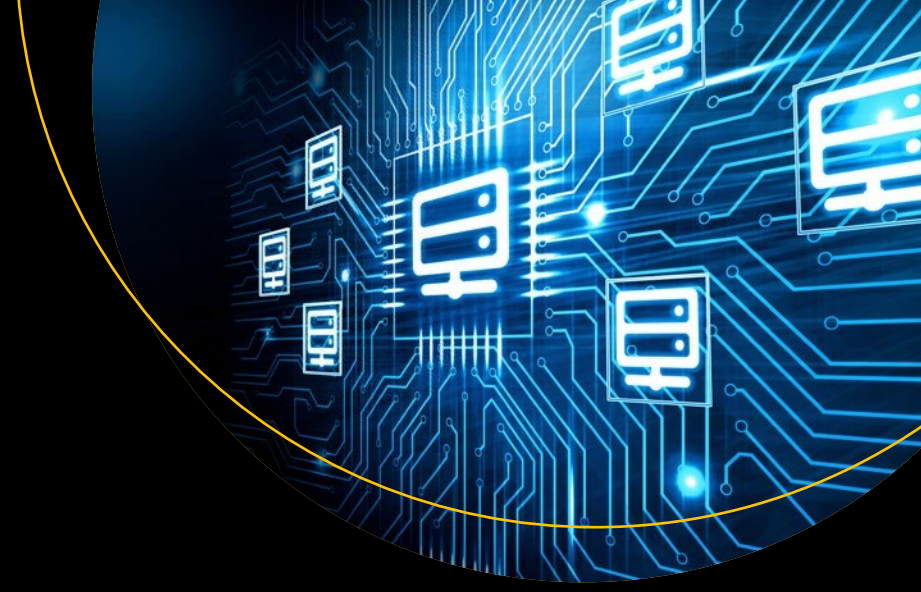

Database-Driven Web Development

Learn to Operate at a Professional Level with PERL and MySQL

—

Second Edition

—

Thomas Valentine

Apress®

Database-Driven Web Development

Learn to Operate at a Professional Level with PERL and MySQL

Second Edition

Thomas Valentine

Apress®

Database-Driven Web Development: Learn to Operate at a Professional Level with PERL and MySQL, Second Edition

Thomas Valentine
Selkirk, MB, Canada

ISBN-13 (pbk): 978-1-4842-9791-9 ISBN-13 (electronic): 978-1-4842-9792-6
https://doi.org/10.1007/978-1-4842-9792-6

Copyright © 2023 by Thomas Valentine

This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

Trademarked names, logos, and images may appear in this book. Rather than use a trademark symbol with every occurrence of a trademarked name, logo, or image we use the names, logos, and images only in an editorial fashion and to the benefit of the trademark owner, with no intention of infringement of the trademark.

The use in this publication of trade names, trademarks, service marks, and similar terms, even if they are not identified as such, is not to be taken as an expression of opinion as to whether or not they are subject to proprietary rights.

While the advice and information in this book are believed to be true and accurate at the date of publication, neither the authors nor the editors nor the publisher can accept any legal responsibility for any errors or omissions that may be made. The publisher makes no warranty, express or implied, with respect to the material contained herein.

Managing Director, Apress Media LLC: Welmoed Spahr
Acquisitions Editor: Divya Modi
Development Editor: James Markham
Editorial Assistant: Divya Modi

Cover designed by eStudioCalamar

Cover image by Freepik (www.freepik.com)

Distributed to the book trade worldwide by Springer Science+Business Media New York, 1 New York Plaza, Suite 4600, New York, NY 10004-1562, USA. Phone 1-800-SPRINGER, fax (201) 348-4505, e-mail orders-ny@springer-sbm.com, or visit www.springeronline.com. Apress Media, LLC is a California LLC and the sole member (owner) is Springer Science + Business Media Finance Inc (SSBM Finance Inc). SSBM Finance Inc is a **Delaware** corporation.

