

## VISUAL BASIC – Diseño de Formularios y manipulación de Datos - Proyecto DBCombo

Este proyecto nos mostrará cómo manipular un DBList o un DBCombo. En los ejemplos se ha utilizado el control DBCombo porque tiene una mayor flexibilidad que el DBList.

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

Como veremos enseguida, permite mucho más, pero empezaremos nomás la primera diferencia que presenta con el control DBList es que permite ingresar un texto, que por ejemplo permitiría efectuar una búsqueda.

Otra capacidad que tienen ambos controles es que permite:

1. **Autollenado** porque se asocian a un campo de la tabla o consulta a la cual está vinculado el control Data (propiedad RecordSource). Estos controles para vincularse al control Data utilizan la propiedad RowSource
2. **Actualización automática**: se enlaza el registro seleccionado en el DBList o DBCombo a un campo específico del objeto Recordset administrado por el control Data.

La principal consecuencia de esto es que al seleccionar un registro en un DBList o en un DBCombo podemos actualizar los datos de otra tabla porque pueden trabajar con dos controles Data: uno para leer los datos origen de la tabla que utilizo como de solo lectura y otro para actualizar una tabla que necesita el valor de la primera. En una palabra el DBCombo es un control ideal vincular y mantener integridad de las claves foráneas..

En la sección teórica tenemos un buen ejemplo:

**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 emparter los valores presentes en las dos tablas

Propiedades - DBCombo1	
Alfabetico	Por categorias
TabIndex	0
TabStop	True
Visible	True
<b>Datos</b>	
DataBindings	
DataField	idcargo
DataFormat	
DataMember	
DataSource	Data1
<b>Fuente</b>	
Font	MS Sans Serif
<b>Lista</b>	
BoundColumn	idcargo
IntegralHeight	True
ListField	nbrecargo
RowSource	Data2
Text	DBCombo1
<b>Otras</b>	
(Acerca de)	
(Nombre)	DBCombo1
(Personalizado)	
Enabled	True
HelpContextID	0
Index	
MouseIcon	(Ninguno)
<b>DragIcon</b>	
Devuelve o establece el icono que se va a mostrar con operación de arrastrar y colocar.	

Vemos dos conjuntos de propiedades fundamentales en el DBCombo superior. Las que tienen que ver con Datos y las que tienen que ver con Listas. Las propiedades de Lista tienen que ver con lo que ve el usuario para poder seleccionar una opción válida y las de Datos tienen que ver con lo que uno quiere hacer. En el primer caso se utilizó un control Data llamado Data2 y en el caso del segundo grupo de propiedades el Data1.

Para que se pueda utilizar eficientemente es necesario comprender las propiedades que intervienen como son:

Nombre de las propiedades	Lista	Datos
<b>RowSource</b> [origen de las filas ó ítems de la lista]: vincula el DBList o DBCombo al control Data asociado para leer los datos de los registros y completar la lista del control DBList o DBCombo  <b>ListField</b> [campo con que llena la lista]: toma un campo del RowSource (uno o más si se efectúa una sentencia SQL es posible agregar más de una columna)  <b>BoundColumn</b> [columna vinculante entre las dos tablas (o sea los dos controles Data)]: esta propiedad es fundamental para poder aplicar en la práctica lo conceptual referido a las claves foráneas	RowSource =Data2 [El Data2 vincula a la tabla Cargos]  ListField=nbrecargo  BoundColumn= idcargo	
<b>DataSource</b> [origen de los datos]:  <b>DataField</b> [campo de datos que se actualiza]:		DataSource=Data1 [El Data1 vincula a la tabla Empleados en el ejemplo] En la tabla Empleados el atributo idgerente es la clave foránea que se vincula a la clave primaria idcargo de la tabla Cargos.  DataField=idcargo Como expresamos arriba lo que queremos hacer es actualizar Empleados.idcargo leyendo directamente Cargos.idcargo
<b>Conclusión:</b> el control Data que utilizemos para leer los datos es el que se vincula con la columna que representa la clave primaria en la tabla de lectura (en este ejemplo la tabla Cargos) y que actualiza los valores a través del otro control data en la tabla que tiene la clave foránea. Por lo tanto es fundamental entender todo lo relativo a normalización y a diseño de bases de datos para poder utilizar este control en forma adecuada		

