất trên cơ sở giá trị của *x*.
FLOOR()	FLOOR(x)	Trả về giá trị nguyên của *x*.
FORMAT()	FORMAT(x,y)	Trả về giá trị *x* được định dạng là một số có phần thập phân *y* và có dấu phẩy đặt vào mỗi khoảng phân cách.
MOD()	Mod(x,y)	Trả về phần dư của phép chia *x* cho *y*.
POWER()	POWER(x, y)	Trả về giá trị của *x* lũy thừa *y*.
RAND()	RAND()	Trả về một số bất kỳ giữa 0 và 1.0.
ROUND()	ROUND(x, y)	Trả về giá trị *x* được làm tròn đến phần thập phân *y*.
SIGN()	SIGN(x)	Trả về giá trị chỉ ra số âm (-1), zero (0), hoặc số dương (+1).
SQRT()	SQRT(x)	Tính căn bậc hai của *x*.

Bảng C.17 liệt kê các hàm tập hợp hoặc các hàm gom nhóm. Bảng C.18 là danh mục các hàm hỗn hợp. Phần lớn các hàm được dùng để lấy giá trị từ cột hoặc nhập giá trị vào một cột, ví dụ: `SELECT ROUND(3.142857,2)`.

Bảng C.17: *Các hàm gom nhóm thường được sử dụng với mệnh đề GROUP BY.*

Hàm	Sử dụng	Mục đích
AVG()	AVG(column)	Trả về giá trị trung bình của cột.
COUNT()	COUNT(column)	Đếm số dòng.
COUNT(DISTINCT)	COUNT(DISTINCT column)	Đếm số giá trị cột phân biệt.
MIN()	MIN(column)	Trả về giá trị nhỏ nhất của cột.
MAX()	MAX(column)	Trả về giá trị lớn nhất của cột.
SUM()	SUM(column)	Trả về tổng các giá trị của cột.

Bảng C.18: *Các hàm xử lý mã hóa và các công việc khác.*

Hàm	Sử dụng	Mục đích
CONCAT_WS()	CONCAT_WS('-', column1, column2)	Liên kết các phần tử với một dấu phân cách chung.
DATABASE()	DATABASE()	Trả về tên của cơ sở dữ liệu đang dùng.

ENCODE()	ENCODE('string', 'salt')	Trả về một phiên bản mã hóa của chuỗi có khả năng được giải mã.
ENCRYPT()	ENCRYPT('string','salt')	Trả về một phiên bản mã hóa của chuỗi sử dụng *salt* (đòi hỏi thư viện mã Unix).
DECODE()	DECODE ('string','salt')	Trả về phiên bản được giải mã của chuỗi.
LAST_INSERT_ID()	LAST_INSERT_ID()	Trả về giá trị tự động gia tăng ngay trước.
PASSWORD()	PASSWORD('string')	Trả về một phiên bản được mã hóa của chuỗi.
USER()	USER()	Trả về tên tài khoản trong session hiện hành.

Ngày, giờ

Tùy thuộc vào nơi phát sinh các giá trị ngày, giờ, chúng ta đều có thể dùng PHP hay MySQL để định dạng các giá trị trả về. Phần trước đã liệt kê các thông số để định dạng ngày trong PHP với hàm ***date()***, sau đó là hàm ***getdate()***. Bảng C.19 liệt kê một số hàm liên quan đến ngày, giờ trong MySQL. Bảng C.20 là định dạng để sử dụng hàm DATE_FORMAT() và TIME_FORMAT().

Bảng C.19: *Các hàm trong MySQL làm việc với các giá trị ngày, giờ.*

Hàm	Sử dụng	Mục đích
HOUR()	HOUR(column)	Trả về giá trị giờ của ngày lưu giữ.
MINUTE()	MINUTE(column)	Trả về giá trị phút của ngày lưu giữ.
SECOND()	SECOND(column)	Trả về giá trị giây của ngày lưu giữ.
DAYNAME()	DAYNAME(column)	Trả về tên ngày của giá trị ngày lưu trữ.
DAYOFMONTH()	DAYOFMONTH(column)	Trả về giá trị ngày trong tháng (bằng số) của ngày lưu giữ.
MONTHNAME()	MONTHNAME(column)	Trả về tên tháng của giá trị ngày lưu trữ.
MONTH()	MONTH(column)	Trả về giá trị tháng bằng số của ngày lưu giữ.

YEAR()	YEAR(column)	Trả về giá trị năm của ngày lưu giữ.
ADDDATE()	ADDDATE(column, INTERVAL x type)	Trả về giá trị của ngày cộng với *x* đơn vị.
SUBDATE()	SUBDATE(column, INTERVAL x type)	Trả về giá trị của ngày trừ đi *x* đơn vị.
CURDATE()	CURDATE()	Trả về ngày hiện hành.
CURTIME()	CURTIME()	Trả về thời gian hiện hành.
NOW()	NOW()	Trả về ngày và thời gian hiện hành.
UNIX_TIMESTAMP()	UNIX_TIMESTAMP(date)	Trả về số giây tính từ mốc thời gian 1/1/1970 hoặc từ ngày tháng cụ thể.

Bảng C.20: *Các thông số dùng trong các hàm* ***DATE_FORMAT()*** *và* ***TIME_FORMAT()****.*

Ký hiệu	Sử dụng	Ví dụ
%e	Ngày của tháng.	1-31
%d	Ngày của tháng, hai chữ số.	01-31
%D	Ngày có hậu tố.	1st - 31st
%W	Tên của ngày trong tuần.	Sunday - Saturday
%a	Tên viết tắt của ngày trong tuần.	Sun - Sat
%c	Số tháng.	1-12
%m	Số tháng, có hai chữ số.	01-12
%M	Tên tháng.	January - December
%b	Tên tháng, viết tắt.	Jan - Dec
%Y	Năm bốn chữ số.	2002
%y	Năm hai chữ số.	02
%l	Giờ.	1-12
%h	Giờ, có hai chữ số.	01-12
%k	Giờ, theo thang 24.	0-23
%H	Giờ, theo thang 24, hai chữ số.	00-23
%i	Phút.	00-59
%S	Giây.	00-59
%r	Thời gian.	8:17:02 PM
%T	Thời gian, theo thang 24.	20:17:02
%p	AM hoặc PM.	AM hoặc PM

Phụ lục D:

NGUỒN THAM KHẢO

Cuốn sách này được viết với mục đích chỉ cho bạn cách phát triển các Website động với PHP và MySQL. Cuốn sách có đề cập đến nhiều nội dung phụ (ví dụ JavaScript và Cascading Style Sheet). Tuy nhiên, còn có một số chủ đề khác chưa được đề cập tới. Phụ lục này sẽ cung cấp cho bạn các nguồn tham khảo để bổ sung những phần còn trống đó và làm phong phú thông tin muốn trình bày.