For information on translations, please e-mail booktranslations@springernature.com; for reprint, paperback, or audio rights, please e-mail bookpermissions@springernature.com.

Apress titles may be purchased in bulk for academic, corporate, or promotional use. eBook versions and licenses are also available for most titles. For more information, reference our Print and eBook Bulk Sales web page at http://www.apress.com/bulk-sales.

Any source code or other supplementary material referenced by the author in this book is available to readers on GitHub (github.com/apress). For more detailed information, please visit https://www.apress.com/gp/services/source-code.

Paper in this product is recyclable

Table of Contents

About the Author

Thomas Valentine has 20 years of experience as both a professional web developer and writer. He is a LAMP, Perl, PHP, and MySQL web developer, programmer, and expert. He writes for various magazines and websites and has been a technical consultant for large-scale, database-driven websites such as FedEx.ca and Texas Instruments (ti.com).

About the Technical Reviewer

Kenneth Fukizi is a software engineer, architect, and consultant with experience in coding on different platforms internationally. Prior to dedicated software development, he worked as a lecturer and was then head of IT at different organizations. He has domain experience working with technology for companies mainly in the financial sector. When he's not working, he likes reading up on emerging technologies and strives to be an active member of the software community.

Kenneth currently leads a community of African developers, through a startup company called AfrikanCoder.

CHAPTER 1

Database-Driven Web Development Fundamentals

The database was first created by software engineers at North American Rockwell in the mid-1970s. They were previously using static files to catalogue the work they were producing. They began to notice that efforts were being duplicated, as a project such as the designing of an Apollo-era rocket generates hundreds of thousands of different parts. Each part was given a static file. Keeping track of these hundreds of thousands of parts was a nightmare before the relational database was put into practice. With it, they were able to find the duplicated efforts and eliminate them.

We'll begin this chapter by reviewing the fundamentals of database-driven websites before moving on to the pros and cons of web hosting, delving into the options that are available. You'll see how the modern database fits into the process of using three different servers (the Perl, Apache, and MySQL servers) to generate a web page. These three servers are what we'll be learning to use together in this book. Read on, it's going to be fun!

The Evolution of the Database-Driven Approach

During the infancy of the Internet, databases weren't used with websites. All pages were static HTML pages. Today, they'd be pretty boring sites to surf, as we've all become accustomed to the modern database-driven website.

The world caught on to the idea that databases were the more efficient way to go, eventually leading to our database-driven world. All of our worldly services from banking to social networks now use a database of some flavor.

© Thomas Valentine 2023
T. Valentine, *Database-Driven Web Development*, https://doi.org/10.1007/978-1-4842-9792-6_1

Of particular note is the Internet. Databases are widely used to serve the pages that we all look at every day. They integrate seamlessly into a web page and have changed the world we live in. Databases are here to stay, so it is of paramount importance that you know a thing or two about them.

Since the database runs on a server, there has to be a way to access these databases. There are many different ways to do this, as there are many different languages used in modern computing. We'll be concentrating on the interactions between the MySQL database and the Perl scripting language.

Perl has great capabilities when it comes to interacting with databases. It is a powerful language that has been around for decades. It is a mature technology and is best suited to working with text. Perl stands for Practical Exporting and Reporting Language and was first used to manipulate text within common static text files. Since the languages we use in web development are textually based, Perl is the way to go when dealing with large textual documents (e.g., HTML markup).

Coupled with a database, Perl is a powerful and capable engine to power the modern website. Perl usually runs on the same server or at least the same network that the database runs on. This allows for fast and efficient interactions comprised of large blocks of textual data, such as the HTML code used in every page on the Internet today.

Databases are evolving to fill the need for newer, faster, and more powerful websites. It is common for one website to use many databases to serve its web pages. This scalable ability is the cornerstone of today's website. As the site grows, more databases are required to run the website. Parceling your website data from one or two databases into, say, a dozen allows you to access these databases from different parts of the website in a timely fashion under a heavy user load.

