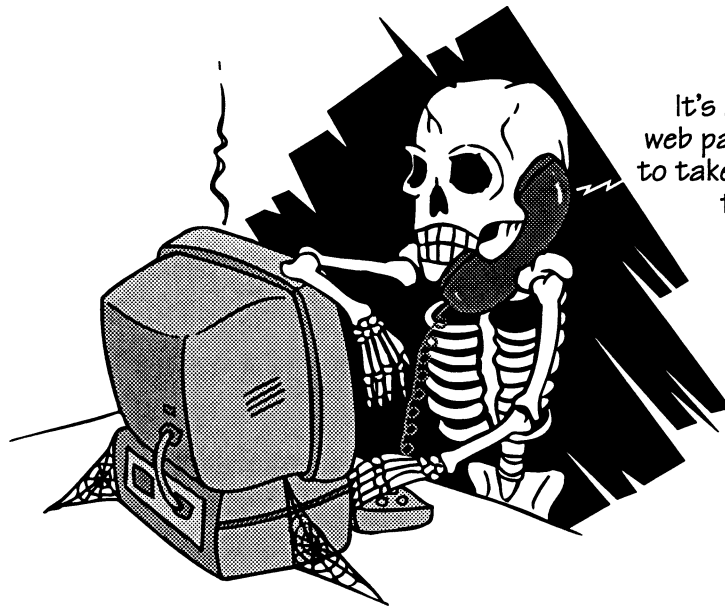


DESIGNING AND MAINTAINING WEB PAGES: A GUIDE FOR NONPROFIT ORGANIZATIONS WITH A FOCUS ON TRANSPORTATION SAFETY



*It's about your
web page...it seems
to take quite a while
to load.*

JOHN S. MILLER
Research Scientist

E. J. DEASY
Video Production Specialist



**DESIGNING AND MAINTAINING WEB PAGES:
A GUIDE FOR NONPROFIT ORGANIZATIONS
WITH A FOCUS
ON TRANSPORTATION SAFETY**

**John S. Miller
Research Scientist**

**E. J. Deasy
Video Publication Specialist**

October 1997
VTRC 98-R22

ACKNOWLEDGMENTS

We thank Nancy Rodrigues of DRIVE SMART Virginia for her interest in Web page development and David Mosley of the Department of Motor Vehicles for the opportunity to pursue this research effort. Peter Massarelli of VTRC and Tom Wise of the University of Virginia provided useful suggestions on how to approach developing a Web page, including server options and database information. Brian Prillaman of VTRC diligently converted the 1995 alcohol data from paper to electronic format and verified the tabulations. Robert Dennis of Virginia Networks figured out the naming convention necessary for the database queries to function on an NT platform.

FOREWORD: HOW TO AVOID READING THIS ENTIRE DOCUMENT

Your time is limited. The fact that you picked this up and read this far places you in a select group of people; an even smaller number will read it from cover to cover. Assuming your reason for opening this in the first place is that you want your nonprofit transportation safety organization to have a presence on the World Wide Web, here's an easy way to figure out which chapters you should examine.

If someone else will be developing your Web page and you just want to know where to place your Web site, then simply read Chapter 1, "Selecting an Internet Service Provider."

If you will be developing your own Web page but just want to keep everything simple for starters, then read Chapters 1 through 4.

If you want to survey your readers, then also read Chapter 5, "Developing Simple Surveys."

If you want to include a database query, then also read Chapter 6, "Posting Data."

If you've learned HTML and are worrying whether your time was wasted because you're hearing so much about Web site development software, then glance at the subsection "Using HTML to Automate Database Queries" in Chapter 6.

To view the DRIVE SMART Virginia Web site that accompanies this manual, go to the Uniform Resource Locator (URL) of <http://www.vanet2.com/vtrc/Default.htm>. This page was designed in July 1997 and is subject to change.

TABLE OF CONTENTS

ACKNOWLEDGMENTS	iii
FOREWORD: HOW TO AVOID READING THIS ENTIRE GUIDE	v
1. SELECTING AN INTERNET SERVICE PROVIDER	1
Where to Learn About the Internet.	1
How to Get Started Showing Your Web Page	1
How to Obtain Information on Internet Service Providers From the Internet ...	6
2. DESIGNING YOUR OWN WEB PAGE	7
Theory	7
Writing in HTML Directly	9
Learning by Example From Other Internet Sites	11
3. BUILDING YOUR WEB PAGE	13
Using the Template	13
Using Tables to Control How Your Text Appears	15
Inserting Text and Hyperlinks	16
Inserting Graphics	17
How to Do "Banners"	18
Citing Your Data Sources	19
Checking Your Web Site	20
4. PUBLISHING TO THE WEB	21
5. DEVELOPING SIMPLE SURVEYS	25
6. POSTING DATA	27
Posting Static Data	27
Posting Live Data	27
Reminders When Posting Data	27
Steps for Establishing a Database Query	29
Using HTML to Automate Database Entries	36
Establishing a Personal Web Server	37
7. CONCLUSION	39



1. SELECTING AN INTERNET SERVICE PROVIDER

Where to Learn About the Internet

If you need background information about what exactly the Web is, there are numerous sites that can help you. Three struck us as particularly useful, although you can find many more by using a search engine and looking under the heading “Computer,” “Internet,” or “Web.” First, *Zen and the Art of the Internet*, published back in 1992 and found at http://www.cs.indiana.edu/docproject/zen/zen-1.0_toc.html, gives a glossary of terms often seen but not defined. For example, the subsection under “domains” explains the significance of the last word following the last period in an address: *.com* means company, *.org* means organization, and so on. Second, Thomas Boutell’s *World Wide Web FAQ* at <http://www.boutell.com/faq/#intro> is also a good introduction to the Web. Third, the *World Wide Web for the Clueless* at <http://www.mit.edu/people/rei/wwwintro.html> gives a very short summary of the notion of Web pages and hypertext.

How to Get Started Showing Your Web Page

If you’re designing your own Web page, your first question will probably be: How do I create a Web page? To answer this, you can begin with Chapter 2. If someone else is designing a Web page for you, this chapter will help you decide how to have it published. You make your Web page available to others by renting space on a *server*.

Intuitively, think of a *server* as a giant billboard next to the roadway. You can rent space on the billboard for a certain amount of time, and you have many options for how you display your message in terms of size, type (color, black and white, photograph, graphic, or text), and upkeep (you can put up your own design or, for

more money, you can have an advertising company design the display for you). You can keep the same display posted year after year or you can make changes periodically.

Before making the decision about which billboard to select, you would want to find out your needs or limitations in terms of cost, color, anticipated audience, and so forth. Hence, before you select an Internet Service Provider, or ISP, from whom you may rent server space, you need to consider a few issues.

1. *Who will be designing your Web page?* You can do it yourself, or you can hire a contractor to do it for you. Your Web page could be designed by someone in your organization who enjoys working with computers and graphic design. In that case, you might only need to rent server space.

