

UNIDAD 2: Bases de Datos en Visual Basic

2. Conocer los controles de acceso a datos: Data Manager, textbox, label, dbgrid.

Bases de Datos en Visual Basic 6.0

Visual Basic versión 6 puede conectarse a base de datos de diferentes tipos utilizando la tecnología OLE DB que permite trabajar con el entorno de desarrollo de la misma manera con distintos motores de bases de datos. Más específicamente permite trabajar con el motor de base de datos Microsoft Jet, que es el mismo que utiliza Microsoft Access.

El motor controla las tres funciones básicas de base de datos:

1. Almacenamiento de datos
2. Recuperación de datos
3. Actualización de datos

O lo que más comúnmente se conoce como ABM (altas, bajas y modificaciones) y también las consultas. Asimismo proporciona una interfaz de programación para datos orientada a objetos que se llama DAO (Data Access Object).

Visual Basic ya en su versión 5 utilizaba dos conceptos importantes: DAO (Data Access Object) y RDO (Remote Data Object). **DAO** es el conjunto de objetos que permiten acceder a una base de datos local y administrarla utilizando los controladores correspondientes como el Motor Jet de Microsoft. En cambio para la conexión a bases de datos remotas utilizaba otro conjunto de objetos llamado RDO (Remote Data Access).

Con la Versión 6 aparece ADO (ActiveX Data Object). **ADO** que reemplaza a las dos anteriores y está formada por un conjunto de objetos que pueden conectarse a bases de datos prácticamente de cualquier tipo (ya sea en forma directa o mediante un origen de datos ODBC). En esta versión 6, por razones de compatibilidad sigue estando presente DAO.

Como ya expresáramos, Visual Basic versión 6.0 puede conectarse a bases de datos de diferentes tipos utilizando la tecnología **OLE DB** haciendo que el entorno de desarrollo pueda utilizarse independientemente de cuál sea el DBMS de la base de datos. Es decir, es independiente del tipo de base de datos.

DAO

El modelo DAO es una colección de clases de objetos que modelan la estructura de un sistema de base de datos relacional. Proporciona las propiedades y los métodos necesarios para poder llevar a cabo las tareas de administración del sistema de bases de datos relacional. Obviamente se basa en SQL que es el lenguaje que permite definir y manipular los datos de las bases de datos relacionales.

La programación está simplificada en cuanto al código a escribir por cuanto básicamente consiste en la creación de objetos de acceso a datos.

DAO es importante porque brinda flexibilidad e integración ya que los mismos objetos sirven para distintos motores, cuestión importante desde el punto de vista del diseño lógico.

Entonces en Visual Basic los mismos objetos de acceso a datos utilizados en una aplicación tienen la capacidad de ser utilizados para acceder a distintas bases de datos. Por ejemplo si se usa Microsoft Access se puede utilizar una base de datos SQL Server u Oracle efectuando unos pocos cambios en el código.

Existen tres tipos de bases de datos que Visual Basic puede reconocer utilizando DAO:

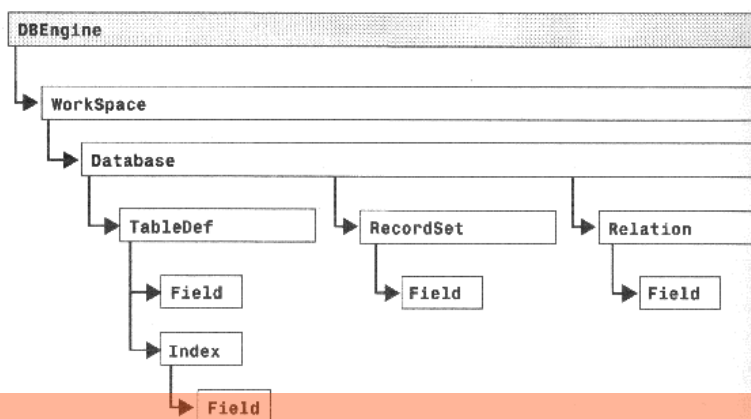
- Bases de datos de Visual Basic: también llamadas bases de datos nativas y que utilizan el mismo formato de Microsoft Access (extensión .mdb);
- Bases de datos externas: son bases de datos populares y que utilizan ISAM (Método de Acceso Secuencial Indexado), por ejemplo Fox Pro, dBase III, dBase IV, etc. Todos estos formatos pueden crearse o manipularse desde Visual Basic y también se puede tener acceso a bases de datos de archivos de texto y hojas de cálculo como Excel;
- Bases de datos ODBC: incluyen las bases de datos cliente servidor que comunican con esta

estandar, por ejemplo SQL Server, ORACLE, etc.