Otra propiedad muy importante pero más útil con los DBCombos es **MatchEntry** que permite que efectuemos búsquedas en el control en base al ListField. Este tipo de búsquedas son más muy útiles.

**MatchEntry:** devuelve o asigna un valor que indica cómo el control DBCombo o DBList realiza búsquedas basándose en la entrada del usuario. La sintaxis es: objeto.**MatchEntry** [= valor ] / valor es una constante o un valor que define el comportamiento de un control cuando tiene el enfoque y el usuario introduce uno o más caracteres.

Estos valores pueden estar en un intervalo = [0/1]:

- 0 = vbMatchEntrySimple o Coincidencia básica: (Predeterminado): el control busca la siguiente coincidencia del carácter introducido usando la primera letra de entradas de la lista. Al escribir repetidamente la misma letra se recorren todas las entradas de la lista que comienzan por esa letra.
- 1= vbMatchEntryExtended o Coincidencia ampliada: El control busca una entrada que coincida

con todos los caracteres introducidos. La búsqueda se realiza a medida que se escriben los caracteres, refinando progresivamente la búsqueda.

En el primer ejemplo se tratará ampliamente cómo utilizar este control y en el segundo cómo realizar una búsqueda.

### Formulario DBCombo1

En este formulario hemos implementado las operaciones ABM sobre la Tabla Departamentos.

Se han utilizado 3 controles Data y una búsqueda en base al iddepto.

Seguidamente se listan los objetos que intervienen en el formulario y figuras de las propiedades de algunos objetos:

Control	Nombre	Finalidad
Array(commandButton)	cmdABM	Para manejar los eventos de las operaciones ABM
Array(cuadros Texto)	txtDeptos[0-3]	Para mostrar los datos y efectuar las modificaciones y o inserciones
ComandButton	cmdBusq	Botón que maneja la interfaz para realizar las búsquedas
Cuadro de Texto	txtBusq	Para llevar a cabo las búsquedas en base al iddepto
Data	catEmp	Vincula a una consulta sobre la tabla Empleados [ver imagen]
Data	catDepartamentos	Vincula a la tabla Departamentos [recordset de tipo Table para poder realizar las búsquedas con sekk]
Data	datCargos	Vincula a la tabla Cargos [recordset de tipo Dynaset]
DBCombo	DBCboGer	Combo que sirve para buscar los datos de los gerentes vinculándose al datEmp y permite actualizar el valor idgerente en la tabla departamentos (datDepartamentos) [ver imagen]
DBCombo	DBCboDom	Combo que sirve para buscar los datos de los domicilios vinculándose al datDomicilios y permite actualizar el valor iddomi en la tabla departamentos(datDepartamentos) [ver imagen]

Figura del diseño del formulario DBCombo1:

Figura de las propiedades del control Data datEmp:

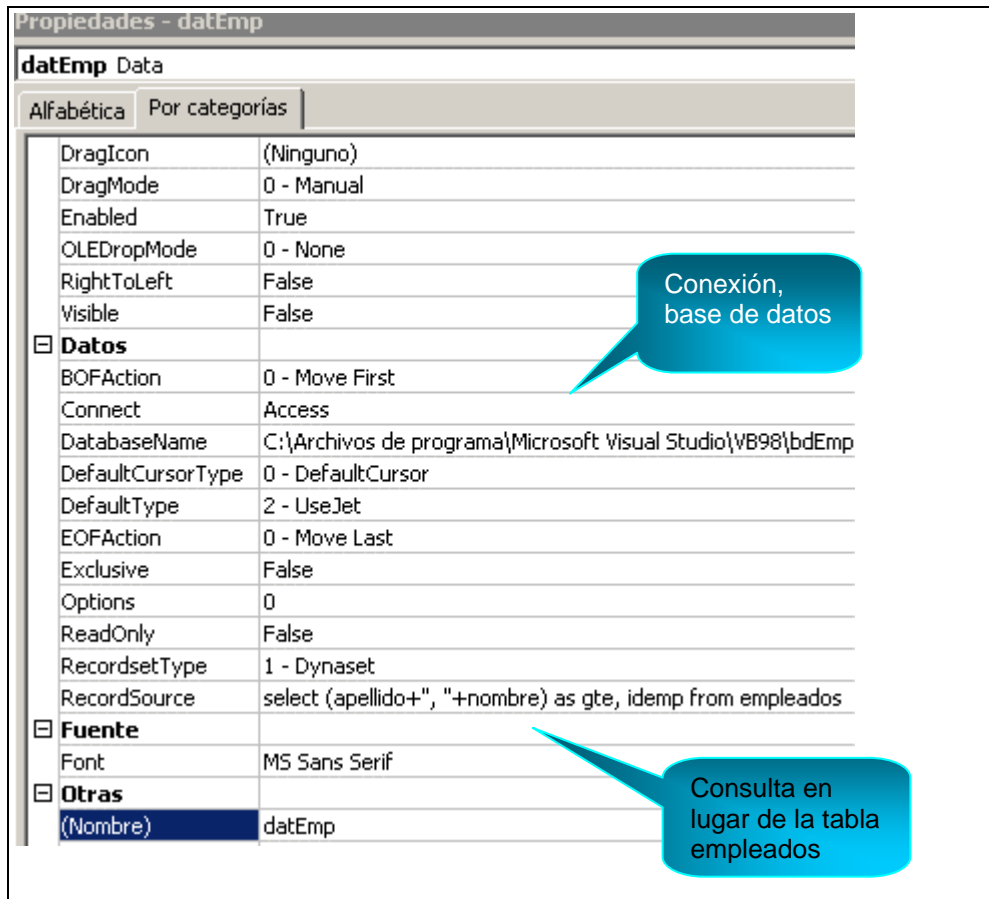
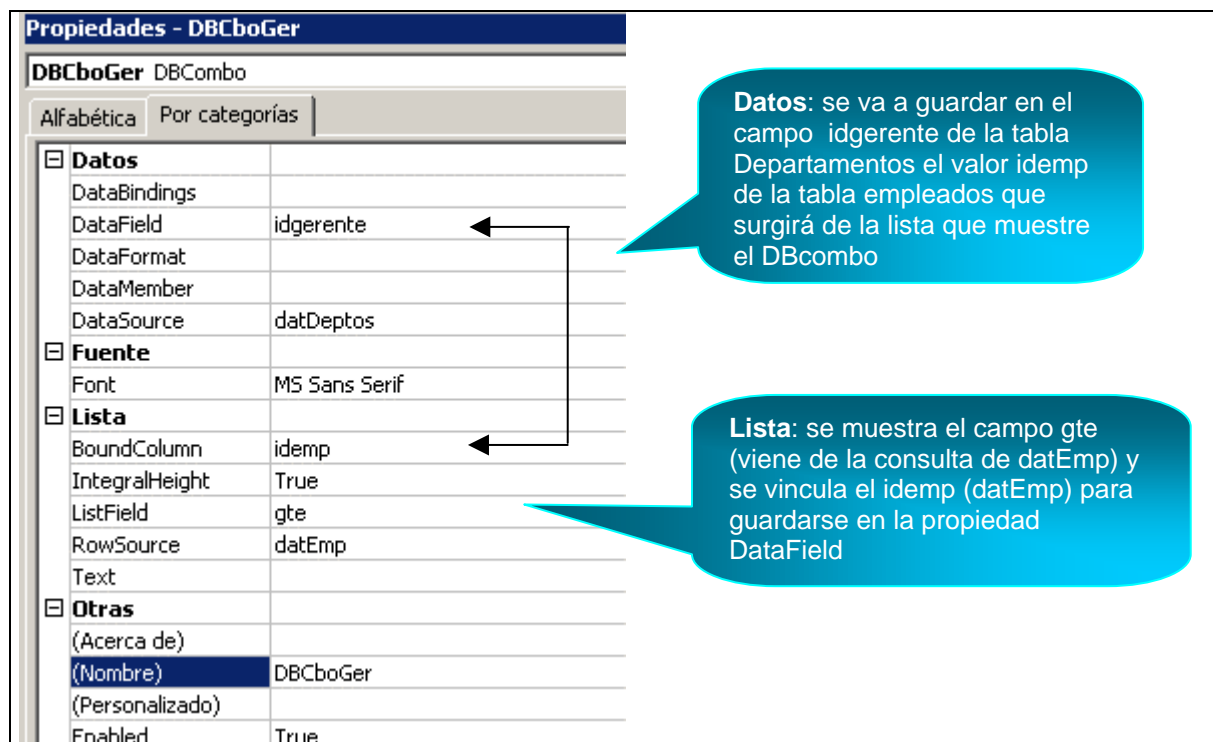
