Untangle Webs Of Data With The BASIS ODBC Driver

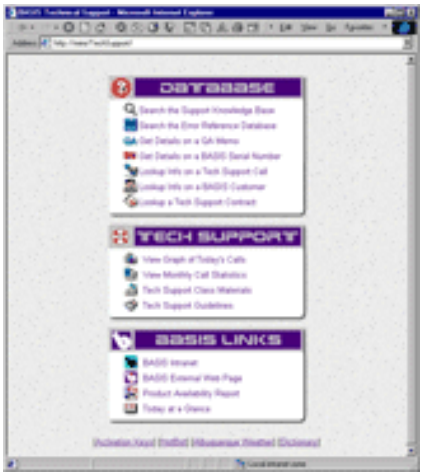
by Ernie Longmire

Any company that maintains its corporate information in more than one database knows that giving access to everyone who needs it can quickly become a nightmare. Often there are few real options available; either everyone needs to be trained on every database application used in the company, or it's necessary to implement and maintain a complete custom software application that ties all the pieces together. The first solution introduces security problems (should your engineers really be able to edit customer billing records?) while the latter gives the company yet another unwieldy internal application with which to wrestle.

Fortunately, there are now two standard tools available that can combine to bridge the gap between corporate data and the audience that needs to see it--the BASIS ODBC Driver® 2.0 and the World Wide Web. This ODBC Driver provides an industry-standard database connection to your existing BBx® and PRO/5® data files. The World Wide Web brings together a number of world-standard interfaces and protocols that can make information available in human-readable format to anyone connected to the Internet. In combination, they form an amazing engine of data retrieval power.

BASIS is using this combination of tools to make detailed information about its customers available to our Technical Support analysts. The data is only accessible to BASIS employees, by way of the internal company network, but the techniques used to set up this powerful internal solution can be applied just as easily to any website, public or private.

This connection between data and the end user is made possible by the current generation of ODBC-to-Web connectivity tools. In our case, BASIS' Technical Services Supervisor, Nick Decker, developed a collection of search pages using Microsoft's Internet Information Server (IIS) webserver and Microsoft FrontPage 97 with the Internet Connector Wizard as the page development tools. However, there is a wide variety of other options available. Any web server that can connect to a Windows ODBC data source can be used with the BASIS ODBC Driver to provide similar access to data sources.



BASIS' Internal Technical Support web page offers access to a wide variety of corporate information sources.

Setting Up For Access

To begin making data available online, the relevant data sources need to be accessible via ODBC. The BASIS ODBC Driver on the company's internal Technical Support web server has at least a half-dozen information sources available from a number of different machines within the company. None of these corporate databases were developed for the Technical Support web application; they were put together for their own specific projects and were later made available for standardized access through the BASIS ODBC Driver.

The two main databases that are accessible through the Technical Support web pages are the Technical Support Call Log database, which is maintained by the Call Log application used every day by BASIS' support staff, and a MAS90-based customer information database named 411. The Call Log database includes a large amount of information directly relevant to day-to-day Technical Support work, including a complete record of all contacts made between the customer and the BASIS support department. The 411 database, on the other hand, includes information on the various product licenses held by each customer, what products they are using on what operating system, and so forth.

By making this information available over the corporate intranet, BASIS has made it much easier to access. For example, managers do not need to use the highly-specialized Call Log application to check on the status of a Technical support issue--they can simply enter a customer name on a search page, select the appropriate incident from the list returned, and view the complete history of the customer's interactions with the support staff, right down to the amount of time spent on each call. From the other side of the fence, Technical Support now has quick and easy access to customer information, such as software revision levels and user license sizes, that can help them narrow down the problem when helping someone on the telephone.

Hooking Up

Connecting to disparate information sources with IIS and FrontPage is simple and straightforward. These Microsoft products offer a number of Wizard-style tools that make issuing queries and returning results point-and-click simple.

The first step is to create the form that will initiate the search. It is possible to initiate an SQL search from a web page using only a handful of lines of embedded HTML code:

```
<form action="/cgi-bin/Detail_Cust.idc" method="GET">
<input type="text" size="35" name="CUST">
<input type="submit" value="Search">
</form>
```

This code displays a text entry box on the page with a Search button next to it. When the button is pressed, the user's web browser will take the text typed in the entry box, label it CUST, and tell the web server hosting the page to search for that text using an SQL query stored in the file **Detail_Cust.idc**.

Before this request will actually *do* anything, however, it's necessary to create the **Detail_Cust.idc** file ("idc" being short for Internet Database Connection). This file defines what Microsoft's IIS will do with the search request when it is submitted.

Using the Microsoft FrontPage Database Connector Wizard, enter the SQL query to run on the specified database.