How to Spot a Database-Driven Website

Most people just take it for granted that their web page will just load with the information they expect to be there. As a developer, you need to know how to spot a database-driven website. There are a few ways you can do this. We'll examine the different ways to spot one and to know, in general, what database is being used, in what way, and why.

The Website Address (URL)

The first, and perhaps easiest, method to use to spot a database-driven website is the URL in the address bar of the browser. If you were to look at a web page that is a static HTML page, you would see an address that ends in a file name with the .html file extension. Examine the following example:

```
http://www.domain.com/index.html
```

You can see the constituent parts of the web address clearly: the address starts with the standard `http://www.domain.com` and ends with the file named index.html.

Most websites use index.html as the default starting point of their website. It is the main page that the Apache web server sees as the base file of any given website, although it is entirely possible to use a different file name as the starting point of your website. Configuring your web server is tackled in a coming chapter of this book.

The .html extension tells you that the page is a static HTML page. There is nothing to the right of the address bar, so it is a good bet that a database isn't being used at all on that website.

When you use Perl to present a database-driven page, the file extension is either .cgi or .pl. If there's a database being used, a question mark will be displayed after the file name extension. What follows the question mark is unique to each website. Examine the following URL:

```
http://www.domain.com/thisfile.cgi?id=1234
```

You can see the basic constituent parts of a URL, including a Perl script named thisfile with the extension .cgi. Then comes a question mark followed by the "id=1234" statement. This is a good indicator that a database is being used. There may be many different statements after the question mark, each separated with an ampersand (&), as follows:

```
http://www.domain.com/thisfile.cgi?id=1234&id2=5678
```

You can see now that the URL shows two different statements called parameters. These parameters show that two sets of data within the parameters are being passed to the server. While it is possible that a database isn't being used with these parameters, it is very uncommon to see more than one parameter that is being used without a database being in the loop somewhere.

Differing Content Areas

You can also see that a database is being used by clicking a few links and noticing that one page looks very much like the previous, but with different content being displayed in the main content areas. This is because a database-driven site commonly uses templates to present their data. By template I mean a set of markup (such as HTML) that is used to present the top and bottom and sides of the page, for example. The only part of the page that changes is, say, the center of the page. This is the database-generated content.

Another easy way to spot a database-driven site is by paying attention to the advertising banners and buttons. If they change with every page or even if you refresh the page, the advertisement changes. This can't be done without using a database on some level.

When working with Perl and a database, you'll notice that a dialog box will be presented by the browser asking if you'd like to "repost form data." This is because you've just finished posting data to a database and have clicked the "Back" button on your browser. The Perl script that posted the information is being run again, making the same database entry you just made again.

On the Perl scripts that don't interact with a database, you won't see the "repost form data" dialog box. As we progress through the chapters of this book, you'll notice that this happens quite often. One of the only ways around this is to provide, within the Perl script, an easy-to-recognize link that takes you back to the previous page without reposting the form data and duplicating the database entry.

Reposting the form data can be detrimental to your site, as it duplicates the data within your database. While the Perl script being rerun may be something as simple as counting a page hit, it may be something as complicated as uploading a dozen images to your server – you'll see that the 12 images are duplicated on your server.

Adding Search Features

One surefire way of detecting that a website is using a database is if the site offers up search features. While it is possible to search a website without using a database, this is costly in terms of server resources and isn't done very often. Searching through dozens or even hundreds of static HTML files would take an inordinate amount of time and server resources.

Searching for content via a database, however, is exactly what the database was created for. MySQL is great for doing this, as it is a relational database. We'll discover more on the term "relational database" in a different section of this chapter.

Search features are a common sight on most database-driven websites. MySQL, along with Perl, can incorporate some very powerful search features into your site. The project we'll be working on in later chapters of this book includes a search feature that finds users and posts based on a word or phrase entered into a search field by you, the user.

Username and Password Considerations

One definite and easy way to spot that you're using a database is if you have to sign in with a username and password. While there is a way to use a username and password without using a database, this method doesn't involve a true database.

