Número 5, desembre 2000

ASP en bibliotecas y centros de documentación: publicación de una base de datos *Microsoft Access* en el web

[versió catalana]

Josep Manuel Rodríguez i Gairín
Facultat de Biblioteconomia i Documentació
Universitat de Barcelona
rzgairin@fbd.ub.es

Resumen

Se describen los elementos necesarios para publicar en Internet una base de datos creada con *Microsoft Access* utilizando la tecnología ASP. Se parte de un ejemplo de base de datos creada para controlar el proceso de adquisiciones y se van resiguiendo, en forma de tutorial, los diferentes pasos que serán necesarios para su consulta desde el web. Finalmente, se indican algunas aplicaciones de la tecnología ASP que pueden ser útiles para bibliotecas y centros de documentación.

Introducción

El presente artículo recoge, a modo de guía, los pasos necesarios para publicar una base de datos de *Microsoft Access* en Internet. Si bien a lo largo del mismo se introducen definiciones y siglas como ASP, ODBC o SQL, se ha pretendido huir de la realización de un redactado teórico y concentrarse en una exposición práctica siguiendo un ejemplo concreto del entorno bibliotecario que permita valorar su utilidad.

Nuestra experiencia demuestra que, en muchas ocasiones, la mejor manera de aprender determinados procesos técnicos es ejecutarlos y por ello, invitamos a leer este artículo al tiempo que se realiza en la práctica el objetivo propuesto de publicar una base de datos.

Si bien son aconsejables conocimientos elementales tanto de *Access* y HTML como de programación básica; la exposición se basará en sencillos ejemplos y trucos que pueden seguirse sin excesiva dificultad por cualquier profesional de la biblioteconomía y la documentación.

Este artículo es interactivo. Si el lector está conectado a Internet podrá ver los resultados de los distintos formularios instalados en un servidor NT e interaccionar directamente con la base de datos de ejemplo. Si se detecta alguna anomalía o disfunción en el servidor agradeceremos nos lo comunique por correo electrónico.

Nuestro ejemplo

Las ventajas e inconvenientes de usar un sistema de gestión de bases de datos relacional como *Microsoft Access* en la gestión del catálogo de una biblioteca ya han sido ampliamente tratados por diversos autores (Salse, 1998; Trigueros 1997; Codina 1994). Sin embargo, muchos de los procesos relacionados con tareas que implican gestión económica —adquisición, obtención de documentos, préstamo interbibliotecario— están siendo tratados con motores de bases de datos relacionales como *Microsoft Access*, *GTBib* o *Dbase*. Incluso grandes centros automatizados con potentes sistemas de gestión de bibliotecas como *VTLS* u otros, usan estas bases de datos relacionales para los procesos administrativos. Por su parte, los pequeños centros, que disponen de escaso presupuesto para automatización, encuentran en el paquete *Office* de Microsoft una de las soluciones de primera línea para todos estos propósitos.

Sin querer entrar más en esta polémica y aunque podríamos haber usado un ejemplo administrativo de otro campo, nos ha parecido más oportuno utilizar uno que pudiera resultar familiar al lector. A este efecto, usando *Microsoft Access,* diseñaremos una base de datos para controlar las adquisiciones de una biblioteca que llamaremos "adqui.mdb". La gestión de las adquisiciones es un proceso complejo que implica la gestión de partidas presupuestarias, proveedores, pedidos, reclamaciones, cierres de ejercicio, etc. Sin embargo y dado que no es el tema principal de este artículo, hemos simplificando el ejemplo usando solamente tres tablas relacionadas entre sí: los usuarios que solicitan la compra por

medio de desideratas o similares, los libros que se desean adquirir y los proveedores de esos documentos. En nuestro ejemplo, cada libro es solicitado por un único usuario y se solicita a un único proveedor; la realidad seria más compleja ya que un libro podría ser solicitado por más de un usuario y en caso de no ser servido podría resolicitarse a otro proveedor. Necesitaríamos, por tanto, otras tablas auxiliares para mejorar todo este proceso.

Gráficamente la estructura de nuestra base de datos simplificada tendrá el siguiente aspecto.

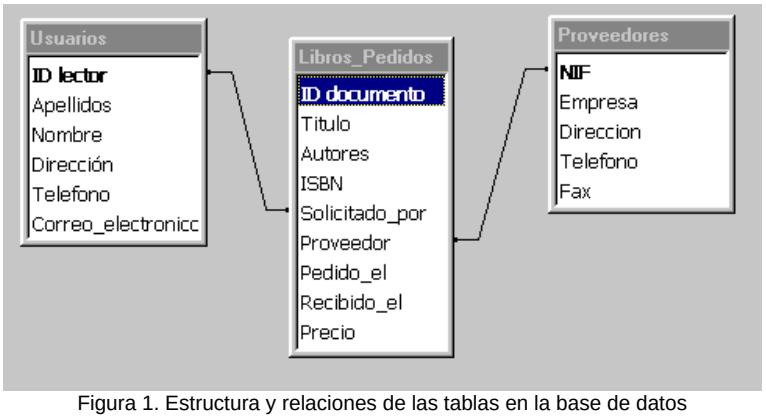

Figura 1. Estructura y relaciones de las tablas en la base de datos

Para realizar los procesos relacionados con la adquisición tal vez no sería necesario publicar esta base en el web. Si los responsables del servicio son una o dos personas, pueden trabajar directamente en un entorno Windows con *Microsoft Access.* El problema se plantea cuando decidimos emplear esta base de datos para ofrecer subproductos al usuario final a través del WWW.

Un ejemplo claro lo constituyen los típicos expositores que encontramos en la entrada de muchas bibliotecas donde se muestran los últimos libros recibidos. Su versión electrónica bien pudiera ser un listado de nuevas adquisiciones ingresadas en nuestro centro que colgara de la página principal de nuestro Web. Incluso podríamos diseñar un sistema que permitiera al usuario consultar el estado de su desiderata, si se ha pedido ya al proveedor, si se ha recibido, etc. Evidentemente, en centros pequeños esta consulta podría convertirse en la interfaz del propio catálogo. Insistiremos en recordar las limitaciones del sistema: campos de longitud fija, no existe posibilidad de repetición, indización por frase, etc.

Estos procesos podrían resolverse en modo local por medio de consultas. Como ejemplo, diseñaremos una consulta de selección sobre la tabla "libros_pedidos" para listar los documentos recibidos hasta una fecha concreta y una consulta de unión para listar los libros pedidos por un usuario o a un proveedor concreto.