2. *Who will be maintaining your Web page?* Probably the ISP will allow you to make changes as often as necessary if you make them yourself. On the other hand, you could have the ISP make changes for you, which might entail some fees. If you plan on maintaining a page designed by someone else, make sure the designer explains any special knowledge you'll need to alter the page. Seemingly simple items, such as changing a few lines of text, can alter the page's appearance depending on the method of the designer.

3. *What kinds of maintenance will you need for your Web page?* Only updating text, such as the upcoming dates of key meetings or names of contacts in your organization, is a lot simpler than modifying graphics, photographs, or a database used by your Web page. Similarly, major design changes require additional work.

4. *Will your Web page require any special needs?* For example, the software used to develop our Web page required special extensions. Although installing them didn't pose problems for some ISPs, others either charged extra for using them or didn't have them available.

5. *Will your ISP work with you?* In our case, we were fortunate that we chose an ISP who was willing to make modifications to the server, including learning a new platform, to allow certain features on our home page to function. Other features, such as the counter, were less important to us, so we didn't base our server selection on the performance of the counter.

6. *How much space will you need?* Fortunately, space is fairly cheap. Unlike with billboards, server space is more plentiful. You should still verify the vendor's pricing scheme, however. Each ISP's pricing scheme may be different. In some cases, you might find a great deal for a small amount of space but pay substantially more to double the space. On the other hand, some ISPs charge a relatively flat rate regardless of how much space you need. For small pages, 5 megabytes (abbreviated as "M" or "Meg") of memory should be plenty. For example, the DRIVE SMART Virginia site, including databases, requires about 3 M of memory.

7. *What types of monitoring for your site will you need?* You may be satisfied simply to have your site posted on the Web. On the other hand, you may wish to obtain information about how often your site is accessed, by whom it is accessed, and so on. Some ISPs include an excellent monitoring program as part of the package when you rent server space.

8. *Does the geographical location of your ISP matter?* An ISP that's a long-distance phone call away may seem unattractive because of long-distance charges. Alternatively, the ISP may offer a toll free number or you may decide that the long-distance costs are relatively small compared to the benefits a particular ISP can offer.

9. *Where do you find an ISP?* You can find ISPs on-line; even the general search term "ISP" reveals citations. Table 1 shows a few. The organizations shown are given as examples only. The fact that an ISP is named or not named is not a judgment against or an endorsement for the ISP. Contacts are given in parentheses.

TABLE 1. SOME INTERNET SERVICE PROVIDERS (ISPs)

Name	Address	Telephone	Fax	Email or Web Site
Cyber Services, Inc.	8027 Leesburg Pike Suite 317 Vienna, VA 22182	(703) 749-9590 (800) 372-2644	(703) 749-9598	http://cyber.cybersrv.com/newpac.html#host,info@cs.com
Erols (Steve Wright)	7921 Woodruff Court Springfield, VA 22151	1-800-376-5772, x2286	(703) 321-8316 (Springfield HQ)	http://www.erols.com/erols/index/price.htm#tone
Global Connect, Inc.	12388 Warwick Blvd. Suite 312 Newport News, VA 23606	(757) 595 3258	(757) 595-6716	http://www.gc.net/sales/Web.html info@gc.net

InterSerF (Larry Burgess)	11901 Main St. Fredericksburg, VA 22408	(540) 371-4195	(540) 371-4197	lburgess@interserf.net
Richmond Net (Wayne Estrada)	1100 Welborne Dr. Suite H-2 P.O. Box 70417 Richmond, VA 23255	(804) 740-6100	(804) 740-2446	http://www.Richmond.net
Virginia Networks (Robert Dennis)	P.O. Box 70594 Richmond, VA 23255-0594	(804) 747-3500 1-800-689- 7571		http://www.vanet.com, admin@vanet.com
Virginia Webs (Gary Haworth)	Route 4, #81 Rustburg, VA 24588	804-821-3246 804-821-4009 804-544-9140	(804) 821-4009	http://www.virginiaWebs. com

10. *How do you compare pricing structures?* Each company has its own method for computing costs, and some may offer discounts for nonprofit organizations. Therefore, consider these questions when you make your price comparisons and all you want is server space:

- Is there a one-time *setup* fee?
- Is this requirement waived for *nonprofit* organizations?
- What are the monthly or yearly *rental* charges?
- Are there any *server space* limitations?
- Are there any *special discounts* available?
- Can you make periodic *updates* to your Web site?

If you do need special services, such as knowing about the audience that examines your site or having your Web page periodically updated by someone else, then make sure these are specified in the agreement you have with the ISP.

Table 2 shows some pricing structures in use as of May-July 1997. They don't include fees for designing or periodically updating Web pages. Where only a yearly fee is given, the monthly rate may be higher than the yearly fee divided by 12.

TABLE 2. WEB PRICING STRUCTURES FOR VARIOUS COMPANIES AS OF JULY 1997

Setup fee	Rates	Limitations or Bonuses	FrontPage Extensions?
\$150 (waived for nonprofits)	\$45/mo	Up to 10 M of space	Yes
\$50	\$380/yr	Up to 5 M of space Half price for nonprofits	Yes, (not done yet but are doing for another customer)
\$0	\$10/mo (1M) \$15/mo (2M) \$30/mo (5M) \$45/mo (10M) \$70/mo (20M) \$80/mo (30M)	Price dependent on space	Yes, but not done yet
\$0	\$55-\$95/mo	Depends on speed and other service options	No (possibly in a couple of months)
\$0	\$20/mo \$200 annually	Includes automated report	Yes
\$100 (if by mo) \$0 (if by yr)	\$30/mo \$324/yr	Up to 25 M of space Includes 20 email boxes	Yes
\$0	\$295 (10 M) \$395 (15 M)	Cheaper options without FrontPage usage	At a higher cost: \$780/yr w/\$125 setup fee; then price drops to \$480/yr

The variation in price can be large. At the time the table was created, for example, costs for the first year of service could range from \$120 to more than \$700, depending on the configuration of the server. Because companies' pricing changes rapidly, we didn't attach names to the prices. We advise you to contact several ISPs to determine your options.

These fees generally reflect a case where you will be using a server that is also used by other companies. For lower traffic sites, this should be sufficient. If you have a large number of people visit your site or if you expect to do a large amount of data transfer from your site to various users, then you may need a dedicated server, which will require a larger investment.

How to Obtain Information on Internet Service Providers From the Internet

Try <http://union.ncsa.uiuc.edu/HyperNews/get/www/leasing.html> for site information nationwide, especially with a focus on lower cost sites. At the time of this writing, Alex Chapman, the author of the site, was updating the list periodically. You can also simply do a search on "Internet Service Provider," or "ISP," for more detailed information.

2. DESIGNING YOUR OWN WEB PAGE


Designing—and maintaining—your Web page can take as much or as little time as you wish to put into it. Most ISPs will create and/or maintain a Web page for a fee. We have seen fees from \$20 to \$80 per hour; one ISP claimed they could develop a decent site in about 8 hours or less. Of course, these fees vary depending on what you want. You may also be able to establish a contract with a provider to develop a site for a fixed amount. If you decide to do this, we recommend you obtain several quotes—just as you would when making any other type of significant purchase.





