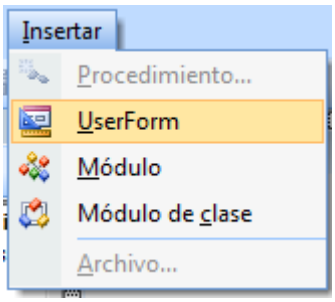
CREANDO FORMULARIOS Y PROGRAMÁNDOLOS

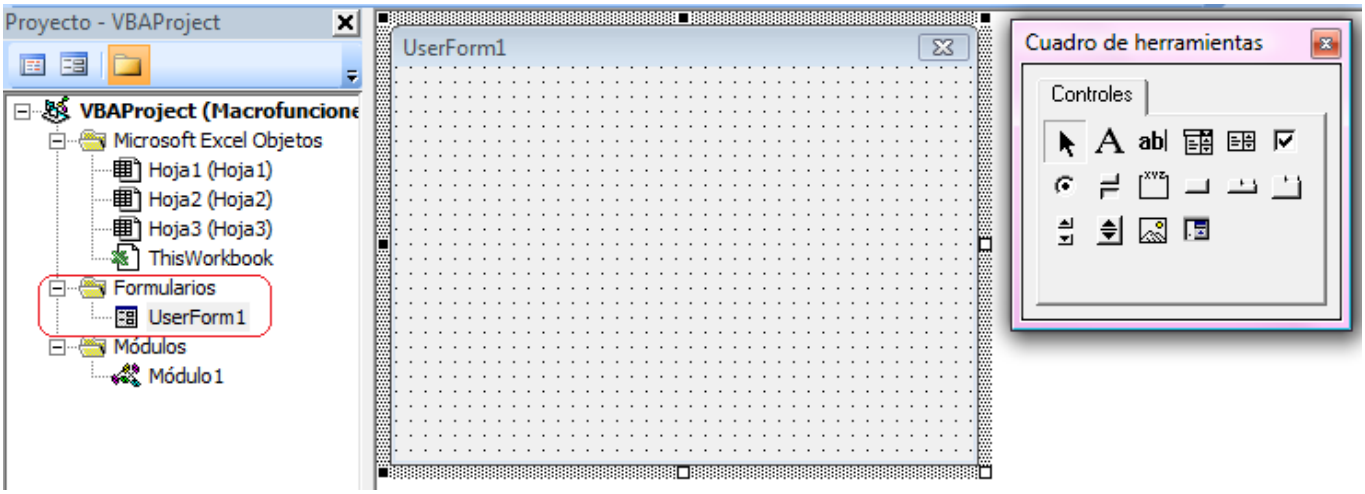
Ahora aprenderemos a dominar lo máximo de Excel que es crear formularios y programarlos, bueno un formulario es una ventana que se programa por medio de controles y estos controles responden a sucesos que nosotros programamos.

Todo esto se encuentra dentro del Editor de Visual Basic. A continuación se muestra como crear un formulario y como programarlo:

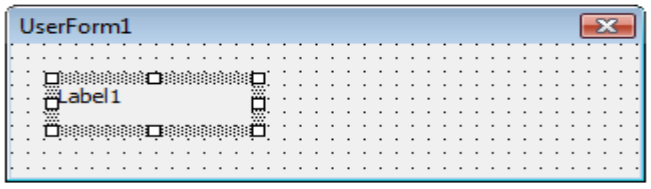
Del **Menú Insertar** elija la Opción **UserForm**, observe:



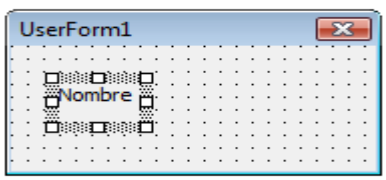
A continuación nos aparece en el explorador la carpeta de formularios y el UserForm que se muestra a la derecha junto con el cuadro de herramientas, observe:



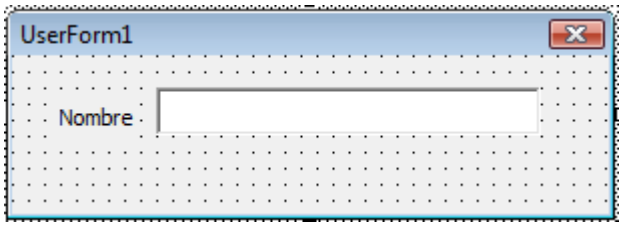
Elija del **Cuadro de Herramientas** el **Control Etiqueta** el que tiene la **A** y Arrastre dibujando en el Formulario **USERFORM1** la etiqueta. Quedara el nombre Label1, observe:



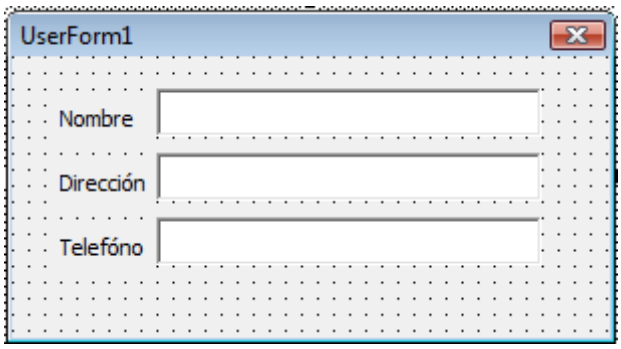
Luego de un clic en la etiqueta dibujada y podrá modificar el nombre de adentro y lo cambiaremos por **Nombre**. Si por error da doble clic en la etiqueta y lo manda a la pantalla de programación de la etiqueta, solo de doble clic en USERFORM1 que se encuentra en el Explorador de Proyecto.



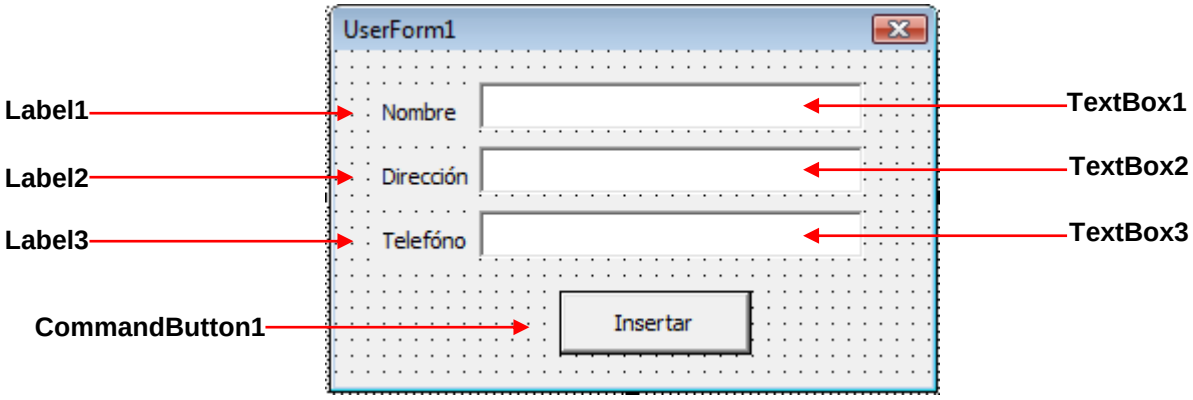
Elija del **Cuadro de Herramientas** el control **Cuadro de Texto** el que tiene **ab|** y arrastre dibujando en el formulario USERFORM1 el cuadro de texto a un lado de la etiqueta que dice Nombre. El cuadro de texto debe de estar vacío y su nombre será Textbox1, el nombre solo aparecerá en el control, observe:



Haga los dos pasos anteriores agregando dos controles de etiqueta y dos cuadros de texto. Esto quedara así después de haberlo hecho:



Por último elija del **Cuadro de Herramientas** el control **Botón de Comando** el que es y arrastre dibujando en el formulario USERFORM1 el botón debe quedar ubicado en la siguiente posición con el nombre Insertar:

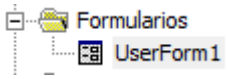


Ahora de doble clic sobre el control **Textbox1** para programarlo y después inserte el siguiente código:

```
Private Sub TextBox1_Change()
Range("A9").Select
ActiveCell.FormulaR1C1 = TextBox1
End Sub
```

Esto indica que se valla a la celda **A9** y escriba lo que hay en el **Textbox1**, es decir cuando ejecute el formulario en Excel en la celda A9 se escribirá el nombre que se digite en el formulario.

Nota: Lo que esta en azul lo genera Excel automáticamente, usted solo escribirá lo que esta en Negrita. Para volver al Formulario y programar el siguiente Textbox de doble clic en **Userform1** que se encuentra en el Explorador de la izquierda:



Ahora de doble clic sobre el control **Textbox2** para programarlo y después inserte el siguiente código:

```
Private Sub TextBox2_Change()
Range("B9").Select
ActiveCell.FormulaR1C1 = TextBox2
End Sub
```

Esto indica que se valla a la celda **B9** y escriba lo que hay en el **Textbox2**.