Algunas de estas consultas deben tener el siguiente aspecto o similar:

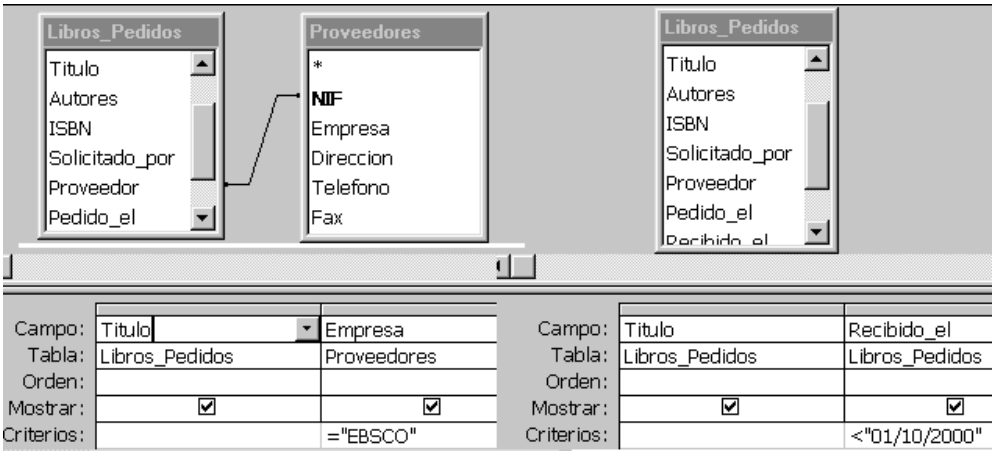

Figura 2. Consultas en *Microsoft Access*

Observemos que en estos casos, si no queremos entrar en programación, las consultas han de ser predeterminadas, es decir, necesitaremos hacer tantas consultas como proveedores o usuarios. Podríamos generar un formulario en *Access* y ligarlo a la consulta pero necesitaríamos conocer más a fondo el sistema. En el caso de la consulta vía Web, como veremos, la solución también pasa por crear un formulario previo que nos permita escoger el proveedor o fijar una fecha concreta en cada caso.

Nota: Aunque recomendamos que sea el propio lector el que genere su propia base de datos y las consultas, si lo desea puede descargar la que hemos confeccionado nosotros, que es la que hemos utilizado como ejemplo a lo largo de este artículo.

La tecnología que describiremos permite no tan sólo consultar los datos vía Web sino también realizar todas las funciones de administración clásicas (añadir, modificar, borrar registros). Sin embargo detallar todos estos procesos excede el propósito de esta exposición y requiere nociones más avanzadas de informática en las que están implicados aspectos de seguridad y control de registros. De todos modos es importante conocer que podríamos publicar igualmente informes o formularios para entrada o modificación de datos.

Los requisitos

Internet se basa en una arquitectura cliente-servidor; los usuarios que consulten nuestra base de datos vía Web simplemente necesitarán disponer de una conexión a Internet y de un navegador, no requiriendo tener instalado *Access* ni ningún otro complemento. Por nuestra parte, necesitamos un servidor de Web conectado permanentemente a Internet para poder disponer de dicha base las 24 horas. Los ficheros *Access* son aplicaciones Windows de 32 bits y por tanto, deben ser instalados en un entorno Windows. Se recomienda *Windows NT* con *Internet Information Server 4.0* o superiores. Sin embargo, para hacer pruebas podemos configurar un sistema en local sin necesidad de conectarnos a Internet simplemente instalando en nuestro PC (*Windows 95/98*) el *Servidor Personal de Web* (*PWS)* que se puede descargar de Internet http://www.microsoft.com/download) aunque también se incluye directamente en el CDROM de *Windows 98*. Se recomienda *Windows 98* con *PWS 4.0* que ofrece ya completo soporte a la tecnología ASP (Active Server Pages).

Evidentemente en nuestro ordenador necesitaremos tener también *Access* y un editor de páginas web (aunque con el bloc de notas tendremos suficiente para nuestro ejemplo). La versión de *Access* que utilizamos es *Access 2000* si bien versiones anteriores son igualmente válidas para configurar la base de datos. *Access 2000* dispone de un asistente para crear las páginas Web pero no lo usaremos sino que se editarán manualmente para aprender lo que realmente hacen.

Los pasos a realizar

Copiar la base de datos al servidor

Los siguientes pasos son válidos tanto si estamos instalando la base de datos en el servidor de Internet (IIS) como en el caso de la instalación local con *Personal Web Server.*

Si ya tenemos creada nuestra base de datos como se ha descrito anteriormente y hemos introducido algunos registros, lo primero que tenemos que hacer es copiarla al servidor. Podemos copiarla en cualquier directorio, no es necesario que sea visible desde el Web, es decir, no es necesario que cuelgue de \inetpub\wwwroot\, es más, se recomienda copiarla fuera de esta estructura para una mayor seguridad.

Establecer una conexión ODBC

ODBC (Open Database Conectivity) es el protocolo que el sistema operativo empleará para comunicarse con la base de datos. Será el propio servidor quien interrogará la base y obtendrá los resultados. Esto es importante ya que implica que por la red sólo circula la pregunta formulada y el conjunto de resultados, disminuyendo notablemente el tiempo de respuesta.

El lenguaje que emplea el servidor para comunicarse con la base se denomina SQL (Structured Query Languaje). Tanto el protocolo como el lenguaje son estándares abiertos, no requieren comprar ningún programa adicional y son compatibles con multitud de formatos de bases de datos entre ellos el MDB (Microsoft Database).

ODBC emplea unos ficheros controladores (drivers) específicos para cada tipo de bases de datos, Microsoft facilita estos controladores y normalmente estarán instalados en nuestro servidor o se instalarán junto a la aplicación de *Office.*

ODBC se instala por defecto con el propio Windows. Simplemente tenemos que definir una conexión o enlace, indicándole al sistema que tenemos una base de datos que queremos hacer accesible a este protocolo. Esto es muy fácil y no requiere más de 5 minutos. Veamos como hacerlo en nuestro ordenador; el proceso es idéntico en el servidor NT, pero en ese caso, por cuestiones de seguridad, posiblemente sólo puede hacerlo el administrador del sistema.

Ir a " Inicio" —> "Configuración" —> "Panel de control" y escogeremos el icono "Fuentes de datos ODBC". Este icono, que es como el que se muestra en la figura adjunta, se instala directamente con Windows y siempre lo encontraremos en nuestro ordenador.
Nos aparecerá un cuadro de diálogo como el que se muestra a continuación:

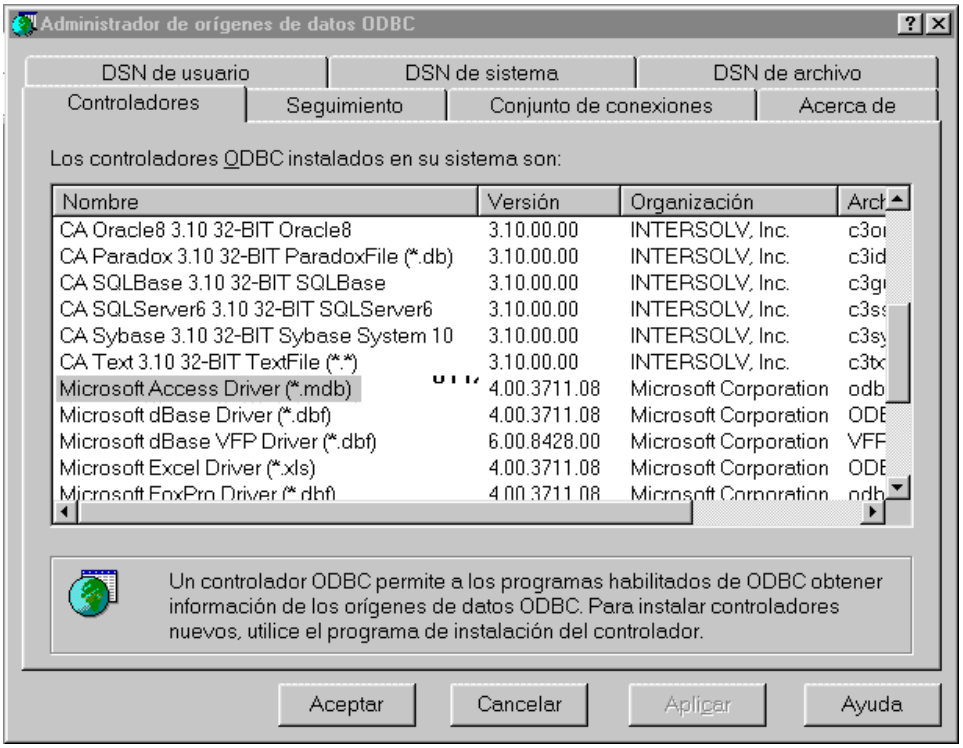

Figura 3. Ventana del administrador de origen de datos ODBC

En este ejemplo se muestran los controladores que nuestro sistema tiene instalados. Cuando se abre este diálogo tal vez aparezca otra pantalla, pulse en la pestaña "Controladores" para acceder a esta pantalla en concreto. En nuestro ejemplo aparecen muchos controladores, obsérvese que ODBC en nuestro ordenador puede interrogar muchos tipos de bases de datos, algunos de estos controladores sí que son propietarios y tienen un coste pero a los efectos que nos ocupa simplemente necesitamos el controlador *Microsoft Access Driver* (*.mdb) que aparecerá si nuestro ordenador tiene instalado Access o que nuestro administrador de red podrá instalar sin coste en el servidor descargándolos de Internet.

Para crear nuestra conexión ODBC simplemente pulsaremos en la pestaña "DSN de sistema" (data source names, nombres de origen de datos de sistema)

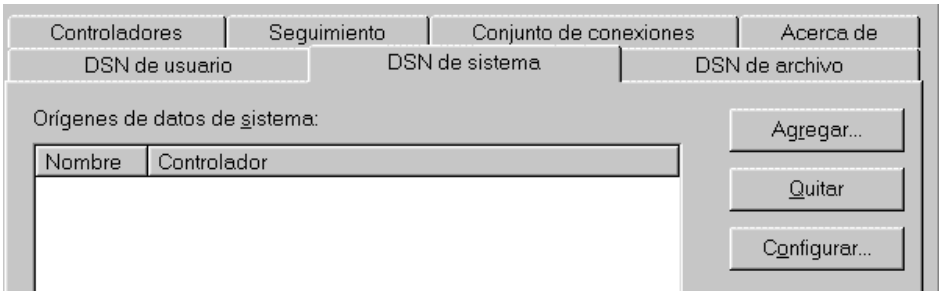

Figura 4. Cuadro de origen de datos de sistema

En el cuadro "Origen de datos de sistema" aparecerán las conexiones ODBC que ya tengamos establecidas. Es necesario crear una conexión/asociación para cada base de datos que queramos publicar. Pulsaremos en "Agregar" para definirla, nos pedirá que escojamos el controlador que vamos a utilizar (en nuestro caso *Microsoft Access Driver)* y al pulsar "Finalizar" nos solicitará una serie de datos:

Nombre del origen de datos: Escogeremos un nombre único para nuestra base, que no necesariamente tiene que coincidir con el nombre del fichero y que será el nombre que utilizaremos desde nuestras páginas cuando queramos acceder a ella con el comando "Open"como se verá más adelante.

Descripción: Una breve descripción, aunque no es imprescindible, puede ser útil sobre todo si tenemos varias bases de datos.

Seleccionar: Al pulsar sobre este botón se nos abrirá un cuadro de diálogo para que escojamos la ubicación del fichero MDB que hemos copiado o generado previamente.

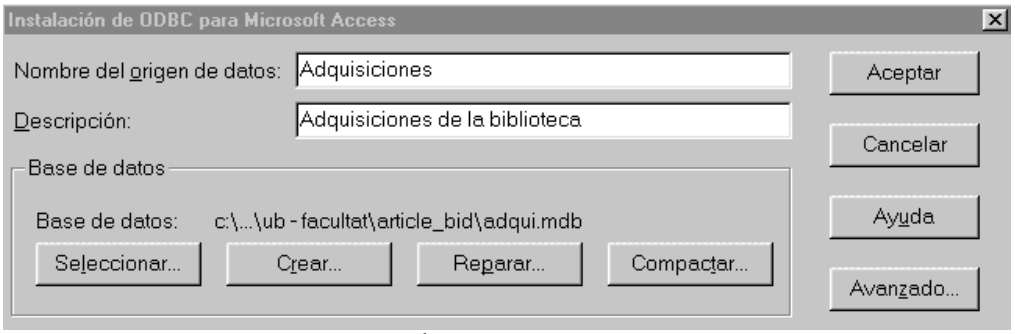

Figura 5. Pantalla para seleccionar la ubicación de la base de datos.

Pulsamos sobre el botón "Aceptar" y ¡ya está!. La asociación ya ha sido creada. Esta operación sólo se ha de ejecutar una vez, salvo que cambiemos de ubicación la base de datos. No es necesario hacer nada cuando modifiquemos o añadamos datos en la base siempre que no cambiemos ni el nombre ni la ubicación.

Usuarios más expertos pueden acceder al botón "Avanzado" para fijar otros parámetros como identificaciones de acceso, etc.

Crear los ficheros necesarios para su consulta

En el caso de bases de datos pequeñas, *Access* permite la creación de páginas estáticas compuestas por tablas HTML que reproducen las propias tablas de la base de datos. El usuario descarga la totalidad de la tabla cada vez que accede vía web y el sistema de búsqueda se basa únicamente en las capacidades de búsqueda que ofrece el propio navegador. Al ser páginas estáticas deben ser regeneradas manualmente en el servidor cada vez que se actualiza la base de datos si se pretende mantener actualizada la consulta.

En el caso de bases grandes o de actualización frecuente, este sistema no resulta nada apropiado y debemos recurrir a consultas dinámicas en las que el servidor interacciona directamente con la base de datos y devuelve al cliente sólo los registros que cumplen la condición solicitada; por ello este sistema resulta muy rápido y totalmente interactivo con los datos actualizados.

Existen dos mecanismos para realizar este tipo de consulta dinámica, uno basado en ficheros IDC/HTX (Internet Database Conector) y otro en ficheros ASP (Active Server Pages). En ambos casos el lenguaje de consulta empleado es SQL (Structured Query Languaje). Pueden encontrarse definiciones ampliadas de estos términos en el glosario que se adjunta al final del artículo.

Microsoft Access dispone de un asistente capaz de generar las páginas con el código necesario. Las versiones de *Office '97* incluyen una opción de "Guardar como página Web" que lanza este asistente, en el caso de *Office 2000* debe accederse a "Crear una nueva página de acceso a datos". En nuestro caso generaremos las páginas manualmente con un editor HTML o incluso con el bloc de notas. Usaremos el método de páginas *ASP*. Generaremos las tres páginas básicas para consultar cualquier base de datos en el web:

- "formulario.htm": Un formulario de consulta.
- "listado.asp": Una página de resultados intermedios de la consulta a modo de listado.
- "ficha.asp": Una página con el resultado final de un registro.