Alternatively, you may wish to develop your own Web page. We wrote this chapter for those of you who do.


Theory


A variety of sources give some good basic concepts regarding Web page development, regardless of the software you plan to use. They range from books, short courses, and Internet sites. By examining several of these sources, you can have a rough feeling for some of the simple, but essential, tenets of Web page development. A few salient points are worth noting:


 **Define your audience.** A site maintained by Sun Microsystems has a subsection entitled *Ten Steps to Developing a World Wide Web Server* at http://www.sun.com/products-n-olutions/hw/servers/netra/netra_i/10steps.html. Although some steps pertain to establishing the server itself, the first step—*Define the Audience*—is critical. In the case of transportation safety, for example, will your audience be high school students, transportation data specialists, civic associations, or some other group? The answer will affect the terminology you use in your Web page.

 **Make sure each page is understandable if read alone, and give the source of your information.** The *Yale C/AIM Web Style Guide* at <http://info.med.yale.edu/caim/manual/contents.html> emphasizes a key difference between the Web and the printed medium: users can often encounter a Web page without any introduction. Thus, if your site will contain multiple pages, make sure each page is understandable if read alone. The *Guide* reinforces a lesson all too common in the world of motor vehicle crash data: give the source of information. In our case, for example, if we give a table of crash statistics, we give the source of the information, whether it be a state database, personal interview, and so forth. More than one Web site highly recommends the *Guide*, which focuses heavily on “graphic page design” (their words).

 **Indicate when your Web page was last updated, and keep information current.** *Composing Good HTML*, copyrighted by Eric Tilton and published by Addison Wesley at http://www.cs.cmu.edu/afs/cs.cmu.edu/user/tilt/public_html/cgh/index.html, suggests the need to indicate when Web pages were last updated. The purpose is to give an estimate of the validity of the information you provide. This is especially useful, for example, for weeding out pages that have outdated information, which puts pressure on you—the author—to keep information current. Keeping information current is also a good way to attract readers. For example, we plan to offer one “interesting site of the month” at the bottom of our Related Links page.

 **Respond to users.** *My Thoughts on Building a Web Page* by Joe Burns at <http://www.htmlgoodies.com/mythoughts.html> highlights the importance of responding to users when they write to you. This is an important lesson that anyone familiar with trying to obtain survey responses will recognize immediately. People are busy, and if they take the time to read what you have and offer a suggestion, whether it be positive or negative, then thanks for their efforts are in order.

 **Put the important information at the beginning of the article.** *Inverted Pyramids in Cyberspace*, by Jakob Nielsen, at <http://www.useit.com/alertbox/9606.html> notes a similarity between Web pages and journalism in that the beginning of the article contains the important information. Yet Nielsen also reminds us that these two media are not identical. A Web page derives some of its value from the ability to link to other sources of information.

 **Be brief.** Nielsen hits on an additional suggestion for Web page developers in *Be Succinct! (Writing for the Web)* at <http://www.useit.com/alertbox/9703b.html>. Brevity is critical. Nielsen advises authors to use half the amount of text they would use if writing a book, article, etc., on paper.

☞ **Provide narrative encompassing the relevant links.** Kevin Werbach, in *What Makes a Good Web Page* at http://werbach.com/Web/page_design.html, reminds us that simply identifying multiple links to other sites is pointless. We initially made this mistake when we identified other transportation safety–related organizations. We have now (we hope) corrected the problem by providing narrative encompassing the relevant links and including only the links that are essential for the reader.

Just as you wouldn't expect a few bullet points to make you a world-class author, you shouldn't expect these sources to contain everything you need to know about Web page design. Indeed, each source contains additional suggestions. We believe, however, that these basic concepts—**identify the audience, anticipate users who will not read every word, keep information current, and be brief**—are a good place to start.

Writing in HTML Directly

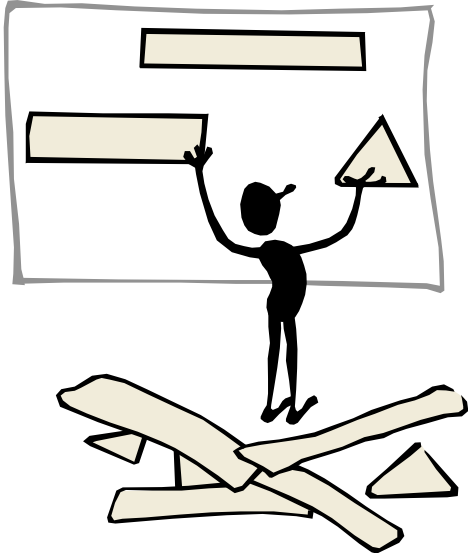
If you don't want to use Web site design software, you can design your Web page in hypertext markup language (HTML), where you manually define how each line of text will appear. A number of Internet locations have good HTML descriptions that will allow you to get started fairly easily with a few basic commands and then allow you to delve into more complex maneuvers once you've made it through the basics. In addition to *Composing Good HTML*, see a *Crash Course on Writing Documents for the Web* at http://www8.zdnet.com/pcweek/eamonn/crash_course.html. It gives a 9-page HTML overview, covering such topics as headings, paragraphs, and hyperlinks. If you're in the Charlottesville area, the Information Technology & Communication Division of the University of Virginia offers a primer entitled *A Beginner's Guide to HTML* (document number TIB-175E). Information in question-and-answer format is available from the *Web Design Group's Web Authoring FAQ (Frequently Asked Questions)* at <http://www.htmlhelp.com/faq/wdgfaq.htm>. Question 25 reminds users to keep in mind that browsers don't present graphics, such as moving text, in the same manner, which is another design consideration. (A *browser* is the instrument with which you view the various pages on the Web; examples are Netscape Navigator, Microsoft Internet Explorer, and Lynx.) Finally, James Powell's *Revised Introduction to HTML* at <http://borg.lib.vt.edu/reports/soasis-slides/HTML-Intro.html> is a comprehensive but easy-to-follow reference for HTML authoring. We recommend that you read it if you have a little more time after skimming a *Crash Course on Writing Documents for the Web* or its equivalent.

We don't recommend writing in HTML directly if your purpose is to design and maintain a Web site that will attract a large audience; instead, we recommend you use Web page development software. A case can be made, however, for using HTML if you view your site as utilitarian and limited to a select group of people. For example, if a group of authors who are located far away from one another wished to co-author a journal article, as might be the case with national research committees, then it would be very reasonable to place the document on the Web page and simply make the address known only to the authors. (This wouldn't offer any kind of security, but it could help keep the document out of the public domain.)

A knowledge of HTML can also be useful for debugging problems even if you do use Web page development software. Invariably, problems will arise with Web page development. One tool for catching errors is to examine a document's HTML code, located within the "view source," or equivalent, button of the browser. The HTML references we mentioned explain the various HTML codes (e.g., the "<p>" indicates the use of a paragraph), but more important, you can pinpoint the source of errors and then use a software package to rectify them. For example, if a particular image is not evident, you can determine whether the file name was transcribed correctly. Knowing HTML can also help you understand or implement some of the Web development software packages, as explained in Chapter 6.