Volver al **UserForm1** doble clic sobre el control **Textbox3** para programarlo y después inserte el siguiente código:

```
Private Sub TextBox3_Change()
Range("C9").Select
ActiveCell.FormulaR1C1 = TextBox3
End Sub
```

Esto indica que se valla a la celda **C9** y escriba lo que hay en el **Textbox3**.

Para crear un comentario en visual Basic debes hacerlo utilizando comilla simple **‘**

Volver al **UserForm1** doble clic sobre el control **CommandButton1** para programarlo y después inserte el siguiente código:

```
Private Sub CommandButton1_Click()
'inserta un renglón
Selection.EntireRow.Insert
'Empty Limpia Los Textbox
TextBox1 = Empty
TextBox2 = Empty
TextBox3 = Empty
'Textbox1.SetFocus Envía el cursor al Textbox1 para volver a capturar los datos
TextBox1.SetFocus
End Sub
```

Nota: El comando **Rem o ‘** son empleados para poner comentarios dentro de la programación, el comando **Empty** es empleado para vaciar los Textbox, es decir dejarlos en blanco para escribir un nuevo dato.

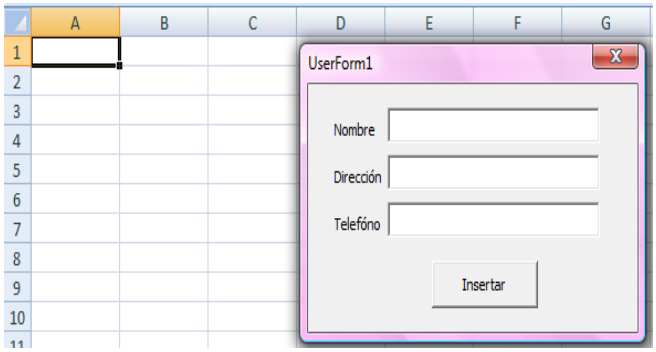
Guarde los cambios para que visual Basic guarde el formulario creado y configurado.

Como Ejecutar el Formulario

Hay 4 formas:

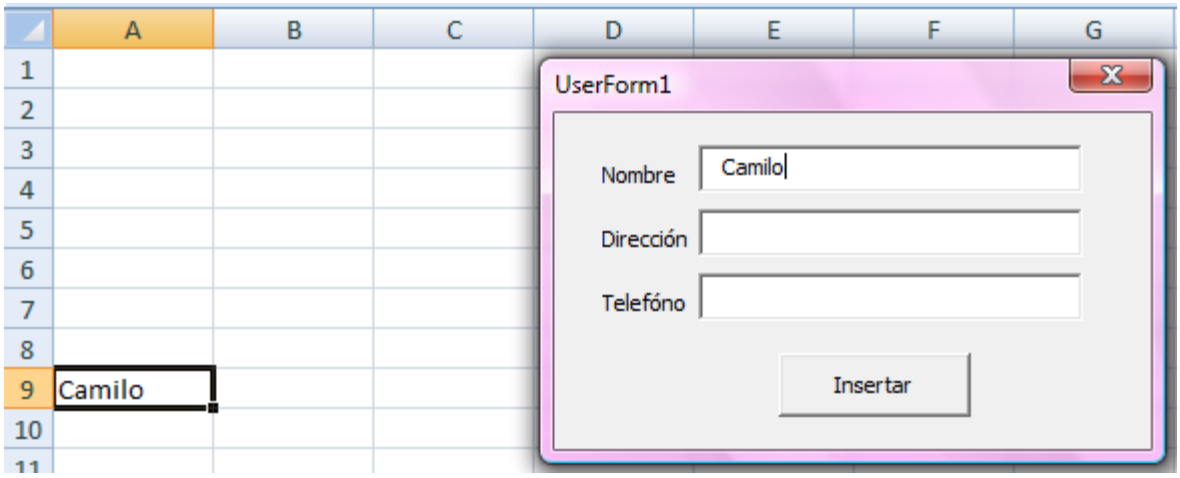
1. Una es pulsando la tecla **F5.**
2. Otra es dar clic en el botón **Ejecutar** .
3. Otra es seleccionar el **Menú Ejecutar** y dar clic en **Ejecutar Sub/UserForm.**
4. La última es ejecutarlo utilizando un botón de formulario en la Hoja de Excel.