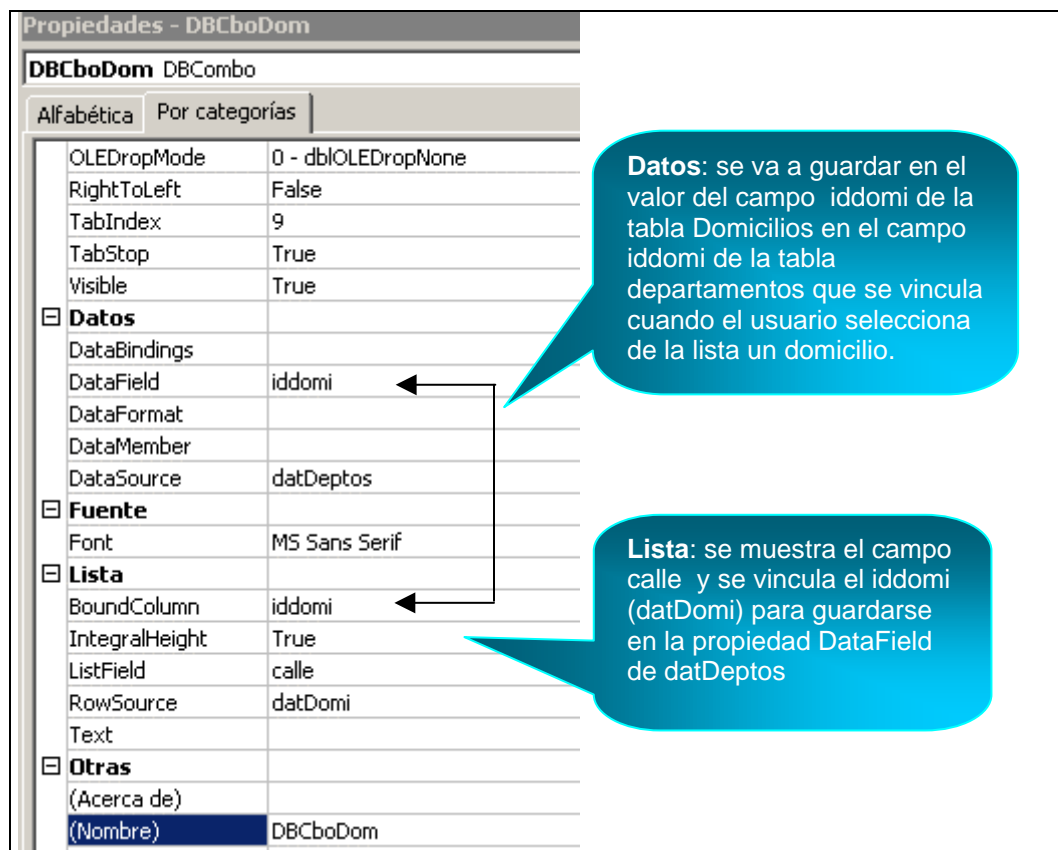