La siguiente imagen muestra cómo está organizada jerárquicamente DAO:

A simple vista podemos ver que unos objetos pueden contener a otros objetos. Por ejemplo un workspace puede contener a una o más bases de datos y un objeto Database puede contener uno o más objetos TableDef (definición de tablas). Es simple: si no tengo un Workspace, no tendré una base de datos.

En el tope de la jerarquía tenemos el objeto DBEngine que es el único objeto que no es una colección y no está contenido en ningún otro. Representa al motor de la base de datos.



Una característica importante de Visual Basic es que tiene propiedades y métodos que pueden separarse como se hace en DML y DDL. En una palabra el Lenguaje de Definición de Datos consta de métodos y propiedades que permiten definir y crear objetos como la misma base de datos, las tablas, etc. El Lenguaje de Manipulación de Datos provee propiedades y métodos que nos permiten operar con la base de datos, como ser: consultas, inserciones, actualizaciones, etc.

Lo más importante de todo es que tiene en cuenta SQL y por ende se puede utilizar el Entorno de Desarrollo independientemente del tipo de motor de base de datos.

Por lo anterior, Utilizar DDL es totalmente posible en VB, sin embargo nosotros trabajaremos de otra manera: diseñaremos y crearemos las bases de datos en Access y luego crearemos las aplicaciones para su manipulación; es decir utilizaremos básicamente DML.

DML y Visual Basic

Básicamente DML presupone que la base de datos ya existe y por lo tanto pueden mencionarse como las tareas básicas de manipulación de datos a las siguientes:

- apertura y cierre de bases de datos
- escritura de datos
- búsqueda de datos

Si bien el manejo de la base de datos mediante DML es muy eficiente requiere un gran esfuerzo en cuanto a codificación. Una alternativa más sencilla para manipular datos es utilizar los controles que Visual Basic trae para ello.

Controles de acceso a datos

El manejo de la base de datos mediante DDL y DML es muy eficiente pero es codificación pura; es decir aplicando los métodos y propiedades mediante código de los objetos vistos anteriormente en la jerarquía. Una alternativa más sencilla para manipular datos es utilizar los controles que Visual Basic trae para ello. Visual Basic contiene un control básico para acceso de datos: el control **Data**. Este control Data forma parte de los objetos DAO.

Básicamente nos permite obtener un origen de datos. Para utilizarlo hay que conocer:

1. qué motor de base de datos utilizará (Connect),
2. qué almacén de datos se usará (DatabaseName) y
3. el origen de recursos que utilizará (RecordSource)

En los siguientes puntos daremos un pantallazo de los principales controles de manejo de datos. En DAO el principal es el Control Data.