El formulario

Diseñemos una página Web con el siguiente código:

```
<html>
<head>
</head>
<body>
<h1 align="center">Formulario de búsqueda</h1>
 <form method="post" action="listado.asp">
    Buscar : <input name="texto">
    en campo :
    <select name="campo">
       <option value="titulo">Titulo</option>
       <option value="Autores">Autores</option>
    </select>
    <input type="submit" value="consultar">
```

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

Figura 6. Código fuente de formulario.htm y aspecto de la página

Ver formulario

Nota: Simplemente visualice el código del formulario, más adelante ya podrá probarlo.

En realidad este código es fácilmente interpretable por cualquier persona con conocimientos básicos de HTML; se trata de un formulario con dos campos, uno de tipo texto (llamado texto) cuyo valor estará definido por el usuario al entrarlo en la casilla correspondiente y uno de tipo selector (llamado campo) cuyo valor está predefinido por el diseñador de la página y el usuario se limitará a escoger entre las opciones, en este caso cada opción contiene el nombre del campo de la base de datos sobre el que queremos buscar el texto.

Tal vez lo más interesante es ver que, si bien habitualmente los formularios son enviados directamente a programas llamados CGI (Common Gateway Interface) que recogen los campos del formulario para realizar acciones sobre las bases de datos y devolver los resultados al navegador; en este caso, el formulario envía los datos directamente a un fichero con extensión ASP.

El listado

En segundo lugar diseñaremos un fichero que llamaremos "listado.asp"que contenga el siguiente código:

```
<%@language="VBScript"%>
<html>
<head>
   <title>Resultados de la búsqueda</title>
</head>
<body>

<%
dim campo, texto
campo = request.form("campo")
texto = request.form("texto")
set cn=Server.createObject("ADODB.Connection")
cn.ConnectionTimeout=20
cn.open "DSN=Adquisiciones;DriverId=25; FIL=MS Access;MaxBufferSize=512;PageTimeout=5;"
set rs=Server.CreateObject("ADODB.Recordset")
rs.Open "SELECT * FROM libros_pedidos WHERE InStr(Ucase(" & campo & "), Ucase('" & texto & "') )>0", cn
%>

<h1>Resultado de la búsqueda</h1>
<ul>

<%
Do until rs.eof
   Response.write "<li><a href=ficha.asp?Id_documento=" & rs.fields("Id_documento") & "> Ver ficha</a> -- "
   Response.write  rs.fields("titulo")
   Response.write chr(10)
rs.MoveNext
loop
%>
</ul>

<%
rs.close
cn.close
set rs = Nothing
set cn = Nothing
%>

</body>
</html>
```

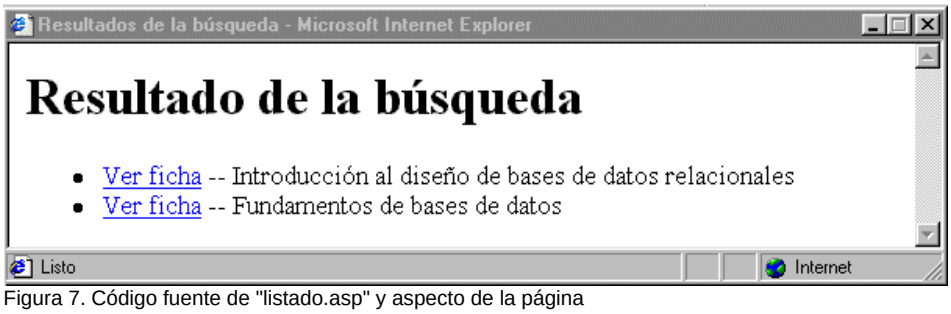

Figura 7. Código fuente de "listado.asp" y aspecto de la página

El uso de los tres colores en el código es para indicar que este fichero contiene tres tipos de lenguajes. el negro representa código HTML, el azul es Visual Basic Script y el rojo es SQL. La interpretación de este código es un poco más complicada pero vamos a intentar explicarlo de una manera sencilla.

En primer lugar hemos de remarcar que todos los ficheros ASP se inician con la línea <% @language="VBScript"%> que indica al navegador que el fichero que se está leyendo contiene código informático en lenguaje Visual Basic Script. ¿Cómo reconoce el navegador qué es HTML y qué es código VBScript? Observemos que cada vez que se inicia código informático va precedido de <% y finaliza con %> El contenido entre ambas marcas no se visualizará como HTML sino que se procesará.

Veamos qué hace este código.

Unos pequeños conocimientos de programación serán muy útiles en este punto. A grandes rasgos diremos que la programación combina la entrada de datos (input), su almacenamiento en variables, su procesamiento y su salida (output). En VBScript existen varias maneras de recibir datos. En este caso request.form es la función que recupera los campos del formulario que ha invocado esta página y los almacena en las variables.

VBScript es un lenguaje de programación orientada a objetos que emplea objetos del tipo ADO (Activex Data Objects). La programación clásica se basa en una serie de instrucciones que el procesador ejecuta secuencialmente. La programación orientada a objetos se basa en definir una serie de objetos con unas características o propiedades que el programa modifica dinámicamente, este tipo de programación es más flexible ya que no está condicionada por una secuencialidad. A efectos de nuestro ejemplo necesitaremos dos objetos, un objeto conexión para comunicarnos con el servidor y un objeto tabla para almacenar y manipular los resultados de la consulta.

La línea set cn=Server.createObject("ADODB.Connection") crea un objeto conexión y lo almacena con el nombre **cn** para su uso posterior. La línea cn.open "DSN=Adquisiciones;DriverId=25; FIL=MS Access;MaxBufferSize=512;PageTimeout=5;" abre la conexión con la base de datos. Obsérvese que le especificamos la base de datos que vamos a utilizar. En realidad, diciéndole simplemente el nombre del origen de datos (DSN) definido en la conexión ODBC cn.open "DSN=Adquisiciones" tendríamos suficiente pero normalmente se añaden el resto de parámetros.

La línea set rs=Server.CreateObject("ADODB.Recordset") crea un objeto tabla y la siguiente línea lo llena con los registros que cumplen una condición concreta. Aunque la sintaxis SQL puede parecer complicada, es interesante ver que se compone simplemente de unas pocas órdenes como SELECT (especifica los campos de las tablas que queremos usar para la visualización o todos si usamos un asterisco) FROM (especifica las tablas de la base de datos que vamos a usar) WHERE (especifica la condición que deben cumplir los registros), ORDER BY (especifica el campo por el que queremos verlos ordenados). Esta instrucción se leería: *selecciona todos los campos de la tabla "libros_pedidos" que contengan el valor de la variable texto en el campo que tiene por nombre el valor de la variable campo.* Para complicarlo un poco más, como la búsqueda es sensible a mayúsculas y minúsculas, previamente se convierte a mayúscula ambos lados de la igualdad. No debemos asustarnos si no entendemos totalmente esta sintaxis, más adelante veremos un pequeño truco con *Access* para usarla sin necesidad de conocerla.

Una vez ya tenemos el conjunto de registros, el siguiente fragmento de código usa lo que en programación se conoce como un bucle Do until/rs.movenext/loop para mostrar en la página Response.write los campos rs.fields que queramos. La instrucción Do until rs.eof se lee: *hacer mientras no se encuentre el final de la base* (eof= end of file).

Observemos la línea Response.write "Ver ficha". En este caso estamos creando un hipervínculo que nos permita enlazar cada uno de los resultados intermedios con la ficha completa. Esta línea combina frases literales entre comillas con campos de la base de datos unidos por el operador &. Esta línea se repite para cada uno de los registros encontrados ya que está dentro del bucle y en cada caso el hipervínculo generado llamará a la página "ficha.asp" pero pasándole el campo "ID_documento" (que es el campo clave) con un valor diferente correspondiente al registro en el que esté situado.