Figura de las propiedades de los controles DBCombo DBCboGer y DBCboDom:





El código fuente se lista más abajo. Algunas consideraciones importantes son que cuando trabajamos con controles Data la base de datos y los recordsets se abren a través de estos controles y se cargan en el formulario en tiempo de ejecución.

```
Option Explicit
Dim i As Integer
Dim j As Integer
Dim intMs As Integer
```

#### Private Sub cmdABM\_Click(Index As Integer)

```
Select Case (Index)
```

```
Case 0:
```

```
txtDeptos(0).Enabled = True
datDeptos.Recordset.AddNew
```

```
With cmdABM
```

```
.Item(0).Enabled = False
.Item(1).Enabled = False
.Item(2).Enabled = False
.Item(3).Enabled = True
.Item(4).Enabled = True
End With
```

```
Case 1:
```

```
With cmdABM
```

```
.Item(0).Enabled = False
.Item(1).Enabled = False
.Item(2).Enabled = False
.Item(3).Enabled = True
.Item(4).Enabled = True
End With
```

```
Case 2:
```

```
datDeptos.Recordset.Delete
```

```
With cmdABM
.Item(0).Enabled = True
.Item(1).Enabled = False
.Item(2).Enabled = False
.Item(3).Enabled = False
.Item(4).Enabled = False
End With
```

Case 3:

```
datDeptos.Recordset.Update
```

```
With cmdABM
.Item(0).Enabled = True
.Item(1).Enabled = False
.Item(2).Enabled = False
.Item(3).Enabled = False
.Item(4).Enabled = False
End With
```

Case 4:

```
datDeptos.Recordset.CancelUpdate
```

```
With cmdABM
.Item(0).Enabled = True
.Item(1).Enabled = False
.Item(2).Enabled = False
.Item(3).Enabled = False
.Item(4).Enabled = False
End With
```

```
End Select
```

```
End Sub
```

#### Private Sub cmdBusq\_Click()

' cuando se va a efectuar una búsqueda el iddepto no puede modificarse \_  
porque ya participa en otras tablas como clave foránea y de lo contrario \_  
se perdería la integridad referencial. Por eso se usa otro cuadro de texto \_  
y se evita así la aparición de errores tipo 3426 o 3421

```
txtBusq.Visible = True
txtDeptos(0).Enabled = False 'se deshabilita para evitar problemas
txtBusq.SetFocus
```

```
End Sub
```