Tất cả nguồn tham khảo được liệt kê trong phụ lục này đều hữu ích cho phần đông độc giả. Những tham chiếu ở đây không có ý xác nhận hoặc ngụ ý rằng đây là nguồn hoặc công cụ tốt nhất.

PHP

Hình D-1: *Tài liệu PHP có ở nhiều dạng khác nhau.*

Bạn nên tham khảo các tài liệu về PHP trước khi làm việc với ngôn ngữ. Nó có sẵn tại site chính thức của PHP, tại địa chỉ *www.php.net/docs.php* (xem hình D-1). Ngoài ra, nó cũng có trong một số Website khác. Bạn có thể tải xuống tài liệu

ở định dạng bất kỳ: PDF, HTML, văn bản thuần túy hoặc dạng tương thích Palm. Website chính thức còn có một phiên bản chú thích của tài liệu tại địa chỉ *www.php.net/manual/en*, trong đó nhiều người dùng đã thêm vào các chú thích và đề nghị hữu ích để giải quyết những vấn đề gặp phải khi dùng một hàm cụ thể.

Bạn có thể truy cập trang tài liệu về một hàm cụ thể nào bằng cách nhập URL http://www.php.net/function_name.

Các Website

Chúng ta sẽ đề cập đến một số Website thường gặp khi lập trình, còn việc chọn site nào là hoàn toàn tùy thuộc vào bạn. Đa số chúng cũng chứa những liên kết đến các site liên quan PHP khác.

Site đáng lưu tâm đầu tiên và hiển nhiên nhất là PHP.net (*www.php.net*). Đây là site chính thức của PHP. Kế đó, bạn nên làm quen với Website Web Zend.com (*www.zend.com*) - (xem hình D-2), trang chủ của những người tạo ra Zend Engine ("linh hồn" của PHP 4). Site này chứa một lượng lớn mã lệnh có thể tải xuống cùng với những nguồn tham khảo phong phú khác.

Hình D-2: *Website tốt nhất kế tiếp sau trang chủ PHP là Zend.com.*

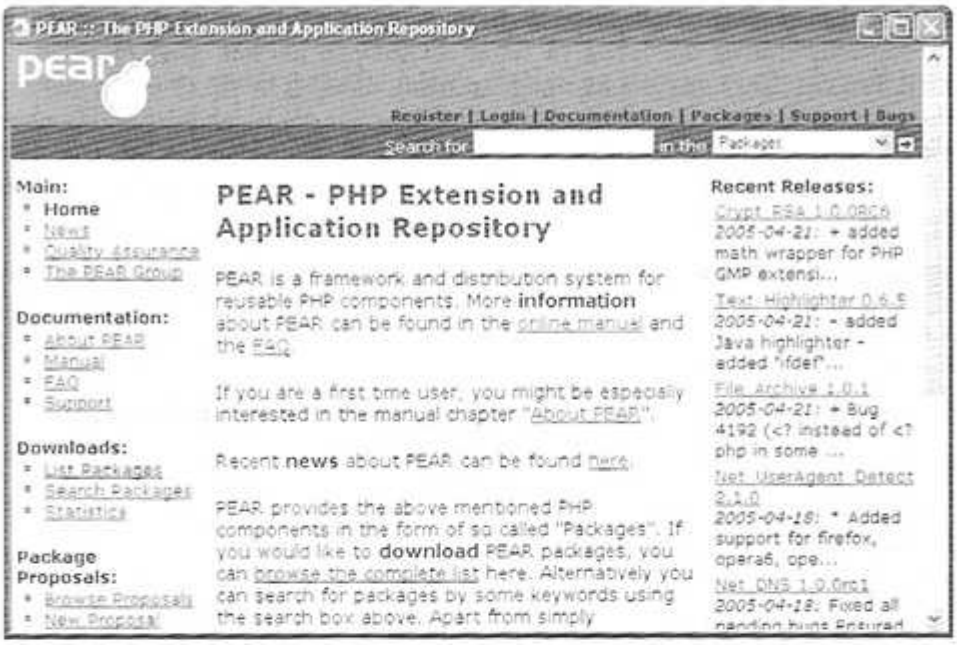
Về thông tin trên các chủ đề cụ thể, PHPBuilder (*www.phpbuilder.com*) là một site đáng quan tâm. Site này có nhiều chủ đề giải thích cách để thực hiện công việc cụ thể khi dùng PHP và MySQL.

Cuốn sách này chưa nói về một kiểu biến có tên là đối tượng (object). PHP có hỗ trợ lập trình hướng đối tượng và thậm chí trở nên nổi bật hơn khi phiên bản PHP 5 ra đời. Khi nâng cao kỹ năng lập trình, đặc biệt là khi thiết lập được một thư viện mã mạnh, có khả năng tạo và ứng dụng các đối tượng, bạn sẽ đẩy nhanh tốc độ lập trình, giảm thiểu các lỗi khi vận hành. Có nhiều bài học trực tuyến liên quan đến đối tượng trong PHP mà bạn có thể tìm thấy tại Zend.com (*www.zend.com/zend/tut/class-intro.php*) - (xem hình D-3).

Hình D-3: *Một trong nhiều bài học bạn có thể tìm thấy trên Zend.com là giới thiệu về OOP.*

Nguồn tham khảo Website cuối cùng là PHP Coding Standard, có tại địa chỉ *http://utvikler.start.no/code/php_coding_standard.html*. Hồ sơ này gợi ý các chuẩn lập trình PHP về định dạng và cú pháp thích hợp cho tên biến, các cấu trúc kiểm tra... Tuy không phải tuân theo những quy tắc này, nhưng bạn nên quan tâm vì chúng vẫn là những gợi ý chắc chắn, được suy nghĩ thấu đáo nhằm giúp giảm thiểu các lỗi khi lập trình.