Por último, el fragmento final de código VBScript cierra la tabla rs.close y la conexión cn.close y destruye los objetos creados set rs = Nothing, set cn = Nothing.

Probar formulario

Nota: Busque el término "datos" ya que el ejemplo contiene bibliografía sobre el tema.

La ficha final

Por último diseñaremos un fichero que llamaremos "ficha.asp" que contenga el siguiente código:

```
<%@language="VBScript"%>
<html>
<head>
  <title>Ficha final</title>
</head>
<body>

<%
dim registro
registro = Request.QueryString("ID_documento")
set cn=Server.createObject("ADODB.Connection")
cn.ConnectionTimeout=20
cn.open "DSN=Adquisiciones;DriverId=25; FIL=MS Access;MaxBufferSize=512;PageTimeout=5;"
set rs=Server.CreateObject("ADODB.Recordset")
rs.Open "SELECT * FROM libros_pedidos WHERE ID_documento=" & registro , cn
%>

<h1 align="center"> Documento <%=rs.fields("ID_documento") %> </h1>
<p>
<hr>
Título : <%=rs.fields("Titulo") %> <br>
Autores: <%=rs.fields("Autores") %> <br>
ISBN: <%=rs.fields("ISBN") %> <br>
<hr>

<%
rs.close
cn.close
set rs = Nothing
set cn = Nothing
%>

</body>
</html>
```

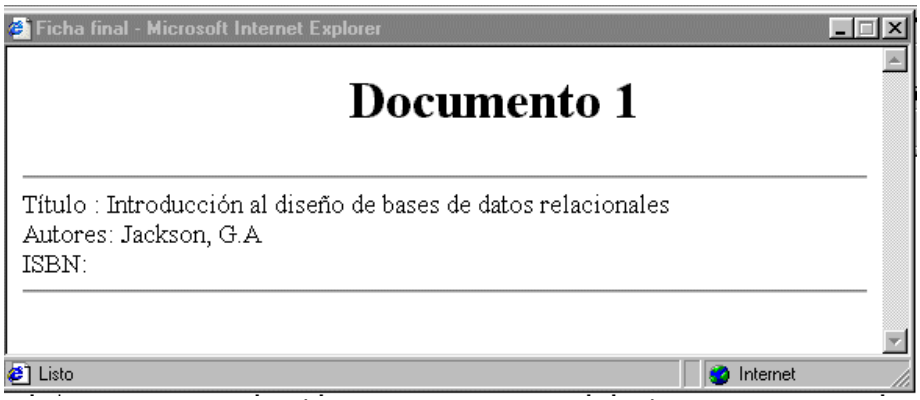

Figura 8. Código de "ficha.asp" y aspecto de la búsqueda

Si hemos comprendido la explicación del punto anterior observaremos que esta página es bastante similar aunque contiene algunas diferencias que es necesario comentar.

Observemos que en este caso sólo definimos una variable registro para almacenar el identificador que recibimos de la página anterior. La función que nos llena esta variable es Request.QueryString en vez de request.form. La razón es que en este caso el identificador nos llega desde la propia cadena dentro de la URL en vez de llegarnos desde un formulario.

request.form recupera variables desde campos de formulario (INPUT, SELECT, RADIO, etc)
request.queryString recupera variables desde URLs (pagina.asp?campo=valor1&campo2=valor2)

La selección SQL en este caso es más sencilla, simplemente recuperaremos el registro que cumple la

condición "ID_documento = valor" recibido de la URL. Al tratarse de un campo clave con valores únicos simplemente recuperaremos un registro.

En esta página no empleamos ningún bucle ya que simplemente tenemos un registro y queremos visualizar los campos. Una diferencia respecto a la página anterior es que aquí para visualizar empleamos <%=rs.fields en vez de Response.write. En realidad podríamos haberlo hecho de la siguiente manera:

```
Response.write "Autor :" & rs.fields('autor') & "<br>"
Response.write "Titulo :" & rs.fields('titulo') & "<br>"
Response.write "ISBN :" & rs.fields('ISBN')& "<br>"
```

ambas formas son válidas, tal vez la primera resulta más sencilla.

Atención si el campo "ID_documento" no fuera clave y la base pudiera tener más de un registro, simplemente visualizaríamos los datos del primer registro encontrado.

Probar formulario

Nota: Busque el término "datos" y pulse sobre "ver ficha" para uno de los resultados intermedios. Visualice el código fuente de la página intermedia para ver qué ha hecho el bucle.

Copiar los ficheros al servidor de Web y probarlo

Una vez creados los tres ficheros simplemente tendremos que copiarlos al servidor de Web y ya tendremos nuestra base de datos consultable. Evidentemente esto es una visión muy simplificada que únicamente nos servirá para comprender vagamente el mecanismo de consulta. En este ejemplo hemos prescindido completamente del control de errores. Supongamos que introducimos en el formulario un texto del cual no existe ningún documento. ¿Qué sucede? La instrucción SQL no recupera ningún registro y posiciona el puntero de la base al final con lo cual el bucle no se ejecuta. En la práctica no aparece nada por pantalla. Una buena programación debe preever esta eventualidad incluyendo código como el siguiente antes o después del bucle:

```
if rs.eof and rs.bof then
Response.write "Lo siento. Ningún registro cumple esa condición"
end if
```

Nota: cuando una tabla está vacía, el puntero está al principio y al final de la base a la vez y por tanto bof (begin of file) y eof (end of file) coinciden.

Probar formulario ANTES de añadir la condición

Probar formulario DESPUÉS de añadir la condición

Nota: Busque el término aspirina que no figura en la base de datos.

Algunas aplicaciones interesantes

Listado de nuevas adquisiciones

El desarrollo anterior se basaba en un modelo en que el usuario interaccionaba con el sistema haciendo una pregunta: ¿Qué documentos hay que tengan tal contenido en un campo? A veces queremos hacer páginas dinámicas con una condición preestablecida. Supongamos que queremos colgar una página web con las nuevas adquisiciones entendiendo por tales las recibidas en los últimos treinta días. Podemos hacerlo de una manera estática, generando una tabla el primer día de cada mes, o de una manera dinámica generando una página ASP que haga la consulta y que no necesitaremos modificar con el paso del tiempo.

¿Cómo sería esto en *Access*?