- **Control Data**
- **Control DBGrid**
- **Control DBCombo**

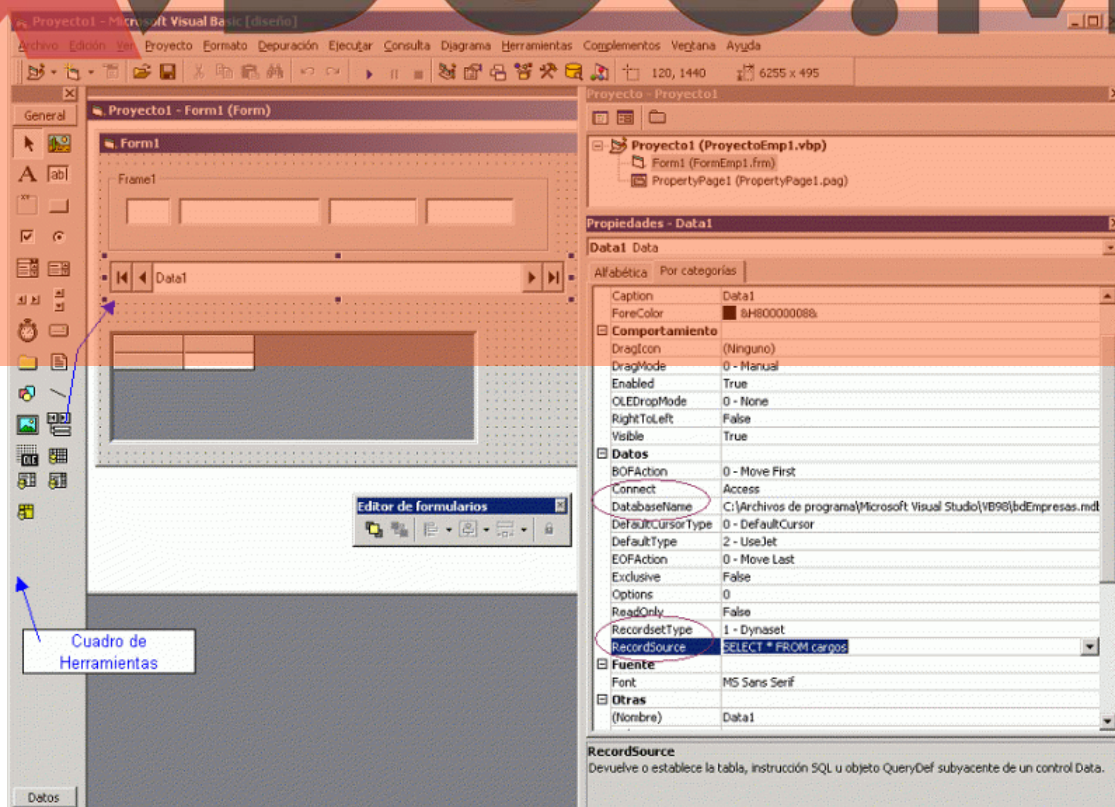
Estos controles específicos para manejo de datos se utilizan con otros que ya conocen como cuadros de texto, botones de comandos, etc. que permiten manipular los datos y eventos dentro de una aplicación. En las prácticas que podrán encontrar en la Sección **Actividades** también encontrarán la teoría pertinente que acompaña a los ejemplos dados por la Cátedra de los siguientes:

- Manejo de Eventos en Visual Basic;
- Utilización de los controles de acceso a datos como el control Data, DBGrid, DBCombo y controles estándar de Visual Basic para captura de datos y o manipulación de los mismos como ser Cuadros de texto, Botones de Comandos, etc.;
- En forma incremental y relacionado con los puntos 3 y 4 siguientes, operaciones que involucran la manipulación de bases de datos: insertar, modificar y eliminar registros; así como la capacidad para realizar consultas;
- El uso y diseño de Arrays de Controles que son muy útiles a la hora de optimizar el código y ordenar el diseño de programación, sobre todo con botones de comandos y cuadros de texto;
- Un adelanto de cómo trabaja Visual Basic con SQL (punto 4).

Es recomendable que el material teórico incluido en cada archivo ZIP de la sección **Actividades** sea abordado con anterioridad al estudio de los distintos Proyectos Visual Basic que están en cada archivo zipeado.

Control Data

El control Data permite que el formulario se vincule a un origen de datos que puede ser una tabla, una vista o una consulta. En la siguiente imagen se muestra un ejemplo práctico de este control.