#### Private Sub txtBusq\_BuscarDatos()

```
datDeptos.Recordset.Index = txtBusq.Tag 'se asignó 'iddepto' a esta propiedad
datDeptos.Recordset.Seek "=", txtBusq.Text
```

```
If datDeptos.Recordset.NoMatch Then
```

```
intMs = MsgBox("No existe el Registro", vbInformation, "Información")
datDeptos.Recordset.MoveFirst 'si no existe mueve el puntero al principio
```

```
Else
```

```
intMs = MsgBox("Registro Encontrado", vbInformation, "Información")
```

```
With cmdABM
.Item(0).Enabled = False
.Item(1).Enabled = True
.Item(2).Enabled = True
.Item(3).Enabled = False
.Item(4).Enabled = False
End With
```

```
For i = 0 To txtDeptos.Count - 1
```

```
txtDeptos(i).Text = datDeptos.Recordset.Fields(i).Value 'para capturar los datos
```

```
Next i
```



```
datDeptos.Recordset.Edit
```

```
End If
```

```
End Sub
```

```
'el usuario sólo puede ingresar numeros
```

```
Private Sub txtBusq_KeyPress(KeyAscii As Integer)
```

```
If (KeyAscii >= 48 And KeyAscii <= 57) Or KeyAscii = 13 Then
```

```
    KeyAscii = KeyAscii
```

```
    If (KeyAscii = 13) Then
```

```
        Call txtBusq_LostFocus
```

```
    End If
```

```
Else
```

```
    j = MsgBox("Debe Ingresar sólo valores numéricos", vbCritical, "Error")
```

```
    KeyAscii = 0
```

```
End If
```

```
End Sub
```

```
Private Sub txtBusq_LostFocus()
```

```
    txtBusq.Visible = False
```

```
End Sub
```

```
Private Sub txtBusq_Validate(Cancel As Boolean)
```

```
If txtBusq.Text = "" Or Len(txtBusq.Text) = 0 Then
```

```
    j = MsgBox("Debe ingresar los datos requeridos", vbInformation, "Alerta")
```

```
    Cancel = True
```

```
Else
```

```
    If (Len(txtBusq.Text) > 2) Then
```

```
        j = MsgBox("Debe ingresar hasta dos números", vbInformation, "Alerta")
```

```
        Cancel = True
```

```
    Else
```

```
        Call txtBusq_BuscarDatos
```

```
        Cancel = False
```

```
    End If
```

```
End If
```

```
End Sub
```

```
'así se valida que el usuario ingrese sólo 2 números
```

```
Private Sub txtDeptos_Validate(Index As Integer, Cancel As Boolean)
```

```
Select Case (Index)
```

```
Case 0:
```

```
'primero se verifica que el campo no esté vacío
```

```
'luego se verifica para cada campo a considerar la longitud de la cadena
```

```
If txtDeptos(Index).Text = "" Or Len(txtDeptos(Index).Text) = 0 Then
```

```
    j = MsgBox("Debe ingresar los datos requeridos", vbInformation, "Alerta")
```

```
    Cancel = True
```

```
Else
```

```
    If (Len(txtDeptos(Index)) > 2) Then
```

```
        j = MsgBox("Debe ingresar hasta dos números", vbInformation, "Alerta")
```

```
        Cancel = True
```

```
    Else
```

```
        Cancel = False
```

```
    End If
```

```
End If
```

```
Case 1:
```

```
If txtDeptos(Index).Text = "" Then
```

```

j = MsgBox("Debe ingresar los datos requeridos", vbInformation, "Alerta")
Cancel = True
Else
If (Len(txtDeptos(Index).Text) > 20) Then
j = MsgBox("Debe ingresar hasta veinte cacteres", vbInformation, "Alerta")
Cancel = True
Else
Cancel = False
'sirve para cambiar toda la cadena a mayúsculas
txtDeptos(Index).Text = UCase(txtDeptos(Index).Text)
End If
End If

Case 2, 3:
If (Len(txtDeptos(Index).Text) > 2) Then
j = MsgBox("Debe ingresar un valor no mayor a 9.999.999,99", vbInformation,
"Alerta")
Cancel = True
Else
Cancel = False

End If

End Select
End Sub

```

### Formulario DBCombo2

En este formulario hemos implementado dos DBCombo con el fin de mostrar algunos datos de la tabla Empleados.

Seguidamente se listan los objetos que intervienen en el formulario y figuras de las propiedades de algunos objetos:

Control	Nombre	Finalidad
Array(cuadros Texto)	txtEmple[0-5]	Para mostrar los datos que surgen de la búsqueda
ComandButton	cmdBusq	Botón que maneja la interfaz para realizar las búsquedas
Data	datEmp	Vincula a una consulta sobre la tabla Empleados [ver imagen]
Data	datTablaEmp	Vincula a la tabla Empleados [Recordset de tipo Table para poder realizar las búsquedas con seek]
DBCombo	DBCboEmp	Combo que sirve para buscar los datos de los empleados y mediante el cual se realizan las búsquedas [ver imagen]

En la siguiente imagen se muestra el diseño del formulario.