Access nos permite ver el código SQL de las consultas generadas con el asistente y, por lo tanto, podemos cortar y pegar ese código desde *Access* a nuestra página ASP. Este sencillo truco nos ahorra conocer SQL. Usando el generador de expresiones podemos hacer una consulta en Access para recoger aquellos documentos en los que la fecha actual menos la fecha de recepción (campo recibido_el) no supere los treinta días. Evidentemente hemos de conocer como funciona el generador de expresiones de *Access* y este ejemplo tal vez no sea de los sencillos. El resultado sería como el siguiente:

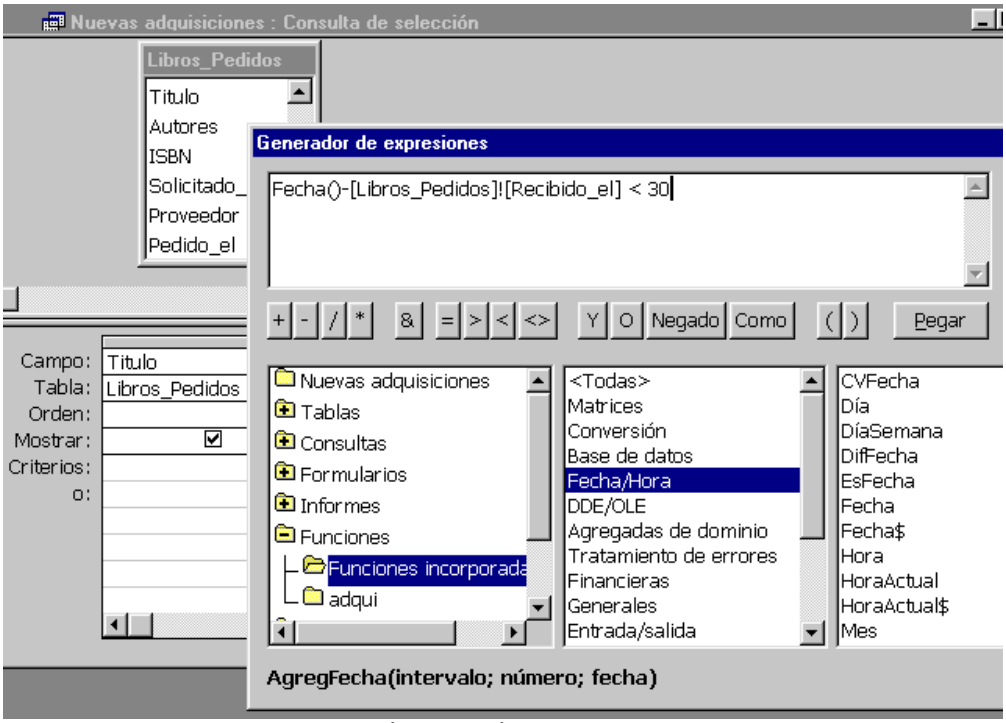

Figura 9. Consulta de libros recibidos los últimos 30 días.

Si pulsamos sobre el icono "vista" y escogemos "vista SQL" vemos el código SQL

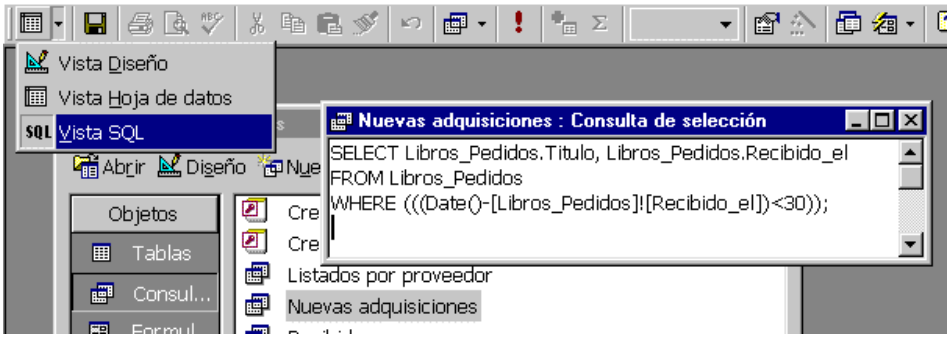

Figura 10. Vista SQL de una consulta.

Con estos datos es fácil diseñar una página que llamaremos "nuevas_adq.asp" cuyo código sería:

```
<%@language="VBScript"%>
<html>
<head>
   <title>Ultimas Adquisiciones</title>
</head>
<body>

<%
set cn=Server.createObject("ADODB.Connection")
cn.ConnectionTimeout=20
cn.open "DSN=Adquisiciones;DriverId=25;
FIL=MS Access;MaxBufferSize=512;PageTimeout=5;"
set rs=Server.CreateObject("ADODB.Recordset")
rs.Open "SELECT * FROM libros_pedidos WHERE (((Date()-[Libros_Pedidos]![Recibido_el])<30)) ORDER BY Recibido_el
DESC;", cn
%>

<h1 align=center> Ultimas adquisiciones recibidas en la biblioteca</h1>
<table width=100%>
<tr><th></th><th>Título</th><th>Recibido el</th></tr>

<%
Do until rs.eof
   Response.write "<tr><td><a href=ficha.asp?Id_documento=" & rs.fields("Id_documento") & ">Ver ficha</a></td>"
   Response.write "<td>" & rs.fields("titulo") & "</td>"
```

```
    Response.write "<td align='right'>" & rs.fields("recibido_el") & "</td></tr>"
    Response.write chr(10)
rs.MoveNext
loop
%>

</ul>

<%
rs.close
cn.close
set rs = Nothing
set cn = Nothing
%>

</body>
</html>
```

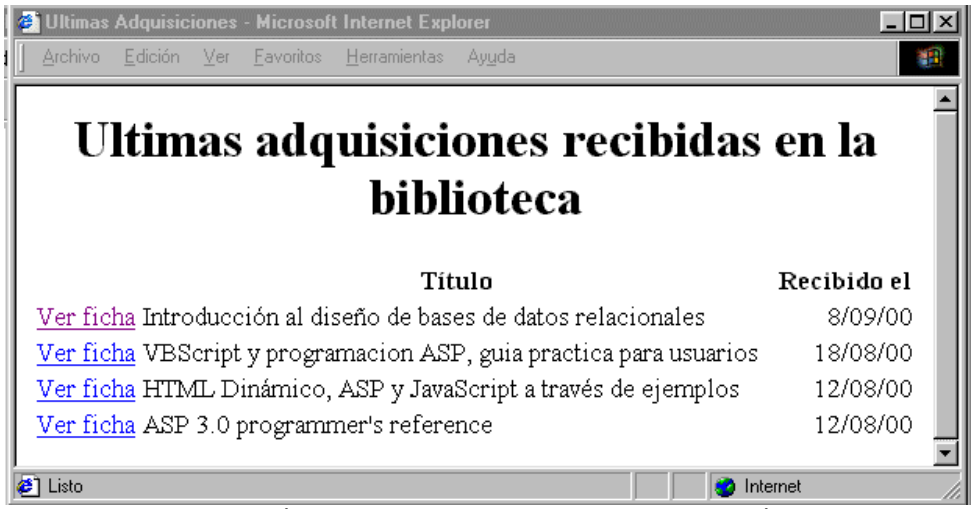

Figura 11. Código fuente de "nuevas_adq.asp" y aspecto de la página

Observemos que el código es incluso más sencillo que los anteriores. Al no haber interacción del usuario no es necesario definir variables ni recibir datos. Está página puede ser llamada directamente como un enlace desde cualquier página HTML y aunque aparentemente parece una página estática, los datos cambiarán en función de la fecha. Es interesante observar que a la instrucción SQL se ha añadido un estamento ORDER BY que permite ordenar los resultados de manera descendente por el campo "recibido_el"**;** de esta manera el último libro recibido aparece en primer lugar.

ver nuevas adquisiciones de la biblioteca

Consultas por proveedor

A partir de aquí no importa lo compleja que sea la instrucción SQL, simplemente tenemos que cortarla desde *Access* y pegarla en nuestra página. Así si miramos la instrucción SQL de la consulta de unión definida para obtener listados por proveedor veremos algo como lo siguiente:

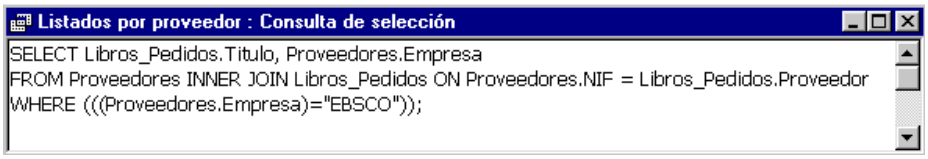

Figura 12. Vista SQL de la consulta *Access*

Sustituyendo "EBSCO" por una variable heredada de un formulario, podemos construir un sistema para obtener listados por subministrador.

