Luke Welling
Laura Thomson

Fourth Edition

PHP and MySQL® Web Development

Developer's Library

PHP and MySQL® Web Development, Fourth Edition

Library of Congress Cataloging-in-Publication Data
Welling, Luke, 1972-
PHP and MySQL Web development / Luke Welling, Laura Thomson. – 4th ed.
p. cm.
ISBN 978-0-672-32916-6 (pbk. w/cd)
1. PHP (Computer program language) 2. SQL (Computer program language) 3. MySQL (Electronic resource) 4. Web sites–Design. I. Thomson, Laura. II. Title.
QA76.73.P224W45 2008
005.2'762–dc22

2008036492

Printed in the United States of America

First Printing: September 2008

ISBN-10: 0-672-32916-6
ISBN-13: 978-0-672-32916-6

Trademarks

All terms mentioned in this book that are known to be trademarks or service marks have been appropriately capitalized. Pearson Education, Inc. cannot attest to the accuracy of this information. Use of a term in this book should not be regarded as affecting the validity of any trademark or service mark.

Warning and Disclaimer

Every effort has been made to make this book as complete and as accurate as possible, but no warranty or fitness is implied. The information provided is on an "as is" basis. The authors and the publisher shall have neither liability nor responsibility to any person or entity with respect to any loss or damages arising from the information contained in this book or from the use of the CD-ROM or programs accompanying it.

Bulk Sales

Pearson Education, Inc. offers excellent discounts on this book when ordered in quantity for bulk purchases or special sales. For more information, please contact

U.S. Corporate and Government Sales
1-800-382-3419
corpsales@pearsontechgroup.com

For sales outside the U.S., please contact

International Sales
international@pearsoned.com

Acquisitions Editor
Mark Taber

Development Editor
Michael Thurston

Managing Editor
Patrick Kanouse

Project Editor
Jennifer Gallant

Copy Editor
Barbara Hacha

Indexer
Tim Wright

Proofreader
Kathy Ruiz

Technical Editor
Tim Boronczyk

Publishing Coordinator
Vanessa Evans

Multimedia Developer
Dan Scherf

Book Designer
Gary Adair

Composition
Bronkella Publishing

Introduction

Welcome to *PHP and MySQL Web Development*. Within its pages, you will find distilled knowledge from our experiences using PHP and MySQL, two of the hottest web development tools around.

In this introduction, we cover

- Why you should read this book
- What you will be able to achieve using this book
- What PHP and MySQL are and why they're great
- What's changed in the latest versions of PHP and MySQL
- How this book is organized

Let's get started.

Why You Should Read This Book

This book will teach you how to create interactive websites from the simplest order form through to complex, secure e-commerce sites or interactive Web 2.0 sites. What's more, you'll learn how to do it using open source technologies.

This book is aimed at readers who already know at least the basics of HTML and have done some programming in a modern programming language before but have not necessarily programmed for the Internet or used a relational database. If you are a beginning programmer, you should still find this book useful, but digesting it might take a little longer. We've tried not to leave out any basic concepts, but we do cover them at speed. The typical readers of this book want to master PHP and MySQL for the purpose of building a large or commercial website. You might already be working in another web development language; if so, this book should get you up to speed quickly.

We wrote the first edition of this book because we were tired of finding PHP books that were basically function references. These books are useful, but they don't help when your boss or client has said, "Go build me a shopping cart." In this book, we have done our best to make every example useful. You can use many of the code samples directly in your website, and you can use many others with only minor modifications.

What You Will Learn from This Book

Reading this book will enable you to build real-world, dynamic websites. If you've built websites using plain HTML, you realize the limitations of this approach. Static content from a pure HTML website is just that—static. It stays the same unless you physically update it. Your users can't interact with the site in any meaningful fashion.

Using a language such as PHP and a database such as MySQL allows you to make your sites dynamic: to have them be customizable and contain real-time information.

We have deliberately focused this book on real-world applications, even in the introductory chapters. We begin by looking at a simple online ordering system and work our way through the various parts of PHP and MySQL.

We then discuss aspects of electronic commerce and security as they relate to building a real-world website and show you how to implement these aspects in PHP and MySQL.

In the final part of this book, we describe how to approach real-world projects and take you through the design, planning, and building of the following projects:

- User authentication and personalization
- Shopping carts
- Web-based email
- Mailing list managers
- Web forums
- PDF document generation
- Web services with XML and SOAP
- Web 2.0 application with Ajax

You should be able to use any of these projects as is, or you can modify them to suit your needs. We chose them because we believe they represent some the most common web-based applications built by programmers. If your needs are different, this book should help you along the way to achieving your goals.

What Is PHP?

PHP is a server-side scripting language designed specifically for the Web. Within an HTML page, you can embed PHP code that will be executed each time the page is visited. Your PHP code is interpreted at the web server and generates HTML or other output that the visitor will see.

PHP was conceived in 1994 and was originally the work of one man, Rasmus Lerdorf. It was adopted by other talented people and has gone through four major rewrites to bring us the broad, mature product we see today. As of November 2007, it was installed on more than 21 million domains worldwide, and this number is growing rapidly. You can see the current number at http://www.php.net/usage.php.

PHP is an Open Source project, which means you have access to the source code and can use, alter, and redistribute it all without charge.

PHP originally stood for *Personal Home Page* but was changed in line with the GNU recursive naming convention (GNU = Gnu's Not Unix) and now stands for *PHP Hypertext Preprocessor*.

The current major version of PHP is 5. This version saw a complete rewrite of the underlying Zend engine and some major improvements to the language.

The home page for PHP is available at http://www.php.net.

The home page for Zend Technologies is http://www.zend.com.

What Is MySQL?

MySQL (pronounced *My-Ess-Que-Ell*) is a very fast, robust, *relational database management system (RDBMS)*. A database enables you to efficiently store, search, sort, and retrieve data. The MySQL server controls access to your data to ensure that multiple users can work with it concurrently, to provide fast access to it, and to ensure that only authorized users can obtain access. Hence, MySQL is a multiuser, multithreaded server. It uses *Structured Query Language (SQL)*, the standard database query language. MySQL has been publicly available since 1996 but has a development history going back to 1979. It is the world's most popular open source database and has won the Linux Journal Readers' Choice Award on a number of occasions.

MySQL is available under a dual licensing scheme. You can use it under an open source license (the GPL) free as long as you are willing to meet the terms of that license. If you want to distribute a non-GPL application including MySQL, you can buy a commercial license instead.

Why Use PHP and MySQL?

When setting out to build a website, you could use many different products. You need to choose the following:

- Hardware for the web server
- An operating system
- Web server software
- A database management system
- A programming or scripting language

Some of these choices are dependent on the others. For example, not all operating systems run on all hardware, not all web servers support all programming languages, and so on.

In this book, we do not pay much attention to hardware, operating systems, or web server software. We don't need to. One of the best features of both PHP and MySQL is that they work with any major operating system and many of the minor ones.

The majority of PHP code can be written to be portable between operating systems and web servers. There are some PHP functions that specifically relate to the filesystem that are operating system dependent, but these are clearly marked as such in the manual and in this book.

Whatever hardware, operating system, and web server you choose, we believe you should seriously consider using PHP and MySQL.

Some of PHP's Strengths

Some of PHP's main competitors are Perl, Microsoft ASP.NET, Ruby (on Rails or otherwise), JavaServer Pages (JSP), and ColdFusion.

In comparison to these products, PHP has many strengths, including the following:

- Performance
- Scalability
- Interfaces to many different database systems
- Built-in libraries for many common web tasks
- Low cost
- Ease of learning and use
- Strong object-oriented support
- Portability
- Flexibility of development approach
- Availability of source code
- Availability of support and documentation

A more detailed discussion of these strengths follows.

Performance

PHP is very fast. Using a single inexpensive server, you can serve millions of hits per day. Benchmarks published by Zend Technologies (http://www.zend.com) show PHP outperforming its competition.

Scalability

PHP has what Rasmus Lerdorf frequently refers to as a "shared-nothing" architecture. This means that you can effectively and cheaply implement horizontal scaling with large numbers of commodity servers.

Database Integration

PHP has native connections available to many database systems. In addition to MySQL, you can directly connect to PostgreSQL, Oracle, dbm, FilePro, DB2, Hyperwave, Informix, InterBase, and Sybase databases, among others. PHP 5 also has a built-in SQL interface to a flat file, called SQLite.

Using the *Open Database Connectivity Standard (ODBC)*, you can connect to any database that provides an ODBC driver. This includes Microsoft products and many others.

In addition to native libraries, PHP comes with a database access abstraction layer called *PHP Database Objects (PDO)*, which allows consistent access and promotes secure coding practices.

Built-in Libraries

Because PHP was designed for use on the Web, it has many built-in functions for performing many useful web-related tasks. You can generate images on the fly, connect to web services and other network services, parse XML, send email, work with cookies, and generate PDF documents, all with just a few lines of code.

Cost

PHP is free. You can download the latest version at any time from http://www.php.net for no charge.

Ease of Learning PHP

The syntax of PHP is based on other programming languages, primarily C and Perl. If you already know C or Perl, or a C-like language such as C++ or Java, you will be productive using PHP almost immediately.

Object-Oriented Support

PHP version 5 has well-designed object-oriented features. If you learned to program in Java or C++, you will find the features (and generally the syntax) that you expect, such as inheritance, private and protected attributes and methods, abstract classes and methods, interfaces, constructors, and destructors. You will even find some less common features such as iterators. Some of this functionality was available in PHP versions 3 and 4, but the object-oriented support in version 5 is much more complete.

Portability

PHP is available for many different operating systems. You can write PHP code on free Unix-like operating systems such as Linux and FreeBSD, commercial Unix versions such as Solaris and IRIX, OS X, or on different versions of Microsoft Windows.

Well-written code will usually work without modification on a different system running PHP.

Flexibility of Development Approach

PHP allows you to implement simple tasks simply, and equally easily adapts to implementing large applications using a framework based on design patterns such as Model–View–Controller (MVC).

Source Code

You have access to PHP's source code. With PHP, unlike commercial, closed-source products, if you want to modify something or add to the language, you are free to do so.

You do not need to wait for the manufacturer to release patches. You also don't need to worry about the manufacturer going out of business or deciding to stop supporting a product.

Availability of Support and Documentation

Zend Technologies (www.zend.com), the company behind the engine that powers PHP, funds its PHP development by offering support and related software on a commercial basis.

The PHP documentation and community are mature and rich resources with a wealth of information to share.

What Is New in PHP 5?

You may have recently moved to PHP 5 from one of the PHP 4.x versions. As you would expect in a new major version, it has some significant changes. The Zend engine beneath PHP has been rewritten for this version. Major new features are as follows:

- Better object-oriented support built around a completely new object model (see Chapter 6, "Object-Oriented PHP")
- Exceptions for scalable, maintainable error handling (see Chapter 7, "Error and Exception Handling")
- SimpleXML for easy handling of XML data (see Chapter 33, "Connecting to Web Services with XML and SOAP")

Other changes include moving some extensions out of the default PHP install and into the PECL library, improving streams support, and adding SQLite.

At the time of writing, PHP 5.2 was the current version, with PHP 5.3 on the near horizon. PHP 5.2 added a number of useful features including:

- The new input filtering extension for security purposes
- JSON extension for better JavaScript interoperability
- File upload progress tracking
- Better date and time handling
- Many upgraded client libraries, performance improvements (including better memory management in the Zend Engine), and bug fixes

Key Features of PHP 5.3

You may have heard about a new major release of PHP, called PHP 6. At the time of this writing, PHP 6 is not in the release candidate stage, and hosting providers won't be

installing it for mass use for quite some time. However, some of the key features planned in PHP 6 have been back-ported to PHP 5.3, which is a minor version release and closer to passing acceptance testing and thus installation by hosting providers (of course, if you are your own server's administrator, you can install any version you like).
Some of the new features in PHP 5.3 are listed below; additional information also appears throughout this book as appropriate:

- The addition of namespaces; for more information see http://www.php.net/language.namespaces
- The addition of the `intl` extension for application internationalization; for more information see http://www.php.net/manual/en/intro.intl.php
- The addition of the `phar` extension for creating self-contained PHP application archives; for more information see http://www.php.net/book.phar
- The addition of the `fileinfo` extension for enhanced ability to work with files; for more information see http://www.php.net/manual/en/book.fileinfo.php
- The addition of the `sqlite3` extension for working with the SQLite Embeddable SQL Database Engine; for more information see http://www.php.net/manual/en/class.sqlite3.php
- The inclusion of support for the MySQLnd driver, a replacement for libmysql; for more information see http://forge.mysql.com/wiki/PHP_MYSQLND

While the list above contains some of the highly-touted features of PHP 5.3, the release also includes a significant number of bug fixes and maintenance performed on existing functionality, such as:

- Removing support for any version of Windows older than Windows 2000 (such as Windows 98 and NT4)
- Ensuring the PCRE, Reflection, and SPL extensions are always enabled
- Adding a few date and time functions for ease of date calculation and manipulation
- Improving the `crypt()`, `hash()`, and `md5()` functionality, as well as improving the OpenSSL extension
- Improving php.ini administration and handling, including better error reporting
- Continuing to fine-tune the Zend engine for better PHP runtime speed and memory usage

Some of MySQLs Strengths

MySQLs main competitors are PostgreSQL, Microsoft SQL Server, and Oracle.
MySQL has many strengths, including the following:

- High performance
- Low cost

- Ease of configuration and learning
- Portability
- Availability of source code
- Availability of support

A more detailed discussion of these strengths follows.

Performance

MySQL is undeniably fast. You can see the developers' benchmark page at http://web.mysql.com/whymysql/benchmarks. Many of these benchmarks show MySQL to be orders of magnitude faster than the competition. In 2002, *eWeek* published a benchmark comparing five databases powering a web application. The best result was a tie between MySQL and the much more expensive Oracle.

Low Cost

MySQL is available at no cost under an open source license or at low cost under a commercial license. You need a license if you want to redistribute MySQL as part of an application and do not want to license your application under an Open Source license. If you do not intend to distribute your application—typical for most web applications, or are working on free or open source Software, you do not need to buy a license.

Ease of Use

Most modern databases use SQL. If you have used another RDBMS, you should have no trouble adapting to this one. MySQL is also easier to set up than many similar products.

Portability

MySQL can be used on many different Unix systems as well as under Microsoft Windows.

Source Code

As with PHP, you can obtain and modify the source code for MySQL. This point is not important to most users most of the time, but it provides you with excellent peace of mind, ensuring future continuity and giving you options in an emergency.

Availability of Support

Not all open source products have a parent company offering support, training, consulting, and certification, but you can get all of these benefits from MySQL AB (www.mysql.com).

What Is New in MySQL 5?

Major changes introduced for MySQL 5 include

- Views
- Stored procedures (see Chapter 13, "Advanced MySQL Programming")
- Basic trigger support
- Cursor support

Other changes include more ANSI standard compliance and speed improvements. If you are still using an early 4.x version or a 3.x version of the MySQL server, you should know that the following features were added to various versions from 4.0:

- Subquery support
- GIS types for storing geographical data
- Improved support for internationalization
- The transaction-safe storage engine InnoDB included as standard
- The MySQL query cache, which greatly improves the speed of repetitive queries as often run by web applications

This book was written using MySQL 5.1 (Beta Community Edition). This version also added support for

- Partitioning
- Row based replication
- Event scheduling
- Logging to tables
- Improvements to MySQL Cluster, information schema, backup processes, and many bug fixes

How Is This Book Organized?

This book is divided into five main parts:

Part I, "Using PHP," provides an overview of the main parts of the PHP language with examples. Each example is a real-world example used in building an e-commerce site rather than "toy" code. We kick off this section with Chapter 1, "PHP Crash Course." If you've already used PHP, you can whiz through this chapter. If you are new to PHP or new to programming, you might want to spend a little more time on it. Even if you are quite familiar with PHP but you are new to PHP 5, you will want to read Chapter 6, "Object-Oriented PHP," because the object-oriented functionality has changed significantly.

Part II, "Using MySQL," discusses the concepts and design involved in using relational database systems such as MySQL, using SQL, connecting your MySQL database to the world with PHP, and advanced MySQL topics, such as security and optimization.

Part III, "E-commerce and Security," covers some of the general issues involved in developing a website using any language. The most important of these issues is security. We then discuss how you can use PHP and MySQL to authenticate your users and securely gather, transmit, and store data.

Part IV, "Advanced PHP Techniques," offers detailed coverage of some of the major built-in functions in PHP. We have selected groups of functions that are likely to be useful when building a website. You will learn about interaction with the server, interaction with the network, image generation, date and time manipulation, and session variables.

Part V, "Building Practical PHP and MySQL Projects," is our favorite section. It deals with practical real-world issues such as managing large projects and debugging, and provides sample projects that demonstrate the power and versatility of PHP and MySQL.

Finally

We hope you enjoy this book and enjoy learning about PHP and MySQL as much as we did when we first began using these products. They are really a pleasure to use. Soon, you'll be able to join the many thousands of web developers who use these robust, powerful tools to easily build dynamic, real-time websites.

11

Accessing Your MySQL Database from the Web with PHP

PREVIOUSLY, IN OUR WORK WITH PHP, WE used a flat file to store and retrieve data. When we looked at this file in Chapter 2, "Storing and Retrieving Data," we mentioned that relational database systems make a lot of these storage and retrieval tasks easier, safer, and more efficient in a web application. Now, having worked with MySQL to create a database, we can begin connecting this database to a web-based front end.

In this chapter, we explain how to access the Book-O-Rama database from the Web using PHP. You learn how to read from and write to the database and how to filter potentially troublesome input data.

Key topics covered in this chapter include

- How web database architectures work
- Querying a database from the Web using the basic steps
- Setting up a connection
- Getting information about available databases
- Choosing a database to use
- Querying the database
- Retrieving the query results
- Disconnecting from the database
- Putting new information in the database
- Using prepared statements
- Using other PHP-database interfaces
- Using a generic database interface: PEAR MDB2

How Web Database Architectures Work

In Chapter 8, "Designing Your Web Database," we outlined how web database architectures work. Just to remind you, here are the steps:

1. A user's web browser issues an HTTP request for a particular web page. For example, the user might have requested a search for all the books written by Michael Morgan at Book-O-Rama, using an HTML form. The search results page is called `results.php`.
2. The web server receives the request for `results.php`, retrieves the file, and passes it to the PHP engine for processing.
3. The PHP engine begins parsing the script. Inside the script is a command to connect to the database and execute a query (perform the search for books). PHP opens a connection to the MySQL server and sends on the appropriate query.
4. The MySQL server receives the database query, processes it, and sends the results—a list of books—back to the PHP engine.
5. The PHP engine finishes running the script. This usually involves formatting the query results nicely in HTML. It then returns the resulting HTML to the web server.
6. The web server passes the HTML back to the browser, where the user can see the list of books she requested.

Now you have an existing MySQL database, so you can write the PHP code to perform the preceding steps. Begin with the search form. The code for this plain HTML form is shown in Listing 11.1.

Listing 11.1 `search.html`— **Book-O-Rama's Database Search Page**

```
<html>
<head>
  <title>Book-O-Rama Catalog Search</title>
</head>

<body>
  <h1>Book-O-Rama Catalog Search</h1>

  <form action="results.php" method="post">
    Choose Search Type:<br />
    <select name="searchtype">
      <option value="author">Author</option>
      <option value="title">Title</option>
      <option value="isbn">ISBN</option>
    </select>
    <br />
    Enter Search Term:<br />
    <input name="searchterm" type=""text" size="40"/>
    <br />
```

Listing 11.1 **Continued**

```
    <input type="submit" name="submit" value="Search"/>
  </form>

</body>
</html>
```

This HTML form is reasonably straightforward. The output of this HTML is shown in Figure 11.1.

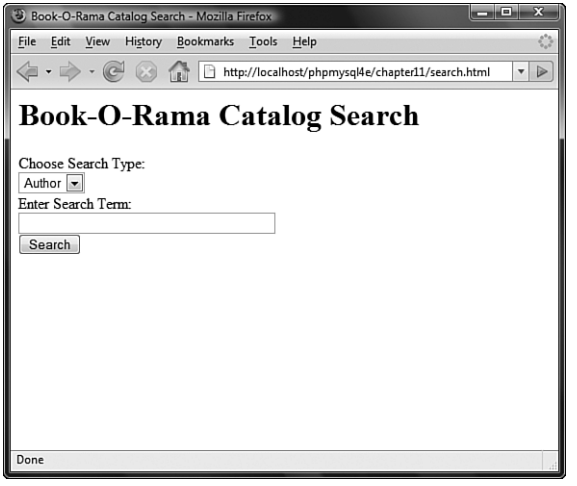

Figure 11.1 The search form is quite general, so you can search for a book by its title, author, or ISBN.

The script that will be called when the Search button is clicked is `results.php`. It is listed in full in Listing 11.2. Through the course of this chapter, we discuss what this script does and how it works.