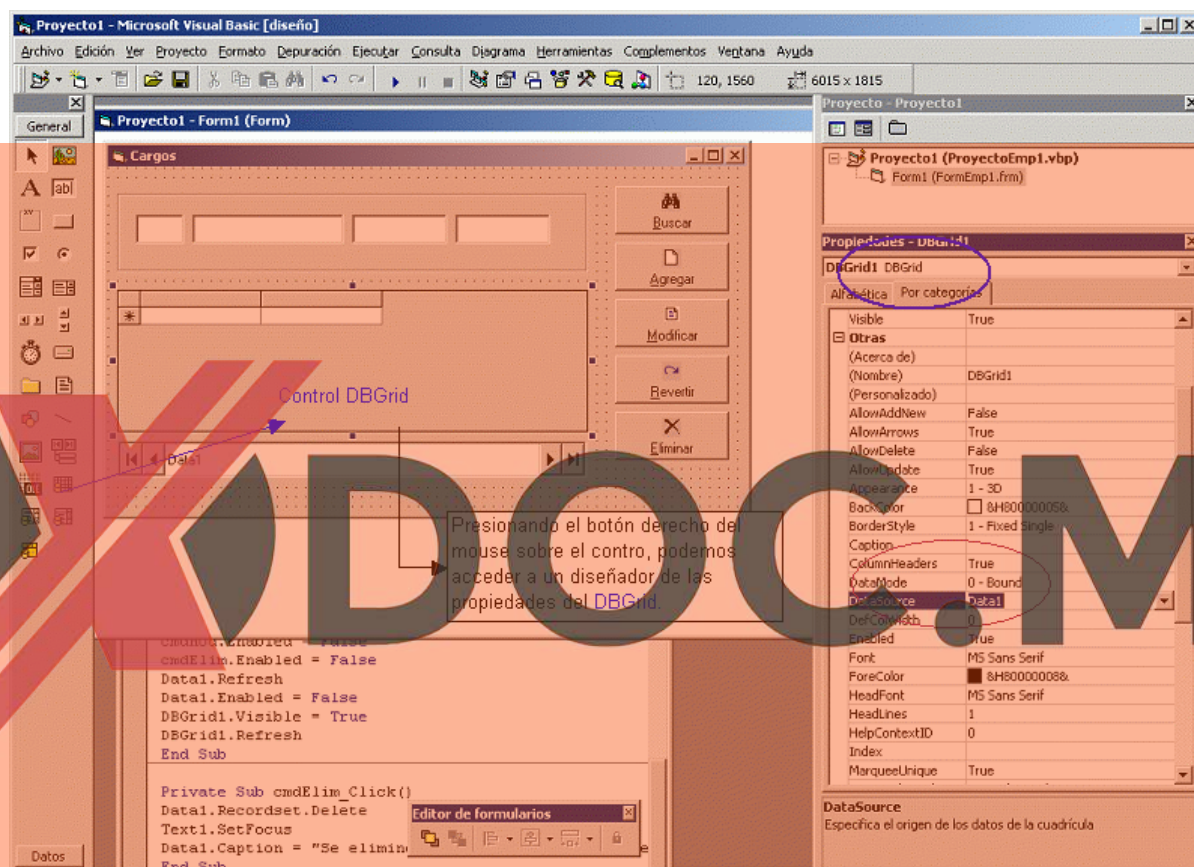
Como puede apreciarse el origen de datos es una consulta mediante selección: `SELECT * FROM cargos`.

En la sección **Actividades** encontrará el **Uso del control Data** que es un archivo ZIP que incluye el Proyecto Cargos (VB) con material teórico complementario en formato PDF.

Los cuatro ejemplos incluidos abordan un distinto nivel de complejidad, incluyendo las operaciones de inserción, actualización y eliminación de registros; así como el manejo y diseño de Arrays de Controles (Unidad 5) y si bien es un adelanto en el programa el manejo de este tema permite diseñar mejor los procedimientos y evitar duplicaciones innecesarias de código.

Control DBGrid

El control DBGrid permite mostrar datos de consultas o tablas.



Es el mismo formulario anterior sólo que hemos agregado el Control DBGrid y unos botones. Como puede apreciarse el origen de datos es el control Data Data1.

Una ejecución del formulario dará una idea acabada de su funcionalidad:

El objetivo es tener una pantalla para poder manipular la tabla Cargos, es decir poder realizar Altas, Bajas y Modificaciones. Un alta no sería problema ya que el Control Data permite directamente generar un nuevo registro que si no se completa se elimina. Además como la tabla tiene restricciones de integridad, en lo que hace a la base de datos, no es necesario ingresar controles de validación.

Si es importante, y lo haremos a medida que avancemos que los datos de la tabla o consulta actualizable no se modifiquen porque sí, ante un error del usuario, por ejemplo.