When supplying a username and password, the script tells the database to first look for the username. Once the username is found, the password is retrieved. Once retrieved, if the password on file mirrors the password provided, the user is logged in. From there on in, a database is most commonly used to present the page to the user, most of the time based on some aspect of the user's personal information.

Protect a Directory with .htaccess Files

The only way to use a username and password without using a traditional database is with an older method of recognizing a user. This older method is by using an .htaccess file. Within this file is a name/value pair that contains the username and password. An .htaccess file is simply a text file with commands and data within it.

Once the username and password are verified, the directory directly below where the .htaccess file resides on your server will be able to be accessed by that particular user. This check is made by the Apache web server, not Perl nor a database.

While this works fine in some cases, you are limited to about 100 users. After the 100-user limit has been reached, it will take an inordinate amount of time to verify the username and password. This is because the supplied username and password must be verified by comparing both pieces of data to the 100 different name/value pairs within the simple .htaccess file.

While the operation of simple user verification is the base operation of an .htaccess file, it should be known that the Apache web server allows you to include special commands that limit or change the conditions of each page covered by that .htaccess file.

If you were a web master, in installing and configuring the Apache web server, you will have to make a certain set of changes in a configuration file known as httpd.conf. Within this file is all of the information the Apache web server requires to successfully serve a web page from that server. Most of these commands, such as for allowing user access, can be put into the .htaccess file. This handy functionality is why the .htaccess file is still being used on the Internet today.

Where and When to Use a Database

You would normally use a database if you have a need to present a certain web page to your users that has as its main point of interest content that is changeable and fluid – a page that differs according to each user such as a user profile page, for example.

Using a database gives you the flexibility to present to your users dynamic content. By dynamic, it is meant data that changes, usually according to user action. The user that generated the data a user is looking at may not be the current user – since there is the potential to have thousands or even millions of users on one website, looking at data generated by a different user is commonplace.

If you see that you have a need to employ features such as displaying individual user information, search features, or are working with HTML forms in any number, you should be using a database.

While working with a database on the Internet, you'll be able to easily see where and how a database is being used within each site that you visit. You'll get an idea of the flexibility and power that a database-driven website can accomplish. Through the use of simple markup code such as HTML, you'll see that a database can extend the creative reach that is possible.

Web Hosting Fundamentals

So you have an idea for a website and wish to pursue it and make it real. One of the first things you'll have to consider is who your web hosting company is going to be.

In a nutshell, there are three options for web hosting – the shared plan, the VPS (Virtual Private Server) plan, and the dedicated server plan – and each has its own pros and cons. We will look at each one in the remainder of the chapter.

The Shared Server

Shared server plans are just what you think they are – you share your server with many different websites. This plan is great for developing your website in its infant stage. You would normally start out with the shared plan in order to develop your website into a finished product. It is the cheapest option that still gives a good amount of functionality, although it usually isn't the option to stay with if you plan on having a large amount of users. It should be noted that there are two ways your shared server will be addressed: name based or IP based. A name-based server address is the way we humans usually understand `www.domain.com`, for example. The other naming convention, IP based, is a numeric form of addressing that uses an "octet" of numbers. IP stands for Internet Protocol, that is, four groups of three numbers separated by a period:

127.34.56.124

The IP address as seen in the preceding example is a valid IP address. It contains four numbers separated by dots. These numbers are the base addressing system on the Internet. In order to match these numbers with words that us humans understand, we would have to use a DNS server. DNS stands for Domain Name Server.

Shared hosting plans usually have all of the functionality you'll need to get your website from nothing to a fully functioning Internet offering. However, there are a few things you have to know of before you sign the contract and pay the company for space on one of its servers.

Web hosting companies only allow a certain amount of resources for your site when using a shared server plan. Resources are things like disk space, a percentage of processor usage, and a limit on bandwidth that can be used, usually measured within a 30-day period.