Listing 11.2 `results.php`**—This Script Retrieves Search Results from the MySQL Database and Formats Them for Display**

```
<html>
<head>
  <title>Book-O-Rama Search Results</title>
</head>
<body>
<h1>Book-O-Rama Search Results</h1>
<?php
  // create short variable names
  $searchtype=$_POST['searchtype'];
  $searchterm=trim($_POST['searchterm']);
```

Listing 11.2 **Continued**

```
  if (!$searchtype || !$searchterm) {
     echo 'You have not entered search details.  Please go back and try again.';
     exit;
  }

  if (!get_magic_quotes_gpc()){
    $searchtype = addslashes($searchtype);
    $searchterm = addslashes($searchterm);
  }

  @ $db = new mysqli('localhost', 'bookorama', 'bookorama123', 'books');

  if (mysqli_connect_errno()) {
     echo 'Error: Could not connect to database.  Please try again later.';
     exit;
  }

  $query = "select * from books where ".$searchtype." like '%".$searchterm."%'";
  $result = $db->query($query);

  $num_results = $result->num_rows;

  echo "<p>Number of books found: ".$num_results."</p>";

  for ($i=0; $i <$num_results; $i++) {
     $row = $result->fetch_assoc();
     echo "<p><strong>".($i+1).". Title: ";
     echo htmlspecialchars(stripslashes($row['title']));
     echo "</strong><br />Author: ";
     echo stripslashes($row['author']);
     echo "<br />ISBN: ";
     echo stripslashes($row['isbn']);
     echo "<br />Price: ";
     echo stripslashes($row['price']);
     echo "</p>";
  }

  $result->free();
  $db->close();

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

Note that this script allows you to enter the MySQL wildcard characters `%` and _ (underscore). This capability can be useful for the user, but you can escape these characters if they will cause a problem for your application.

Figure 11.2 illustrates the results of using this script to perform a search.

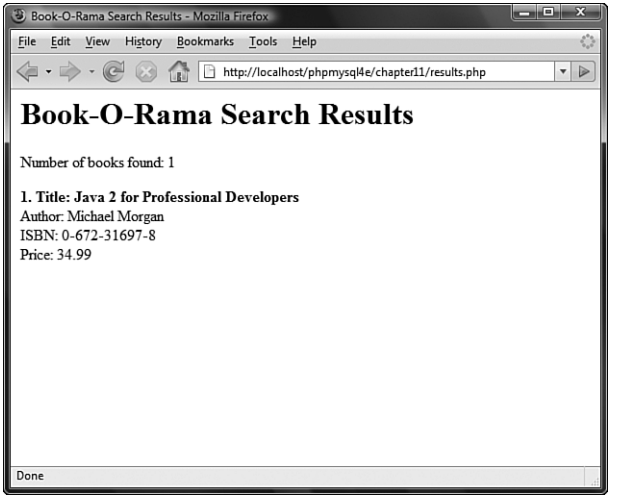

Figure 11.2 The results of searching the database for books about Java are presented in a web page using the `results.php` script.

Querying a Database from the Web

In any script used to access a database from the Web, you follow some basic steps:

1. Check and filter data coming from the user.
2. Set up a connection to the appropriate database.
3. Query the database.
4. Retrieve the results.
5. Present the results back to the user.

These are the steps we followed in the script `results.php`, so now let's go through each of them in turn.

Checking and Filtering Input Data

You begin the script by stripping any whitespace that the user might have inadvertently entered at the beginning or end of his search term. You do this by applying the function `trim()` to the value of `$_POST['searchterm']` when giving it a shorter name:

```
$searchterm=trim($_POST['searchterm']);
```

The next step is to verify that the user has entered a search term and selected a search type. Note that you check whether the user entered a search term after trimming whitespace from the ends of `$searchterm`. If you arrange these lines in the opposite order, you could encounter situations in which a user's search term is not empty and therefore does not create an error message; instead, it is all whitespace, so it is deleted by `trim()`:

```
if (!$searchtype || !$searchterm) {
   echo "You have not entered search details.  Please go back and try again.";
   exit;
}
```

You check the `$searchtype` variable, even though in this case it's coming from an HTML `SELECT`. You might ask why you should bother checking data that has to be filled in. It's important to remember that there might be more than one interface to your database. For example, Amazon has many affiliates who use its search interface. Also, it's sensible to screen data in case of any security problems that can arise because of users coming from different points of entry.

When you plan to use any data input by a user, you need to filter it appropriately for any control characters. As you might remember, in Chapter 4, "String Manipulation and Regular Expressions," we described the functions `addslashes()`, `stripslashes()`, and `get_magic_quotes_gpc()`. You need to escape data when submitting any user input to a database such as MySQL.

In this case, you check the value of the `get_magic_quotes_gpc()` function. It tells you whether quoting is being done automatically. If it is not, you use `addslashes()` to escape the data:

```
if (!get_magic_quotes_gpc()) {
  $searchtype = addslashes($searchtype);
  $searchterm = addslashes($searchterm);
}
```

You also use `stripslashes()` on the data coming back from the database. If the magic quotes feature is turned on, the data will have slashes in it when it comes back from the database, so you need to take them out.

Here you use the function `htmlspecialchars()` to encode characters that have special meanings in HTML. The current test data does not include any ampersands (`&`), less than (`<`), greater than (`>`), or double quotation mark (`"`) symbols, but many fine book titles contain an ampersand. By using this function, you can eliminate future errors.

Setting Up a Connection

The PHP library for connecting to MySQL is called mysqli (the *i* stands for improved). When using the mysqli library in PHP, you can use either an object-oriented or procedural syntax.

You use the following line in the script to connect to the MySQL server:

```
@ $db = new mysqli('localhost', 'bookorama', 'bookorama123', 'books');
```

This line instantiates the `mysqli` class and creates a connection to host `localhost` with username `bookorama`, and password `bookorama123`. The connection is set up to use the database called `books`.

Using this object-oriented approach, you can now invoke methods on this object to access the database. If you prefer a procedural approach, mysqli allows for this, too. To connect in a procedural fashion, you use

```
@ $db = mysqli_connect('localhost', 'bookorama', 'bookorama123', 'books');
```

This function returns a resource rather than an object. This resource represents the connection to the database, and if you are using the procedural approach, you will need to pass this resource in to all the other mysqli functions. This is very similar to the way the file-handling functions, such as `fopen()`, work.

Most of the mysqli functions have an object-oriented interface and a procedural interface. Generally, the differences are that the procedural version function names start with `mysqli_` and require you to pass in the resource handle you obtained from `mysqli_connect()`. Database connections are an exception to this rule because they can be made by the mysqli object's constructor.

The result of your attempt at connection is worth checking because none of the rest of code will work without a valid database connection. You do this using the following code:

```
if (mysqli_connect_errno()) {
   echo 'Error: Could not connect to database.  Please try again later.';
   exit;
}
```

(This code is the same for the object-oriented and procedural versions.) The `mysqli_connect_errno()` function returns an error number on error, or zero on success.

Note that when you connect to the database, you begin the line of code with the error suppression operator, `@`. This way, you can handle any errors gracefully. (This could also be done with exceptions, which we have not used in this simple example.)

Bear in mind that there is a limit to the number of MySQL connections that can exist at the same time. The MySQL parameter `max_connections` determines what this limit is. The purpose of this parameter and the related Apache parameter `MaxClients` is to tell the server to reject new connection requests instead of allowing machine resources to be completely used up at busy times or when software has crashed.

You can alter both of these parameters from their default values by editing the configuration files. To set `MaxClients` in Apache, edit the `httpd.conf` file on your system. To set `max_connections` for MySQL, edit the file `my.conf`.

Choosing a Database to Use

Remember that when you are using MySQL from a command-line interface, you need to tell it which database you plan to use with a command such as

```
use books;
```

You also need to do this when connecting from the Web. The database to use is specified as a parameter to the `mysqli` constructor or the `mysqli_connect()` function. If you want to change the default database, you can do so with the `mysqli_select_db()` function. It can be accessed as either

```
$db->select_db(dbname)
```

or as

```
mysqli_select_db(db_resource, db_name)
```

Here, you can see the similarity between the functions that we described before: The procedural version begins with `mysqli_` and requires the extra database handle parameter.

Querying the Database

To actually perform the query, you can use the `mysqli_query()` function. Before doing this, however, it's a good idea to set up the query you want to run:

```
$query = "select * from books where ".$searchtype." like '%".$searchterm."%'";
```

In this case, you search for the user-input value (`$searchterm`) in the field the user specified (`$searchtype`). Notice the use of `like` for matching rather than `equal`: it's usually a good idea to be more tolerant in a database search.

> **Tip**
> Remember that *the query you send to MySQL does not need a semicolon at the end of it*, unlike a query you type into the MySQL monitor.

You can now run the query:

```
$result = $db->query($query);
```

Or, if you want to use the procedural interface, you use

```
$result = mysqli_query($db, $query);
```

You pass in the query you want to run and, in the procedural interface, the database link (again, in this case `$db`).

The object-oriented version returns a result object; the procedural version returns a result resource. (This is similar to the way the connection functions work.) Either way,

you store the result in a variable (`$result`) for later use. This function returns `false` on failure.

Retrieving the Query Results

A large variety of functions is available to break the results out of the result object or identifier in different ways. The result object or identifier is the key to accessing the rows returned by the query.

In this example, you counted the number of rows returned and also used the `mysqli_fetch_assoc()` function.

When you use the object-oriented approach, the number of rows returned is stored in the `num_rows` member of the result object, and you can access it as follows:

```
$num_results = $result->num_rows;
```

When you use a procedural approach, the function `mysqli_num_rows()` gives you the number of rows returned by the query. You should pass it the result identifier, like this:

```
$num_results = mysqli_num_rows($result);
```

It's useful to know this if you plan to process or display the results, because you now know how many there are and can loop through them:

```
for ($i=0; $i <$num_results; $i++) {
  // process results
}
```

In each iteration of this loop, you call `$result->fetch_assoc()` (or `mysqli_fetch_assoc()`). The loop does not execute if no rows are returned. This is a function that takes each row from the resultset and returns the row as an array, with each key an attribute name and each value the corresponding value in the array:

```
$row = $result->fetch_assoc();
```

Or you can use a procedural approach:

```
$row = mysqli_fetch_assoc($result);
```

Given the array `$row`, you can go through each field and display it appropriately, as shown in this example:

```
echo "<br />ISBN: ";
echo stripslashes($row['isbn']);
```

As previously mentioned, you call `stripslashes()` to tidy up the value before displaying it.

Several variations can be used to get results from a result identifier. Instead of an array with named keys, you can retrieve the results in an enumerated array with `mysqli_fetch_row()`, as follows:

```
$row = $result->fetch_row($result);
```

or

```
$row = mysqli_fetch_row($result);
```

The attribute values are listed in each of the array values `$row[0]`, `$row[1]`, and so on. (The `mysqli_fetch_array()` function allows you to fetch a row as either or both kinds of array.)

You could also fetch a row into an object with the `mysqli_fetch_object()` function:

```
$row = $result->fetch_object();
```

or

```
$row = mysqli_fetch_object($result);
```

You can then access each of the attributes via `$row->title`, `$row->author`, and so on.

Disconnecting from the Database

You can free up your resultset by calling either

```
$result->free();
```

or

```
mysqli_free_result($result);
```

You can then use

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

or

```
mysqli_close($db);
```

to close a database connection. Using this command isn't strictly necessary because the connection will be closed when a script finishes execution anyway.

Putting New Information in the Database

Inserting new items into the database is remarkably similar to getting items out of the database. You follow the same basic steps: make a connection, send a query, and check the results. In this case, the query you send is an `INSERT` rather than a `SELECT`.

Although this process is similar, looking at an example can sometimes be useful. In Figure 11.3, you can see a basic HTML form for putting new books into the database.

The HTML for this page is shown in Listing 11.3.

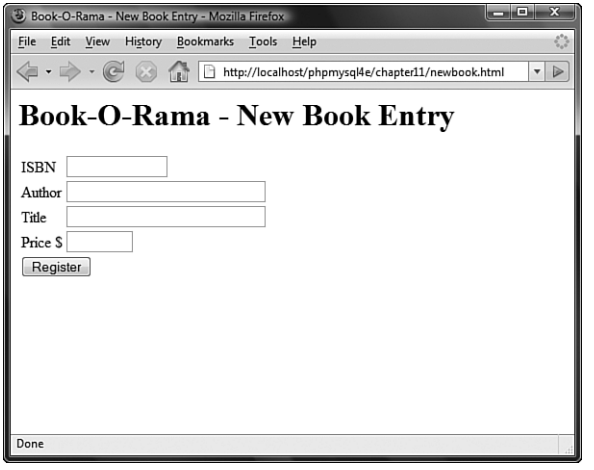

Figure 11.3 This interface for putting new books into the database could be used by Book-O-Rama's staff.

Listing 11.3 `newbook.html`— **HTML for the Book Entry Page**

```
<html>
<head>
  <title>Book-O-Rama - New Book Entry</title>
</head>

<body>
  <h1>Book-O-Rama - New Book Entry</h1>

  <form action="insert_book.php" method="post">
    <table border="0">
      <tr>
        <td>ISBN</td>
         <td><input type="text" name="isbn" maxlength="13" size="13"></td>
      </tr>
      <tr>
        <td>Author</td>
        <td> <input type="text" name="author" maxlength="30" size="30"></td>
      </tr>
      <tr>
        <td>Title</td>
        <td> <input type="text" name="title" maxlength="60" size="30"></td>
      </tr>
      <tr>
        <td>Price $</td>
        <td><input type="text" name="price" maxlength="7" size="7"></td>
```

Listing 11.3 **Continued**

```
      </tr>
      <tr>
        <td colspan="2"><input type="submit" value="Register"></td>
      </tr>
    </table>
  </form>
</body>
</html>
```

The results of this form are passed along to `insert_book.php`, a script that takes the details, performs some minor validations, and attempts to write the data into the database. The code for this script is shown in Listing 11.4.

Listing 11.4 `insert_book.php` **— This Script Writes New Books into the Database**

```
<html>
<head>
  <title>Book-O-Rama Book Entry Results</title>
</head>
<body>
<h1>Book-O-Rama Book Entry Results</h1>
<?php
  // create short variable names
  $isbn=$_POST['isbn'];
  $author=$_POST['author'];
  $title=$_POST['title'];
  $price=$_POST['price'];

  if (!$isbn || !$author || !$title || !$price) {
     echo "You have not entered all the required details.<br />"
          ."Please go back and try again.";
     exit;
  }

  if (!get_magic_quotes_gpc()) {
    $isbn = addslashes($isbn);
    $author = addslashes($author);
    $title = addslashes($title);
    $price = doubleval($price);
  }

  @ $db = new mysqli('localhost', 'bookorama', 'bookorama123', 'books');

  if (mysqli_connect_errno()) {
     echo "Error: Could not connect to database.  Please try again later.";
     exit;
  }
```

Listing 11.4 **Continued**

```
  $query = "insert into books values
            ('".$isbn."', '".$author."', '".$title."', '".$price."')";
  $result = $db->query($query);

  if ($result) {
      echo  $db->affected_rows." book inserted into database.";
  } else {
          echo "An error has occurred.  The item was not added.";
  }

  $db->close();
?>
</body>
</html>
```

The results of successfully inserting a book are shown in Figure 11.4.

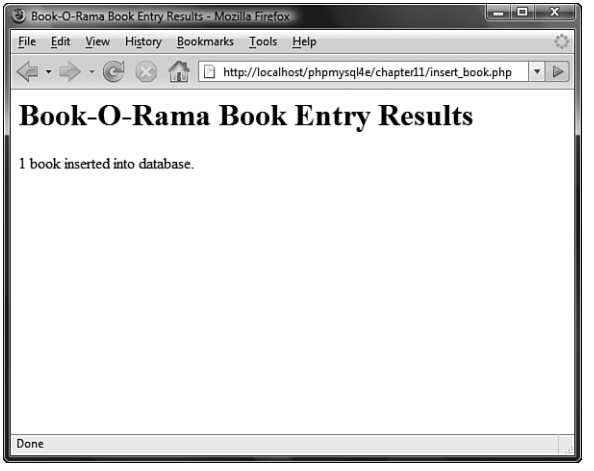

Figure 11.4 The script completes successfully and reports that the book has been added to the database.

If you look at the code for `insert_book.php`, you can see that much of it is similar to the script you wrote to retrieve data from the database. You check that all the form fields were filled in, and you format them correctly for insertion into the database (if required) with `addslashes()`:

```
if (!get_magic_quotes_gpc()) {
  $isbn = addslashes($isbn);
```

```
  $author = addslashes($author);
  $title = addslashes($title);
  $price = doubleval($price);
}
```

Because the price is stored in the database as a float, you don't want to put slashes into it. You can achieve the same effect of filtering out any odd characters on this numerical field by calling `doubleval()`, which we discussed in Chapter 1, "PHP Crash Course." This also takes care of any currency symbols that the user might have typed into the form.

Again, you connect to the database by instantiating the `mysqli` object and setting up a query to send to the database. In this case, the query is an SQL `INSERT`:

```
$query = "insert into books values
          ('".$isbn."', '".$author."', '".$title."', '".$price."')";
$result = $db->query($query);
```

This query is executed on the database by calling `$db->query()` (or `mysqli_query()` if you want to do things procedurally).

One significant difference between using `INSERT` and `SELECT` is in the use of `mysqli_affected_rows()`. This is a function in the procedural version or a class member variable in the object-oriented version:

```
echo  $db->affected_rows." book inserted into database.";
```

In the previous script, you used `mysqli_num_rows()` to determine how many rows were returned by a `SELECT`. When you write queries that change the database, such as `INSERT`s, `DELETE`s, and `UPDATE`s, you should use `mysqli_affected_rows()` instead.

We've now covered the basics of using MySQL databases from PHP.

Using Prepared Statements

The mysqli library supports the use of prepared statements. They are useful for speeding up execution when you are performing large numbers of the same query with different data. They also protect against SQL injection-style attacks.

The basic concept of a prepared statement is that you send a template of the query you want to execute to MySQL and then send the data separately. You can send multiple lots of the same data to the same prepared statement; this capability is particularly useful for bulk inserts.

You could use prepared statements in the `insert_book.php` script, as follows:

```
$query = "insert into books values(?, ?, ?, ?)";
$stmt = $db->prepare($query);
$stmt->bind_param("sssd", $isbn, $author, $title, $price);
$stmt->execute();
echo  $stmt->affected_rows.' book inserted into database.';
$stmt->close();
```

Let's consider this code line by line.

When you set up the query, instead of substituting in the variables as done previously, you put in question marks for each piece of data. You should not put any quotation marks or other delimiters around these question marks.

The second line is a call to `$db->prepare()`, which is called `mysqli_stmt_prepare()` in the procedural version. This line constructs a statement object or resource that you will then use to do the actual processing.

The statement object has a method called `bind_param()`. (In the procedural version, it is called `mysqli_stmt_bind_param()`.) The purpose of `bind_param()` is to tell PHP which variables should be substituted for the question marks. The first parameter is a format string, not unlike the format string used in `printf()`. The value you are passing here (`"sssd"`) means that the four parameters are a string, a string, a string, and a double, respectively. Other possible characters in the format string are `i` for integer and `b` for blob. After this parameter, you should list the same number of variables as you have question marks in your statement. They will be substituted in this order.

The call to `$stmt->execute()` (`mysqli_stmt_execute()` in the procedural version) actually runs the query. You can then access the number of affected rows and close the statement.

So how is this prepared statement useful? The clever thing is that you can change the values of the four bound variables and re-execute the statement without having to reprepare. This capability is useful for looping through bulk inserts.

As well as binding parameters, you can bind results. For `SELECT` type queries, you can use `$stmt->bind_result()` (or `mysqli_stmt_bind_result()`) to provide a list of variables that you would like the result columns to be filled into. Each time you call `$stmt->fetch()` (or `mysqli_stmt_fetch()`), column values from the next row in the resultset are filled into these bound variables. For example, in the book search script you looked at earlier, you could use

```
$stmt->bind_result($isbn, $author, $title, $price);
```

to bind these four variables to the four columns that will be returned from the query. After calling

```
$stmt->execute();
```

you can call

```
$stmt->fetch();
```

in the loop. Each time this is called, it fetches the next result row into the four bound variables.

You can also use `mysqli_stmt_bind_param()` and `mysqli_stmt_bind_result()` in the same script.

Using Other PHP-Database Interfaces

PHP supports libraries for connecting to a large number of databases, including Oracle, Microsoft SQL Server, and PostgreSQL.

In general, the principles of connecting to and querying any of these databases are much the same. The individual function names vary, and different databases have slightly different functionality, but if you can connect to MySQL, you should be able to easily adapt your knowledge to any of the others.

If you want to use a database that doesn't have a specific library available in PHP, you can use the generic ODBC functions. ODBC, which stands for Open Database Connectivity, is a standard for connections to databases. It has the most limited functionality of any of the function sets, for fairly obvious reasons. If you have to be compatible with everything, you can't exploit the special features of anything.

In addition to the libraries that come with PHP, available database abstraction classes such as `MDB2` allow you to use the same function names for each type of database.

Using a Generic Database Interface: PEAR MDB2

Let's look at a brief example using the PEAR MDB2 abstraction layer. This is one of the most widely used of all the PEAR components. Instructions for installing the MDB2 abstraction layer can be found in the "PEAR Installation" section in Appendix A, "Installing PHP and MySQL."

For comparative purposes, let's look at how you could write the search results script differently using MDB2.

Listing 11.5 `results_generic.php`**—Retrieves Search Results from the MySQL Database and Formats Them for Display**