Ver formulario

Incluso podríamos diseñar un formulario dinámico en el que el usuario en vez de tener que introducir el nombre del proveedor lo seleccionara de un desplegable generado a partir de la tabla proveedores. El código sería:

```
<%@language="VBScript"%>

<%
set cn=Server.createObject("ADODB.Connection")
cn.ConnectionTimeout=20
cn.open "DSN=Adquisiciones;DriverId=25;
FIL=MS Access;MaxBufferSize=512;PageTimeout=5;"
set rs=Server.CreateObject("ADODB.Recordset")
rs.Open "SELECT * FROM proveedores", cn
%>

<html>
<head>
</head>
<body>
<h1 align="center"> Listado por proveedores</h1>
   <center>
   <form method="post" action="proveedor.asp">
      Seleccione proveedor
      <select name="texto">

<%
Do until rs.eof
   Response.write "<option value='" & rs.fields("empresa") & "'>" & rs.fields("empresa") & "</option>"
   Response.write chr(10)
rs.MoveNext
loop
%>

      </select>
      <input type="submit" value="consultar">
      </center>
   </form>
</body>
</html>

<%
rs.close
cn.close
set rs = Nothing
set cn = Nothing
%>
```

Figura 13. Código fuente del desplegable dinámico

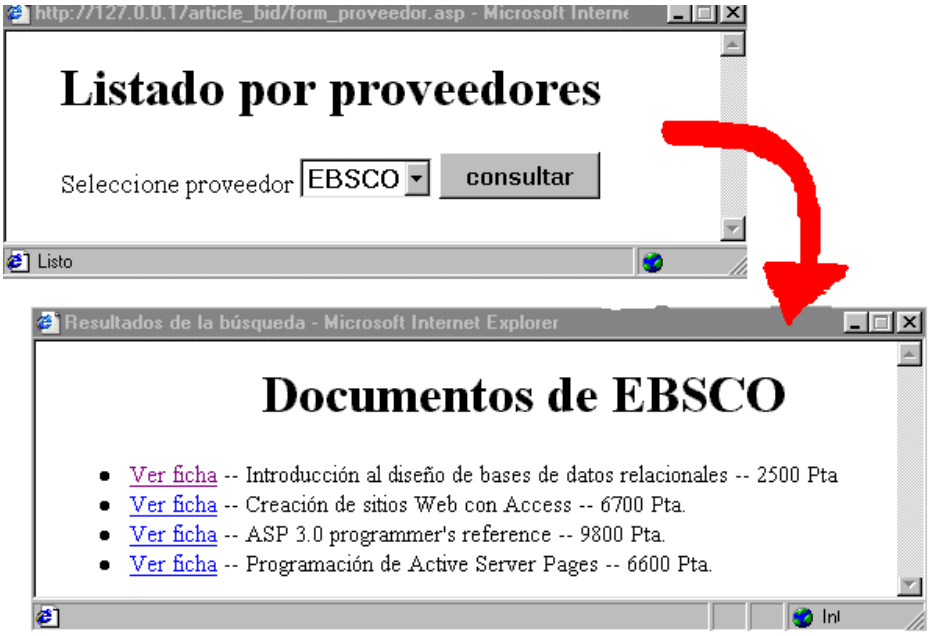

Figura 14. Aspecto de las pantallas del listado por proveedores

Ver formulario

Los usuarios más avanzados pueden usar Javascript para sumar los costes de los documentos.

Captura de desideratas / buzón de sugerencias

Nos resistimos a finalizar este conjunto de ejemplos sin mostrar al lector un ejemplo de administración de bases de datos desde Web, es decir permitir al usuario añadir registros a la base desde el propio navegador. En una biblioteca el ejemplo pudiera ser el libro de sugerencias o de reclamaciones; en nuestro ejemplo usaremos un formulario para recoger la opinión o comentarios del lector sobre este artículo. Los datos del formulario quedarán recogidos en una base "opinion.mdb" que contiene una tabla con cinco campos (día, hora, nombre, correo, comentario). El sistema simplemente consta de un formulario con tres campos que es enviado a un fichero "opinion.asp" similar a los descritos. El código relevante del mismo es:

```
<%
set cn=Server.createObject("ADODB.Connection")
cn.ConnectionTimeout=20
cn.open "DSN=opinion;DriverId=25;FIL=MS Access;MaxBufferSize=512;PageTimeout=5;"
set rs=Server.CreateObject("ADODB.Recordset")
rs.Open "comentarios", cn, 1, 2, 2
rs.addnew
rs.fields("dia")= date
rs.fields("hora")= time
rs.fields("nombre")=request.form("nombre")
rs.fields("correo")=request.form("correo")
rs.fields("comentario")=request.form("comentario")
rs.update
%>
```

Figura 15. Fragmento del código fuente de "opinion.asp"

Observemos que después de abrir la tabla comentarios (sin necesidad de seleccionar registros) simplemente tenemos un método rs.addnew para añadir un registro en blanco seguido de unas instrucciones para llenar los campos con los valores del formulario y un método rs.update para actualizar realmente el registro. Las constantes date y time sirven para llenar los campos respectivos con la fecha y hora en que se produce la actualización.

Observemos igualmente que el método rs.Open incluye unos parámetros finales especiales que permiten usar la base de datos para añadir registros (véase http://www.asp-magazine.com/fr/blitz/bdf5.asp).

Registrar su opinión

Ver las opiniones registradas

El formulario para listar las opiniones registradas es similar a "listado.asp" descrito anteriormente.

Conclusiones

Llegados a este punto podemos intuir que las posibilidades enlazando tablas y formularios son prácticamente infinitas y que, en muchas ocasiones, lo importante en el diseño de los procesos no es la técnica sino la imaginación, las ideas y el conocimiento de la materia. En este artículo se han expuesto cuatro nociones básicas de técnica y algunas ideas o soluciones prácticas. Si el lector pertenece al mundo de la biblioteconomía es posible que encuentre la técnica excesivamente complicada mientras que los procesos expuestos lógicos e intuitivos. Por el contrario, un informático acostumbrado a la programación en C++, u otros sistemas casi seguro que habrá considerado las nociones técnicas muy elementales mientras que no necesariamente habría imaginado las posibles aplicaciones. Tal vez ése es el problema de muchas de las soluciones existentes en el mercado; por un lado, programas creados por expertos informáticos que poco conocen del mundo de las bibliotecas y por otro, la ausencia de una formación informática básica en el bibliotecario que le impide la autosuficiencia. La solución se basa en la colaboración. Sólo si técnica y conocimiento de la materia se unen podremos obtener unos resultados prácticos para nuestros fines.