Propiedades de datTablaEmp:

**Propiedades - datTablaEmp**

**datTablaEmp Data**

Alfabética Por categorías

DragIcon	(Ninguno)
DragMode	0 - Manual
Enabled	True
OLEDropMode	0 - None
RightToLeft	False
Visible	True
<b>Datos</b>	
BOFAction	0 - Move First
Connect	Access
DatabaseName	D:\VB_PCAS\Ejemplos\bdEmpresas.mdb
DefaultCursorType	0 - DefaultCursor
DefaultType	2 - UseJet
EOFAction	0 - Move Last
Exclusive	False
Options	0
ReadOnly	False
RecordsetType	0 - Table
RecordSource	empleados
<b>Fuente</b>	
Font	MS Sans Serif
<b>Otras</b>	
(Nombre)	datTablaEmp
Index	

Propiedades de datEmp:

**Propiedades - datEmp**

**datEmp Data**

Alfabética Por categorías

RightToLeft	False
Visible	False
<b>Datos</b>	
BOFAction	0 - Move First
Connect	Access
DatabaseName	D:\VB_PCAS\Ejemplos\bdEmpresas.mdb
DefaultCursorType	0 - DefaultCursor
DefaultType	2 - UseJet
EOFAction	0 - Move Last
Exclusive	False
Options	0
ReadOnly	True
RecordsetType	1 - Dynaset
RecordSource	select (apellido+", "+nombre) as emple,idemp from empleados order by apellido,nombre
<b>Fuente</b>	
Font	

Propiedades de DBCboEmp:

Propiedades - DBCboEmp	
DBCboEmp DBCombo	
Alfabética Por categorías	
Locked	False
MatchEntry	1 - dbExtendedMatching
OLEDragMode	0 - dbOLEDragManual
OLEDropMode	0 - dbOLEDropNone
RightToLeft	False
TabIndex	12
TabStop	True
Visible	True
<b>Datos</b>	
DataBindings	
DataField	idemp
DataFormat	
DataMember	
DataSource	datEmp
<b>Fuente</b>	
Font	MS Sans Serif
<b>Lista</b>	
BoundColumn	idemp
IntegralHeight	True
ListField	emple
RowSource	datEmp
Text	

El código fuente de este formulario es:

Option Explicit

#### Private Sub Buscar()

```

datTablaEmp.Recordset.Index = DBCboEmp.BoundColumn
'si el valor ingresado no está en la lista no puede buscar el idemp _
entonces no tiene un valor equivalente a numérico sino string
If IsNumeric(DBCboEmp.BoundText) Then
    datTablaEmp.Recordset.Seek "=", DBCboEmp.BoundText
    If datTablaEmp.Recordset.NoMatch Then
        MsgBox ("No Existe el registro")
        datTablaEmp.Recordset.MoveFirst
    End If
End If
End Sub

```

#### Private Sub DBCboEmp\_Click(Area As Integer)

```

If (Area = 2 Or Area = 1) Then
    Call Buscar 'areaConstant es texto o lista
End If
End Sub

```

#### Private Sub DBCboEmp\_KeyPress(KeyAscii As Integer)

```

If KeyAscii = 13 Then
    Call DBCboEmp_Validate(True)
End If
End Sub

```

**Private Sub DBCboEmp\_Validate(Cancel As Boolean)**

```
If (DBCboEmp.Text <> "") Then
    DBCboEmp.Text = UCase(DBCboEmp.Text)
    Call Buscar
    Cancel = False
Else
    Cancel = True
End If
End Sub
```