En las dos siguientes imágenes podemos ver que si nos desplazamos por el DBGrid se modifican los valores de los cuadros de Texto:

Cargos

1 GCIA GRAL 5000 15000

Buscar

Agregar

Modificar

Revertir

Eliminar

idcargo	nbrecargo	minsal	maxsal
1	GCIA GRAL	5000	15000
2	GTE ADMINIST	4000	12000
3	GTE FINANZAS	4000	12000
4	GTE COMERC	4000	12000
5	GTE COMPRAS	4000	12000
10	JEFE CONTAB	2000	8000

Inicio de Sesión

Cargos

4 GTE COMERC 4000 12000

Buscar

Agregar

Modificar

Revertir

Eliminar

idcargo	nbrecargo	minsal	maxsal
1	GCIA GRAL	5000	15000
2	GTE ADMINIST	4000	12000
3	GTE FINANZAS	4000	12000
4	GTE COMERC	4000	12000
5	GTE COMPRAS	4000	12000
10	JEFE CONTAB	2000	8000

4 / 18

En esta imagen se puede apreciar cómo se modifican los datos, sin utilizar ningún botón de comando, directamente accediendo al DBGrid.

Cargos

4 GTE COMERC 4000 12000

Buscar

Agregar

Modificar

Revertir

Eliminar

idcargo	nbrecargo	minsal	maxsal
1	GCIA GRAL	5000	15000
2	GTE ADMINIST	4000	12000
3	GTE FINANZAS	4000	12000
4	GTE	4000	12000
5	GTE COMPRAS	4000	12000
10	JEFE CONTAB	2000	8000

4 / 18

Los cuadros de texto para ingresar datos y/o modificar se han deshabilitado para su edición, pero sí muestran los datos.

Se está modificando el valor en la tabla, eso indica el lápiz. Habría que dejar no habilitado el DBGrid como el caso de los cuadros de texto superiores para que no se alterase el valor de la tabla en forma directa.

Cargos

4 GTE 4000 12000

Buscar

Agregar

Modificar

Revertir

Eliminar

idcargo	nbrecargo	minsal	maxsal
1	GCIA GRAL	5000	15000
2	GTE ADMINIST	4000	12000
3	GTE FINANZAS	4000	12000
4	GTE	4000	12000
5	GTE COMPRAS	4000	12000
10	JEFE CONTAB	2000	8000

4 / 18

Puede apreciarse cómo se han modificado los datos directamente al escribir sobre el DBGrid.

Esto suele ser peligroso o no. Depende de para qué queramos utilizar el DBGrid. Si es para una consulta o bien para poder desplazarnos por los datos para un búsqueda, pues ello no resulta del todo conveniente.