Bibliografía

Anderson R, Denault D, Francis, B. (2000). *ASP 3.0 programmer's reference*. ISBN 1-86100-323-4.

Bobadilla Sancho, J [et al.] (1999) . HTML Dinámico, ASP y JavaScript a través de ejemplos. Madrid: Ra-ma. 511 p.

Codina, L. “Bases de datos relacionales: qué son y qué aportan a la gestión documental”. *Information World en español*, núm. 29 (1994), p. 18-19.

Cornell, G (1998). Aprenda Microsoft Visual Basic Script ya. Madrid: McGraw-Hil. 290 p. ISBN 84-481-2004-3.

Figuerola C.G, Alonso Berrocal J.L, Zazo Rodríguez A.F. “Disseny d'un motor de recuperació de la informació per a ús experimental i educatiu”. *BiD,* núm 4, (juny 2000). [en línea] <http://www.ub.edu/bid/04figue1.htm> [consulta 14 Agosto 2000]

Gonzalez Moreno, O. (1997). *VBScript y programación ASP, guía practica para usuarios*.

Hillier, S. (1998). Programación de Active Server Pages. Madrid: MacGraw-Hill; Interamericana de España. ISBN 84-481-1466-3.

Hobuss, James J.(1998). Creación de sitios Web con Access. Madrid: Prentice Hall. 497 p. ISBN 84-8322-131-4.

Kilcullen M. “Publishing a newspaper index on the World Wide Web using Microsoft Access 97”. *Indexer,* vol. 20, núm. 4 (1997) p. 195-6.

Pereda García A, Cruz Álvarez D. “Acceso a bases de datos remotas con Microsoft RDO”. *RPP: Revista Profesional para programadores*, vol. 3, núm 7, (1996) p. 41-44.

Salse, M. “Limitacions i possibilitats de Microsoft Access en la gestió documental”. *BiD,* núm. 2 (març 1999) [en línea] <http://www.ub.edu/bid/02salse.htm> [consulta 14 Agosto 2000]

Trigueros Díaz, J. L.; Higuera Matas, R. “Bases de datos relacionales versus bases de datos documentales”. *Boletín de la Asociación Andaluza de Bibliotecarios*, vol. 13, núm. 49, (1997) p. 43-57.

Glosario

En el siguiente apartado se presentan las definiciones de los principales términos empleados a lo largo del artículo así como direcciones de manuales o información accesible en Internet.

ActiveX

Conjunto de tecnologías de interoperabilidad creadas por Microsoft, independientes del lenguaje que permiten que distintos componentes de software escritos en diferentes lenguajes funcionen juntos en entornos de red. Los elementos fundamentales de la tecnología ActiveX son el Modelo de objetos componentes (COM) y el Modelo de objetos componentes distribuido (DCOM).

ADO

ActiveX Data Objects. Un conjunto de interfaces de acceso a datos basadas en objetos optimizadas para las aplicaciones basadas en Internet y centradas en datos. ADO está basado en una especificación publicada y se incluye con Microsoft Internet Information Server y con Microsoft Visual InterDev.

ASP

Active Server Pages. Páginas que combinan secuencias de comandos ActiveX que se ejecutan en el servidor con secuencias HTML u otros componentes permitiendo crear aplicaciones basadas en Web.

Podemos encontrar manuales e información de de ASP en la red en http://webtech.metropoli2000.com/asp/index.html, http://www.montreat.edu/docs/aspdocs/roadmap.asp, http://www.learnasp.com, http://www.stardeveloper.com/databases.asp y algunas revistas electrónicas gratuitas de ASP en http://www.asptoday.com y http://www.asp-magazine.com.

CGI

Common Gateway Interface. Dícese de aquellos programas diseñados para interaccionar directamente con un servidor de Web. Un programa convencional recibe los datos de entrada desde el teclado u otro dispositivo de entrada y envía los resultados a la interfaz Windows. Un programa CGI recibe los datos de entrada desde el navegador y devuelve la respuesta al propio navegador.

IDC

Internet Database connector. IDC es un componente básico del servidor de web Internet Information Server que proporciona conectividad de bases de datos entre las aplicaciones de IIS y las bases de

datos compatibles con ODBC.

ODBC

Open Database Connectivity. Una interfaz de programación que permite a las aplicaciones tener acceso a datos desde diversas especificaciones estándar de orígenes de datos para acceso a bases de datos multiplataforma. Los vendedores de bases de datos suministran controladores de bajo nivel en conformidad con la interfaz para ODBC.DLL. Los programadores de aplicaciones hacen llamadas estándar a ODBC.DLL para accesar a cualquier base de datos sin importar su propio formato.

SQL

Structured Query Languaje. Lenguaje de consulta y programación de bases de datos ampliamente utilizado para tener acceso, consultar, actualizar y administrar datos en sistemas de bases de datos relacionales.

Podemos encontrar algunos manuales de SQL en la red en las direcciones http://www.lobocom.es/~claudio/sql.html y http://www.pntic.mec.es/ies2000/iessql.htm.

VBScript

Microsoft Visual Basic Scripting Edition. Subconjunto del lenguaje Microsoft Visual Basic, un intérprete rápido, portátil y ligero para su uso en exploradores de World Wide Web y otras aplicaciones que usan controles ActiveX, servidores de Automatización OLE y subprogramas Java.

Nota: Existen gran cantidad de manuales, incluso en castellano, tanto de ASP como de SQL accesibles de manera gratuita en Internet, Se aconseja emplear algún multibuscador como Copernic (http://www.copernic.com) u otro y buscar secuencias como SQL, ASP junto a términos como curso o manual, en muchos casos encontraremos información sobre cursos presenciales pero también hallaremos manuales electrónicos.

Fecha recepción: 15/09/2000. Fecha aceptación: 2/10/2000.

Articles similars a BiD

- El catàleg d'arquitectura del Centre de Documentació de Projectes d'Arquitectura de Catalunya. Pouplana, Xavier; Poves, Lluís. (2005)
- Més enllà de la usabilitat : característiques mínimes exigibles per a les interfícies de bases de dades web. Rodríguez Yunta, Luis; Giménez Toledo, Elea. (2004)
- RCLIS : cap a una biblioteca digital de biblioteconomia i documentació. Barrueco Cruz, José Manuel; Subirats Coll, Imma. (2003)
- Revistes font en el SCI-E. Badia Segura, Maria; Alonso, José; Juan, Joan. (2003)
- Distribució de bases de dades en el web amb Knosys Internet. Abadal, Ernest; Martínez Rivas, Raúl. (2000)

Articles similars a Temària

- Bases de datos documentales en el web : análisis del software para su publicación. Abadal, Ernest. (2005)
- Fuentes de información para el análisis de la publicidad : las bases de datos publicitarias en Internet. Malalana Ureña, Antonio. (2004)
- Apuntes para una historia de la documentación deportiva. Arquesolo Vegas, José. (2000)
- Una propuesta metodológica para el diseño de bases de datos documentales (parte II). Codina, Lluís. (1997)
- Automatización de la información clínica odontológica : un proyecto de software íntegramente español. Sánchez Hernández, María F.. (1996)

Articles del mateix autor a Temària

Rodríguez Gairín, Josep Manuel

Facultat de Biblioteconomia i Documentació
Universitat de Barcelona
Barcelona, desembre de 2000
http://www.ub.edu/biblio • Comentaris

Citació recomanada • Metadades
UB • Facultat • BiD