Một nguồn khác về chuẩn mã hóa có trong tài liệu về PEAR (http://pear.php.net) - (xem hình D-4).



Hình D-4: *PEAR là một công cụ tốt được dùng khi bạn đã sẵn sàng đi vào lập trình ở những bước kế tiếp.*

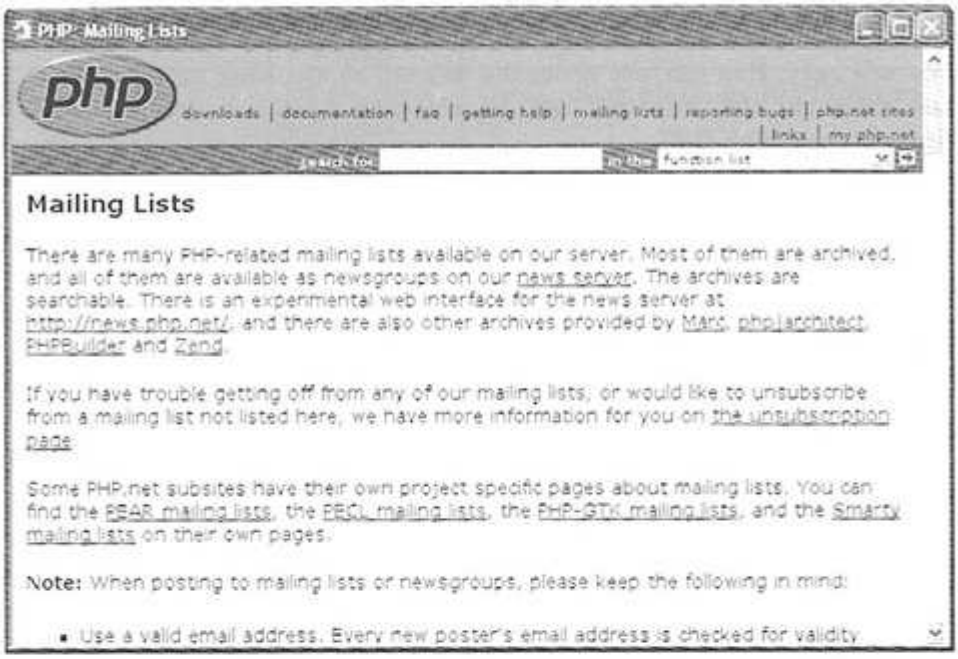
PHP.net, Zend.com và PHPBuilder là ba "đại gia" của các Website có liên quan PHP. Một tìm kiếm trực tuyến nhanh sẽ giúp bạn khám phá ra nhiều ứng dụng khác (với chất lượng khác nhau).

Nhóm tin (newsgroup) và nhóm thư (mailing list)

Nếu có tham gia các nhóm tin, bạn nên dùng chúng như một nguồn để tìm ra các ý tưởng. Đó cũng là nơi mà những câu hỏi khó nhất của bạn sẽ được giải đáp. Dĩ nhiên, bạn luôn có thể phản hồi lại cho nhóm bằng cách đưa ra những ý kiến của riêng mình khi được yêu cầu.

Nhóm tin PHP bằng ngôn ngữ tiếng Anh lớn nhất là *comp.lang.php*. Bạn có thể truy cập *comp.lang.php* qua ISP của mình hoặc thông qua tổ chức Usenet. Còn có nhiều nhóm tin có sẵn những ngôn ngữ khác ngoài tiếng Anh.

Website PHP liệt kê nhiều nhóm thư mà bạn có thể đăng ký tại địa chỉ *www.php.net/mailing-list.php* (xem hình D-5).



Hình D-5: *Có nhiều nhóm thư trong PHP mà bạn có thể đăng ký sử dụng.*

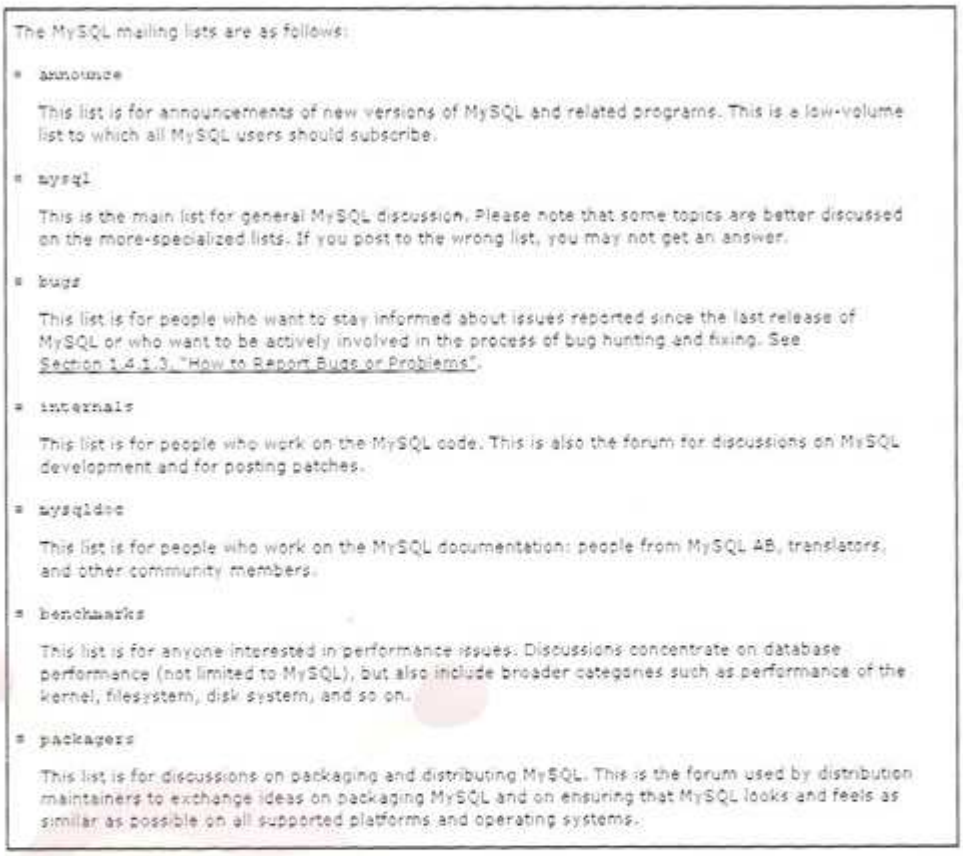
MySQL

Nguồn đầu tiên bạn nên tham khảo là tài liệu MySQL có sẵn tại Website của công ty (*www.mysql.com*). Phiên bản trực tuyến chính có ưu điểm là có thể tìm kiếm và các phiên bản trực tuyến khác đưa ra những bình luận của người dùng đôi khi cũng có ích. Bạn nên lưu một bản sao của tài liệu trên ổ cứng cho phiên bản MySQL đang dùng (vì tài liệu phản ánh phiên bản hiện hành của MySQL cũng chứa đầy đủ những nội dung của phiên bản cũ).