Learning by Example From Other Internet Sites

Another way to develop useful Web pages is to examine other Internet sites and determine whether they have ideas you find applicable to transportation safety. For example, on our Related Links page, we initially simply listed transportation-related organizations with a short description. On other pages, however, our sponsor had observed the use of such links placed within the narrative such that if users didn't click on the links, they learned something by reading the narrative. Thus, we modified the design of the Related Links page to what it is now, using the same principle of placing hyperlinks within regular text. You can view the HTML for any page by clicking on the **View source** command in the browser, which will show you the HTML code. This isn't as useful if you're using Web design software, but you can still get a feel for what types of things were created with HTML and what types of features required specialized programming.



3. BUILDING YOUR WEB PAGE

There are a number of ways to design your Web site. Word processors and spreadsheets offer the capability to create a Web page directly within the software itself, or at least post the text or table created with the package to a specific Web page. You can also develop a Web page in HTML as mentioned in the last chapter. We developed our Web page using a software package designed expressly for creating Web pages. Several packages are on

the market. A free example is at <http://www.aolpress.com/press/2.0/usrguide/preface.htm> and is called *AOLPress*.

The remainder of these instructions concern a template we created for DRIVE SMART Virginia with Microsoft FrontPage 97, a Web page development software package. Hence, except where noted otherwise, the instructions presume you'll be using this package. However, you may read the HTML files, shown as having an *.htm* extension, with any Web development software package or word processor. To edit the HTML files themselves without using a Web development software package, see the references provided in "Writing in HTML Directly," in Chapter 2.

Using the Template

The template is at <http://www.vanet2.com/vtrc/Default.htm>. Our instructions illustrate how to do various manipulations with your Web page, such as writing text, changing fonts, adding hyperlinks, and modifying a survey form. The best way to learn is simply to download the template to your PC and begin modifying various elements. If you're using FrontPage, you should be able to import the entire site using the **File/Import** command. After you import it, you can, for example, delete the image on the top page, called *Default.htm*, and replace it with your own image or simply the title of your organization. Alternatively, you may use the instructions to modify the template gradually, making it less and less a Web page for DRIVE SMART Virginia and making it more and more a Web page for your organization. For additional FrontPage documentation beyond the scope of this document, see the FrontPage user's manual, entitled *Getting Started with Microsoft FrontPage 97*, which accompanies the software.

If you're not using FrontPage, you may obtain the HTML files directly using the **Save As** command in your browser. To obtain the images that are stored as file names with *.gif* or *.jpg* suffixes, examine the HTML file. You can see this file if you use the **View source** command in your browser. Then, read through the various commands in the file and you will see something along the lines of *img src=*, meaning *the image source is . . .*. This will be followed by a filename, such as *bullet.gif*. You can then change the Web address (Uniform Resource Locator, or "URL") to reflect only this image.

For example, the button shown in the *Default.htm* page is stored as *bullet.gif*. Recall that the normal URL is <http://www.vanet2.com/vtrc>. Therefore, to grab the *bullet.gif* file, you would go to the URL <http://www.vanet2.com/vtrc/bullet.gif> and use the **Save As** command to save the graphic to your computer.

The template is divided into eight pages. The first page, known as *Default.htm*, simply introduces the user to the organization of the Web site and indicates where to find information. The remaining seven pages are referenced from the first page. The user can click on any button to study a particular topic. To learn *About DRIVE SMART Virginia*, therefore, the user can go to the appropriate button. Not all the text is on the first page: users don't want to be drowned with information. There's nothing magic about having seven subdivisions. In fact, you probably don't want to have more sections than that on the first page.


You can then go to the other pages from the first page. Organizationally, we found it easier to have a "tree" of information rather than a single vertical list. For example, we have three separate pages with alcohol information. Yet, unless the user is interested in obtaining alcohol statistics, the fact that we have one, three, or thirty pages of alcohol data is meaningless. So, on the Web page, we show only one reference to "alcohol data." If users click on that page, they will find additional detailed information depending on whether they want 1994, 1995, or 1996 data.


To edit a particular page, open FrontPage and then double click on the appropriate *.htm* file. For example, to modify the initial page users will see, double click on the page *Default.htm*.


Using Tables to Control How Your Text Appears

One way you can format results and control how your text appears is to build your Web page within a *table*, which we did with these templates. For users of common word processing packages, however, there are some striking differences between table management in FrontPage and table management with a word processing package. First, the height of each cell is **not** explicitly specified as you might expect; rather, you may insert, merge, or delete cells to obtain a desired layout. Second, the cell boundaries themselves are flexible. It can be the case, for example, that adding text to one cell will enlarge or shrink other cells. You can overcome these challenges by having a good idea of how your Web page will appear before you begin to key it onto your computer screen, as well as some trial and error with the software.

The appearance of your Web page while you're designing is not necessarily what it will look like when viewed with a browser. To keep text formatted in the way you originally intended, consider the following options:

-  Use hard returns by using the **Insert/Break/Normal Line Break** command.

-  Build tables within tables by using the **Table/Insert Table** command.

-  Specify the width of the table or individual cells by using the **Table/Table Properties or Table/Cell Properties** command.

For more discussion on nesting tables and fixing their width, see the Web sites at <http://www.killersites.com/tutorial/index.html> and <http://www.killersites.com/tutorial/sizing.html>, which give tips for adjusting the table size, both in absolute pixel values and as a percentage that varies as a function of the browser. These sites also explain why tables on the Web behave differently from tables in a word processor. Tables on the Web try to accommodate different browsers, which adds a level of complexity not found in word processing tables.

Inserting Text and Hyperlinks

Once your table is established, you can begin to add text. The *color palette* and the *font options* in FrontPage allow you to control the appearance of any text you present. You can also change the background of your page to a specific color using the **File/Page Properties/Background** command. This will allow you to set the

default text color and default background color. With the use of tables, you can also modify these properties for a particular cell. For example, on the page *survey.htm*, the entire table was set to yellow, whereas on page *cgi-bin/detailed.htm*, specific cells were set to yellow. Finally, just as is the case with a word processor, you can highlight specific text sections and change the fonts for them using the command *Format/Font*.

Rather than simply listing all hyperlinks in sequence, consider using the links in a paragraph. It is tempting to tell the world everything about your organization, but average users will need to have their attention grabbed by your page. So, consider using links in a sentence, and then give the user the option to click on them as appropriate. You can do this simply by inserting a “pointer” to another URL. For example, in FrontPage, you may simply highlight the appropriate word(s) and then use the **Edit/Hyperlink** command to insert the appropriate hyperlink.

One feature of hyperlinks is that they can whisk a user to another site as instantly as the user arrived at your site in the first place. If this is a concern, then consider alerting users when they're leaving your site. You can do this by either placing all of the links to other sites in a specific subsection of your page or adding explanatory text (e.g., “To leave this site and learn about related organizations, go to . . .”). The ability to go from site to site quickly isn't inherently bad; in fact, it offers a new style of learning. It's simply a feature of which Web page designers should be aware.