Creating an **.idc** file is as easy as selecting the New option on the File menu in the FrontPage 97 editor. Just select Database Connector Wizard from the menu presented and enter three pieces of information:

- The ODBC Data Source to search.
- The name of the template file to use when displaying the query results (the name of the file should end in **.htx**).
- The actual SQL query to execute when a request comes in.

From this information, the Wizard creates the **.idc** file in the server's **cgi-bin** directory, a special directory that the server uses to hold executable applications. This is really just a simple text file, as shown by the sample **.idc** file:

```
Datasource: 411
Template: Detail_Cust.htx
SQLStatement: SELECTCUSTOMER.CUST_NUM,
+CUSTOMER.NAME,CUSTOMER.ADDR_ONE,
+CUSTOMER.ADDR_TWO,CUSTOMER.CITY,
+CUSTOMER.STATE,CUSTOMER.ZIP_CODE,
+
+STR(CUSTOMER.PHONE_NUM,'(XXX) XXX-XXXX')
+ as CUSTOMER.PHONE_NUM,
+
+CUSTOMER.CONTACT
+FROM CUST_NAME_SORT, CUSTOMER
+WHERE (CUST_NAME_SORT.NAME LIKE '%CUST%%%')
+ AND (CUST_NAME_SORT.CUST_DIV=CUSTOMER.CUST_DIV)
+ AND (CUST_NAME_SORT.CUST_NUM=CUSTOMER.CUST_NUM)
```

When a search request comes in for an **.idc** file, IIS expands the query with any parameters passed from the request form (such as the **'%CUST%%%'** parameter above, which is replaced with the contents of the CUST text entry box mentioned earlier) and runs the query against the data source specified. The result is a set of zero or more records that matches the SQL query.

Making Your Results Look Good

This is all very well and good, but how does the data get from the query results onto a web page? That's why the **.htx** template file is used. This file is a normal HTML file with a few extras that IIS uses to

help present the results of an SQL query. Here are a few examples of what's special about **`.htx`** files:

- Output can be formatted with tags representing specific fields in the records being retrieved from the data source. Each of these tags is replaced with the actual contents of the field when the result page is generated.

- Different pieces of HTML can be displayed depending on the results of a query. For example, it is possible to display a formatted result if one or more records is returned but show a "no records found" message if the query came up emptty.

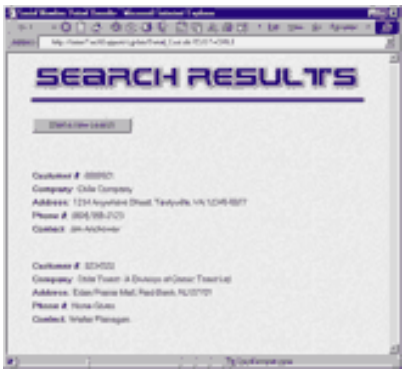
- A "detail" template can be defined that will automatically be filled once for each record returned as a result of the query.

All of these options come together in this sample template, which is based on one of the templates used by our Technical Support department. (The HTML formatting tags have been removed to make it easier to see what the special template-related tags are doing.)

```
<%if CUST_NUM GT "0"%>

<%begindetail%>
Customer #: <%CUST_NUM%>
Company: <%NAME%>
Address: <%ADDR_ONE%>
<%if ADDR_TWO GT "0"%>
<%ADDR_TWO%>
<%endif%>
<%CITY%>, <%STATE%> <%ZIP_CODE%>
Phone #:
<%if PHONE_NUM GT "( ) -"%>
<%PHONE_NUM%>
<%else%>
 None Given
<%endif%>
<%enddetail%>

<%else%>

Sorry, no information matched your criteria.

<%endif%>
```

The outer **`if-else-endif`** code checks to see whether any records were actually returned by the query. If there were, they are displayed with the template defined between the pair **`begindetail-enddetail`**, but if there were no records, the "**`Sorry, no information matched...`**" message is displayed.

The code between **`begindetail`** and **`enddetail`** is fairly straightforward, showing the contents of the various fields present in each record returned by the query. There are two special **`if-endif`** and **`if-else-endif`** cases, which tell IIS to display the second address line and the telephone number only if they are present. If more than one record is returned from the query, IIS automatically fills this "detail" template once for each record returned, displaying all of the results on the same output page.



Searching for companies with "chile" in the name returns a list like this one.

It is a simple matter to extend the techniques described above to almost any kind of data, from a corporate parts inventory to an internal company telephone/email directory. The BASIS ODBC Driver has the power to connect to your legacy data sources, and the Web has the power to make those data available virtually anywhere. Together they can make the power of your corporate information be felt worldwide.