Khi đã nghiên cứu kỹ tài liệu MySQL, bạn nên tham khảo thêm một trong những nhóm thư dành riêng cho MySQL (không có nhóm tin tức chính thức của MySQL). Mỗi nhóm thư có một chủ đề khác nhau:

- Thông báo
- Tổng quát
- Java
- Windows
- ODBC
- C++
- Perl

Tất cả những chủ đề trên, ngoại trừ phần thông báo, đều có sẵn ở dạng lưu trữ (archive) để bạn có thể nhận được hai thư điện tử lớn thay vì nhận 50 thư riêng biệt mỗi ngày. Hơn nữa, các nhóm thư này còn có sẵn bằng ngôn ngữ khác. Để biết thêm chi tiết, hãy truy cập địa chỉ *www.mysql.com/doc/M/a/Mailing-list.html* (xem hình D-6).



The MySQL mailing lists are as follows:

- announce

  This list is for announcements of new versions of MySQL and related programs. This is a low-volume list to which all MySQL users should subscribe.

- mysql

  This is the main list for general MySQL discussion. Please note that some topics are better discussed on the more-specialized lists. If you post to the wrong list, you may not get an answer.

- bugs

  This list is for people who want to stay informed about issues reported since the last release of MySQL or who want to be actively involved in the process of bug hunting and fixing. See Section 1.4.1.3, "How to Report Bugs or Problems".

- internals

  This list is for people who work on the MySQL code. This is also the forum for discussions on MySQL development and for posting patches.

- mysqldoc

  This list is for people who work on the MySQL documentation: people from MySQL AB, translators, and other community members.

- benchmarks

  This list is for anyone interested in performance issues. Discussions concentrate on database performance (not limited to MySQL), but also include broader categories such as performance of the kernel, filesystem, disk system, and so on.

- packagers

  This list is for discussions on packaging and distributing MySQL. This is the forum used by distribution maintainers to exchange ideas on packaging MySQL and on ensuring that MySQL looks and feels as similar as possible on all supported platforms and operating systems.

Hình D-6: *MySQL cũng có rất nhiều nhóm thư, mỗi nhóm phục vụ cho một chủ đề cụ thể.*

Thông qua liên kết khác tại site MySQL *http://lists.mysql.com* (xem hình D-7), bạn có thể tìm kiếm thông tin lưu trữ của nhóm thư. Việc thực hiện một tìm kiếm nhanh ở đây trước khi gửi câu hỏi cho nhóm thư (sau khi đã tìm kiếm trong tài liệu MySQL) sẽ giúp bạn tiết kiệm thời gian và tránh những phiền toái cho những thành viên khác trong nhóm thư.

Hình D-7: *Nhóm thư MySQL có sẵn ở định dạng có thể tìm kiếm được. Rất có thể mọi thắc mắc của bạn đã được trả lời nhiều lần.*

Website Bit By Bit của Đan Mạch có một số nguồn tham khảo MySQL tốt, bao gồm MySQL FAQ, được tạo và duy trì bởi Carsten Pedersen. Bạn có thể tìm thấy tại địa chỉ *www.bitbybit.dk/mysqlfaq*.

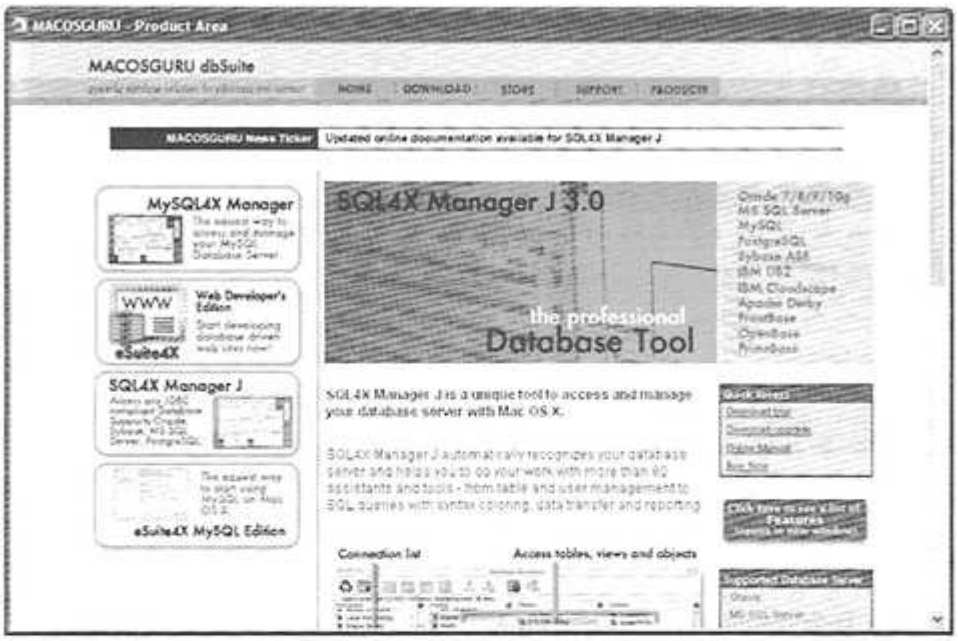
Các công cụ MySQL

Trong tương tác với MySQL, chúng ta thường dùng trình quản lý mysql hoặc phpMyAdmin, nhưng chúng không phải là lựa chọn duy nhất. Còn có nhiều ứng dụng mã nguồn mở và các ứng dụng thương mại có chất lượng khác nữa cho mỗi hệ điều hành.