Inserting Graphics

It's easy to insert cartoon-like images, commonly known as “clip art,” into your Web page. For example, the rotating button shown in the template was simply an image available from FrontPage. A variety of Internet sites have similar art available; one such site, <http://www.microsoft.com/gallery/default.asp>, provided the champagne glass, truck, and warning signs on the alcohol Web page. There are two important things to remember when using clip art. *First*, make sure the source of your art doesn't object to you using the art in your Web page. It would be inappropriate, for example, for designers to immediately begin using the DRIVE SMART Virginia logo in their own Web page unless they'd obtained permission from DRIVE SMART Virginia. The one exception to this rule is when you link to a page: adding a logo usually helps viewers recognize the organization to which the link is pointing. *Second*, be careful not to overdo the images and lose your original point. Clip art can attract attention, but too much may distract the user. Photographs require a bit more work than clip art since you must consider their

resolution; hence, we recommend not delving into that area until you're comfortable with Web page development overall.

We developed the images shown in the template from two sources. We developed some with specialized software packages. We developed others using a simpler route; we used the tools available with the commercially available Web page package. We used a product called Image Composer, which comes with FrontPage. It allows you to select various fonts, colors, and sizes for your image. You can select the size of the images as well as the size of the composition. You do the latter by using the **Edit/Properties** command.

To create images such as the "Are Governments the Only Players? NO!" graphic on the Related Links page, do the following steps with the FrontPage Image Composer:

1. Use *Edit/Properties* to select the width and height of the overall image.
2. Use the *Text* button and fonts with the message.
3. Use *Shapes/Geometry* to add the background, setting the volume as appropriate.
4. Repeat steps 1 through 3 for the different colored letters.
5. Save the image file within the appropriate subdirectory.
6. *Insert* the image within the appropriate place in your page.

You can easily do this entire process in 10 minutes. The challenge doesn't come from the mechanics of the steps but, rather, from visualizing the image itself, including varying the font and colors so they complement one another. Regarding step 6, on our computer, we found that we sometimes had to save the image, exit from FrontPage, and then restart FrontPage before the image would be directly recognizable in a *.gif* format.

How to Do "Banners"

Banners, or marquees, are blocks of text that flash, roll, or otherwise move across the screen. Although there is a *marquee* option in FrontPage, it wasn't animated when we used a Netscape browser. This highlights another potential problem with Web page development: not all browsers view text in the same manner.

To create the banners shown under the Related Links page, we used GIF Construction Set software. This is available over the Internet for \$20 from a company called Alchemy Mindworks at <http://www.mindworkshop.com/alchemy/gifcon.html>. Again, a number of options exist for developing animated images such as those shown in the Related Links page. We simply chose one.

To develop the banners as shown, do the following steps with Gif Construction Set:

1. Use *Edit/Banner*.
2. Use *Banner Text* (insert the text and fonts desired).
3. Use *Text Color* (change as appropriate).
4. Save the file.
5. In Microsoft FrontPage, change the *image properties* to the desired size.

One thing to keep in mind is that the scaled image may appear different. For example, increasing the image size from about 450 pixels to 550 pixels also enlarged the font. This increase, however, didn't appear to adversely affect the image. You may do more complex images with GIF Construction Set if you're so inclined.

Citing Your Data Sources

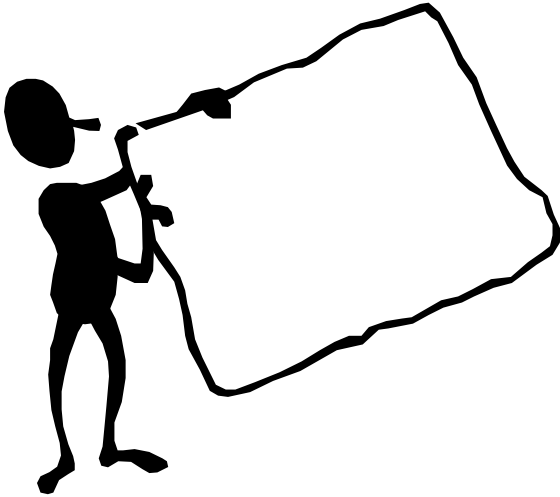
A disadvantage with any medium that allows easy communication, such as the World Wide Web, is that it's easy for rumors or conflicting accounts to be spread on any given topic. Unfortunately, crash data aren't exempt from this problem. To ensure your organization's credibility, we recommend that you cite, whenever appropriate, the source of your information. In our example, the alcohol involvement

and crash data were cited as originating from the *1994 Virginia Traffic Crash Facts* (and later the *1995* and *1996 Virginia Traffic Crash Facts*). Although no data source is guaranteed to be accurate all of the time, citations help you minimize the risk of using fabricated data and at least let your users know where you obtained your information.

If you're planning to display Virginia crash data, we strongly recommend that you consider using DMV's *Virginia Traffic Crash Facts*, as this data source has credibility and is subject to verification.

Checking Your Web Site

Keep in mind that how your Web page appears on your computer isn't necessarily how it will appear on another person's computer. Even two users who have identical browsers, such as Netscape Version 3.01, may see text and images formatted differently. Hence, once the site is operational, make sure you get feedback from other users about the appearance of certain features, especially any text or graphics that require unique formatting. In our case, the counter's appearance needed substantial modification even though it looked fine on the computers we were using. In addition, you'll need to check hyperlinks periodically. Features of Web page development software are available to accomplish this, but it's also good to go through the process of trying the links yourself as would a user. Why? The quality or content of the sites to which you're linking may have changed.



4. PUBLISHING TO THE WEB

If the server you use has **FrontPage extensions**, then publishing to the Web is easy. Simply use the **File/Publish FrontPage Web** command, which will then prompt you for the address. If you don't know this address, then ask your ISP.

One very simple thing to keep in mind is that FrontPage differentiates between the *Editor* and the *Explorer*. Up until now, if you've been modifying the template by clicking on the various *.htm* pages, such as *Default.htm*, then you have been using the *Editor*. For the **Publish** command, however, use the *Explorer*, which activates immediately after starting FrontPage.

If the server you use doesn't have **FrontPage extensions**, a problem we initially encountered, then it won't be possible to publish directly to the Web. Instead, you'll need to use a program such as FTP (File transfer protocol) to transfer your files. This will be easier if you place all images in the same subdirectory as your HTML files. Then, using FTP, execute the following commands at the <FTP> prompt:

```
lcd Web path (in our case, we entered lcd c:\Webshare\wwwroot\  
drive_smart)  
ascii  
prompt  
mput *.htm  
mput *.htx  
bin  
mput *.jpg  
mput *.gif  
bye
```

The **lcd** command ensures that you copy files from the appropriate directory, which in this case is the subdirectory where the Web is stored in your PC. The **prompt** command turns off the prompt, such that files are passed without you having

to press return for each file. The **bin** command ensures that binary files are transferred correctly.