```
<html>
<head>
  <title>Book-O-Rama Search Results</title>
</head>
<body>
<h1>Book-O-Rama Search Results</h1>
<?php
  // create short variable names
  $searchtype=$_POST['searchtype'];
  $searchterm=trim($_POST['searchterm']);

  if (!$searchtype || !$searchterm) {
     echo "You have not entered search details.  Please go back and try again.";
     exit;
  }

  if (!get_magic_quotes_gpc()) {
    $searchtype = addslashes($searchtype);
```

Listing 11.5 **Continued**

```
    $searchterm = addslashes($searchterm);
}

// set up for using PEAR MDB2
require_once('MDB2.php');
$user = 'bookorama';
$pass = 'bookorama123';
$host = 'localhost';
$db_name = 'books';

// set up universal connection string or DSN
$dsn = "mysqli://".$user.":".$pass."@".$host."/".$db_name;

// connect to database
$db = &MDB2::connect($dsn);

// check if connection worked
if (MDB2::isError($db)) {
  echo $db->getMessage();
  exit;
}

// perform query
$query = "select * from books where ".$searchtype." like '%".$searchterm."%'";

$result = $db->query($query);

// check that result was ok
if (MDB2::isError($result))  {
  echo $db->getMessage();
  exit;
}

// get number of returned rows
$num_results = $result->numRows();

// display each returned row
for ($i=0; $i <$num_results; $i++) {
   $row = $result->fetchRow(MDB2_FETCHMODE_ASSOC);
   echo "<p><strong>".($i+1).". Title: ";
   echo htmlspecialchars(stripslashes($row['title']));
   echo "</strong><br />Author: ";
   echo stripslashes($row['author']);
   echo "<br />ISBN: ";
   echo stripslashes($row['isbn']);
```

Listing 11.5 **Continued**

```
    echo "<br />Price: ";
    echo stripslashes($row['price']);
    echo "</p>";
  }

  // disconnect from database
  $db->disconnect();
?>
</body>
</html>
```

Let's examine what you do differently in this script.

To connect to the database, you use the line

```
$db = MDB2::connect($dsn);
```

This function accepts a universal connection string that contains all the parameters necessary to connect to the database. You can see this if you look at the format of the connection string:

```
$dsn = "mysqli://".$user.":".$pass."@".$host."/".$db_name;
```

After this, you check to see whether the connection was unsuccessful using the `isError()` method and, if so, print the error message and exit:

```
if (MDB2::isError($db)) {
  echo $db->getMessage();
  exit;
}
```

Assuming everything has gone well, you then set up a query and execute it as follows:

```
$result = $db->query($query);
```

You can check the number of rows returned:

```
$num_results = $result->numRows();
```

You retrieve each row as follows:

```
$row = $result->fetchRow(DB_FETCHMODE_ASSOC);
```

The generic method `fetchRow()` can fetch a row in many formats; the parameter `MDB2_FETCHMODE_ASSOC` tells it that you would like the row returned as an associative array.

After outputting the returned rows, you finish by closing the database connection:

```
$db->disconnect();
```

As you can see, this generic example is similar to the first script.

The advantages of using MDB2 are that you need to remember only one set of database functions and that the code will require minimal changes if you decide to change the database software.

Because this is a MySQL book, we use the MySQL native libraries for extra speed and flexibility. You might want to use the MDB2 package in your projects because sometimes the use of an abstraction layer can be extremely helpful.

Further Reading

For more information on connecting MySQL and PHP together, you can read the appropriate sections of the PHP and MySQL manuals.

For more information on ODBC, visit http://www.webopedia.com/TERM/O/ODBC.html.

Next

In the next chapter, we go into more detail about MySQL administration and discuss how to optimize databases.

Index

SYMBOLS

! (logical operator), 38
!= (comparison operator), 37
!= (inequality operator), 40, 87
!== (comparison operator), 37
!== (non-identity operator), 40, 87
$result->fetch_assoc() function, 275
$type parameter, 829
% (modulus operator), 33
% (wildcard character), 293
%= (combined assignment operator), 35
& (bitwise operator), 38
& (reference operator), 36
&& (logical operator), 38
+ (addition operator), 33
+ (plus symbol), 126, 748
+ (union operator), 40, 87
++ (increment operator), 35-36
+= (combined assignment operator), 35
, (comma operator), 39
- (subtraction operator), 33
— (decrement operator), 35-36
-= (combined assignment operator), 35
. (string concatenation operator), 26-27
.= (combined assignment operator), 35
/ (division operator), 33
/= (combined assignment operator), 35
< (comparison operator), 37
<< (bitwise operator), 38
<= (comparison operator), 37
= (assignment operator), 34
== (comparison operator), 37
== (equality operator), 40, 87
== (equals comparison operator), 37
=== (comparison operator), 37
=== (identity operator), 40, 87
?: (ternary operator), 39
@ (error suppression operator), 39

A

a file mode, 63
a+ file mode, 63
about.php files (Tahuayo application), 819
absolute paths, 62
abstract classes, 186
access
  associative array contents, 85
  control (authentication)
    implementing, 392-395
    multiple pages, protecting, 399
    passwords, 395-399
  modifiers, 166-167
  MySQL, 219-220
  numerically indexed array contents, accessing, 83-84
  to .php files, restricting, 374-375
  restricting to sensitive data, 364
  substrings, substr() function, 118-119
accounts
  creating, 668-669
  deleting, 670
  modifying, 670
  online newsletters, 702-705, 719
  selecting, 671, 673
  setting up, 666-668
ACID compliance transactions, 313
Acrobat Web site, 776
actions
  Amazon, 826
  MLM, 701
  script architecture, 694
Add to Cart link, 817
add_bm() function, 597-598, 881
add_bms.php files (PHPBookmark application), 572
add_bm_form.php files (PHPBookmark application), 572
add_quoting() function, 766
addBMResponse() function, 874
adding Ajax elements to PHPbookmark projects, 871-880
addition (+) operator, 33
addNewBookmark() function, 873
addslashes() function, 114, 272, 296, 417
addToCart() function, 852
admin.php files (Shopping Cart application), 611
admin.php script (Shopping Cart application), 641-643
admin_fns.php files (Shopping Cart application), 612
administration, 609, 643-650
  administration menu (admin.php), 641-643
  edit_book_form.php script, 646
  functions, online newsletters, 721
  insert_book.php script, 644-645
  insert_book_form.php script, 644
  show_book.php script, 646
  users, 226-227
  views, 609
Adobe
  FDF Web site, 789
  PostScript, 774-775
  Web site, 776
Advanced Maryland Automated Network Disk Archiver (AMANDA), 358
advanced OO features, 184-186, 191
aggregate functions, MySQL, 256
aggregating data, 256-257
Ajax, 856
  bookmarks, adding, 872
  developer Web sites, 885
  elements, 871-880
  JavaScript libraries, 884
  servers
    communication, 863-864
    response, 866
  XMLHTTPRrequest object, 860-862
ajaxServerTime.html, 866-869
aliases, tables, 253-254
ALL privilege, 227
ALTER privilege, 225
ALTER TABLE statement, 261-263
AMANDA (Advanced Maryland Automated Network Disk Archiver), 358
Amazon
  actions, 826
  Associate ID, 813
  books, showing in categories, 826-828
  browse nodes, 816
  caching, 814-815, 846-849
  checking out, 852-853

connecting, 807-808
constants.php file, 823
developer token, 813
index.php file, 820-826
PHP SOAP libraries, 814
project codes, installing, 853-854
REST/XML, 838-839, 844
sessions, creating, 823
shopping carts, building, 813, 849-852
SOAP (Simple Object Access Protocol), 845-846
solution overview, 815-820
Web Services interfaces, 813-814
XML, parsing, 814

AmazonResultSet class, 819, 828-829
Analog Web site, 330
anchoring strings, 126-127
and operator, 38
anomalies, avoiding (Web databases), 213
anonymous login (FTP), 462
ANSI Web site, 265
anti-aliasing text, 489
Apache
configurations, PHP installations, 904
HTTP server, 380-381
installation
binary installations, 890
source installations, 891
Windows, 902
parameters, MaxClients, 273
resources, 909
running, 897
Software Web site, 909
Today Web site, 909
Web server
basic authentication (HTTP), 400-406
htpasswd program, 405
mod_auth module, 402
mod_auth_mysql module, 406-408
Web site, 891
Week Web site, 909

applications
archives, 7
Bob's Auto Parts, 14, 17, 199-202
Book-O-Rama application
Database Search page, 268
schema, 219, 230
content, 546
development environment, 544
documentation, 544-545
internationalization, 7
layer protocols, 414
logic, 546
optimizations, 546-547
PHPBookmark
creating, 569
extensions, 606
files, 572
planning, 536-537
prototypes, 545-546
rewriting code, 537-538
running, 536-537
Shopping Cart. *See* Shopping Cart application
Smart Form Mail
creating, 107-109
regular expressions, 128-129
software engineering, 536
Tahuayo (Amazon), 815-820
testing code, 548
tiers, 218
version control, 542-543
Web forum. *See* Web forum application
writing maintainable code, 538
breaking up, 541-542
code standards, 538
commenting, 540
directory structures, 542
function libraries, 542
indenting, 540-541
naming conventions, 538-540

architecture, Web databases, 216-218
archives
applications, 7
BUGTRAQ, 437

arguments, 22
Array data type (variables), 29
array() language construct, 82
array_count_values() function, 104

array_push() function, 713
array_reverse() function, 97-98
array_walk() function, 103-104
arrays, 81-82
associative, 85
contents, accessing, 85
each() function, 85-87
initializing, 85
list() function, 85-87
looping through, 85-87
bounding box contents, 497
categoryList, 827
converting to scalar variables, 105-106
elements, 82
applying functions, 103-104
counting, 104
functions, passing by reference, 104
indexes, 82
key-value pairs, getdate() function, 473
loading from files, 98-101
medium style form variable, 24
multidimensional, 81, 88-92
sorting, 93-95
three-dimensional arrays, 90, 92
two-dimensional arrays, 88-90
navigating within an array, 102
numerically indexed arrays
accessing with loops, 84
contents, accessing, 83-84
initializing, 82-83
operators, 33-34, 87-88
reordering, 96
array_reverse() function, 97-98
shuffle() function, 96
set cardinality, 104
sorting, 92
asort() function, 93
ksort() function, 93
reverse order, 93
sort() function, 92
superglobal, 24
article list (Web forum application), 747, 749
adding new articles, 762-769
displaying articles, 752-753
plus symbols, 748
threads
collapsing, 748-752
expanding, 748-751
treenode class, 753, 757-760
viewing individual articles, 760-762
ASCII, 772
ASINSearch() method, 829
asort() function, 93
ASP style (PHP tags), 19
assignment operators, 28, 34
combination assignment operators, 35
decrement operators, 35-36
equal sign (=), 25
increment operators, 35-36
reference operator, 36
returning values, 34-35
Associate ID (Amazon), 813
associative arrays, 85. *See also* arrays
contents, accessing, 85
each() function, 85-87
initializing, 85
list() function, 85-87
looping through, 85, 87
associativity, operators, 42-44
asterisk symbol (*), regular expressions, 126
atomic column values (databases), 214-215
attachments, email, 689
attributes, 162-166
creating, 162-164
overriding, 170-173
auditing, 357
authentication, 343, 350-351, 383-384, 401-406. *See also* security
access control
encrypting passwords, 397-399
implementing, 392-395
multiple pages, protecting, 399
storing passwords, 395
basic authentication (HTTP), 399-400
with Apache .htaccess files, 402-406
in PHP, 400-402
custom, creating, 408
digest authentication (HTTP), 400
identifying users, 391-392

mod_auth_mysql module, 406–408
documentation Web sites, 408
installing, 406–407
testing, 407
passwords, 350–351
session control, 517–524
authmain.php script, 517–522
logout.php script, 523–524
members_only.php script, 522–523
users
input data, validating, 580
logging in, 584–587
logging out, 587–588
passwords, 588–595
registering, 577, 580–583
Web sites, 408
authmain.php script (authentication), 517-522
auto_append_file (php.ini file), 142-143
AUTO_INCREMENT keyword, 231
auto_prepend_file (php.ini file), 142-143
autocommit mode, 314
autoload()function, 187
automatic generation of images, 490-491
AVG(column) function, 256

B

b file mode, 63
backing up, 358
AMANDA (Advanced Maryland Automated Network Disk Archiver), 358
databases, 305–306
FTP functions, 459–463
checking update times, 464–465
closing connections, 466
downloads, 465–466
logins, 463
remote connections, 463
MySQL databases, 358
backticks, 448
backtrace (functions), 196
base canvases, 495
baseline descenders, 497
basename() function, 442, 445
basic authentication (HTTP), 399-400
with Apache .htaccess files, 402–406
in PHP, 400–402
binary installations, 890
MySQL, 892–893
binary large objects (BLOB types), 239, 241
bind_param() method, 281
bindings, late static bindings, 185-186
bitwise operators, 38
BLOB types (binary large objects), 239-241
blocks, 47
catch (exception handling), 194
try (exception handling), 193
Bob's Auto Parts application, 14, 17
exception handling, 199–202
book details page (Shopping Cart application), 616, 622-623, 646
Book-O-Rama application
Database Search page, 268
schema, 219, 230
setting up, 243
tables, SQL code, 245
book_fns.php files (Shopping Cart application), 612
book_sc database (Shopping Cart application), 612, 614-615
book_sc.sql files (Shopping Cart application), 612
book_sc_fns.php files (Shopping Cart application), 612
bookdisplayfunctions.php files (Tahuayo application), 819
bookmark_fns.php files (PHPBookmark application), 572
bookmarks, 571
adding, 596–599, 872
bookmark.gif files (PHPBookmark application), 572
deleting, 600–602
displaying, 599
recommending, 571
storing, 571
bookmarks.sql files (PHPBookmark application), 572
books, Amazon categories, 826, 828
Boolean data type (variables), 29
bottom-up approach to security, 363
bottom.php files (Tahuayo application), 819

bounding boxes
array contents, 497
coordinates, 496
Boutell Web site, 508
branching (regular expressions), 127
break statement, 56
breaking up code, 541-542
brochureware sites, 328
common pitfalls, 330
limitations, 328
answering feedback, 329
lack of information, 328
poor presentation, 329
tracking success, 330-331
updated information, 329
browse nodes (Amazon), 816
browsedir.php file, 439
browseNode variable, 824
browseNodeSearch() function, 845
browseNodeSearch() method, 829, 835
browsers
authentication, 351
directories, 439
secure transactions, 410-411
Web database architecture, 216
bugs, 376-377
PHP 5.3 fixes, 7
regression, 377
BUGTRAQ archives, 437
building MLM, 687
buttons
colors, 495
make_button.php scripts, 492
scripts, calling, 493
text
colors/fonts, 492
fitting onto, 495-498
positioning, 498-499
writing, 499

C

cache() function, 847-848
cached() function, 847-848
cachefunctions.php files (Tahuayo application), 820
caching Amazon, 814-815, 846-849
calculate_items() function, 630-631
calculate_price() function, 630-631
calculating dates
in MySQL, 478-480
in PHP, 477-478
calendar functions, 480-481
calling
button scripts, 493
class operations, 167
functions, 22, 143
case sensitivity, 146
errors, 66
parameters, 143-144
prototypes, 144
undefined functions, 145-146
canvas
base, 495
images, creating, 487
caret symbol (^), regular expressions, 126-127
Cartesian product, 250, 254
cartfunctions.php files (Tahuayo application), 820
CAs (Certifying Authorities), 355
cascading style sheets (CSS), 546
case sensitivity
calling functions, 146
MySQL statements, 221
strings, changing, 113-114
casts (variable types), 30
catalog scripts (Shopping Cart application), 615-617
index.php, 615-620
show_book.php, 616, 622-623, 646
show_cat.php, 615, 620-622
catch blocks (exception handling), 194
categories, Amazon books, 826, 828
category page (Shopping Cart application), 615, 620-622
categoryfunctions.php files (Tahuayo application), 820
categoryList array, 827
Certificate Signing Request (CSR), 356-357
certification projects, personalized documents, 779
files, 779
index.html file, 780-781
PDF, 788-796
PDFlib, 796, 802-804

RTF, 784–787
score.php file, 782–784
Certifying Authorities (CAs), 355
CGI Interpreter, 890
CGI specification Web site, 450
change_passwd.php files (PHPBookmark application), 572
change_passwd_form.php files (PHPBookmark application), 572
change_password() function, 590, 720
change_password.php files (Shopping Cart application), 611
change_password_form.php files (Shopping Cart application), 611
characters
classes, 125
escaping, 114
padding, 112
reading, 75
sets, 124–125
check_admin_user() function, 700
check_auth_user() function, 665
check_logged_in() function, 700
check_normal_user() function, 700
check_valid_user() function, 585
checkdate() function, 370, 474
Checkout links, 818
checkout.php files (Shopping Cart application), 611
checkout.php script (Shopping Cart application), 633-638
chgrp() function, 446
child nodes (Web forum tree structure), 743
chmod() function, 446
chown() function, 446
ciphertext (encryption), 351
classes, 160-161
abstract, 186
AmazonResultSet, 828–829
attributes, 164–166
calling, 167
character (regular expressions), 125
converting to strings, 190
creating, 162–164
CSS, 859
designing, 174–175
Exception, 195–196
extending, 196–197
methods, 195
exceptions, creating, 196
inheritance, 162
instantiating, 164
polymorphism, 161
Product, 839
tree_node class, 743
treenode class (Web forum application), 753–760
type hinting, 184
writing code for, 175–183
clauses
GROUP BY, 256–257
HAVING, 257
LIMIT, 258
ORDER BY, 255
SELECT, 255–257
throw, 196
WHERE, 248
comparison operators, 248–249
join condition, 250
client-side programming, 859
cloning objects, 186
closedir($dir) function, 440
closing tags (XML), 810
code
blocks, 47
content, 546
execution directives, 57
indenting, 47
logic, 546
naming conventions, 539
organizing, 374
optimizations, 546–547
prototypes, 545–546
reusing. *See* reusing code
rewriting, 537–538
Shopping Cart application, 610
testing, 548
version control, 542–543
CVS (Concurrent Versions System), 543
multiple programmers, 543
repository, 542–543
writing, 538–541
Codewalkers Web site, 909
collapsing threads (Web forum application), 748, 752

colors
buttons, 495
RGB (red, green, and blue), 488
text, 492
columns, 232-236
date and time types, 238-239
DESCRIBE statement, 299
keys, 209
creating, 215
foreign keys, 210
primary keys, 210
numeric types, 236-238
floating point data types, 237-238
integral data types, 237
string, 239-241
values, 209
atomic column values, 214-215
EXPLAIN statement, 303
columns_priv table, 288, 292-293
combination assignment operators, 35
comma (,) operator, 39
command line, 531
commands
DESCRIBE, 233-234
GRANT, 223-228
LOCK TABLES, 305
mysql, 221
mysql_dump, 305
phpinfo(), 31
REVOKE, 227-228
SHOW, 233-234
SQL, 229-231
traceroute (UNIX), 344
Web server functions, 447-450
comments, 20-21, 540
commercial Web sites, 327, 336
adding value to goods or services, 335
authentication, 343
cutting costs, 335-336
firewalls, 357-358
importance of stored information, 342
online brochures, 328
common pitfalls, 330
limitations, 328-331
orders for goods or services, 331-334
privacy policies, 333
providing services and digital goods, 334-335
return policies, 333
risks, 336
competition, 338
crackers, 337
failure to attract business, 337-338
hardware failure, 337
legislation and taxes, 339
service provider failures, 338
software errors, 338
system capacity limits, 339
security, 342
auditing, 357
authentication, 350-351
backing up data, 358
Certificate Signing Request (CSR), 356-357
Certifying Authorities (CAs), 355
compromises, 349
digital certificates, 355
digital signatures, 354-355
encryption, 351-354
hash function, 354
log files, 357
passwords, 350-351
physical security, 359
Secure Web servers, 356-357
security policies, creating, 349-350
threats, 342-348
SSL (Secure Sockets Layer), 333
strategies, selecting, 339
types, 327-328
committed transactions, 314
comparing strings, 119
length, testing, 120
strcasecmp() function, 119
strcmp() function, 119
strnatcmp() function, 119
comparison operators, 36-37
equals operator, 37
WHERE clauses, 248-249
compatibility of commercial Web sites, 334
components, user personalization, 570
compression, SSL (Secure Sockets Layer), 416

compromised servers, 365
Concurrent Versions System (CVS), 543
conditionals, 46
code blocks, 47
comparing, 51
else statements, 47
elseif statements, 48-49
if statements, 46-47
indenting code, 47
switch statements, 49-51
conditions, join, 250
configuration
DMZs, 386-387
PHP, 894
sessions, 516-517
web servers
Apache HTTP server, 380-381
Microsoft IIS, 381
connections
Amazon, 807-808
Database servers, 384-385
FTP servers, closing, 466
MySQL database, 293
networks, runtime errors, 557-558
remote FTP servers, 463
Web databases, 273
constants, 31
error reporting, 562
constants.php files
Amazon, 823
Tahuayo application, 819
constructors, 163-164
content (code), 546
continuation symbol (MySQL), 220
continue handlers, 320
continue statements, 56
control
characters
\n (newline), 68
\t (tab), 68
structures, 46, 49
alternate syntax, 56
breaking out of, 56
conditionals, 46-51
declare, 57
loops, 51-56
stored procedures, 319-323
version (code), 542-543
CVS (Concurrent Versions System), 543
multiple programmers, 543
repository, 542-543
conversion
arrays to scalar variables, 105-106
calendars, 481
classes to strings, 190
format strings, 112-113
printf() function, 112-113
type codes, 112-113
cookies, 510-511
session IDs, 511-512
setting, 510-511
coordinates, bounding boxes, 496
copy() function, 447
correlated subqueries, 260
cos() function, 804
count() function, 104
COUNT(items) function, 256
counting array elements, 104
crackers, 337, 366
CREATE privilege, 226
CREATE TABLE command (SQL), 229-231
CREATE TEMPORARY TABLES privilege, 226
create_database.php files (Warm Mail application), 655
create_database.sql files, 745
MLM application, 691
Web forum application, 744
credit card numbers, storing, 419
criteria, retrieving from databases, 248-249
cross join, 254
crypt() function, 397-398
PHP 5.3, functionality in, 7
cryptography, 352
CSR (Certificate Signing Request), 356-357
CSS (cascading style sheets), 546, 858
classes, 859
curly braces ({}), regular expressions, 126
current() function, 102
cursors (stored procedures), 319-323
custom authentication, creating, 408
cutting costs (commercial Web sites), 335-336
CVS (Concurrent Versions System), 543

D

data
aggregating, 256-257
encrypting, 418
graphing, 499-507
grouping, 256-257
input
checking, 558
user authentication validation, 580
inserting into databases, 244-245
joins, 254-255
loading from files, 311
redundant data, avoiding (Web databases), 212-213
retrieving
from databases, 246-247
from multiple tables, 249-250
in a particular order, 255-256
with specific criteria, 248-249
rows, returning, 258
sensitive data
credit card numbers, storing, 419
storing, 417-418
storing, 59
tables
aliases, 253-254
joining, 251-252
rows unmatched, 252-253
two-table joins, 250-251
transfer, 306-308
types
BLOB types (binary large objects), 239
date and time data types, 238-239
floating point data types (numeric column types), 237-238
integral data types (numeric column types), 237
TEXT types, 239
variables, 29
Data Definition Languages (DDL), 244
Data Encryption Standard (DES), 353
Data Manipulation Languages (DML), 244
data_valid_fns.php files
MLM application, 691
PHPBookmark application, 572
Shopping Cart application, 612
Warm Mail application, 655
Web forum application, 744
databases, 208
backing up, 305-306
benefits, 80, 207
Book-O-Rama
setting up, 243
tables, SQL code, 245
book_sc database (Shopping Cart application), 612-615
creating with MySQL, 222
data
aggregating, 256-257
grouping, 256-257
inserting, 244-245
joins, 254-255
loading from files, 311
retrieving, 246-256
rows unmatched, 252-253
tables, 251-254
two-table joins, 250-251
DDL (Data Definition Languages), 244
DML (Data Manipulation Language), 244
dropping, 264
front page, 574-577
information gathering, 296
EXPLAIN statement, 299-303
SHOW statement, 296-297
keys, 209
foreign keys, 210
primary keys, 210
lists, 688
MySQL, 287
aggregate functions, 256
backing up, 358
columns_priv table, 292
connection verification, 293
db table, 290-291
host table, 291
join types, 254-255
request verification, 293
results.php script, 269
tables_priv table, 292

user table, 289
Web database architecture, 268-271
optimizing, 304-305
default values, 305
designs, 304
indexes, 305
permissions, 304
tables, 304
passwords
encrypting, 295
storing, 295
PEAR, 284-285
privilege system, 287-288
columns_priv table, 293
db table, 290-291
grant table, 293
host table, 290-291
privileges, updating, 293-294
tables_priv table, 293
user table, 289-290
queries, indexes, 304
records
deleting, 264
updating, 261
relational databases, 210
relationships, 211
many-to-many relationships, 211
one-to-many relationships, 211
one-to-one relationships, 211, 216
replication, 306-307
data transfer, 306-308
master servers, 306-307
slaves, 306-308
restoring, 306
rows, returning, 258
runtime errors, 555-557
schemas, 210, 573-574
security, 294
operating system, 294
passwords, 295
user privileges, 295-296
Web issues, 296
selecting in MySQL, 229
servers
connecting to, 384-385
security, 383-385
setting up (online newsletters), 692-694
Shopping Cart application, 615
SQL (Structured Query Language), 243-244
subqueries, 258-259
correlated, 260
operators, 259
row, 260
temporary tables, 260
subscribers, 688
tables, 208
altering, 261-263
Cartesian product, 250
columns, 209, 232-241
creating in MySQL, 229-231
dropping, 264
equi-joins, 251
indexes, creating, 234-235
joins, 250
keywords, 231
left joins, 252-253
rows, 209
types, 216, 229
values, 209
viewing, 233-234
Warm Mail application (email client), 655-656
Web forum application, 744-745, 747

date and time

calendars, 481
column types, 238-239
converting between PHP and MySQL formats, 476-477
data types, 238-239
in MySQL
calculations, 478-480
DATE_FORMAT() function, 476-477
MySQL Web site, 481
UNIX_TIMESTAMP() function, 476-477

in PHP, 469, 474
calculations, 477–478
calendar functions, 480–481
checkdate() function, 474
date() function, 469–472
floor() function, 478
getdate() function, 473
microseconds, 480
mktime() function, 471–472
PHP Web site, 481
date() function, 21-22, 445, 469-471
format codes, 469–471
Unix timestamps, 471–472
date_add() function, 478
DATE_FORMAT() function, 476-477
date_sub() function, 478
db table, 288, 290-291
db_connect() function, 583
db_fns.php files
MLM application, 691
PHPBookmark application, 572
Shopping Cart application, 612
Warm Mail application, 655
Web forum application, 744