Người dùng Mac OS X có thể tìm thấy những sản phẩm tốt tại MacOSGuru (*www.macosguru.de*) - (xem hình D-8) hoặc dùng YourSQL (*www.mludi.net/YourSQL*) miễn phí.

MySQL AB, công ty đứng sau MySQL đã tạo ra ứng dụng GUI gọi là MySQLCC (MySQL Control Center). Nó có sẵn miễn phí trên Windows và Linux tại địa chỉ *www.mysql.com/products/mysqlcc/index.html*.

Cuối cùng, MySQL Manager của EMS (*http://ems-hitech.com/mymanager/*) là một công cụ quản trị MySQL mạnh, chạy trên Windows và Linux.

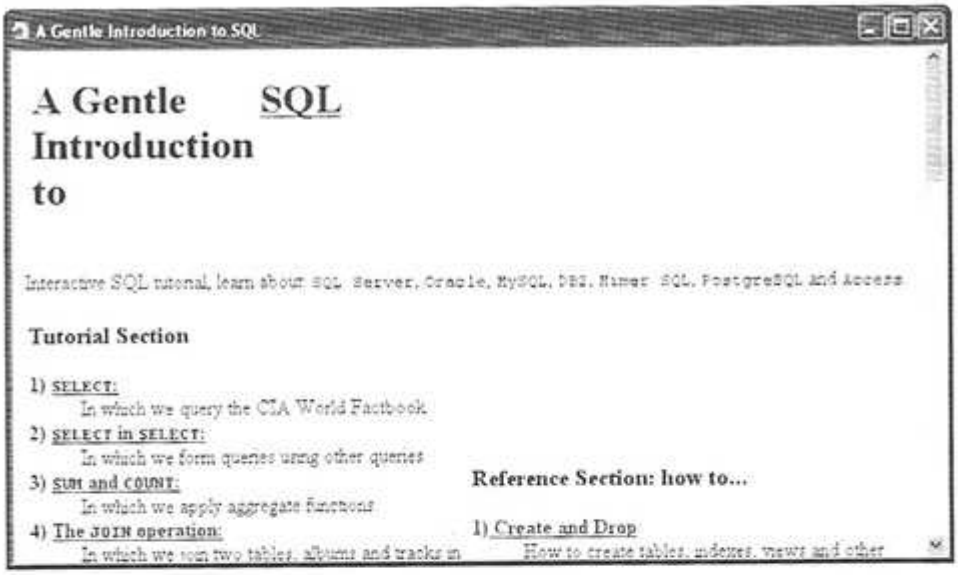


Hình D-8: *Website MacOSGuru phát triển một vài ứng dụng để làm việc với các cơ sở dữ liệu.*

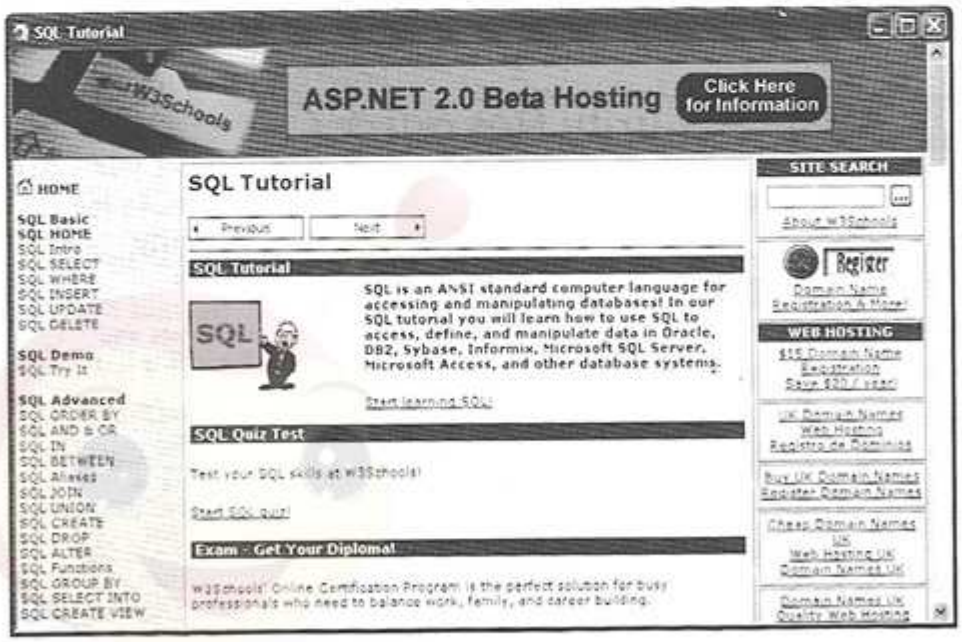
SQL

Vì SQL được dùng bởi MySQL và các cơ sở dữ liệu khác, bạn có thể tìm thấy những nguồn tham khảo vô tận cho ngôn ngữ này. Trong khi các tham chiếu SQL nói chung không trực tiếp chỉ cho bạn cách làm việc với cơ sở dữ liệu MySQL, nhưng nó sẽ cung cấp những kiến thức căn bản. Một vài tham chiếu SQL trực tuyến gồm:

- SQL Course (*www.sqlcourse.com*).
- A Gentle Introduction to SQL (*www.sqlzoo.net*) - (xem hình D-9).
- SQL Course 2 (*www.sqlcourse2.com*).
- W3Schools.com (*www.w3schools.com/sql/default.asp*) - (xem hình D-10).
- SearchDatabase.com (*www.searchdatabase.com*): nhà cung cấp nội dung cơ sở dữ liệu.
- SQL.org (*www.sql.org*): cổng truy cập cho phần lớn những gì có liên quan với cơ sở dữ liệu.
- Web Developer's Virtual Library: (*http://wdvl.internet.com/Authoring/DB/*).