Most of the template will work fine without the FrontPage extensions, but they're necessary for the counter, survey, and database query.

Whether your server has FrontPage extensions or not, you'll want to register your site with the various search services users employ to query various sorts of information. We found all but two of these companies by going to the "net search" or its equivalent in our browser. Examples of major search services are Yahoo, WebCrawler, and Infoseek. You may submit your site to any or all of the search services shown in Table 3.

TABLE 3. SEARCH SERVICES TO WHICH YOU SHOULD SUBMIT YOUR WEB PAGE

Search Service	Address (URL)
Add-It	http://www.liquidimaging.com/submit/ (all in one site submission)
AltaVista	http://altavista.digital.com/av/content/addurl.htm
AOL NetFind	http://www.aol.com/netfind/info/addurl.html
Excite	http://www.excite.com/Info/add_url.html
Hotbot	http://www.hotbot.com/addurl.html
Infoseek	http://www.infoseek.com/AddUrl?pg=DCaddurl.html
LookSmart	http://www.looksmart.com/h/info/submsite.html
Lycos	http://www.lycos.com/addasite.html
Submit-It	http://free.submit-it.com/ (all-in-one site submission)
WebCrawler	http://www.Webcrawler.com/WebCrawler/Help/GetListed/HelpAddURL.html
Yahoo	http://add.yahoo.com/fast/add?

If you click on the URL and find that it goes nowhere, don't panic. You can still find the search service by simply entering its name. For example, if the address **<http://add.yahoo.com/fast/add?>** doesn't work, then simply enter *yahoo* in the URL, which will get you to the Yahoo company site. Two of the sites listed in **bold, Add-It, and Submit-It** are not search services themselves but are free services that will update the other search services regarding the existence of your site. Thus, simply completing the forms with either of these two organizations may cover the remaining search services, but if you decide to select this option, then we recommend that you verify with the other search services that the search for your Web site is fruitful. For information on promoting your Web site, the organization at **<http://www.cnet.com/Content/Features/Howto/Promote/ss01f.html>** has good suggestions.

Each search engine may search the Web in a different manner. For example, WebCrawler doesn't use key words submitted by the user but rather uses text from your Web page, which underscores the need for your Web page to have some brief but meaningful text that conveys the type of information users can find at your site.

At the time of this writing, many of the search companies were indicating that once you submit your Web site, it may take 2 to 4 weeks for their search to include your site on their list. Last, the process of submissions took us about an hour; we found that one of the all-in-one submission sites did not have all of the search services in which we were interested, so we also had to submit our site to a couple of search services individually.

5. DEVELOPING SIMPLE SURVEYS

We created the survey shown as part of the *survey.htm* page by using the command **File/New/Feedback Form** in the FrontPage Editor. You can modify the wording of the questions as well as the type of questions by changing the *form field properties* for each piece of information entered by the respondent. For example, consider the second question of the survey, shown here:



What type of organization do you represent?

Federal Government	▶
--------------------	---

The user can then click on the *down arrow button* and find other choices, such as state government, private sector, and so on. If you wanted to add a selection to this list, such as the name of your organization, then you would right-click on the box, select *form field properties*, and add your company. If you wanted to add additional questions, then you would add the text along with the **Insert/Form Field** command, which then lets you select the type of box in which the user can enter a response: multiple choice, drop-down, free response, and so on. In a nutshell, the **form field properties** command allows you to control how the user returns survey data. At the very end of the survey, you need to add a *submit* button through which the user can send these data to the server.

Once the survey is designed, you need to ensure that (1) you can get the data the user submits, and (2) the user receives an acknowledgment. If you right-click anywhere within the survey template and select *form properties*, a dialog box that reads *Form Handler* will appear with the *name WebBot Save Results Component*. Go to *Settings*, and first press on the *Results* tab. You'll see a file called *feedback.htm*. This is the file to which user responses are stored. Note that *File Format* is "Text database using comma as a separator." There's nothing magical

about this format. If, for example, you want to use SPSS (which prefers tabs to commas), then select the option “*Text database using Tab as a separator.*”

If at the same point you click on the *Confirm* tab, the page *confirm.htm* will be shown. This is the letter sent to the user immediately after the user submits the data. We urge you to customize this letter so that the user feels some gratification as a result of going to the trouble to respond to the survey!

Going back to the *Results* tab with the confirmation file (*feedback.htm*), note that you can store these data in a variety of formats, including HTML. You can quickly see that you can use this capability to create a live “comments” page where readers can submit ideas on a variety of topics. If you do this, you want to make sure that the confirmation file’s URL is made available to the users who want to see comments sent by other users.

Finally, you can treat the *feedback.htm* file as an ASCII file and sort it accordingly. For example, you can import the file into Microsoft Word, remove all the hard returns except for each survey, and then import the file into Microsoft Excel and tabulate each question. In our case, we can find how many respondents are from local governments.



6. POSTING DATA

Posting Static Data

Placing “static” data, such as the entire contents of a spreadsheet, on the Web is fairly easy, and word processing or spreadsheet packages sometimes contain step-by-step instructions for doing it. For example, the first table of alcohol data was created in Microsoft Excel and then saved as an HTML file. From there, we imported it directly into FrontPage.

Posting Live Data

It's possible to store data such that they may be queried directly from a database. This is potentially very useful for two reasons. *First*, it saves you—the data provider—the time and effort of customizing an entire data set for Web display, although you must do some preprocessing. *Second*, the user can query select data elements that only he or she is interested in, thereby avoiding information overload. In our case, we believe this is useful for displaying alcohol statistics by county. With 135 cities and counties in our Virginia database, the user may find it much more aesthetically pleasing to select only a single jurisdiction rather than having to read statistics for the entire state.

Reminders When Posting Data

There are four basic concepts that are good to keep in mind when you're posting data:

1. **Give the source of the data.**

2. Explain any limitations or quirks that a new user might not grasp without explanation. For example, in our case, a user can query a database for a specific jurisdiction and then obtain crash data. For jurisdictions such as Fairfax, however, that represent both a city and a county, we established our database such that the user enters *FairfaxCi* for Fairfax City and *FairfaxCo* for Fairfax County. This information is given before the user begins the search.

3. Tell users what to do if their search fails. For example, if they need to change case, check spelling, or so on, then this should be communicated when the search fails. (See step 7 in the next subsection where the appropriate message can be added to the *.htx* file.) Users can also alert you to problems if you respond to your error messages. For example, when our search fails, users are prompted to email us a description of the problem along with the jurisdiction and year for which they're doing the search.

Many of these mistakes came to our attention because we made them in the first place! In the live data application, the cities and counties that were scanned electronically in some cases used abbreviations we found nonnutritive. Initially, the database returned *That jurisdiction is not in Virginia* when *Virginia Beach* was entered. This is because the entry used the term *Va. Beach*. Needless to say, we changed the entry. Another way to address this problem is to create a “drop down” list from which users automatically select the database entry, which we later did for all three data sets.