db_result_to_array() function, 619
DDL (Data Definition Languages), 244
DDoS (Distributed Denial of Service), 346, 364
preparing for, 387
debugging variables, 559-561
declare control structure, 57
declare handlers, 320
declaring
functions, 146–147
stored functions, 318–319
stored procedures, 316–317
decoct() function, 446
decrement operators, 35-36
decryption, 352
default values, database optimization, 305
DELETE privilege, 225
DELETE statement, 264
delete_account() function, 670
delete_bm() function, 601
delete_bms.php files
PHPBookmark application, 572
Shopping Cart application, 612
delete_category.php files (Shopping Cart application), 612
delete_message() function, 681-682
deletion anomalies, avoiding (Web databases), 213
Denial of Service (DoS), 346-347, 364
deploying new software versions, 379
deregistering variables, 513
DES (Data Encryption Standard), 353
DESC keyword, 255
descenders (letters), 497
DESCRIBE command, 233-234
DESCRIBE statement, 299
describe user; statement, 289
design. *See also* configuration
classes, 174–175
database optimization, 304
Web databases, 211
anomalies, avoiding, 213
atomic column values, 214–215
keys, creating, 215
null values, avoiding, 216
questions, formulating, 215
real-world objects, modeling, 211–212
redundant data, avoiding, 212–213
table types, 216
design_button.html file, 492-493
destroying sessions, 513
destructors, 163-164
Details link, 817
developer token (Amazon), 813
development environments, 544
Devshed Web site, 508, 908
DHTML (Dynamic HTML), 857
diagrams
entity relationship, 210
online newsletters, 689–691
die() language construct, 526
digest authentication (HTTP), 400
digital certificates, 355
digital goods (commercial Web sites), providing, 334-335
digital signatures, 354-355
directives
execution, 57
magic_quotes_gpc, 417
magic_quotes_runtime, 417
php.ini file, editing, 529–530

directories
browsing, 439
extensions, copying libpdf_file, 899
files, write permissions, 418
functions, 439
creating directories, 443
deleting directories, 443
file paths, 442–443
reading from directories, 439–441
structures, 542
dirname() function, 442, 445
disabling unnecessary OS applications, 388
disaster recovery, 364
planning, 388–389
disconnecting Web databases, 276
discussion board application, 741-742, 763-764
article list, 747–749
collapsing threads, 748–752
displaying articles, 752–753
expanding threads, 748–751
individual articles, viewing, 760–762
new articles, adding, 762–769
plus symbols, 748
treenode class, 753, 757–760
database design, 744–747
extensions, 769
files, 744
posters, 744
solutions, 742–744
tree structure, 742–743
tree_node class, 743
discussion boards, 741
discussion_fns.php files (Web forum application), 744
disgruntled employees, 366
disk_free_space($path) function, 443
display() function, 758
display_account_form() function, 667, 703, 719
display_account_select() function, 672-673
display_account_setup() function, 667, 669
display_book_form() function, 647
display_button() function, 733
display_cart() function, 627-630
display_categories() function, 618-619
display_information() function, 713-714
display_items() function, 709
display_list() function, 674
display_list_form() function, 722
display_mail_form() function, 726
display_message() function, 678
display_password_form() function, 719
display_post() function, 762
display_preview_button() function, 733
display_registration_form() function, 577
display_tree() function, 752-753
display_user_menu() function, 585
display_user_urls() function, 599
displaying articles (Web forum application), 752-753
Distributed Denial of Service (DDoS), 346
division operator, 33
DML (Data Manipulation Languages), 244
DMZs (demilitarized zones), 366
configuring, 386–387
dns_get_mx() function, 459
do..while loops, 55-56
do_html_header() function, 632, 672, 700
Document Type Definition. *See* DTD
documentation
gd, Web site, 508
Web application projects, 544–545
documents
headers, 779, 804
personalized, 771
RTF, 784–787
certification project, 779–781, 788–796, 802–804
creating, 771–772
extensions, 805
formats, 772
requirements, 776–777
DOM (Document Object Model), 857
web resources, 884
DoS (Denial of Service), 346-347, 364
preparing for, 387
double data type (variables), 29
doubleval() function, 296
downloading
files from FTP servers, 465–466
FreeType library, 484
jpeg-6b, 484

PostScript Type 1 fonts, 484
t1lib, 484
draw_star() function, 804
drawing
figures, 499–507
functions, 488
images, scripts, 486
text, 487–489
DROP DATABASE statement, 264
DROP privilege, 226
DROP TABLE statement, 264
dropping
databases, 264
tables, 264
DTD (Document Type Definition), 810
dump variables.php file, 559
dynamic content, 21
date() function, 21–22
functions, 22

E

e-commerce Web sites, 327, 336
adding value to goods or services, 335
authentication, 343
cutting costs, 335–336
online brochures, 328
common pitfalls, 330
limitations, 328–331
orders for goods or services, 331–334
privacy policies, 333
providing services and digital goods, 334–335
return policies, 333
risks, 336
competition, 338
crackers, 337
failure to attract business, 337–338
hardware failure, 337
legislation and taxes, 339
service provider failures, 338
software errors, 338
system capacity limits, 339
security, 342
auditing, 357
authentication, 350–351
backing up data, 358
Certificate Signing Request (CSR), 356–357
Certifying Authorities (CAs), 355
compromises, 349
digital certificates, 355
digital signatures, 354–355
encryption, 351–354
firewalls, 357–358
hash function, 354
importance of stored information, 342
log files, 357
passwords, 350–351
physical security, 359
Secure Web servers, 356–357
security policies, creating, 349–350
threats, 342–348
SSL (Secure Sockets Layer), 333
strategies, selecting, 339
types, 327–328
each() function, 85, 87, 102
echo statements, 26-27
edit_book.php files (Shopping Cart application), 612
edit_book_form.php files (Shopping Cart application), 612, 646
edit_category.php files (Shopping Cart application), 611
edit_category_form.php files (Shopping Cart application), 611
elements, 82
applying functions, 103–104
counting, 104
root elements (XML), 811
else statements, 47
elseif statements, 48-49
email
accounts
creating, 668–669
deleting, 670
modifying, 670
selecting, 671, 673
setting up, 666–668
attachments, 689
encryption, 419–420
GPG (Gnu Privacy Guard), 419–427
PGP (Pretty Good Privacy), 419

reading, 452
sending, 452
Warm Mail application
database, 655-656
deleting email, 681-682
email, 681-682
extensions, 686
files, 654-655
forwarding/replying, 684-685
IMAP function library, 652-653
interface, 654
logging in, 663-666
logging out, 666
reading mail, 671-681
script architecture, 657, 662-663
sending, 682-685
solutions, 652-654
embedding PHP in HTML, 17-18
comments, 20-21
PHP
statements, 19-20
tags, 18-19
whitespace, 20
empty() function, 45
encryption, 351-352, 419-420
algorithm, 351
ciphertext, 351
cryptography, 352
data, 418
Data Encryption Standard (DES), 353
decryption, 352
digital certificates, 355
digital signatures, 354-355
GPG (Gnu Privacy Guard), 419
installing, 420-422
key pairs, 420-421
testing, 422-427
hash functions, 354
passwords, 295, 397-399
PGP (Pretty Good Privacy), 419
plain text, 351
private keys, 353
public keys, 353-354
RSA, 353
end() function, 102
engineering software, 536
entities (HTML), 372
entity relationship diagrams, 210
ENUM type, 241
envelopes, SOAP, 812
environments
development, 544
PHP functions, 450
EPA Web site, 359
equal sign (=) assignment operator, 25
equality operator, 87
equals operator, 37
equi-joins, 251, 255
Equifax Secure, 355
ereg() function, 129-130
eregi() function, 129
ereg_replace() function, 130
eregi_replace() function, 130
errors
401 errors (HTTP), 404
exception handling, 565-567
exit statement, 56
function calling, 66
handling, 202
logic, 558-559
messages, 145-146
PHP 5.3, 7
programming, 551-554
logic errors, 558-559
runtime errors, 553-555
syntax errors, 552-553
reporting levels, 562-563
runtime, 553-554
database interaction, 555-557
functions that don't exist, 554-555
input data, checking, 558
network connections, 557-558
reading/writing files, 555
settings, 563-564
software, 338, 347
developer assumptions, 347
poor specifications, 347
poor testing, 348
suppression operator, 39
syntax, 552-553
triggering, 564
escapeshellcmd() function, 378, 417, 449
escaping characters, 114

escaping output, 371
eval() function, 525-526
evaluating strings, 525-526
Evil Walrus Web site, 909
examining php.ini file, 380
Exception class, 195-196
extending, 196–197
methods, 195
exceptions, 193-195, 565-567
Bob's Auto Parts application, 199–202
catch blocks, 194
classes, 196
Exception class, 195–197
I/O (input/output) files, 199
throwing, 193
try blocks, 193
tutorials, 203
user-defined exceptions, 196–199
exec() function, 447
executable content (stored data), 417
execution
command line, 531
directives, 57
operator, 39–40
quotes, 377–378
exit
handlers, 321
language constructs, 526
statements, 56
expand_all() function, 751
expanding threads (Web forum application), 748-751
EXPLAIN statement, 299-303
column values, 303
join types, 301–302
explode() function, 100-101, 116-117, 459
expressions
regular, 123–124
* symbol, 126
+ symbol, 126
branching, 127
caret symbol (^), 126–127
character sets, 124–125
curly braces ({}), 126
Perl, 123
slash (\), 127
Smart Form Mail application, 128–129
special characters, 127–128
splitting strings, 130
string anchoring, 126–127
subexpressions, 126
substrings
finding, 129–131
extending
Exception class, 196–197
syntax, 257
Extensible Markup Language. *See* XML
extensions
copying libpdf_file, 899
loading, 528
online newsletters, 740
personalized documents, 805
PHPBookmark application, 606
require() statement, 136
Shopping Cart application, 650
Warm Mail application, 686
Web forum application, 769
extract() function, 105-106
extract_type parameter, 105
extract_type parameter, 105
Extreme Programming Web site, 549

F

f file mode, 63
FastTemplate Web site, 546
fclose() function, 69, 440
FDF Web site, 789
fdf_create() function, 789
fdf_set_file() function, 789
fdf_set_value() function, 789
Fedex Web site, 335
feof() function, 73
fetchRow() method, 284
fgetc() function, 75
fgetcsv() function, 73-74
fgets() function, 73
fgetss() function, 73
fields
scope, 290
tables, 209
figures, drawing, 499-507
File Details view, 445

FILE privilege, 226-227, 295
File Transfer Protocol. *See* FTP
file() function, 74
file_exists() function, 76
fileatime() function, 445
filedetails.php file, 444-445
filegroup() function, 444, 446
fileinfo extension, 7
filemtime() function, 445
filename extensions, require() statement, 136
fileowner() function, 444, 446
fileperms() function, 446
files, 59-61, 443
backing up, 459-466
browsedir.php, 439
checking, 76
closing, 69
constants.php (Amazon), 823
create_database.sql, 745
creating, 447
data, loading from, 311
deleting, 76, 447
design_button.html, 492-493
dump variables.php, 559
filedetails.php, 444-445
formats, 68-69
handle.php, 566
htaccess files (Apache Web server), 402-406
httpd.conf, 896-897
I/O (input/output), 199
index.html (certification application), 780-781
index.php
MLM online newsletters, 694
Tahuayo application, 820-826
Warm Mail application, 657
libpdf_php, copying, 899
limitations, 79
loading arrays from, 98-101
locking, 78-79
log files, 357
lookup.php, 453
make_button.php, 493
mirroring FTP functions, 459-466
MLM, 690
modes, 61-62
moving, 447
multiple, uploading (online newsletters), 727, 731
navigating, 76-77
new_post.php, 763
newbooks.txt, 311
opening, 61
file modes, 61-62
fopen() function, 62-64
FTP (File Transfer Protocol), 64-65
HTTP (Hypertext Transfer Protocol), 64-65
potential problems, 65-66
paths, directories, 442-443
pdflib.php, 796
personalized documents, 779
php.ini file
auto_append_file, 142-143
auto_prepend_file, 142-143
directives, editing, 529-530
PHPBookmark application, 572
pollsetup.sql, 500
progex.php, 448-449
properties, changing, 446
reading, 61, 71-72, 444-446
feof() function, 73
fgetc() function, 75
fgetcsv() function, 73-74
fgets() function, 73
fgetss() function, 73
file() function, 74
fopen() function, 72
fpassthru() function, 74
fread() function, 75
readfile() function, 74
reading/writing, 555
rtf.php, 786
score.php (certification project), 782-784
Shopping Cart application, 611-612
showpoll.php, 502-506
simplegraph.php, 486
status function results, 448-449
Tahuayo application, 819-820

topbar.php, 825
uploading, 431-432
displaying, 437
FTP (File Transfer Protocol), 466
HTML, 431-433
online newsletters, 688-689
PHP, writing, 434-438
security, 434, 438
troubleshooting, 438-439
utilityfunctions.php, 825
vote.html, 500
Warm Mail application (email client), 654-655
Web forum application, 744
write permissions, 418
writing, 61
file formats, 68-69
fputs() function, 67
fwrite() function, 67
fwrite() function, parameters, 68
filesize() function, 76, 446
filetype() function, 446
filled_out() function, 580-581
filtering
input data (Web databases), 272
user input, 367-371
final keyword, 172
find and replace substrings, 122-123
finding substrings, 120-121
numerical position, 121
regular expressions, 129-130
strchr() function, 121
stristr() function, 121
strpos() function, 121
strrchr() function, 121
strrpos() function, 122
strstr() function, 121
findstr.exe, 377
firewalls, 357-358, 386
FishCartSQL, 650
fitting text onto buttons, 495-498
flat files, 59-61
checking, 76
closing, 69
deleting, 76
formats, 68-69
limitations, 79
locking, 78-79
navigating, 76-77
opening, 61
file modes, 61-62
fopen() function, 62-64
FTP (File Transfer Protocol), 64-65
HTTP (Hypertext Transfer Protocol), 64-65
potential problems, 65-66
reading, 61, 71-72
feof() function, 73
fgetc() function, 75
fgetcsv() function, 73-74
fgets() function, 73
fgetss() function, 73
file() function, 74
fopen() function, 72
fpassthru() function, 74
fread() function, 75
readfile() function, 74
writing, 61
file formats, 68-69
fputs() function, 67
fwrite() function, 67
fwrite() function, parameters, 68
Float data type (variables), 29
floating point data types (numeric column types), 237-238
flock() function, 78
floor() function, 478
focus groups, 330-331
fonts
button text, 492
descenders, 497
FreeType library, downloading, 484
images, creating, 491-499
PDF readers, 794-795
PostScript Type 1 fonts, downloading, 484
TrueType, 492
footers, script architecture, 694
fopen() function, 61-64, 72, 440, 454
for loops, 54-55
foreach loops, 54-55
iteration, 191

foreign keys
databases, 210
InnoDB tables, 315-316
forgot_form.php files (PHPBookmark application), 572
forgot_passwd.php files (PHPBookmark application), 572
formats
codes, date() function, 469-471
images, 484
GIF (Graphics Interchange Format), 485
JPEG (Joint Photographic Experts Group), 485
PNG (Portable Network Graphics), 485
WBMP (Wireless Bitmap), 485
personalized documents, 772
ASCII, 772
HTML, 773
paper, 772
PDF, 775
PostScript, 774-775
RTF, 774
word processors, 773
strings, 110
case, changing, 113-114
conversion specifications, 112-113
HTML formatting, 110-111
ltrim() function, 110
nl2br() function, 110-111
printing, 110, 112-113
rtrim() function, 110
storage, 114-116
trim() function, 110
trimming whitespace, 110
forms
HTML, 268-269, 431
Bob's Auto Parts application, 14, 17
creating, 14-16
processing, 16
totaling with operators, 41-42
variables, accessing, 23-27
forum application. *See* Web forum application
forwarding to email, Warm Mail application, 684-685
fpassthru() function, 74
FPDF function library, 778
fputs() function, 67
fread() function, 75
FreeType library, downloading, 484
fseek() function, 77
ftell() function, 76
FTP (File Transfer Protocol), 459
anonymous login, 462
backing up files, 459-463
checking update times, 464-465
closing connections, 466
downloads, 465-466
logins, 463
remote connections, 463
FTP transfer modes, 466
ftp_get() function, 466
ftp_mdtm() function, 464
ftp_nlist() function, 467
ftp_size() function, 467
mirroring files, 459-463
checking update times, 464-465
closing connections, 466
downloads, 465-466
logins, 463
remote connections, 463
opening files, 64-65
set_time_limit() function, 467
timeouts, avoiding, 467
uploading files, 466
ftp_connect() function, 463
ftp_get() function, 466
ftp_mdtm() function, 464
ftp_nlist() function, 467
ftp_size() function, 467
full joins, 250, 254
functions, 143-144, 151, 158. *See also* commands
$result->fetch_assoc(), 275
add_bm(), 597-598
add_quoting(), 766
addslashes(), 114, 272, 296, 417
addToCart(), 852
aggregate, MySQL, 256
applying to array elements, 103-104
arguments, 22

array_count_values(), 104
array_push(), 713
array_reverse(), 97-98
array_walk(), 103-104
arsort(), 93
asort(), 93
autoload(), 187
AVG(column), 256
backtrace, 196
basename(), 442, 445
browseNodeSearch(), 845
cache(), 847-848
cached(), 847-848
calculate_items(), 630-631
calculate_price(), 630-631
calendars, 480-481
calling, 22, 143
  case sensitivity, 146
  errors, 66
  parameters, 143-144
  prototypes, 144
  undefined functions, 145-146
change_password(), 590, 720
check_admin_user(), 700
check_auth_user(), 665
check_logged_in(), 700
check_normal_user(), 700
check_valid_user(), 585
checkdate(), 370, 474
chgrp(), 446
chmod(), 446
chown(), 446
closedir($dir), 440
copy(), 447
cos(), 804
count(), 104
COUNT(items), 256
creating, 146
crypt(), 397-398
current(), 102
date(), 21-22, 445, 469-471
  format codes, 469-471
  Unix timestamps, 471-472
DATE_FORMAT(), 476-477
db_connect(), 583
db_result_to_array(), 619
declaring, 146-147
decoct(), 446
delete bm(), 601
delete_account(), 670
delete_message(), 681-682
directories, 439
  creating, 443
  deleting, 443
  file paths, 442-443
  reading from, 439-441
dirname(), 442, 445
disk_free_space($path), 443
display() function, 758
display_account_form(), 667, 703, 719
display_account_select(), 672-673
display_account_setup(), 667, 669
display_book_form(), 647
display_button(), 733
display_cart(), 627-630
display_categories(), 618-619
display_information(), 713-714
display_items(), 709
display_list(), 674
display_list_form(), 722
display_mail_form(), 726
display_message(), 678
display_password_form(), 719
display_post(), 762
display_preview_button(), 733
display_registration_form(), 577
display_tree(), 752-753
display_user_menu(), 585
display_user_urls(), 599
do_html_header(), 632, 672, 700
doubleval(), 296
drawing parameters, 488
draw_star(), 804
each(), 85-87, 102
empty(), 45
end(), 102
ereg(), 129-130
eregi(), 129
ereg_replace(), 130
eregi_replace(), 130

escapeshellcmd(), 378, 417, 449
eval(), 525-526
exec(), 447
expand_all(), 751
explode(), 100-101, 116-117
extract(), 105-106
fclose(), 69, 440
fdf_create(), 789
fdf_set_file(), 789
fdf_set_value(), 789
feof(), 73
fgetc(), 75
fgetcsv(), 73-74
fgets(), 73
fgetss(), 73
file(), 74
file_exists(), 76
fileatime(), 445
filegroup(), 444, 446
filemtime(), 445
fileowner(), 444, 446
fileperms(), 446
files
  creating, 447
  deleting, 447
  moving, 447
  properties, changing, 446
  reading, 444-446
  status results, 448-449
filesize(), 76, 446
filetype(), 446
filled_out(), 580-581
flock(), 78
floor(), 478
fopen(), 61-64, 72, 440, 454
fpassthru(), 74
fputs(), 67
fread(), 75
fseek(), 77
ftell(), 76
FTP functions, 459-466
  backing up files, 459-466
  ftp_get(), 466
  ftp_mdtm(), 464
  ftp_nlist(), 467
  ftp_size(), 467
  mirroring files, 459-466
  set_time_limit(), 467
  timeouts, avoiding, 467
  uploading files, 466
ftp_connect(), 463
fwrite(), 67-68
get_accounts(), 668
get_account_list(), 671
get_archive(), 716
get_categories(), 618
get_category_name(), 621
get_current_user(), 529
get_email(), 707
get_extension_funcs(), 528
get_loaded_extensions(), 528
get_magic_quotes_gpc() function, 272
get_post(), 761-762
get_post_title(), 765
get_random_word(), 593
get_unsubscribed_lists(), 712
get_user_urls(), 585, 599
getARS(), 828, 844-846
getCategoryName(), 826-827
getdate(), 473
getenv(), 450
getlastmod(), 529
gettype(), 44
Header(), 489-490, 787
highlight_file(), 530-531
htmlentities, 372-374
htmlspecialchars(), 272, 372-374, 417
ImageColorAllocate(), 488
ImageCreate(), 487
ImageCreateFromGIF(), 487, 495
ImageCreateFromJPEG(), 487, 495
ImageCreateFromPNG(), 487, 495
ImageDestroy(), 490
ImageFill(), 488
ImageFilledRectangle(), 505-507
ImageGetTTFBBox(), 496
ImageJPEG(), 490
ImageLine(), 505
ImagePNG(), 490, 495
ImageRectangle(), 507

images, 507
ImageString(), 488
ImageTTFBBox(), 497
ImageTTFText(), 496, 507
IMAP function library, 652-653
imap_body(), 679-680
imap_delete(), 682
imap_expunge(), 682
imap_fetchheader(), 679
imap_header(), 679
imap_headers(), 676, 679
imap_open(), 675-676
implode(), 117
ini_get(), 529-530
ini_set(), 529-530
insert_order(), 636
intval(), 101
is_uploaded_file(), 438
isset(), 45, 155
join(), 117
krsort(), 93
ksort(), 93
libraries, 542
  FPDF, 778
  mail_fns.php, 668
  output_fns.php, 664
  PHPBookmark application, 572
list(), 85-87
load_list_info(), 714
login(), 585, 706
lstat(), 446
ltrim(), 110
mail(), 109, 452, 595, 689
max() function, 155-156
MAX(column), 256
microtime(), 480
MIN(column), 256
mkdir(), 443
mktime(), 471-472
my_error_handler (), 565
mysql_affected_rows(), 280
mysql_connect(), 274, 555
mysql_query(), 274-275
mysql_select_db(), 274
mysqli_errno(), 556
mysqli_error(), 556
mysqli_fetch_assoc(), 275
mysqli_query(), 556
naming, 147-148, 539
network lookup, 455-459
  dns_get_mx(), 459
  explode(), 459
  gethostbyaddr(), 458
  gethostbyname(), 456-458
  getmxrr(), 456
  parse_url(), 458
next(), 102
nl2br(), 110-111
notify_password(), 592-594
number_of_accounts(), 671
ODBC (Open Database connectivity), 282
open_mailbox(), 675
opendir(), 440
overloading, 147
parameters, 22, 148-150
  pass by reference, 153-154
  pass by value, 153-154
passing functions by reference, 104
passthru(), 448
pdf_add_outline(), 794
pdf_begin_page(), 793
pdf_close(), 796
pdf_fill(), 804
pdf_rect(), 802
pdf_replace(), 789
pdf_setlinewidth(), 802
pdf_set_info(), 793
pdf_show(), 795
pdf_show_xy(), 803
pdf_stringwidth(), 803
pdf_stroke(), 802
PHP environment variables, 450
phpinfo(), 450, 778
posix_getgrgid(), 446
posix_getpwuid(), 444-446
pretty(), 714
prev(), 102
print(), 110
printf(), 111-112

prototypes, 144
putenv(), 450
range(), 83
readdir($dir), 440
readfile(), 74
recommend_urls(), 603-605
recursive functions, 156-158
register(), 582
rename(), 447
reset_password(), 592
reset(), 102
retrieve_message(), 678
returning from, 154-155
rewind(), 76
rewinddir($dir), 441
rmdir(), 443
rsort(), 93
rtrim(), 110
runtime errors, 554-555
safeString(), 825
scope, 151
send(), 734
send_message(), 683-684
serialize(), 526-527, 848
session_get_cookie_params(), 511
session_register(), 513
session_start(), 512-515
session_unregister(), 513
set_error_handler(), 565
setcookie(), 510-511
settype(), 44
shal1(), 398
shopping carts (Amazon), 849
show_source(), 530-531
showBrowseNode(), 826-827
showCart(), 852
showCategories(), 826
ShowSmallCart(), 825
showSummary(), 828, 844
shuffle(), 96
sin(), 804
sizeof(), 104
sort(), 92
split(), 130
sprintf(), 111
stat(), 446
STD(column), 256
STDDEV(column), 256
store_account(), 704
store_account_settings(), 668-669
store_list(), 723
store_new_post(), 767
stored, declaring, 318-319
str_replace(), 122, 787
strcasecmp(), 119
strchr(), 121
strcmp(), 119
strings
  case, 113-114
  versus regular expressions, 131
strip_tags(), 417
stripslashes(), 116, 272, 296
stristr(), 121
strlen(), 120
strnatcmp(), 119
strpos(), 121
strrchr(), 121
strrpos(), 122
strstr(), 121, 597
strtok() function, 117
strtolower(), 113
strtoupper(), 113
subscribe(), 717
substr(), 118-119
substr_replace(), 123
SUM(column), 256
system(), 448
touch(), 447
trim(), 110, 271
uasort(), 95
ucfirst(), 113