Hình D-9: *A Gentle Introduction to SQL là hướng dẫn về ngôn ngữ SQL thích hợp cho những người mới bắt đầu.*

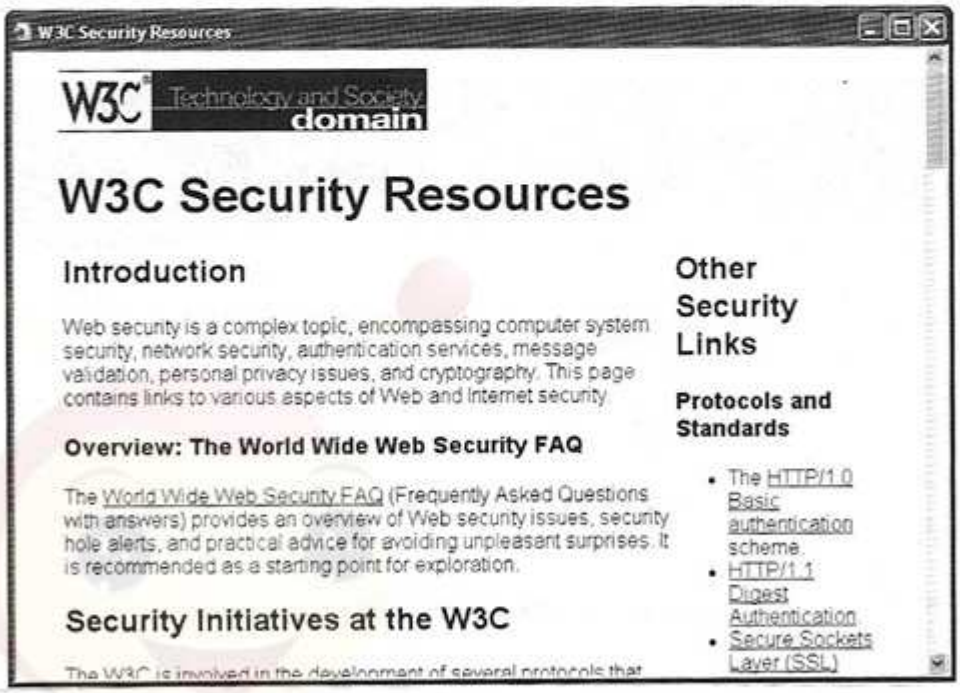


Hình D-10: *Website W3School có nhiều thông tin hữu ích về SQL, HTML, CSS và nhiều thông tin khác.*

An toàn

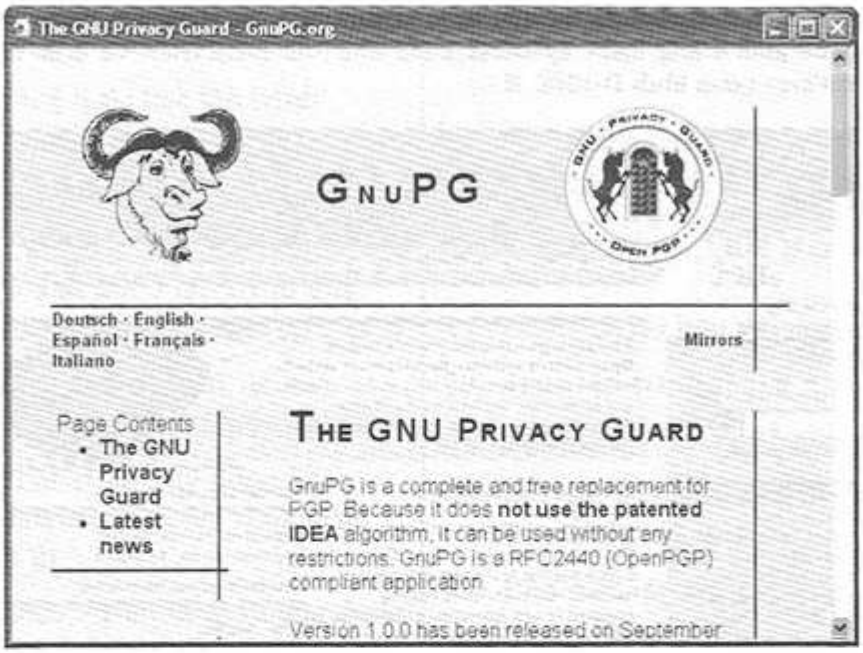
An toàn máy chủ Web, hệ điều hành và PHP là những đề mục đáng được đề cập đến trong những quyển sách riêng nói về chúng. Sẽ không có lợi nếu nói về an toàn mà không có được thông tin cập nhật. Vì thế, cách tốt nhất để nắm được vấn đề an toàn mới nhất là theo dõi các Website sau:

- CERT (*www.cert.org*) là một viện liên kết với Trường Đại học Carnegie Mellon, làm nhiệm vụ kiểm tra và báo cáo các vấn đề an toàn thích hợp trên Internet.
- SecurityFocus (*www.securityfocus.com*) liệt kê các vấn đề có liên quan đến an toàn chủ yếu cho hệ điều hành Windows và Unix. Nó cũng kiểm tra các lỗi và vi rút mà những người phụ trách Web cần phải biết.
- Insecure.org (*www.insecure.org*) được phát triển từ quan điểm của một hacker. Nó cung cấp nhiều công cụ và thông tin an toàn, chủ yếu cho Unix.
- The W3C's Security Resources (*www.w3.org/Security/*) là bản tóm tắt thông tin an toàn Web của World Wide Web Consortium (xem hình D-11).



Hình D-11: *Một số các tham chiếu liệt kê tại site World Wide Web Consortium là những liên kết nói đến vấn đề an toàn.*

- GnuPG (the Gnu Privacy Guard) (*www.gnupg.org*) là một dạng miễn phí của PGP (Pretty Good Privacy). Nó được dùng để mã hóa và giải mã các thông tin (xem hình D-12).



Hình D-12: *GnuPG chạy trên hầu hết các hệ điều hành, bao gồm Windows, Linux và Mac OS X.*

- A Study in Scarlet (*www.secureReality.com.au/studyinscarlet.txt*) là một trang giới thiệu về một số các vấn đề an toàn cụ thể của PHP.
- Tài liệu MySQL có một phần dành riêng về an toàn. Nó có tại địa chỉ *www.mysql.com/doc/G/e/General_security.html*.

Kể từ phiên bản 4.0, MySQL có sử dụng SSL (Secure Sockets Layer) để kết nối an toàn với cơ sở dữ liệu. Trong tài liệu về MySQL cũng có mô tả việc dùng SSH (Secure Shell) với cách thức tương tự. Cả hai đều đáng được xem xét mỗi khi cần thực hiện các giao dịch an toàn.