4. Remember that proofreading never hurts. In one instance, we found that when users clicked on a certain jurisdiction from the 1995 data, they received an error message indicating that the jurisdiction was not in Virginia. The Web page was correct, but the jurisdiction appeared in the database being queried as:

Winchester.

The problem, as you might have guessed, arose when we scanned a page of data electronically. The “V” was not scanned as a letter at all but as a series of four slash marks: “\” and “/” and “\” and “/”. The only way we caught this error was by trying out the database query on Winchester, finding that the query was unsuccessful, and working from there.

Steps for Establishing a Database Query

We didn't invent the steps shown here. We pulled them from other sources, applied them to the DRIVE SMART Virginia Web page, and then used the lessons we learned to compile the steps. For more complete information, see <http://www.microsoft.com/frontpage/wpp/kit/primer.htm>, which has a good FrontPage introduction. Also see an article called *Sample Web: Using Internet Database Connectivity with FrontPage 97* at http://www.microsoft.com/FrontPageSupport/content/IDC/idc_ex.htm, which provides illustrative database examples. This last file allows you to install a sample Web page and study your own database queries.

The data need to be placed in what is known as an open database connectivity (ODBC) database. This is a standard database protocol that in our case allows Web pages to access the database using a format known as Structured Query Language (SQL). In our case, for example, the data were first scanned electronically into Microsoft Word, converted to Microsoft Excel, verified by hand, and then pasted into a Microsoft Access database. (For more information on the significance of ODBC-compliant databases, see *Start/settings/Control Panel/ODBC/About* in Windows 95.)

You need two software packages to create a Web page database query from scratch. One houses the database. For this one, we used a software package called Microsoft Access, although any ODBC-compliant database should be acceptable. The second software package is Microsoft FrontPage, although other Web site development software packages may be able to accomplish similar functions.

The steps aren't difficult, but initially they're tedious. Therefore, the first time you do them, we recommend that you allow yourself time to focus on this single task alone. Thus, it's good to do this section once everything else concerning the Web is in order. For the DRIVE SMART Virginia template, these steps refer to the alcohol statistics query, as shown in the pages *detailed.htm* and *cgi-bin/detailed.htx*. The other data pages, *livedata.htx* and *cgi-bin/livedata.htx*, were built using the same principles.

Here are the steps:

1. Create the Access database.

2. Assign the ODBC driver.
3. Create a Web page from which to access the database.
4. Create a Web page that will return the search results.
5. Create the Internet database connector page.
6. Ensure that all files within the *cgi-bin* subdirectory can be executed.
7. Add an If-then clause to the *.htx* file.
8. Update the *.htm* file regarding the Internet database connector.
9. In Access, create a query for your database (optional).
10. Make final updates (optional).
11. Make changes for your specific server (optional).

Step 1. Create the Access Database

1. Select the subdirectory in which you want to store your new database.
2. Give the database a title.
3. Enter the values.
4. Name each column by the type of data stored in it. For now, we can call these field names.
5. Save the work as Table1.

We stored the new database in the *_private* subdirectory and gave it the title *alc.mdb*, such that the path name in the Web was *private/alc.mdb*. Then, we pasted values from the Excel worksheet (you could enter them directly). We then named each column by the type of data stored in it. We used the field names *juris*, *crash*, *fatal*, *injury*, *acrash*, *afatal*, *ainjury*, *driver*, and *rate*. Finally, we saved the work as *Table1*.

Step 2. Assign the ODBC Driver

1. For a PC in Windows 95, click on the following in the order shown:

Start/Settings/Control Panel/32bit ODBC/System DSN/Add/Access

2. Under the *Database* heading, press **Select** until the file name, which in our case was *alc.mdb*, appears. Later, when queried in step 5, you will use the three-letter name *alc*.

Step 3. Create a Web Page from Which to Access the Database

1. Create a file name with an *.htm* extension. We used *detailed.htm*.
2. Use the **table** command to create a table that will store the variable a user will use to search the database. In our case, users will be searching for alcohol statistics by *jurisdiction*, so the table contained a column for the user to enter the jurisdiction.
3. Insert a form field in the second column. In our case, the name under *form field properties* was *juris*, since the jurisdiction will be used to query the database.
4. Insert *Submit* and *Reset* buttons in the same table where the query appears.

Step 4. Create a Web Page That Will Return the Search Results

1. Place the page in the *cgi-bin* subdirectory.
2. Enter the name of file from step 3 with an *.htx* extension. We used *cgi-bin/detailed.htx*. Therefore, the path name in the Web is *cgi-bin/detailed.htx*.
3. Create a table as in step 3. This time, the table will contain the search results and will have two columns, one with the field name and one with the field value. For

example, the left column might contain the word “crashes,” and the right column would contain the number of crashes for the jurisdiction selected.

4. Click on **Edit/Database/Detail** only once such that the entire table is placed as a detail area.
5. While in each column in which you want a database result to appear, click on **Edit/Database/Database Column Value** and enter the database name. For our example, each cell had the field names from step 1: *juris, crash, fatal, injury, acrash, afatal, ainjury, driver, and rate*.
6. Adjust the table borders and table size as appropriate using the **Table/Table Properties** command. As illustrated with the 1994 and 1995 data, you may arrange data by row or column.

Step 5. Create an Internet Database Connector Page to Link the Web Pages to the Database

1. Place it in the *cgi-bin* subdirectory with an *.idc* extension. In our example, the path name in the Web is *cgi-bin/detailed.idc*.
2. Go to *New page*, and then, *Database Connector Wizard*.
3. Under *Database Connector*, select the Access database, which in the example is *alc* from step 2, as the ODBC datasource. **Even though *alc* refers to *alc.mdb*, don't type the *.mdb* extension! Type only the three letters *alc*!**
4. Under *query results template*, press the **browse** button until you obtain *detailed.htx*.
5. Press the **Next** button, and in the SQL box, enter the following text: *SELECT juris, crash, fatal, injury, acrash, afatal, ainjury, driver, rate FROM Table1 WHERE juris = '%juris%'*. This means to print the values of crash, fatal, and so on only for the jurisdiction identified by the user. The *SELECT* statement means we are selecting the field names we used in step 1 where we created the Access

database. The *WHERE* statement means that we are searching by the field name used in step 2, substep 2, which is the jurisdiction.

6. Press the **Finish** and **Save** buttons.

Step 6. Ensure That All Files in the cgi-bin Subdirectory Can Be Executed

In the FrontPage Explorer, click on the *cgi-bin* subdirectory, then click **Edit**, and then check the box that indicates “allow scripts or programs to be run.”

Step 7. Add an If-then Clause to the .htx File

In case users search for a county or city that doesn't exist, they'll want to know why their database query returned nothing. Instead of presenting them with a blank table, you may have the search return a phrase, such as “That city or county is not in Virginia. Please check the spelling.”

1. Open the *.htx* page, which in our example is *detailed.htx*. Go to *Edit/Database/If then/*, change the *type* to *CurrentRecord*, and at the bottom of the screen, enter the number *0* for the Value.
2. Place the message you want your users to receive within the markers that appeared on the page. Although a drop-down list should render this step unnecessary, it's a good intermediate measure until you've created the list or finished debugging the query.