ucwords(), 114
uksort(), 95
umask(), 443
undefined, calling, 145-146
UNIX_TIMESTAMP(), 476-477
unlink(), 76, 447
unserialize(), 527, 848
unset(), 45
unsubscribe(), 717

urlencode(), 399, 455
usort(), 94
valid_email(), 581
values, returning, 155–156
variables, 44, 148
reinterpreting, 46
scope, 151–153
status, testing, 45
type, setting/testing, 44–45
fwrite() function, 67-68

G

gd documentation Web site, 508
generating images automatically, 490-491
get_account_list() function, 671
get_accounts() function, 668
get_archive() function, 716
get_categories() function, 618
get_category_name() function, 621
get_current_user() function, 529
get_email() function, 707
get_extension_funcs() function, 528
get_loaded_extensions() function, 528
get_magic_quotes_gpc() function, 272
get_post() function, 761-762
get_post_title() function, 765
get_random_word() function, 593
get_unsubscribed_lists() function, 712
get_user_urls() function, 585, 599
getARS() function, 828, 844, 846
getCategoryName() function, 826-827
getdate() function, 473
getenv() function, 450
gethostbyaddr() function, 458
gethostbyname() function, 456-458
getlastmod() function, 529
getmxrr() function, 456
getServerTime() function, 862-868
gettype() function, 44
Ghostscript Web site, 775
GIF (Graphics Interchange Format), 485
global privileges, 224
global scope, 151
global variables, 151
GMT (Unix Epoch), 471
Gnu Privacy Guard (GPG), 419
installing, 420–422
key pairs, 420–421
testing, 422–427
Web site, 419
goods (commercial Web sites)
adding value to, 335
digital goods, providing, 334–335
taking orders for, 331–334
Google Web site, 811
GPG (Gnu Privacy Guard), 419
installing, 420–422
key pairs, 420–421
testing, 422–427
Web site, 419
GRANT command, 223-228
GRANT privilege, 295
GRANT statements, 287, 297
grant tables, 288, 293
Graphics Interchange Format (GIF), 485
graphs
data, 499–507
tutorials, 508
Gregorian calendar, 480-481
GROUP BY clause, 256-257
grouping data, 256-257

H

-h switch (mysql command), 221
hackers, 366
handle.php file, 566
handlers
continue, 320
declare, 320
exit, 321
handling
errors, 202
exceptions, 193–195, 565–567
Bob's Auto Parts application, 199–202
catch blocks, 194
classes, creating, 196
Exception class, 195–197
I/O (input/output) files, 199
throwing exceptions, 193
try blocks, 193
tutorials, 203
user-defined exceptions, 196–199
handshaking, 414-415

hardware
failure (commercial Web sites), 337
thieves, 366
hash() function, 354
PHP 5.3, functionality in, 7
HAVING clause, 257
header bar summaries, printing (Shopping Cart application), 632
Header() function, 489-490, 787
headers
generating certificates, 804
message headers (Warm Mail application), 680-681
script architecture, 694
HEAP tables, 312
Hello World scripts, PDFlib, 792-796
heredoc syntax, 27
highlight_file() function, 530-531
highlighting syntax, 530-531
host table, 288, 290-291
hosting services, 382-383
HotScripts.com Web site, 908
htaccess files (Apache Web server), 402-406
HTML (Hypertext Markup Language), 773
embedding PHP, 17-18
comments, 20-21
statements, 19-20
tags, 18-19
whitespace, 20
entities, 372
file upload, 433
formatting (strings), 110-111
forms
file upload, 431
order, creating, 14-16
processing, 14, 17
htmlentities() function, 372-374
htmlspecialchars() function, 272, 372-374, 417
htpasswd program (Apache Web server), 405
HTTP (Hypertext Transfer Protocol), 414, 856
authentication Web sites, 408
basic authentication, 399-400
401 errors, 404
with Apache .htaccess files, 402-406
in PHP, 400-402
digest authentication, 400
handshaking, 414-415
opening files, 64-65
requests, 856-857
Secure Sockets Layer (SSL), 414
XML Amazon connections, 808
httpd.conf, 896-897
HTTPResponse() function, 866
Hypertext Markup Language. *See* HTML
Hypertext Transfer Protocol. *See* HTTP

I

I/O (input/output) files, exception handling, 199
IDE (Integrated Development Environment), 544
identifiers, 28
images, destroying, 490
MySQL identifiers, 235-236
results identifiers, 275-276
identity operator, 87
if statements, 46-47
ImageColorAllocate() function, 488
ImageCreate() function, 487
ImageCreateFromGIF() function, 487, 495
ImageCreateFromJPEG() function, 487, 495
ImageCreateFromPNG() function, 487, 495
ImageDestroy() function, 490
ImageFill() function, 488
ImageFilledRectangle() function, 505, 507
ImageGetTTFBBox() function, 496
ImageJPEG() function, 490
ImageLine() function, 505
ImageMagick library, 483
ImagePNG() function, 490, 495
ImageRectangle() function, 507
images
canvas, creating, 487
colors, RGB (red, green, and blue), 488
creating, 486-487
fonts, 491-495
text, 491-495, 499
data, graphing, 499-507
drawing scripts, 486
figures, drawing, 499-507

formats, 484
GIF (Graphics Interchange Format), 485
JPEG (Joint Photographic Experts Group), 485
PNG (Portable Network Graphics), 485
WBMP (Wireless Bitmap), 485
functions, 507
generating automatically, 490-491
identifiers, destroying, 490
inline, dynamically produced, 491
outputting, 489-490
supporting PHP, 484
text
drawing/printing, 487-489
fitting onto buttons, 495-498
positioning, 498-499
writing onto buttons, 499
ImageString() function, 488
ImageTTFBBox() function, 497
ImageTTFText() function, 496, 507
IMAP (Internet Message Access Protocol), 452, 651-652
client Web site, 891
function library, 652-653
imap_body() function, 679-680
imap_delete() function, 682
imap_expunge() function, 682
imap_fetchheader() function, 679
imap_header() function, 679
imap_headers() function, 676, 679
imap_open() function, 675-676
implementing
inheritance, 167-168
login (online newsletters), 702
PHPBookmark database, 573-577
recommendations, 602-605
implode() function, 117
importing public keys (Gnu Privacy Guard), 422
include() statement, 134
auto_append_file (php.ini file), 142-143
auto_prepend_file (php.ini file), 142-143
include_fns.php files
MLM application, 691
Warm Mail application, 655
Web forum application, 744
increment operators, 35-36
indenting code, 47, 540-541
INDEX privilege, 225
index.html files (certification application), 779-781
index.php file
MLM application, 691
MLM online newsletters, 694
Shopping Cart application, 611
Tahuayo application, 819-826
Warm Mail application, 655-657
Web forum application, 744
index.php script (Shopping Cart application), 615-620
indexes
arrays, 304
creating (MySQL), 234-235
database optimization, 305
queries, 304
inequality operator, 87
inheritance, 162
implementing, 167-168
multiple inheritance, 172-174
preventing, 172
ini_get() function, 529-530
ini_set() function, 529-530
initializing
associative arrays, 85
numerically indexed arrays, 82-83
inline images, dynamically produced, 491
inner join, 254
InnoDB tables
foreign keys, 315-316
transactions, 314-315
input data
checking, 271, 558
filtering, 272
validating, 580
input/output (I/O) files, exception handling, 199
insert book.php script, prepared statements, 280
INSERT privilege, 225
INSERT queries, 276-280

INSERT statement, 244
insert_book.php files (Shopping Cart application), 611
insert_book.php script, 278-279
insert_book.php script (Shopping Cart application), 644-645
insert_book_form.php files (Shopping Cart application), 611
insert_book_form.php script (Shopping Cart application), 644
insert_category.php files (Shopping Cart application), 611
insert_category_form.php files (Shopping Cart application), 611
insert_order() function, 636
insertion anomalies, avoiding (Web databases), 213
install program (Apache), 902
installation
Apache, Windows, 902
binary installations, 890-893
GPG (Gnu Privacy Guard), 420-422
MIME mail package, 905
mod_auth_mysql module, 406-407
MySQL, 900-901
PEAR (PHP Extension and Application Repository), 905-906
PHP, 14, 894, 903-905
project codes, Amazon, 853-854
source installations, 891-896
instanceof type operator, 40
instances, SOAP (Simple Object Access Protocol), 845
instantiating classes, 164
integer data types
numeric column types, 237
variables, 29
Integrated Development Environment (IDE), 544
interfaces
administration interface (Shopping Cart application), 643-650
administrator, Shopping Cart application, 609
PHP database interfaces, 282
Warm Mail application (email client), 654
Web Services (Amazon), 813-814
internationalization (applications), 7
Internet Message Access Protocol (IMAP), 452, 651-652
Internet Protocol (IP), 414
Internet, secure transactions, 411-412
intl extension, 7
intval() function, 101
IP (Internet Protocol), 414
is_uploaded_file() function, 438
isset() function, 45, 155
iteration. *See* loops

J

JavaScript
add_BM() function, 881
addBMResponse() function, 874
addNewBookmark() function, 873
libraries for Ajax applications, 884
XMLHTTPRequest, 860, 862
JD (Julian Day) Count calendar, 480-481
join() function, 117
joins, 254-255
Cartesian product, 254
conditions, WHERE clause, 250
cross, 254
equi-joins, 251, 255
EXPLAIN statement, 301-302
full, 250, 254
inner, 254
left, 252-255
strings
implode() function, 117
join() function, 117
tables, 251-252
two-table, 250-251
types, MySQL, 254-255
JPEG (Joint Photographic Experts Group), 485, 778, 891
jpeg-6b, downloading, 484
Julian calendar, 480-481

K

keys
arrays, 82, 473
databases
creating, 215
foreign keys, 210
primary keys, 210

pairs, installing, 420–421
private keys, 420
public keys, 420–422
keywords
AUTO_INCREMENT, 231
DESC, 255
LIKE, 249
NOT NULL, 231
PRIMARY KEY, 231
REGEXP, 249
return, 154–155
UNSIGNED, 231
krsort() function, 93
ksort() function, 93

L

languages
constructs
array(), 82
die(), 526
exit, 526
DDL (Data Definition Languages), 244
DML (Data Manipulation Language), 244
late static bindings, 185-186
leaf nodes (Web forum tree structure), 743
left joins, 252-255
length of strings, testing, 120
letters, descenders, 497
libpdf_php file, copying, 899
libraries. *See also* functions, libraries
FreeType, downloading, 484
function, 542, 572
ImageMagick, 483
mysqli prepared statements, 280–281
PDFlib
certificates, 796, 802–804
PDF documents, 792–796
PECL (PHP Extension Class Library), 483
PHP, 891
database interfaces, 282
SOAP libraries (Amazon), 814
SOAP, 812
LIKE keyword, 249
LIMIT clause, SELECT statement, 258
links
Add to Cart, 817
Checkout, 818
Details, 817
Web forum tree structure, 742
list() function, 85, 87
lists
creating (online newsletters), 722–724
databases, 688
viewing (online newsletters), 708–716
literals, 27
special characters (regular expressions), 127
LOAD_DATA_INFILE statement, 311
load_list_info() function, 714
loading
arrays from files, 98–101
data from files, 311
extensions, 528
local variables, 151
stored procedures, 319
LOCK TABLES command, 226, 305
locking files, 78-79
logging in
log files, 357
MySQL, 221–222
online newsletters, 705–707
user authentication, 584–587
Warm Mail application (email client), 663–666
logging out
MySQL, 229
online newsletters, 721
user authentication, 587–588
Warm Mail application (email client), 666
logic, 546
errors, 558–559
separating from content, 546
logical operators, 38
login
anonymous login (FTP), 462
FTP servers, 463
implementing (online newsletters), 702
login() function, 585, 706

login.php files
PHPBookmark application, 572–577
Shopping Cart application, 611
logout.php files
PHPBookmark application, 572
Shopping Cart application, 611
logout.php script (authentication), 523-524
long style form variable, 24-26
lookup functions
dns_get_mx(), 459
explode(), 459
networks, 455–459
gethostbyaddr(), 458
gethostbyname(), 456–458
getmxrr(), 456
parse_url(), 458
lookup.php file, 453
loops, 51-53
accessing numerically indexed arrays, 84
associative arrays, 85–87
break statement, 56
do..while loops, 55–56
for loops, 54–55
foreach loops, 54–55
iteration, 188–191
while loops, 53–54
lstat() function, 446
ltrim() function, 110

M

magic quotes, 115
magic_quotes_gpc directive, 417
magic_quotes_runtime directive, 417
Mail Exchange (MX) records, 459
mail() function, 109, 452, 595, 689
mail_fns.php files (Warm Mail application), 655
mail_fns.php function library, get_accounts() function, 668
mailbox (Warm Mail application), viewing contents of, 674-676
mailing list manager. *See* MLM
main page (Shopping Cart application), 615-620
maintainability of code, 538
breaking up, 541–542
code standards, 538
commenting, 540
directory structures, 542
function libraries, 542
indenting, 540–541
naming conventions, 538–540
make_button.php file, 492-493
malicious code injection, 365
many-to-many relationships (databases), 211
master servers, database replication, 306-307
matching
regular expressions, 123–124
* symbol, 126
+ symbol, 126
branching, 127
caret symbol (^), 126–127
character classes, 125
character sets, 124–125
curly braces ({}), 126
finding substrings, 129–130
literal special characters, 127
replacing substrings, 130
slash (\), 127
special characters, 127–128
splitting strings, 130
string anchoring, 126–127
subexpressions, 126
Web references, 131
substrings, 120–121
find and replace, 122–123
numerical position, 121
regular expressions, 129–130
strchr() function, 121
stristr() function, 121
strpos() function, 121
strrchr() function, 121
strrpos() function, 122
strstr() function, 121
max() function, 155-156
MAX(column) function, 256
max_connections parameter, 273
MaxClients parameter (Apache), 273

md5(), PHP 5.3, functionality in, 7
medium style form variable, 24
member.php files (PHPBookmark application), 572
members_only.php script (authentication), 522-523
MEMORY tables, 312
MERGE tables, 312
messages. *See also* errors, messages
sending, 733, 737-739
viewing, 680-681
methods. *See also* functions
ASINSearch(), 829
bind_param(), 281
browseNodeSearch(), 829, 835
_call(), 186-187
Exception class, 195
fetchRow(), 284
overloading, 186-187
parseXML(), 838
static, 184
microseconds, measuring, 480
Microsoft
IIS, configuring, 381
Web site, 773
Word, RTF, 774
microtime() function, 480
MIME mail package, installing, 905
MIN(column) function, 256
mirroring
FTP functions, 459-463
checking update times, 464-465
closing connections, 466
downloads, 465-466
logins, 463
remote connections, 463
RAID (Redundant Array of Inexpensive Disks), 358
mktime() function, 471-472
MLM (mailing list manager), 687
actions, 701
building, 687
files, 690
online newsletters, 687
account settings, 702-705, 719
administrative functions, 721
databases, 688, 692-694
diagrams, 689-691
email attachments, 689
extensions, 740
file upload, 688-689
lists, 708-717, 722-724
logging in, 705-707
logging out, 721
login, implementing, 702
passwords, 719-721
previewing, 732-733
requirements, 688
script architecture, 694, 700-701
sending messages, 733, 737-739
solution overview, 689-691
subscribing, 717-718
unsubscribing, 717-718
uploading, 724-731
mlm_fns.php files (MLM application), 691
mod_auth module (Apache Web server), 402
mod_auth_mysql module, 406-408
documentation Web sites, 408
installing, 406-407
testing, 407
mode variable, 824
modeling real-world objects (Web databases), 211-212
modes
autocommit, 314
file modes, 61-62
modification
anomalies, avoiding (Web databases), 213
dates (scripts), 529
modules
code, 539, 610
mod_auth (Apache Web server), 402
mod_auth_mysql, 406-408
installing, 406-407
testing, 407
operator, 33
PHP, running, 890
monitors
MySQL, 220-221
security, 363
moving files, 447

multidimensional arrays, 81-92
sorting, 93
reverse sorts, 95
user-defined sorts, 93-95
three-dimensional arrays, 90-92
two-dimensional arrays, 88-90
multiline comments, 21
multiple files, 727, 731
multiple inheritance, 172-174
multiple programmers, 543
multiplication operator, 33
MX (Mail Exchange) records, 459
my_error_handler() function, 565
MyISAM table, 312
myisamchk utility, 303, 306
MySQL
access, 219-220
aggregate functions, 256
continuation symbol, 220
database, 287, 292
backing up, 358
connection verification, 293
creating, 222
db table, 290-291
host table, 291
request verification, 293
results.php script, 269
selecting, 229
tables, creating, 229-231
tables_priv table, 292
user table, 289
viewing, 233-234
Web database architecture, 268-271
date and time
converting between PHP and MySQL formats, 476-477
date calculations, 478, 480
DATE_FORMAT() function, 476-477
MySQL Web site, 481
UNIX_TIMESTAMP() function, 476-477
GRANT command, 223-228
identifiers, 235-236
installation
binary installations, 890-893
source installations, 891
Windows, 900
Windows, setting PATH, 900-901
join types, 254-255
logging in, 221-222
logging out, 229
max_connections parameter, 273
mod_auth_mysql module, 406-408
documentation Web sites, 408
installing, 406-407
testing, 407
mysql command, 221
online manual, 241
passwords, 418
privileges, 223
global privileges, 224
GRANT command, 223-228
principle of least privilege, 223
REVOKE command, 227-228
types, 225-227
resources, 909
REVOKE command, 227-228
runtime errors, 555-557
semicolon (;), 220
statements, 221
syntax, 257
users
GRANT command, 224-228
REVOKE command, 227-228
setting up, 223, 227-229
Web site, 220
mysql command, 221
mysql_affected_rows() function, 280
mysql_dump command, 305
mysql_query() function, 274-275
mysql_select_db() function, 274
mysqladmin facility, 235
mysqlhotcopy script, 306
mysqli_connect() function, 274, 555
mysqli_errno() function, 556
mysqli_error() function, 556
mysqli_fetch_assoc() function, 275
mysqli_query() function, 556

N

namespaces, 158
PHP 5.3, 7
XML, 811
naming
conventions, 538–540
functions, 147–148
Natural Order String Comparison Web site, 119
navigating
within arrays, 102
files, 76–77
Netcraft, 382
Netscape Web site
cookie specification, 511
SSL 3.0 Specification, 427
Network News Transfer Protocol (NNTP), 452
networks
connecting, 557–558
lookup functions, 455–459
dns_get_mx(), 459
explode(), 459
gethostbyaddr(), 458
gethostbyname(), 456–458
getmxrr(), 456
parse_url(), 458
TCP/IP security, 343
new operator, 39
New York Times Web site, 392
new_post.php files, 763
Web forum application, 744
newbooks.txt file, 311
newline control sequence (\n), 68
newsletters, 687
accounts
configuring, 719
creating, 702–705
administrative functions, 721
databases
configuring, 692–694
lists/subscribers, 688
diagrams, 689–691
email attachments, 689
extensions, 740
file upload, 688–689
lists
archives, viewing, 716–717
creating, 722–724
viewing, 708–716
logging in, 705–707
logging out, 721
login, implementing, 702
passwords, 719–721
previewing, 732–733
requirements, 688
script architecture, 694, 700–701
sending messages, 733, 737–739
solution overview, 689–691
subscribing, 717–718
unsubscribing, 717–718
uploading, 724–731
next() function, 102
nl2br() function, 110-111
NNTP (Network News Transfer Protocol), 452
nodes
browse nodes (Amazon), 816
Web forum tree structure, 742
child nodes, 743
leaf nodes, 743
parent nodes, 743
root nodes, 743
non-identity operator, 87
NOT NULL keyword, 231
notify_password() function, 592-594
NULL data type (variables), 29
null values, avoiding (Web databases), 216
number_of_accounts() function, 671
numeric column types, 236-238
date and time, 238–239
floating point data types, 237–238
integral data types, 237
string, 239–241
numerical position of substrings, finding, 121
numerically indexed arrays
accessing with loops, 84
contents, accessing, 83–84
initializing, 82–83

O

Object data type (variables), 29
objects, 160-161
cloning, 186
real-world modeling (Web databases), 211-212
throwing, 196
ODBC (Open Database Connectivity) functions, 282
one-to-many relationships (databases), 211
one-to-one relationships (databases), 211, 216
online brochures (commercial Web sites), 328
common pitfalls, 330
limitations, 328
answering feedback, 329
lack of information, 328
poor presentation, 329
tracking success, 330-331
updated information, 329
online catalogs (Shopping Cart application), 608
online newsletters, 687
accounts
configuring, 719
creating, 702-705
administrative functions, 721
databases
configuring, 692-694
lists/subscribers, 688
diagrams, 689-691
email attachments, 689
extensions, 740
file upload, 688-689
lists
archives, viewing, 716-717
creating, 722-724
viewing, 708-716
logging in, 702, 705-707
logging out, 721
passwords, 719-721
previewing, 732-733
requirements, 688
script architecture, 694, 700-701
sending messages, 733, 737-739
solution overview, 689-691
subscribing, 717-718
unsubscribing, 717-718
uploading, 724-727, 731
OOP (object-oriented programming)
classes, 160-161
objects, 160-161
polymorphism, 161
Open Database Connectivity (ODBC) functions, 282
open_mailbox() function, 675
opendir() function, 440
opening files, 61
file modes, 61-62
fopen() function, 62-64
FTP (File Transfer Protocol), 64-65
HTTP (Hypertext Transfer Protocol), 64-65
potential problems, 65-66
tags (XML), 810
OpenSSL
configuring, 894
Web site, 891
operating systems
database security, 294
unnecessary applications, disabling, 388
updating, 387-388
operations
creating, 162-164
overriding, 173
operators, 32
arithmetic operators, 33-34
arrays, 40, 87-88
assignment (=), 25, 28, 34
combination assignment operators, 35
decrement operators, 35-36
increment operators, 35-36
reference operator, 36
returning values, 34-35
associativity, 42-44
bitwise operators, 38
comma operator, 39
comparison operators, 36-37
equals operator, 37
WHERE clauses, 248-249

error suppression operator, 39
execution operator, 39-40
logical operators, 38
new operator, 39
precedence, 42-44
strings
concatenation operator, 26-27
operators, 34
subqueries, 259
ternary operator, 39
totaling forms, 41-42
type operator, 40
unary operators, 33
optimizing
code, 546-547
databases, 304-305
default values, 305
designs, 304
indexes, 305
permissions, 304
tables, 304
Zend Optimizer, 547
or operator, 38
ORDER BY clause, 255
order forms
creating, 14-16
processing, 16
order.fns.php files (Shopping Cart application), 612
ordered data, retrieving, 255-256
ordering strings
strcasecmp() function, 119
strcmp() function, 119
strnatcmp() function, 119
orders for goods or services (commercial Web sites), 331-332
compatibility, 334
trust, 333
unanswered questions, 332
user interfaces, 333-334
organizing code, 374
output_fns.php files
MLM application, 691
PHPBookmark application, 572
Shopping Cart application, 612
Warm Mail application, 655
Web forum application, 744
output_fns.php function library, 664
outputting images, 489-490
overloading
functions, 147
methods, 186-187
overriding, 170-173
owners (scripts), identifying, 529

P

-p switch (mysql command), 221
padding characters, 112
pages. *See* Web pages
parameters, 22
$type, 829
Apache, MaxClients, 273
drawing functions, 488
extract() function, 105
function parameters, 148-150
calling functions, 143-144
pass by reference, 153-154
pass by value, 153-154
max_connections parameter, 273
startup, 900
parent nodes (Web forum tree structure), 743
parse_url() function, 458
parseXML() method, 838
parsing XML (Amazon), 814
pass by value (function parameters), 153-154
passing by reference, 104, 153-154
passthru() function, 448
passwords, 350-351, 362, 570
databases
access, 383-384
security, 295
encrypting, 295, 397-399
logging in to MySQL, 221-222
MySQL, 418
online newsletters, 719-721
storing, 295, 395
user authentication, 588-595
PATH settings, MySQL installations, 900-901

paths
absolute, 62
file, 442–443
relative, 62
payments
modules, 639–641
systems, 608–609
PCRE extension, 7
PDF (Portable Document Format), 771–775
generating certificates, 788–791
headers, 804
PDFlib, 792–804
personalized documents, 772
readers, 794–795
templates, creating, 776–777
Web site, 775
pdf.php files (certification application), 779
pdf_add_outline() function, 794
pdf_begin_page() function, 793
pdf_close() function, 796
pdf_fill() function, 804
pdf_rect() function, 802
pdf_replace() function, 789
pdf_set_info() function, 793
pdf_setlinewidth() function, 802
pdf_show() function, 795
pdf_show_xy() function, 803