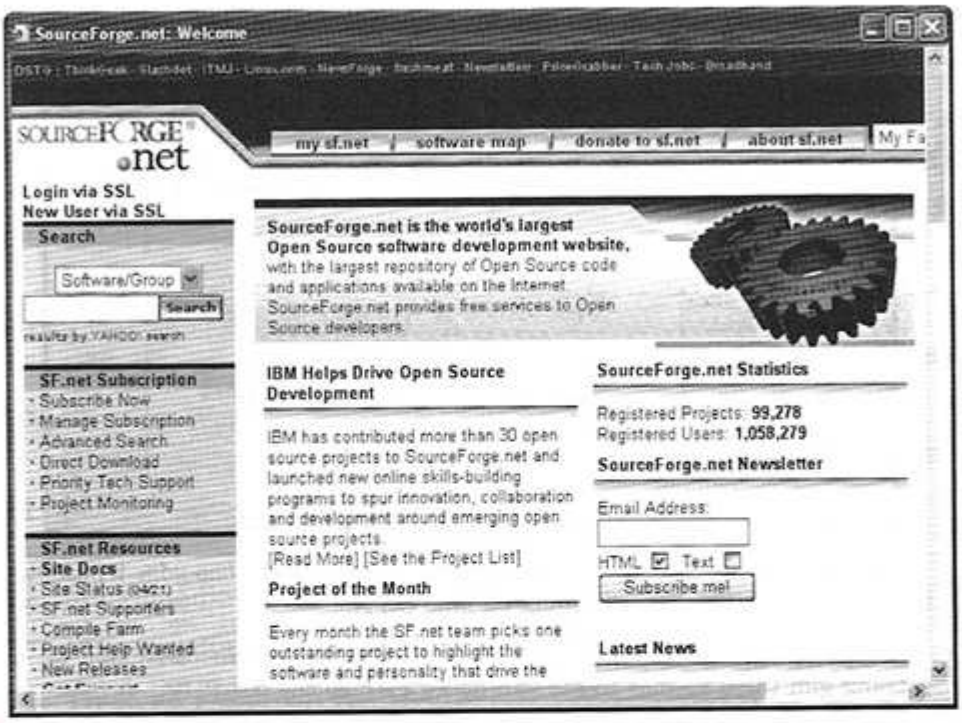
Các nội dung khác

Có một số các liên kết Web cũng đáng được để ý đến mặc dù nó không phù hợp với một trong số đề mục được nêu trong phụ lục này. Một số liên kết đề cập đến sự phát triển Web nói chung và một số có liên quan đặc biệt với những kỹ thuật được đề cập trong sách.

Tổng quát

E-Gineer (*www.e-gineer.com*) là một site thông tin Internet. Dù không phải là một site đặc biệt về PHP, nó vẫn có những thông tin bổ ích cho việc cài đặt và sử dụng ngôn ngữ.

SourceForge (*www.sourceforge.net*) được xem là kho ứng dụng mã nguồn mở lớn nhất thế giới. Hàng ngàn kỹ thuật khác nhau đã phát triển và được đặt vào SourceForge (xem hình D-13).



Hình D-13: *SourceForge là trang chủ có gần 60.000 dự án nguồn mở.*

Phát triển Web

Developer Shed (*www.devshed.com*) là site phát triển Web nói chung, bao gồm các chủ đề về các kỹ thuật và ngôn ngữ lập trình khác nhau mà bạn có thể sử dụng.

WebMonkey (*www.webmonkey.com*) rất giống vớiDevShed dù lãnh vực sử dụng rộng hơn.

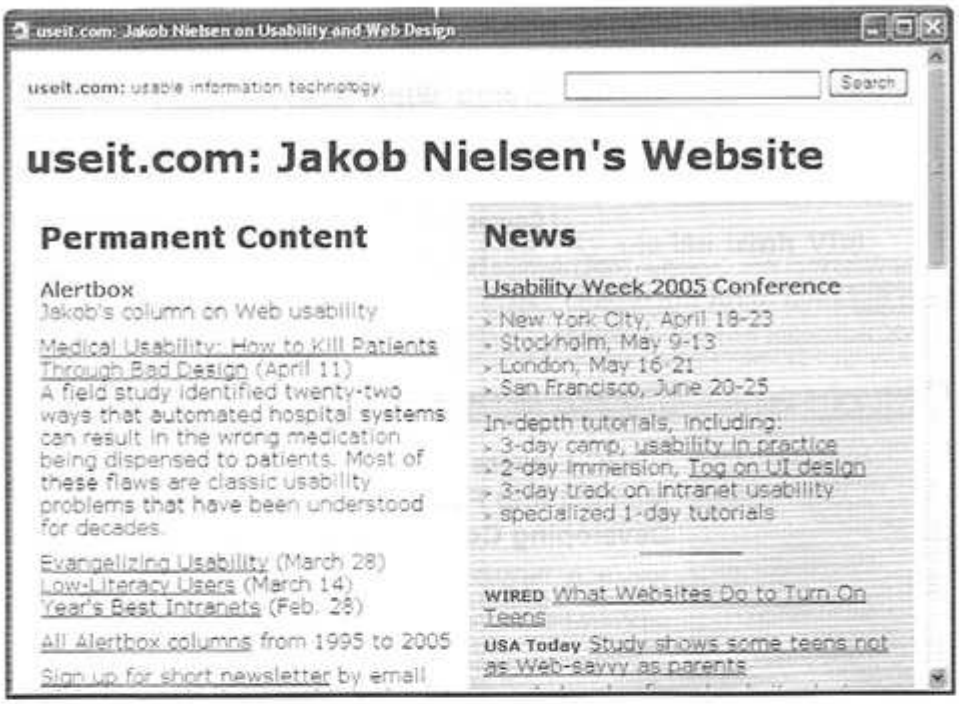
HTML

Nguồn hiển nhiên nhất đáng quan tâm về HTML là trang chủ của World Wide Web Consortium (*www.w3.org*) - (xem hình D-14). Sau đó, bạn cũng nên quan tâm đến một số các nguồn khác được liệt kê trước đây, bao gồm *W3School.com*, *WebMonkey.com* và *DevShed.com*.