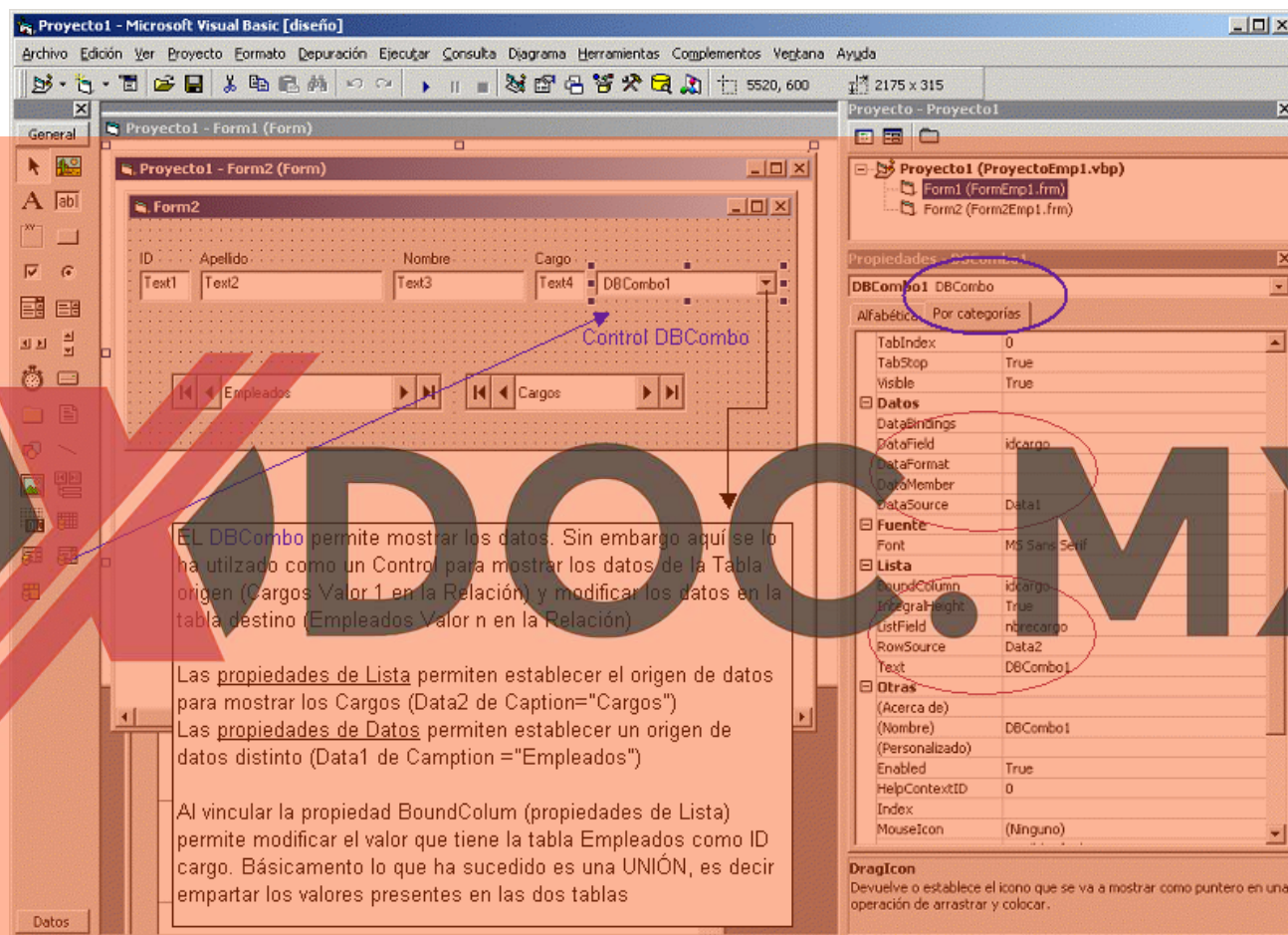
En la sección **Actividades** encontrará el **Uso del control DBGrid** que es un archivo ZIP que incluye el Proyecto DBGrid (VB) con material teórico complementario en formato PDF.

Los ejemplos incluidos abordan un distinto nivel de complejidad, incluyendo temas que tienen que ver con el manejo de SQL en Visual Basic e igualmente el uso y diseño de Arrays de Controles.

Control DBCombo

El control DBCombo básicamente permite seleccionar un valor desde una consulta o una tabla.

Como veremos enseguida, permite mucho más.



Control DBCombo

EL DBCombo permite mostrar los datos. Sin embargo aquí se lo ha utilizado como un Control para mostrar los datos de la Tabla origen (Cargos Valor 1 en la Relación) y modificar los datos en la tabla destino (Empleados Valor n en la Relación)

Las propiedades de Lista permiten establecer el origen de datos para mostrar los Cargos (Data2 de Caption="Cargos")

Las propiedades de Datos permiten establecer un origen de datos distinto (Data1 de Caption="Empleados")

Al vincular la propiedad BoundColumn (propiedades de Lista) permite modificar el valor que tiene la tabla Empleados como ID cargo. Básicamente lo que ha sucedido es una UNIÓN, es decir emparar los valores presentes en las dos tablas

Propiedades de DBCombo1

Propiedad	Valor
TabIndex	0
TabStop	True
Visible	True
DataField	idcargo
DataFormat	
DataMember	
DataSource	Data1
Font	MS Sans Serif
BoundColumn	idcargo
IntegralHeight	True
ListField	idcargo
RowSource	Data2
Text	DBCombo1
Otras	
(Acerca de)	
(Nombre)	DBCombo1
(Personalizado)	
Enabled	True
HelpContextID	0
Index	
MouseIcon	(Ninguno)

En la sección **Actividades** encontrará el **Uso del control DBCombo** que es un archivo ZIP que incluye el Proyecto DBCombo (VB) con material teórico complementario en formato PDF.

Los ejemplos incluidos abordan un distinto nivel de complejidad, incluyendo el manejo y diseño de Arrays de Controles, operaciones de inserción, actualización y eliminación de registros; vinculación de más de dos controles Data a un mismo formulario, etc.