pdf_stringwidth() function, 803
pdf_stroke() function, 802
PDFlib
certificates, 796, 802–804
PDF documents, 792–796
pdflib.php files, 796
certification application, 779
PEAR (PHP Extension and Application Repository)
Databases, 284–285
installing, 905–906
Web site, 907
PECL (PHP Extension Class Library), 483
Web site, 907
per-class constants, 184
Perl regular expressions, 123
permissions
database optimization, 304
write files, 418
personalization, 771
certification project, 779
files, 779
index.html file, 780–781
PDF, 788–791
PDF, PDFlib, 792–796
PDFlib, 796, 802–804
RTF, 784–787
score.php file, 782–784
creating, 771–772
extensions, 805
formats, 772
ASCII, 772
HTML, 773
paper, 772
PDF, 775
PostScript, 774–775
RTF, 774
word processors, 773
headers, 804
requirements
questions/answers, 776
software, 776–777
users
bookmarks, 571, 596–602
defined, 569
passwords, 570
recommendations, implementing, 602–603, 605
solutions, 570–572
system requirements, 570
usernames, 570
PGP (Pretty Good Privacy), 419
phar extension, 7
Philip and Alex's Guide to Web Publishing Web site, 910
Phorum web forums project, 770
PHP
advanced OO features, 184–186, 191
Application Tools Web site, 909
Base Library Web site, 908
basic authentication (HTTP), 400–402
calling functions, 22
Center Web site, 908
Classes Repository Web site, 908
Club Web site, 908

command line, 531
configuring, 894
constants, 31
control structures, 46, 49
alternate syntax, 56
breaking out of, 56
conditionals, 46-51
declare, 57
loops, 51-56
database interfaces, 282
date and time, 469, 474
calendar functions, 480-481
checkdate() function, 474
converting between PHP and MySQL formats, 476-477
date calculations, 477-478
date() function, 469-472
floor() function, 478
getdate() function, 473
microseconds, 480
mktime() function, 471-472
PHP Web site, 481
date() function, 21-22
Developer Web site, 909
Developer's Network Unified Forums Web site, 909
development environments, 544
embedding in HTML, 17-18
comments, 20-21
statements, 19-20
tags, 18-19
whitespace, 20
environment variable functions, 450
evaluating strings, 525-526
extensions directory, copying libpdf_files, 899
functions
eval() function, 525-526
get_current_user() function, 529
get_extension_funcs(), 528
get_loaded_extensions() function, 528
getlastmod() function, 529
highlight_file(), 530-531
ini_get() function, 529-530
ini_set() function, 529-530
my_error_handler() function, 565
mysql connect() function, 555
mysqli_errno() function, 556
mysqli_error() function, 556
mysqli_query() function, 556
names in code, 539
serialize() function, 526-527
set_error_handler() function, 565
show_source() functions, 530-531
unserialize() function, 527
variables, 44-46
gd documentation Web site, 508
highlighting syntax, 530-531
Homepage Web site, 908
images
canvas, creating, 487
creating, 486-499
formats, 484
generating automatically, 490-491
GIF (Graphics Interchange Format), 485
identifiers, destroying, 490
JPEG (Joint Photographic Experts Group), 485
outputting, 489-490
PNG (Portable Network Graphics), 485
supporting, 484
text, 487-499
WBMP (Wireless Bitmap), 485
installation, 14, 894
binary installations, 890
source installations, 891, 893-896
Windows, 903-905
jpeg-6b, downloading, 484
Kitchen Web site, 909
language constructs
die(), 526
exit, 526
libraries, 891
Magazine Web site, 907
modular names in code, 539
network lookup functions, 455-459
dns_get_mx(), 459
explode(), 459

gethostbyaddr(), 458
gethostbyname(), 456-458
getmxrr(), 456
parse_url(), 458
online manual, 80
operators, 32
arithmetic operators, 33-34
array operator, 40
assignment operators, 28-36
associativity, 42-44
bitwise operators, 38
comma operator, 39
comparison operators, 36-37
error suppression operator, 39
execution operator, 39-40
logical operators, 38
new operator, 39
precedence, 42-44
string operators, 34
ternary operator, 39
totaling forms, 41-42
type operator, 40
unary operators, 33
optimizations, 546-547
PHP 5.3
bug fixes in, 7
crypt() functionality in, 7
date/time functions in, 7
date_add() function, 478
date_sub() function, 478
error reporting in, 7
fileinfo extension, 7
hash() functionality in, 7
intl extension, 7
md5() functionality in, 7
MySQLnd drivers, 7
namespaces, 7
new features of, 7
PCRE extension, 7
phar extension, 7
php.ini administration in, 7
Reflection extension, 7
SPL extension, 7
sqlite3 extension, 7
time/date functions in, 7
Windows support for, 7, 900
Zend engine improvements, 7
resources, 907-909
Resource Web site, 908-909
running
as CGI Interpreter, 890
as modules, 890
scripts, 551
debugging variables, 559, 561
errors, 562-567
modification dates, 529
programming errors, 551-558
MySQL passwords, 418
owners, identifying, 529
terminating execution, 526
serialization, 526-527
sessions. *See* sessions
SOAP libraries (Amazon), 814
statements, 19-20
tags, 18-19
ASP style, 19
require() statement, 136
SCRIPT style, 19
Short style, 19
variables
form variables, accessing, 23-27
identifiers, 28
names in code, 539
scope, 31-32
superglobal, 32
types, 29-30
user declared variables, 28
values, assigning, 28
Web site, 481, 537, 891
writing, 434-438
XML style, 19
PHP Extension and Application Repository (PEAR)
installing, 905-906
Web site, 907
PHP, Hypertext Preprocessor Web site, 106
php.ini file
administration in PHP 5.3, 7
auto_append_file, 142-143
auto_prepend_file, 142-143

directives, editing, 529-530
examining, 380
phpautodoc Web site, 545
PHPBookmark application
Ajax elements, adding, 871
creating, 569
database schema, 573-574
front page, 574-577
function libraries, 572
extensions, 606
files, 572
project, 870-883
PHPBuilder.com Web site, 908
PHPCertifcation.pdf files (certification application), 779
PHPCertification.rtf files (certification application), 779
PHPCommunity Web site, 907
phpdoc Web site, 544
PHPDocumentor Web site, 544
PHPIndex.com Web site, 908
phpinfo() command, 31
phpinfo() function, 450, 778
PHPMyAdmin.Net Web site, 908
PHPWizard.net Web site, 908
php|architect Web site, 907
physical security, 359, 388
plain text (encryption), 351
plus symbols (+)
regular expressions, 126
Web forum articles, 748
PNG (Portable Network Graphics), 485
library Web site, 891
pollsetup.sql file, 500
polymorphism, 161
POP (Post Office Protocol), 452
POP3 (Post Office Protocol version 3), 651-652
populate.sql files (Shopping Cart application), 612
Portable Document Format. ***See*** **PDF**
Portable Network Graphics. ***See*** **PNG**
positioning text buttons, 498-499
POSIX regular expressions. ***See*** **regular expressions**
posix_getgrgid() function, 446
posix_getpwuid() function, 446
posix_getpwuid() functions, 444
Post Office Protocol (POP), 452
Post Office Protocol version 3 (POP3), 651-652
post-decrement operator, 35-36
post-increment operator, 35-36
posters (Web forum application), 744
Postnuke Web site, 909
PostScript, 774-775
Downloading fonts, 484
power failures, 359
pre-decrement operator, 35-36
pre-increment operator, 35-36
precedence, operators, 42-44
prepared statements, 280-281
preparing for DoS/DDoS attacks, 387
preprocessing script architecture, 694
Pretty Good Privacy (PGP), 419
pretty() function, 714
prev() function, 102
preventing
inheritance, 172
overriding, 172
previewing online newsletters, 732-733
PRIMARY KEY keyword, 231
primary keys (databases), 209-210
principle of least privilege, 223
print() function, 110
printf() function, 111-112
printing
header bar summaries (Shopping Cart application), 632
strings, 110-113
print() function, 110
printf() function, 111-112
sprintf() function, 111
text images, 487-489
privacy policies
commercial Web sites, 333
SSL (Secure Sockets Layer), 333
private access modifier, 166-167
visibility, controlling, 169-170
private keys
encryption, 353
Gnu Privacy Guard (GPG), 420
privileges
FILE, 295
GRANT, 295

MySQL, 223
global privileges, 224
GRANT command, 223-228
principle of least privilege, 223
REVOKE command, 227-228
PROCESS, 295
types, 225-227
system, 287-288
columns_priv table, 293
db table, 290-291
grant table, 293
host table, 290-291
privileges, updating, 293-294
slaves, 307
tables_priv table, 293
user table, 289-290
updating, 293-294
user database security, 295-296
PROCESS privilege, 226, 295
process.php files (Shopping Cart application), 611
process.php script (Shopping Cart application), 639
processing HTML forms, 14, 17
Product class, 839
Product.php files (Tahuayo application), 819
progex.php file, 448-449
programming errors, 551, 553-554
logic errors, 558-559
runtime errors, 553-554
database interaction, 555-557
functions that don't exist, 554-555
input data, checking, 558
network connections, 557-558
reading/writing files, 555
syntax errors, 552-553
programs. *See also* applications
install (Apache), 902
running command line, 531
project codes, installing (Amazon), 853-854
property files, changing, 446
protocols, 451-452
application layer protocols, 414
File Transfer Protocol (FTP), 459
anonymous login, 462
backing up files, 459-465
ftp_get() function, 466
ftp_mdtm() function, 464
ftp_nlist() function, 467
ftp_size() function, 467
mirroring files, 459-465
set_time_limit() function, 467
timeouts, avoiding, 467
uploading files, 466
FTP (File Transfer Protocol), 64-65
HTTP (Hypertext Transfer Protocol), 414
handshaking, 414-415
opening files, 64-65
Secure Sockets Layer (SSL), 414
IMAP (Internet Message Access Protocol), 452, 651-652
IP (Internet Protocol), 414
NNTP (Network News Transfer Protocol), 452
POP (Post Office Protocol), 452
POP3 (Post Office Protocol version 3), 651-652
RFCs (Requests for Comments), 451-452
SMTP (Simple Mail Transfer Protocol), 452, 652
SOAP (Simple Object Access Protocol), 845-846
stacks, 413-414
TCP (Transmission Control Protocol), 414
Web Services
SOAP (Simple Object Access Protocol), 811-812
WSDL (Web Services Description Language), 812
prototypes
code, 545-546
functions, 144
public access modifier, 166-170
public keys
encryption, 353-354
Gnu Privacy Guard (GPG), 420-422
purchase.php files (Shopping Cart application), 611
purchase.php script (Shopping Cart application), 634, 639

putenv() function, 450
PX-PHP Code Exchange Web site, 908

Q

queries
EXPLAIN statement, 299-303
indexes, 304
INSERT, 276-280
subqueries, 258-259
correlated, 260
operators, 259
row, 260
temporary tables, 260
Web databases, 271
adding data, 276-280
connections, setting up, 273
disconnecting from databases, 276
input data, 271-272
mysql_query() function, 274-275
prepared statements, 280-281
retrieving results, 275-276
selecting databases, 274
quotes, magic quotes, 115

R

r+ file mode, 63
RAID (Redundant Array of Inexpensive Disks), 358
range() function, 83
RDBMS (relational database management systems), 80, 243
readdir($dir) function, 440
readers, PDF, 794-795
readfile() function, 74
reading
from directories, 439-441
files, 61, 71-72, 444-446
feof() function, 73
fgetc() function, 75
fgetcsv() function, 73-74
fgets() function, 73
fgetss() function, 73
file() function, 74
fopen() function, 72
fpassthru() function, 74
fread() function, 75
readfile() function, 74
runtime errors, 555
Warm Mail application, 671, 681
mailbox contents, viewing, 674-676
messages, 677-678, 680-681
selecting accounts, 671, 673
real-world objects, modeling (Web databases), 211-212
recommend.php files (PHPBookmark application), 572
recommend_urls() function, 603, 605
recommendations
bookmarks, 571
implementing, 602-603, 605
records
deleting, 264
updating, 261
tables, 209
recursive functions, 156-158
red, green, and blue (RGB), 488
Redundant Array of Inexpensive Disks (RAID), 358
redundant data, avoiding (Web databases), 212-213
reference operator, 36
reflection API, 190-191
Reflection extension, 7
REGEXP keyword, 249
register() function, 582
register_form.php files (PHPBookmark application), 572
register_new.php files (PHPBookmark application), 572
registering
session variables, 513
user authentication, 577, 580-583
regression, 377
regular expressions, 123-124
* symbol, 126
+ symbol, 126
branching, 127
caret symbol (^), 126-127
characters
classes, 125
sets, 124-125
curly braces ({}), 126
functions versus string functions, 131

Perl, 123
slash (\), 127
Smart Form Mail application, 128–129
special characters, 127–128
splitting strings, 130
string anchoring, 126–127
subexpressions, 126
substrings
finding, 129–130
replacing, 130
Web references, 131
reinterpreting variables, 46
relational database management systems. *See* RDBMS
relational databases, 208, 210
benefits, 207
keys, 209
foreign keys, 210
primary keys, 210
relationships, 211
many-to-many relationships, 211
one-to-many relationships, 211
one-to-one relationships, 211, 216
schemas, 210
tables, 208
columns, 209
rows, 209
values, 209
relationships (databases), 211
many-to-many relationships, 211
one-to-many relationships, 211
one-to-one relationships, 211, 216
relative paths, 62
RELOAD privilege, 226
remote FTP connections, 463
rename() function, 447
reordering arrays, 96
array_reverse() function, 97–98
shuffle() function, 96
repetitive tasks. *See* loops
replacing substrings, 122-123
with regular expressions, 130
replication, databases, 306-307
data transfer, 306–308
master servers, 306–307
slaves, 306–308
REPLICATION CLIENT privilege, 226
REPLICATION SLAVE privilege, 226
replying to email, Warm Mail application, 684-685
repository (version control, code), 542
repudiation, 348-349
requests
HTTP, 856–857
MySQL database, 293
server response, 866
Requests for Comments (RFCs), 451-452
require() statement, 135-136
auto_append_file (php.ini file), 142–143
auto_prepend_file (php.ini file), 142–143
filename extensions, 136
PHP tags, 136
Web site templates, 137–142
reset password() function, 592
reset() function, 102
resetting passwords, user authentication, 593
resources, 907
Apache, 909
data types, 29
MySQL and SQL, 909
PHP, 907–909
Web development, 910
responses (HTTP), 866
REST/XML (Amazon), 838-839, 844
restoring databases, 306
restricting access
to .php files, 374–375
to sensitive data, 364
result identifiers, retrieving query results (Web databases), 275-276
results.php script, 269
retrieve_message() function, 678
returning
assignment operator, 34–35
from functions, 154–155
keywords, 154–155
policies, 333
rows, 258
statements, 154
values, 94, 155–156

reusing code
benefits, 133-134
consistency, 134
cost, 134
reliability, 134
include() statement, 134, 142-143
require() statement, 135-136, 142-143
auto_prepend_file (php.ini file), 142-143
filename extensions, 136
PHP tags, 136
Web site templates, 137-142
reverse sort order
arrays, 93
multidimensional arrays, 95
reverse spam, 346
REVOKE command, 227-228
rewind() function, 76
rewinddir($dir) function, 441
rewriting code, 537-538
RFCs (Requests for Comments), 451-452
RFC Editor Web site, 451, 468
RGB (red, green, and blue), 488
Rich Text Format (RTF), 771, 774
risks for commercial Web sites, 336
competition, 338
crackers, 337
failure to attract business, 337-338
hardware failure, 337
legislation and taxes, 339
service provider failures, 338
software errors, 338
system capacity limits, 339
rmdir() function, 443
rolled back transactions, 314
root elements (XML), 811
root nodes (Web forum tree structure), 743
rows
returning, 258
subqueries, 260
unmatched, 252-253
values, 209
RSA, 353
rsort() function, 93
RTF (Rich Text Format), 771, 774
generating certificates, 784-787
templates, creating, 776
rtf.php files, 779, 786
rtrim() function, 110
running
Apache, 897
command line programs, 531
PHP
as CGI Interpreter, 890
as modules, 890
runtime errors, 553-554
database interaction, 555-557
functions that don't exist, 554-555
input data, checking, 558
network connections, 557-558
reading/writing files, 555

S

S-HTTP (Secure Hypertext Transfer Protocol), 412
safeString() function, 825
scalar variables, 81,
converting arrays to, 105-106
schemas
Book-O-Rama application, 219, 230
database (PHPBookmark application), 573-577
scope
fields, 290
function scope, 151
global scope, 151
variable scope, 31-32, 150-153
score.php files (certification project), 779-784
screening user input, 417
SCRIPT style (PHP tags), 19
scripts
admin.php script (Shopping Cart application), 641, 643
architecture
footers, 694
headers, 694
online newsletters, 694-701
performing actions, 694
preprocessing, 694
authmain.php (authentication), 517-522
breaking out of, 56

buttons, calling, 493
catalog scripts (Shopping Cart application), 615-617
index.php, 615-620
show_book.php, 616, 622-623, 646
show_cat.php, 615, 620-622
checkout.php script (Shopping Cart application), 633-638
edit_book_form.php (Shopping Cart application), 646
executing, 531
Hello World, 792-796
images, drawing, 486
insert_book.php, 278-279, 644-645
prepared statements, 280
insert_book_form.php script (Shopping Cart application), 644
logout.php (authentication), 523-524
make_button.php, 492
members_only.php (authentication), 522-523
modification dates, 529
mysqlhotcopy, database backup, 306
owners, identifying, 529
PHP, MySQL passwords, 418
process.php script (Shopping Cart application), 639
purchase.php script (Shopping Cart application), 634, 639
querying Web databases, 271
adding data, 276-280
connections, setting up, 273
disconnecting from databases, 276
input data, 271-272
mysql_query() function, 274-275
prepared statements, 280-281
retrieving results, 275-276
selecting databases, 274
results.php, 269
servertime.php, 863-864
show_book.php (Shopping Cart application), 646
show_cart.php script (Shopping Cart application), 623-627
adding items to cart, 630-631
header bar summary, printing, 632
updated carts, saving, 631-632
viewing contents of cart, 627-630
terminating execution, 526
Warm Mail application (email client), 657, 662-663
Web database architecture, 217
SearchDatabase.com Web site, 909
searching substrings, 120-121
find and replace, 122-123
numerical position, 121
regular expressions, 129-130
strchr() function, 121
stristr() function, 121
strpos() function, 121
strrchr() function, 121
strrpos() function, 122
strstr() function, 121
Secure Hypertext Transfer Protocol (S-HTTP), 412
Secure Socket Layer. *See* SSL
secure storage, 417-419
secure transactions, 409-410
Internet, 411-412
screening user input, 417
Secure Sockets Layer (SSL), 413-415
compression, 416
handshaking, 414-415
protocol stacks, 413-414
sending data, 415-416
secure storage, 417-419
systems, 412-413
user machines, 410-411
Web browsers, 410-411
Secure Web servers, 355-357
security, 362
authentication, 343, 401-406
access control, implementing, 392-395
basic authentication. *See* basic authentication
custom, creating, 408
digest authentication, 400
encrypting passwords, 397-399
identifying users, 391-392
mod_auth_mysql module, 406-408
multiple pages, protecting, 399

passwords, 350-351
storing passwords, 395
Web sites, 408
bottom-up approach, 363
bugs, testing for, 376-377
Certifying Authorities (CAs), 355
code organization, 374
commercial Web sites, 342
auditing, 357
authentication, 350-351
backing up data, 358
Certificate Signing Request (CSR), 356-357
compromises, 349
crackers, 337
digital certificates, 355
digital signatures, 354-355
encryption, 351-354
firewalls, 357-358
hash function, 354
importance of stored information, 342
log files, 357
passwords, 350-351
physical security, 359
Secure Web servers, 356-357
security policies, creating, 349-350
threats, 342-349
databases, 294, 384
authentication, 383-384
connecting to servers, 384-385
operating system, 294
passwords, 295
servers, 385
user privileges, 295-296
Web issues, 296
denial of service, 364
disaster recovery, 364, 388-389
DMZ, 386-387
DoS attacks, preparing for, 387
effect on usability, 362
encryption, 352-353, 419-420
Data Encryption Standard (DES), 353
GPG (Gnu Privacy Guard), 419-427
PGP (Pretty Good Privacy), 419
files
system considerations, 375-376
uploads, 434, 438
firewalls, 386
hosting services, 382-383
malicious code injection, 365
monitoring, 363
output, escaping, 371
passwords, 362
.php files, restricting access to, 374-375
physical security, 388
restricting access to sensitive data, 364
Secure Socket Layer (SSL), 344
SQL injection attacks, 371
TCP/IP networks, 343
top-down approach, 363
transactions, 409-410
Internet, 411-412
screening user input, 417
Secure Sockets Layer (SSL), 413-416
secure storage, 417-419
systems, 412-413
user machines, 410-411
Web browsers, 410-411
user input, filtering, 367-371
SELECT clauses, 257
SELECT privileges, 225
SELECT statements, 246
LIMIT clause, 258
ORDER BY clause, 255
selecting
databases in MySQL, 229
Web databases, 274
selectors (CSS), 858
semicolon (;), MySQL, 220, 274
send() function, 734
send_message() function, 683-684
sending
email, 452
messages, online newsletters, 733, 737-739

Warm Mail application
forwarding/replying, 684-685
new messages, 682-684
sensitive data, storing, 417-419
serialization, 526-527
session variables, 514
serialize() function, 526-527, 848
server-side programming, 860
servers
Apache. *See* Apache, Web server
authentication, 351
communication with Ajax, 863-864
database servers, Web database architecture, 217
master, database replication, 306-307
response to HTTP requests, 866
secure storage, 417-419
Secure Web servers, 355-357
Web servers, Web database architecture, 216
servertime.php script, 863-864
services
adding, 335, 452-454
providing, 334-335
taking orders for, 331-334
session_get_cookie_params() function, 511
session_register() function, 513
session_start() function, 512, 514-515
session_unregister() function, 513
sessions, 509, 512
authentication, 517-524
configuring, 516-517
cookies, 510-511
creating (Amazon), 823
destroying, 513
example session, 514-516
IDs, 509-512
Shopping Cart application, 608, 623
starting, 512
variables, 510
deregistering, 513
implementing, 513
registering, 513
serializing, 514
set cardinality (arrays), 104
SET type, 241
set_error_handler() function, 565
set_time_limit() function, 467
setcookie() function, 510-511
setting up
Book-O-Rama, 243
databases of lists, 688
settype() function, 44
SGML (Standard Generalized Markup Language), 808
shal1() function, 398
shell command executor, 377-378
shell script-style comments, 20
Shopping Cart application, 607, 617, 624, 643, 650
administration
interfaces, 609
views, 609-610
administration interface, 643-647, 650
administration menu (admin.php), 641, 643
edit_book_form.php script, 646
insert_book.php script, 644-645
insert_book_form.php script, 644
show_book.php script, 646
book_sc database, 612-615
catalog scripts, 615-617
index.php, 615-620
show_book.php, 616, 622-623, 646
show_cat.php, 615, 620-622
code modules, 610
database, 615
extensions, 650
files, 611-612
online catalogs, building, 608
payments
modules, 639-641
systems, 608-609
session variables, 608, 623
shopping cart module
adding items, 630-631
checkout.php script, 633-638
header bar summary, printing, 632
purchase.php script, 634, 639
show_cart.php script, 623-627

updates, saving, 631-632
viewing contents of, 627-630
solution overview, 609-612
tracking user's purchases, 608
user view, 609-610
shopping carts, 607
building (Amazon), 813, 849-852
Short style (PHP tags), 19
short style form variable, 23-24
SHOW COLUMNS statement, 297
SHOW command, 233-234
SHOW DATABASES privilege, 226
SHOW statement, 296-297
SHOW TABLES statement, 297
show_book.php files (Shopping Cart application), 611
show_book.php script (Shopping Cart application), 616, 622-623, 646
show_cart.php files (Shopping Cart application), 611
show_cart.php script (Shopping Cart application), 623, 625, 627
adding items to cart, 630-631
header bar summary, printing, 632
updated carts, saving, 631-632
viewing contents of cart, 627-630
show_cat.php files (Shopping Cart application), 611
show_cat.php script (Shopping Cart application), 615, 620-622