Hình D-14: *Website W3C liệt kê tất cả các mô tả kỹ thuật của HTML.*

Có rất nhiều ứng dụng kiểm tra HTML có sẵn trên mạng và được sử dụng miễn phí. Chúng sẽ cho bạn biết (ngay cả với những trang do PHP tạo ra) mã HTML kết quả của một liên kết có vấn đề gì không. Khái niệm này ngày càng trở nên quan trọng khi XHTML và XML được chấp thuận. Hai ứng dụng kiểm tra tính hợp lệ miễn phí tốt nhất là của W3C (*http://validator.w3.org*) và Bobby (*www.cast.org/bobby*).

UseIt.com (*www.useit.com*) là Website thảo luận về tính hữu dụng của Web, giới thiệu những điều nên làm và không nên làm khi phát triển Web (xem hình D-15).



Hình D-15: *UseIt.com đề cập đến tính hữu dụng của Web.*

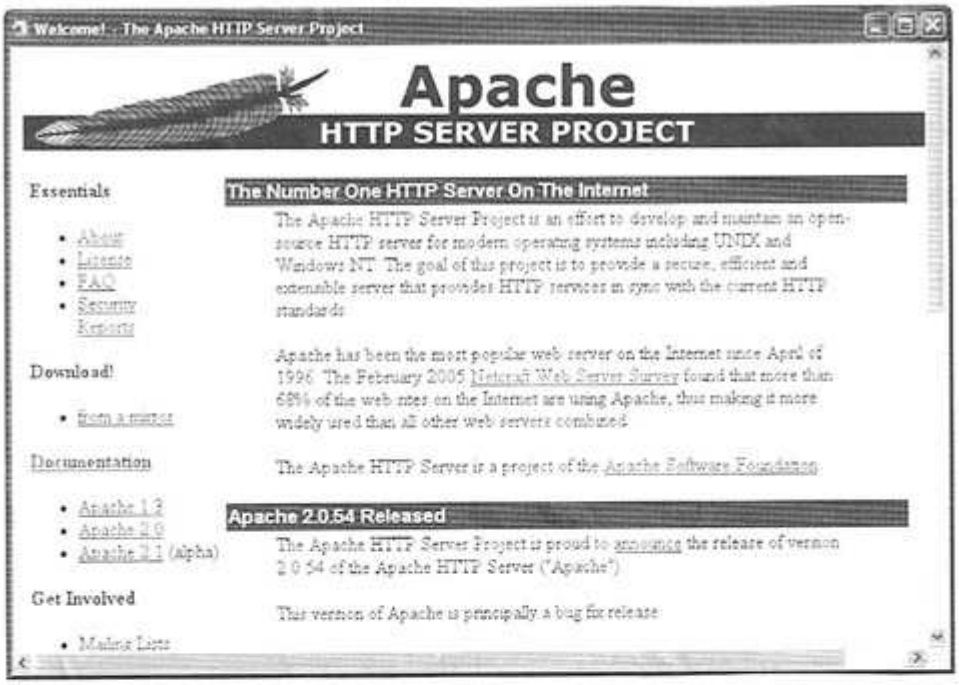
JavaScript

Các nguồn JavaScript trên Web có khuynh hướng không nhất quán. Bạn cần thường xuyên tham khảo nhiều nguồn để tìm ra những thông tin cần thiết. Khi nhu cầu phát sinh, hãy bắt đầu với:

- *www.javascript.com*
- *javascript.internet.com*
- *www.jworld.com*

Máy chủ Apache

Vì Apache là máy chủ Web được sử dụng phổ biến nhất, đặc biệt là trên hệ điều hành không phải Windows, nên có một số Website được dành riêng cho phần mềm này. Bạn nên tham khảo tài liệu có trên trang chủ của Apache (*http://http.apache.org*) - (xem hình D-16). Ngoài ra, còn có nhiều thông tin đáng quan tâm tại Apache Week (*www.apacheweek.com*) và Apache Today (*www.apachetoday.com*).



Hình D-16: *Trang chủ của máy chủ Web Apache.*

Thương mại điện tử

Có nhiều điều về thương mại điện tử trên Internet mà bạn có thể phát triển, nhưng có hai điều cần phải biết: các giao dịch phải an toàn và phải có khả năng thanh toán.

Để làm được những yêu cầu trên, cần phải cài đặt SSL và có một chứng nhận điện tử. Bạn có thể lấy chứng nhận điện tử (một cách chứng minh tính an toàn khi mua sắm trên site) từ VeriSign (*http://www.verisign.com*) hoặc Thawte (*http://thawte.com*).

Sử dụng PHP và MySQL
Thiết kế Web động

Chịu trách nhiệm xuất bản:
CÁT VĂN THÀNH

Chịu trách nhiệm bản thảo:
TS. THÁI THANH BẢY

Biên soạn:
NGUYỄN TRƯỜNG SINH

Biên tập:
THANH DUY

Sửa bản in:
NXB THỐNG KÊ

Trình bày bìa:
VIỆT DŨNG

Thực hiện liên doanh: **Công ty TNHH Minh Khai S.G**
E-mail: mk.book@minhkhai.com.vn – mk.pub@minhkhai.com.vn
Website: www.minhkhai.com.vn

Tổng phát hành

- **TP.HCM:** Nhà sách Minh Khai - 249 Nguyễn Thị Minh Khai - Quận 1
  ĐT: (08) 9.250590 - 9.250591. Fax: (08) 8.331124
- **Hà Nội:** Nhà sách Minh Châu - Nhà 30 - Ngõ 22 - Tạ Quang Bửu - Bách khoa
  ĐT: (04) 8.692785. Fax: (04) 8.683995

In 4.000 cuốn, khổ 16 x 24 cm, tại Xí nghiệp in Machinco
Số 21 Bùi Thị Xuân, Q.1, TP.Hồ Chí Minh
Giấy chấp nhận đăng ký kế hoạch xuất bản số 54/XB-QLXB
do Cục xuất bản cấp ngày 17/01/2005
Mã số : $\frac{6T7 - 6T73}{TK2005}$ 337 – 54 – 2005
In xong và nộp lưu chiểu tháng 8 năm 2005