Step 8. Update the .htm File Regarding the Internet Database Connector

1. Open *detailed.htm*, select the **submit** button with the left mouse button, right click, and change the *form* property to *Internet Database Connector* and also change the settings to reflect the *idc* file created in step 5.

2. "Browse" until you reach *cgi-bin/detailed.idc*. If your query fails, browse again, select your **submit** or **reset** button, and ensure that the *Internet Database Connector* is your *form* property with the appropriate IDC file named in the *settings*.

Step 9. Create a Query for Your Database

A query can be useful if you wish to add new variables to your database. In our case, we accomplished this with the 1994 crash data to compute a percentage of crashes that are alcohol related.

1. Return to the database you created, and click on *Table/query/create* using wizard. At this stage you may add variables. For example, you may do *table/query/design/percentage: [crash alcohol]/[crash total]*.
2. Then *format*, and then save as *Table2.query*. This step is not essential; the 1995 and 1996 data didn't use a specific query created in Access.

Step 10. Make Final Updates

In the Internet Database Connector file, such as *livedata.idc*, you may have to revise the name of the table used in step 9. For example, in the *livedata.idc* file, we replaced the name *Table1* with *Table2query*. In addition, in the query, you may use SQL (Standard Query Language) commands to format the data as appropriate. For example, in our query, we wanted a variable that computes a percentage based on two variables and then rounded it off to the nearest integer. In that case, the relevant portion of the SQL command is given in bold :

```
SELECT Table1.ID, Table1.County, Table1.CrashTotal, Table1.CrashAlcohol,  
Int([CrashAlcohol]/[CrashTotal]*100) AS Percentage  
FROM Table1;
```

The **Int** command means that the variable computed as "percentage" will be rounded to an integer. You can find a list of SQL reserved words beginning on page 3-5 in the *ORACLE SQL Language Reference Manual*, by D. Cheu, Part Number 778-V6.0, 1988.

Step 11. Make Changes for Your Specific Server

Once everything works on your PC and you're ready to publish the command to a commercial server, there are at least two changes that need to occur.

1. Your ISP administrator will need to update the DSN name as appropriate. This corresponds on an NT Server, for example, to the tasks that you did for your personal Web server back in step 2.
2. You'll need to update the *detailed.idc* file that was created in step 5 to match what your system administrator does. In our case, for example, the system administrator had given the DSN name of *vanetc.alc*; hence, in the *detailed.idc* file, we changed the *datasource* from *alc* to *vanetc.alc*.

Using HTML to Automate Database Entries

If you've gone to the trouble of learning HTML, you haven't wasted your time. Knowing both HTML and FrontPage can help automate otherwise tedious database queries. For the 1995 data, shown in the file *detailed.htm*, there's a drop-down menu with the various cities and counties. One way to place these entries is to type them in one line at a time – a process that can become tiresome very quickly. Unfortunately, you can't globally paste all 135 entries directly into the *form field properties* dialog box.

There is, however, an easy alternative. Establish the *form field properties* with just a couple of sample entries. We put in Albemarle and Charlottesville. Then, using the command *View HTML*, you can see the HTML structure, which includes

```
<option>Albemarle</option>  
<option>Charlottesville</option>
```

Then, from your database, grab all 135 entries, paste them into a word processor, globally insert the `<option>` and `</option>` after each county or city, and

paste all this information into the HTML code. You'll have created a drop down menu for the user with very little effort.

More important, this is a way to use HTML and any site development software together. Use the site development software to start what you want on a very small scale, then see how it's accomplished in HTML, and then either edit the HTML or apply the software – whichever is easier for the task at hand.

Establishing a Personal Web Server

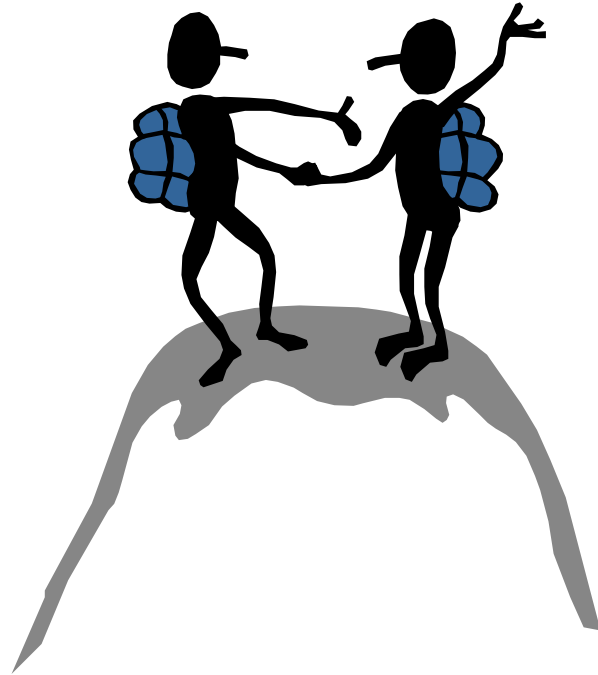
Creating a personal Web server isn't detailed in this document since it's not essential to building your Web page. For some users, however, this option may be worthwhile until you obtain a commercial server, especially if you want to test certain interactive capabilities of the Web page, such as surveys, data queries, and "chat rooms." If your organization has a firewall, you may be interested in using the Settings/Control Panel/Network/TCPIP icon to change your Internet Provider (IP) address to an address that isn't within the firewall. See your Systems Administrator for details. For more information on how to establish your personal Web server, should this not already be installed, see

<http://www.microsoft.com/frontpage/upgrade/engupgrade.htm>, which contains a document entitled *Upgrading your FrontPage 1.1 Personal Web Server and content to the new Microsoft Personal Web Server*.

7. CONCLUSION

The purpose of a Web page, as is the case with a word processor, spreadsheet, database, geographic information system, or any communications medium, is to enhance your organization's mission. Be selective about how you spend your resources, *including our suggestions*. If your users don't need data from your site, for example, then don't waste time posting them. If your users do need data from your site but don't need graphics, then dropping fancy graphics from your site is fine. If your users need information on how they can become

actively involved with highway safety efforts, then keeping your page current is essential. If another site already has the same information as yours, then unless you think you can do a better job of presenting or managing the information, consider finding a niche that no one else has occupied rather than repeating what's already available.



Last, knowing a little bit of the theory behind how Web pages function can be useful for getting along with the administrator of the commercial server who maintains your site. In our case, we used a "counter" that recorded the number of hits on the home page. The counter worked fine on a personal Web server but failed on the commercial server, even after several hours of debugging. The problem, as it turned out, was that although the commercial server had installed certain files (FrontPage extensions), the administrator had decided to install only the files necessary for that particular counter for a newer version of FrontPage. Once the administrator realized this was the problem, he kindly updated our home page counter software for us. Yet, had we realized that there was an incompatibility problem between FrontPage 97 and FrontPage 98 regarding the installation of the extensions for this particular counter, we could have saved ourselves a few working hours.