Some limit the amount of MySQL databases you are able to create and use. You should ask your web host if your MySQL databases are on the same server you're using or accessed via the local network on a server dedicated to running MySQL databases. This is important because in order to use the database, you'll have to know whether to use a web address as the location if the database is on a dedicated server or "localhost" if the database resides on the same server the rest of your website is stored on. This addressing will be reflected in every Perl script that uses the database, so knowing this is important.

Since the primary scripting language of this book is Perl, it is important that you ask the web hosting company if they allow you to install more Perl modules that don't come with the standard Perl distribution, such as DBI.pm. In later chapters of this book, you will also need to use size.pm and resize.pm. size.pm and resize.pm are Perl modules that are used to return the size of an image (size.pm) and resize the image (resize.pm). More on those two Perl modules in a later chapter.

You should also ask your web host where the Perl executables are located. This piece of information is the one line of code at the very top of every Perl script, known as the "shebang." If the script can't find the Perl executables, the script won't run. The most common location of Perl is as follows:

```
#!/usr/bin/perl
```

I have encountered web hosting companies that list every recent version of Perl as the shebang, giving you the option of what version to use. Some companies use an unusual shebang. Look to the FAQ or support section of the web host's website for the location of Perl.

I have run into a web hosting company that doesn't offer Perl as a scripting language. Their excuse was that Perl is not scalable, which couldn't be further from the truth. Perl is infinitely scalable – you just have to know how to do it. Between the use of subdomains and files, you're able to spread an extremely complicated site over an entire array of servers.

The VPS Server

The Virtual Private Server (VPS) plan is a step up from the shared plan in terms of server ability and customization. A VPS plan gives a guaranteed percentage of processor use, memory (RAM), and hard drive space. It closely emulates a dedicated server without the higher cost of a dedicated sever. Most plans also allocate more bandwidth per 30-day period than you would get with a shared server plan.

A VPS plan is one that emulates the functionality of a dedicated server. They're pricier than a shared plan, but the extra expense is usually worth the cost. Use a VPS server when you're in the final beta testing stage of your site's life cycle. Again, be careful about what is and isn't allowed on your VPS server. The VPS server is a good solution for low-traffic, low-bandwidth sites that require the functionality of a dedicated server without the added cost.

The Dedicated Server

Dedicated plans are the costliest of the three common plan types and are also the most feature-rich, since it is usually only the one website on a single server. Some web hosting companies allow you to have more than one website on each dedicated server.

You would normally use a dedicated server after your site is complete and is already taking on users. They're the most expensive plan that a web hosting company provides, as yours is the only site using that server. Use a dedicated server when your traffic starts to bog down or you run low on hard drive space on your shared or VPS server.

Dedicated servers usually are offered on a sliding scale of functionality and capability. If you're going to lease a dedicated server, make sure it has hard drive redundancy, such as RAID 1. RAID stands for Redundant Array of Independent Disks.

The two most common flavors of RAID are RAID 1 and RAID 5. RAID 1 involves two separate hard drives that mirror each other's data. Only two hard drives are involved, with the idea being that if one hard drive fails, the data is still in a usable form on the other mirrored disk. This is the most common form of RAID.

RAID 5 is known as disk striping. Up to seven disks may be involved in storing your data. If one disk is lost, enough information remains on the other functioning hard drives to rebuild the data on the hard drive that had failed and was replaced.

Email Options

Another basic service you should query your web host about is Linux's sendmail feature. While most people are familiar with Windows, it is rarely used on the Internet as a server. This is where Linux comes in. Linux is the prevalent server used to serve the web pages you see every day.

In order to send email via a script, you'll need sendmail to do it with. sendmail is a very basic email server that has been in use for decades. With it, you're able to send email on the fly according to scripted action.

You'll also want to ask about your online email client. There are many email clients to choose from, and all offer pretty much the same functions. What you're looking for, though, is a client that is stable and time tested. The most common email client is Horde, and it works very well. It is one of the oldest and most developed online email clients available. It is a mature technology. With it, you'll be able to retrieve and send