show_source() function, 530-531
showBrowseNode() function, 826-827
showCart() function, 852
showCategories() function, 826
showpoll.php file, 502-504, 506
ShowSmallCart() function, 825
showSummary() function, 828, 844
shuffle() function, 96
SHUTDOWN privilege, 226
signature.png files (certification application), 779
Simple Mail Transfer Protocol (SMTP), 452, 652
Simple Object Access Protocol. *See* SOAP
simplegraph.php file, 486
sin() function, 804
single-line comments, 21
sites. *See* commercial Web sites; Web sites
sizeof() function, 104
slash (\), 311
regular expressions, 127
Slashdot Web site, 392, 741
slaves
database replication, 306-308
replication, 307
Smart Form Mail application
creating, 107-109
regular expressions, 128-129
SMTP (Simple Mail Transfer Protocol), 452, 652
SOAP (Simple Object Access Protocol), 808-812
Amazon, 807-808, 845-846
envelopes, 812
example, 811
instances, 845
libraries, 812
PHP SOAP libraries (Amazon), 814
software
engineering, 536
errors, 338, 347
developer assumptions, 347
poor specifications, 347
poor testing, 348
personalized documents, 776
PDF, 776-777
RTF, 776
updating, 378-379
solutions, user personalization, 570-572
sort() function, 92
sorting arrays, 92
asort() function, 93
ksort() function, 93
multidimensional, 93
reverse sorts, 95
user-defined sorts, 93-95
reverse order, 93
sort() function, 92
source installations, 891-896
SourceForge Web site, 545, 909
spam, 346

special characters
literal special characters (regular expressions), 127
regular expressions, 127–128
special privileges, 227
specifications, CGI Web site, 450
speed of queries, 304
SPL extension, 7
split() function, 130
splitting strings
explode() function, 116–117
regular expressions, 130
strtok() function, 117
substr() function, 118–119
sprintf() function, 111
SQL (Structured Query Language), 243
ANSI standard Web site, 265
Book-O-Rama database
setting up, 243
tables, code to populate, 245
Course Web site, 909
CREATE TABLE command, 229–231
databases, 246–256
defined, 243–244
dropping, 264
joins, 254–255
records, 261, 264
rows, 252–253, 258
subqueries, 258–260
tables, 251–254, 261–263
two-table joins, 250–251
DDL (Data Definition Languages), 244
DML (Data Manipulation Language), 244
MySQL
aggregate functions, 256
join types, 254–255
RDBMS (relational database management systems), 243
resources, 909
strings, security, 371
sqlite3 extension, 7
SSL (Secure Sockets Layer), 344, 412-415, 889
commercial Web sites, 333
compression, 416
handshaking, 414–415
protocol stacks, 413–414
sending data, 415–416
testing, 899
stability, planning for, 376-377
Standard Generalized Markup Language. ***See*** **SGML**
starting sessions, 512
startup parameters, 900
stat() function, 446
statements
ALTER TABLE, 261–263
break statement, 56
continue statement, 56
DELETE, 264
DESCRIBE, 299
describe user;, 289
DROP DATABASE, 264
DROP TABLE, 264
echo statements, 26–27
else statements, 47
elseif statements, 48–49
exit statement, 56
EXPLAIN, 299–303
column values, 303
join types, 301–302
GRANT, 287, 297
if statements, 46–47
include() statement, 134
auto_append_file (php.ini file), 142–143
auto_prepend_file (php.ini file), 142–143
INSERT, 244
LOAD_DATA_INFILE, 311
MySQL case-sensitivity, 221
PHP statements, 19–20
prepared, 280–281
require() statement, 135–136
auto_append_file (php.ini file), 142–143

auto_prepend_file (php.ini file), 142-143
filename extensions, 136
PHP tags, 136
Web site templates, 137-140, 142
return statement, 154
SELECT, 246
LIMIT clause, 258
ORDER BY clause, 255
SHOW, 296-297
SHOW COLUMNS, 297
SHOW TABLES, 297
switch statements, 49-51
UPDATE, 261
static bindings, 185
static methods, implementing, 184
STD (column) function, 256
STDDEV (column) function, 256
storage engines, 312-313
InnoDB tables
foreign keys, 315-316
transactions, 314-315
MEMORY tables, 312
MERGE tables, 312
MyISAM, 312
store_account() function, 704
store_account_settings() function, 668-669
store_list() function, 723
store_new_post() function, 767
store_new_post.php files (Web forum application), 744
stored functions, declaring, 318-319
stored procedures, 316
control structures, 319-323
cursors, 319-323
declaring, 316-317
local variables, 319
stored functions, declaring, 318-319
storing
bookmarks, 571
data, 59. *See also* files
passwords, 295, 395
redundant data (Web databases), 212-213
secure storage, 417-419
session IDs, cookies, 511-512
strings, 114-116
addslashes() function, 114
stripslashes() function, 116
str_replace() function, 122, 787
strategies, commercial Web sites, 339
strcasecmp() function, 119
strchr() function, 121
strcmp() function, 119
strings
anchoring, 126-127
case functions, 113-114
column types, 239-241
comparing, 119
length of strings, testing, 120
strcasecmp() function, 119
strcmp() function, 119
strnatcmp() function, 119
concatenation operator, 26-27
data type (variables), 29
evaluating, 525-526
formatting, 110
case, changing, 113-114
conversion specifications, 112-113
HTML formatting, 110-111
ltrim() function, 110
nl2br() function, 110-111
printing, 110-113
rtrim() function, 110
storage, 114-116
trim() function, 110
trimming whitespace, 110
functions versus regular expression functions, 131
joining
implode() function, 117
join() function, 117
length, testing, 120
operators, 34
ordering
strcasecmp() function, 119
strcmp() function, 119
strnatcmp() function, 119

printing, 110-113
print() function, 110
printf() function, 111-112
sprintf() function, 111
securing, 371
specifying, 27
splitting
explode() function, 116-117
regular expressions, 130
strtok() function, 117
substr() function, 118-119
substrings
accessing, substr() function, 118-119
finding, 120-121, 129-130
numerical position of, finding, 121
replacing, 122-123, 130
tokens, 117
strip_tags() function, 417
stripslashes() function, 116, 272, 296
stristr() function, 121
strlen() function, 120
strnatcmp() function, 119
Stronghold Web site, 356
strpos() function, 121
strrchr() function, 121
strrpos() function, 122
strstr() function, 121, 597
strtok() function, 117
strtolower() function, 113
strtoupper() function, 113
Structured Query Language. *See* SQL
structures, directory, 542
style sheets, CSS, 859
subexpressions, 126
subqueries, 258-259
correlated, 260
operators, 259
row, 260
temporary tables, 260
subscribe() function, 717
subscribers
databases, 688
online newsletters, 717-718
substr() function, 118-119
substr_replace() function, 123
substrings
accessing, 118-119
finding, 120-121
numerical position, 121
regular expressions, 129-130
strchr() function, 121
stristr() function, 121
strpos() function, 121
strrchr() function, 121
strrpos() function, 122
strstr() function, 121
replacing, 122-123, 130
subtraction operator, 33
SUM(column) function, 256
Summary Web site, 330
SUPER privilege, 226
superglobal arrays, 24
superglobal variables, 32
switch statements, 49-51
switches
-h switch (mysql command), 221
-p switch (mysql command), 221
-u switch (mysql command), 221
syntactic sugar, 537
syntax, 552
ALTER TABLE statement, 262-263
control structures, 56
DESCRIBE statement, 299
errors, 552-553
extended, 257
heredoc, 27
highlighting, 530-531
system() function, 448
systems
capacity limits (commercial Web sites), 339
operating, 294
secure transactions, 412-413
user personalization, 570

T

t file mode, 63
t1lib, downloading, 484
tab control sequence (\t), 68

tables
aliases, 253-254
altering, 261-263
Book-O-Rama database, 245
Cartesian product, 250
columns, 209
atomic column values, 214-215
DESCRIBE statement, 299
types, 232-233
columns_priv, 288-293
creating in MySQL, 229-231
indexes, creating, 234-235
keywords, 231
table types, 229
viewing tables, 233-234
databases
backup, 305
optimization, 304
db, 288-291
dropping, 264
equi-joins, 251
grant, 288, 293
host, 288-291
InnoDB
foreign keys, 315-316
transactions, 314-315
joins, 250-255
keys, 209
creating, Web databases, 215
primary keys, 210
left joins, 252-253
MEMORY, 312
MERGE, 312
MyISAM, 312
rows, 209
returning, 258
unmatched, 252-253
values, 209
schemas, 210
scope fields, 290
tables_priv, 288-293
temporary, 260
two-table joins, 250-251
user, 288-290
tables_priv table, 288-293
tags
closing/opening (XML), 810
PHP tags, 18-19
ASP style, 19
require() statement, 136
SCRIPT style, 19
Short style, 19
XML style, 19
Tahuayo application (Amazon), 815-820
TCP (Transmission Control Protocol), 414
TCP/IP (Transmission Control Protocol/Internet Protocol), 386
security, 343
templates
PDF, creating, 776-777
RTF, creating, 776
Web sites, 137-142
temporary tables, subqueries, 260
terminating execution (scripts), 526
ternary operator, 39
testing
code, 548
GPG (Gnu Privacy Guard), 422-427
mod_auth_mysql module, 407
PHP
installations, 904-905
support, 897
regression, 377
SSL, 899
string length, 120
variable status, 45
text, 59-61
anti-aliasing, 489
baseline, 497
buttons, colors/fonts, 492
checking, 76
ciphertext (encryption), 351
closing, 69
deleting, 76
fitting onto buttons, 495-498
formats, 68-69
images
creating, 491-499
drawing or printing on, 487-489

limitations, 79
locking, 78-79
navigating, 76-77
opening, 61
file modes, 61-62
fopen() function, 62-64
FTP (File Transfer Protocol), 64-65
HTTP (Hypertext Transfer Protocol), 64-65
potential problems, 65-66
plain text (encryption), 351
positioning, 498-499
reading, 61, 71-72
feof() function, 73
fgetc() function, 75
fgetcsv() function, 73-74
fgets() function, 73
fgetss() function, 73
file() function, 74
fopen() function, 72
fpassthru() function, 74
fread() function, 75
readfile() function, 74
writing, 61, 499
file formats, 68-69
fputs() function, 67
fwrite() function, 67-68

TEXT type, 239-241
Thawte Web site, 348, 355
threaded discussion group application, 741-742, 763-764
article list, 747, 749
collapsing threads, 748, 752
displaying articles, 752-753
expanding threads, 748-751
individual articles, viewing, 760-762
new articles, adding, 762-769
plus symbols, 748
treenode class, 753-760
database design, 744-745, 747
extensions, 769
files, 744
posters, 744
solutions, 742-744
tree structure, 742-743
tree_node class, 743

threads, 741
collapsing, 748, 752
expanding, 748-753

threats to security
commercial Web sites, 342
DDoS (Distributed Denial of Service), 346
DoS (Denial of Service), 346-347
exposure of confidential data, 343-344
loss of data, 344-345
modification of data, 345-346
repudiation, 348-349
software errors, 347-348
crackers, 366
disgruntled employees, 366
hardware thieves, 366
infected machines, 366

three-dimensional arrays, 90-92
throw clause, 196
throwing exceptions, 193
tiers (applications), 218
TIFF library Web site, 778, 891

time and date
converting between PHP and MySQL formats, 476-477
in MySQL
date calculations, 478-480
DATE_FORMAT() function, 476-477
MySQL Web site, 481
UNIX_TIMESTAMP() function, 476-477
in PHP, 7, 469, 474
calendar functions, 480-481
checkdate() function, 474
date calculations, 477-478
date() function, 469-472
floor() function, 478
getdate() function, 473
microseconds, 480
mktime() function, 471-472
PHP Web site, 481

timeouts, avoiding, 467
timestamps, Unix, 471-472
tokens (strings), 117
top-down approach to security, 363
topbar.php file, 819, 825
totaling forms with operators, 41-42
touch() function, 447
traceroute command (UNIX), 344
tracking user's purchases (Shopping Cart application), 608
Transmission Control Protocol. *See* TCP
Transmission Control Protocol/Internet Protocol. *See* TCP/IP
transactions, 313
ACID compliance, 313
autocommit mode, 314
committed, 314
defined, 313
InnoDB tables, 314–315
rolled back, 314
secure transactions, 409–410
Internet, 411–412
screening user input, 417
Secure Sockets Layer (SSL), 413–416
secure storage, 417–419
systems, 412–413
user machines, 410–411
Web browsers, 410–411
transfer modes, FTP, 466
transferring data, database replication, 306-308
tree structure (Web forum application), 742-743
tree_node class, 743
treenode class (Web forum application), 753, 757-760
treenode_class.php files (Web forum application), 744
triggering errors, 564
trim() function, 110, 271
Tripwire Web site, 346
troubleshooting
errors, 66. *See also* errors
file uploads, 438–439
opening files, 65–66
TrueType fonts, 492
try blocks (exception handling), 193
tuples (tables), 209
tutorials
exception handling, 203
graphs, 508
two-dimensional arrays, 88-90
two-table joins, 250-251
type
conversion specification type codes, 112–113
hinting, 184
operator, 40

U

-u switch (mysql command), 221
uasort() function, 95
ucfirst() function, 113
ucwords() function, 114
uksort() function, 95
umask() function, 443
unary operators, 33
undefined functions, calling, 145-146
uninterruptible power supply (UPS), 359
union operator, 87
Unix
binary installations, 890–893
date() function, 471–472
Epoch (GMT), 471
httpd.conf file, 896–897
libpdf_php file, copying, 899
PHP, testing, 897
source installations, 891, 893–896
SSL, testing, 899
traceroute command, 344
UNIX_TIMESTAMP() function, 476-477
unlink() function, 76, 447
unmatched rows, 252-253
unnecessary OS applications, disabling, 388
unserialize() function, 527, 848
unset() function, 45
UNSIGNED keyword, 231
unsubscribe() function, 717
unsubscribing online newsletters, 717-718
update anomalies (Web databases)

UPDATE privilege, 225
UPDATE statement, 261
updating
avoiding, 213
FTP servers, 464–465
operating systems, 387–388
privileges, 293–294
records, 261
Shopping Cart application, 631–632
software, 378–379
upload.php files (MLM application), 691
uploading
files, 431–432
displaying, 437
HTML, 433
HTML forms, 431
PHP, writing, 434–438
security, 434, 438
troubleshooting, 438–439
FTP (File Transfer Protocol), 466
online newsletters, 724–731
UPS (uninterruptible power supply), 359
url_fns.php files (PHPBookmark application), 572
urlencode() function, 399, 455
USAGE privilege, 227
user authentication
input data, validating, 580
logging in, 584–587
logging out, 587–588
passwords
resetting, 591–595
setting, 588–591
registering, 577, 580–583
user declared variables, 28
user input, screening, 417
user interfaces, commercial Web sites, 333–334
user personalization
bookmarks
adding, 596–599
deleting, 600–602
displaying, 599
recommending, 571
storing, 571
defined, 569
passwords, 570
recommendations, 602–605
solutions, 570–572
system requirements, 570
usernames, 570
user privileges, database security, 295–296
user tables, 288–290
user views (Shopping Cart application), 609–610
user-defined exceptions, 196–197, 199
user-defined sorts, multidimensional arrays, 93–95
user_auth_fns.php files
MLM application, 691
PHPBookmark application, 572
Shopping Cart application, 612
Warm Mail application, 655
user_auth_fns.php library
check_auth_user() function, 665
usernames, 570
users
administrative user privileges, 226–227
authentication, 391, 401–406
access control, implementing, 392–395
basic authentication, 399
digest authentication, 400
encrypting passwords, 397–399
identifying users, 391–392
mod_auth_mysql module, 406–408
multiple pages, protecting, 399
storing passwords, 395
Web sites, 408
MySQL, setting up, 223
privileges, 223
global privileges, 224
GRANT command, 223–228
principle of least privilege, 223
REVOKE command, 227–228
types, 225–227
secure transactions, 410–411
setting up in MySQL, 223–229
Using mkdir() function, 443
usort() function, 94

utilities, myisamchk, 303
utilityfunctions.php file, 820, 825

V

valid email() function, 581
validating user authentication input data, 580
values
array elements, 82
assigning to variables, 28
atomic column values (databases), 214-215
columns, EXPLAIN statement, 303
default, database optimization, 305
null values, avoiding (Web databases), 216
returning, 94
assignment operator, 34-35
functions, max() function, 155-156
tables, 209
variables, 27, 30, 150-153, 539
arrays, 81-82
applying functions to elements, 103-104
associative arrays, 85
converting to scalar variables, 105-106
counting elements, 104
elements, 82
functions, passing by reference, 104
indexes, 82
loading from files, 98-101
multidimensional arrays, 88-95
navigating within an array, 102
numerically indexed arrays, accessing contents, 83-84
operators, 87-88
reordering, 96-98
set cardinality, 104
sorting, 92-93
browseNode, 824
debugging, 559-561
environment functions, 450
form variables, 23-27
functions, 44, 148
reinterpreting, 46
status, testing, 45
types, setting/testing, 44-45
global variables, 151
identifiers, 28
local stored procedures, 319
local variables, 151
mode, 824
page, 824
scalar variables, 81, 105-106
scope, 31-32
sessions, 510
deregistering, 513
implementing, 513
registering, 513
serializing, 514
Shopping Cart application, 623
superglobal, 32
types, 29
casts, 30
data types, 29
strength, 29-30
variable variables, 30
user declared variables, 28
values, assigning, 28
verifications
connections, 293
requests, 293
VeriSign, 355
Web site, 348
version control (code), 542-543
CVS (Concurrent Versions System), 543
multiple programmers, 543
repository, 542-543
view_post.php files (Web forum application), 744
viewing
databases in MySQL, 233-234
individual articles (Web forum application), 760-762
lists (online newsletters), 708-717
message headers (Warm Mail application), 680-681
tables in MySQL, 233-234

views, File Details, 445
visibility, controlling, 169-170
vote.html file, 500

W

w file mode, 63
w+ file mode, 63
W3C Web site, 808
Warm Mail application (email client)
accounts
creating, 668-669
deleting, 670
modifying, 670
selecting, 671-673
setting up, 666-668
databases, setting up, 655-656
email, deleting, 681-682
extensions, 686
files, 654-655
IMAP function library, 652-653
interface, 654
logging in, 663-666
logging out, 666
reading mail, 671, 681
mailbox contents, viewing, 674-676
messages, 677-681
selecting accounts, 671, 673
script architecture, 657, 662-663
sending mail
forwarding/replying, 684-685
new messages, 682-684
solutions
components, 652-653
overview, 654-655
WBMP (Wireless Bitmap), 485
Web application projects
content, 546
database security, 296
development environment, 544
documentation, 544-545
logic, 546
planning, 536-537
prototypes, 545-546
rewriting code, 537-538
running, 536-537
software engineering, 536
testing code, 548
version control, 542-543
writing maintainable code, 538
breaking up, 541-542
code standards, 538
commenting, 540
directory structures, 542
function libraries, 542
indenting, 540-541
naming conventions, 538-540
Web browsers
authentication, 351
secure transactions, 410-411
Web database architecture, 216
Web databases
architecture, 216-218, 268-271
designing, 211
anomalies, avoiding, 213
atomic column values, 214-215
keys, creating, 215
null values, avoiding, 216
questions, formulating, 215
real-world objects, modeling, 211-212
redundant data, avoiding, 212-213
table types, 216
update anomalies, avoiding, 213
querying, 271
adding data, 276-280
connections, setting up, 273
disconnecting from databases, 276
input data, 271-272
mysql_query() function, 274-275
prepared statements, 280-281
retrieving results, 275-276
selecting databases, 274
selecting in MySQL, 229
tables
column types, 232-241
creating, 229-231
indexes, creating, 234-235

keywords, 231
types, 229
viewing, 233-234
transaction process, 217
users, setting up, 228-229
viewing in MySQL, 233-234
Web development, 910
Web forum application, 741-742, 763-764
article list, 747-749
collapsing threads, 748-752
displaying articles, 752-753
expanding threads, 748-751
individual articles, viewing, 760-762
new articles, adding, 762-769
plus symbols, 748
treenode class, 753-760
database design, 744-747
extensions, 769
files, 744
posters, 744
solution components, 742-743
solution overview, 743-744
tree structure, 742-743
tree_node class, 743
Web forums
Phorum, 770
threads, 741
Web pages
authentication, 399
services, adding, 452, 454
Web resources for DOM, 884
Web servers
Apache. *See* Apache, Web server
authentication, 351
commands, 447-450
file upload, 434-438
Microsoft IIS, configuring, 381
secure storage, 417-419
Secure Web servers, 355-357
Web database architecture, 216
Web Services. *See also* SOAP
adding to Web pages, 452-454
defined, 811
interfaces (Amazon), 813-814
protocols
SOAP (Simple Object Access Protocol), 811-812
WSDL (Web Services Description Language), 812
Web Services Description Language (WSDL), 812
Web sites
Adobe, FDF, 789
Adobe Acrobat, 776
Ajax development, 885
AMANDA (Advanced Maryland Automated Network Disk Archiver), 358
Analog, 330
ANSI, 265
Apache, 891
Apache Software, 909
Apache Today, 909
Apache Week, 909
authentication documentation, 408
Boutell, 508
BUGTRAQ archives, 437
CGI specification, 450
Codewalkers, 909
CVS (Concurrent Versions System), 543, 549
Devshed, 508, 908
EPA, 359
Equifax Secure, 355
Evil Walrus, 909
Extreme Programming, 549
FastTemplate, 546
FDF, 789
Fedex, 335
FishCartSQL, 650
FPDF function library, 778
gd documentation, 508
Ghostscript, 775
GNU Privacy Guard, 419
Google, 811
HotScripts.com, 908
IMAP c client, 891

JPEG (Joint Photographic Experts Group), 485
JPEG library, 778, 891
Microsoft Word, 773
MySQL, 220, 309, 891, 909
  date and time functions, 481
  online manual, 241
Natural Order String Comparison, 119
Netscape
  cookie specification, 511
  SSL 3.0 Specification, 427
New York Times, 392
OpenSSL, 891
PDF, 775
PEAR (PHP Extension and Application Repository), 907
PECL, 907
Philip and Alex's Guide to Web Publishing, 910
PHP, 537, 891
  Application Tools, 909
  Base Library, 908
  calendar functions, 481
  Center, 908
  Classes Repository, 908
  Club, 908
  Developer, 909
  Developer's Network Unified Forums, 909
  Homepage, 908
  Hypertext Preprocessor, 106
  Kitchen, 909
  Magazine, 907
  online manual, 80
  Resource, 908-909
phpautodoc, 545
PHPBuilder.com, 908
PHPCommunity, 907
phpdoc, 544
PHPDocumentor, 544
PHPIndex.com, 908
PHPMyAdmin.Net, 908
PHPWizard.net, 908
php|architect, 907
PNG (Portable Network Graphics), 485
PNG library, 891
Postnuke, 909
PX-PHP Code Exchange, 908
RFC Editor, 451, 468
SearchDatabase.com, 909
Slashdot, 392, 741
SourceForge, 545, 909
SQL Course, 909
Stronghold, 356
Summary, 330
templates, 137-142
Thawte, 348, 355
TIFF library, 778, 891
Tripwire, 346
UPS, 335
VeriSign, 348, 355
W3C, 808
Webalizer, 330
WeberDev.com, 908
WebMonkey.com, 908
Zend, 131, 508
Zend.Com, 907
zlib library, 891
Webalizer Web site, 330
WeberDev.com Web site, 908
WebMonkey.com Web site, 908
WHERE clause, 248
  comparison operators, 248-249
  join condition, 250
while loops, 53-54
whitespace, 20, 110
wildcard character (%), 293
Windows
  Apache, 902
  MySQL, 900-901
  PHP, 903-904
    Apache configurations, 904
    testing, 904-905
  support, 7, 900
Wireless Bitmap (WBMP), 485
word processor formats, 773

writing
code for classes, 175–183
files, 61, 418
file formats, 68–69
fputs() function, 67
fwrite() function, 67–68
maintainable code, 538
breaking up, 541–542
code standards, 538
commenting, 540
directory structures, 542
function libraries, 542
indenting, 540–541
naming conventions, 538–540
PHP file uploads, 434–438
runtime errors, 555
Text buttons, 499
WSDL (Web Services Description Language), 812

X-Y

x file mode, 63
x+ file mode, 63
XHTML (Extensible Hypertext Markup Language), 858
XML (Extensible Markup Language), 807, 860
Amazon connections, 807–808
defined, 808–810
DTD (Document Type Definition), 810
example, 808
namespaces, 811
parsing (Amazon), 814
REST/XML (Amazon), 838–839, 844
root elements, 811
SGML (Standard Generalized Markup Language), 808
styles, 19
tags (closing and opening), 810
XMLHTTPRequest object, 860, 862
XSLT (XSL Transformations), 860
XSS (Cross Site Scripting) attacks, 365

Z

Zend engines
Optimizers, 547
PHP 5.3, improvements for, 7
Web site, 131, 508, 907
zlib library Web site, 891