Pulsemos la tecla **F5**, y observe que el formulario aparece activado en la hoja de Excel, observe:

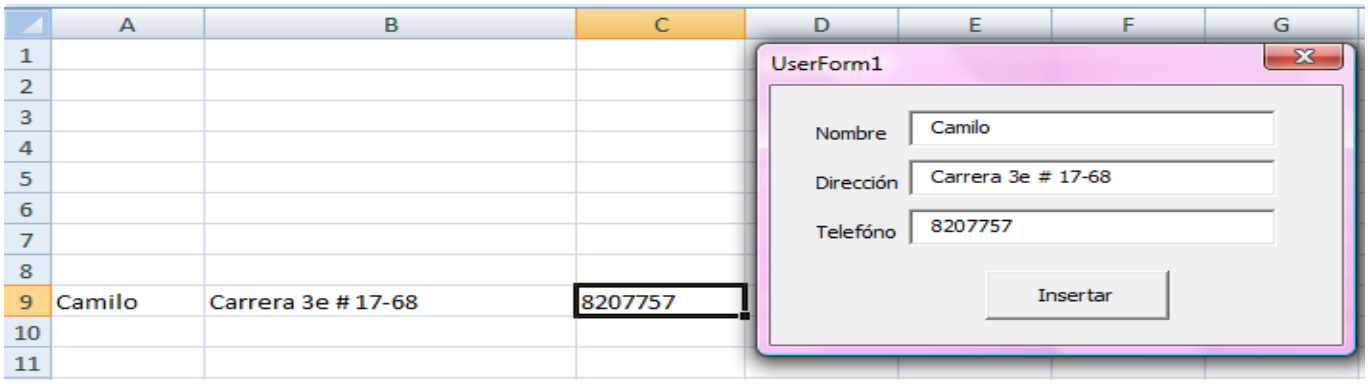


Se activará el **Userform1** y todo lo que escriba en los Textbox se escribirá en Excel y cuando presione el botón Insertar, se insertara un renglón y se vaciaran los Textbox y después se mostrara el cursor en el **Textbox1**.

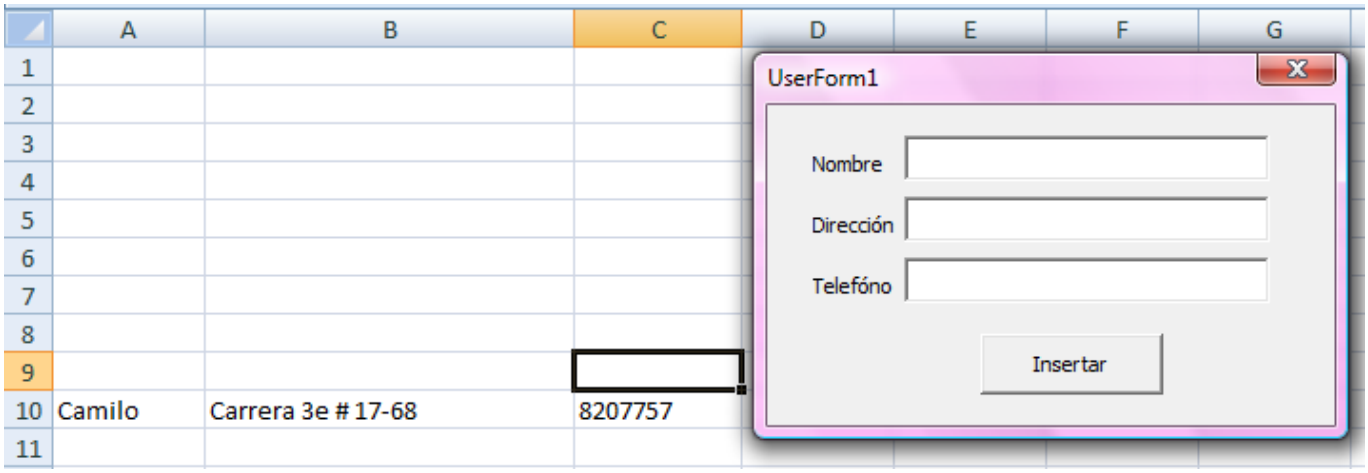
Veamos insertaremos el siguiente registro en el formulario, observe que automáticamente cuando se escribe el nombre en el formulario este se escribe inmediatamente en la celda A9, mire:



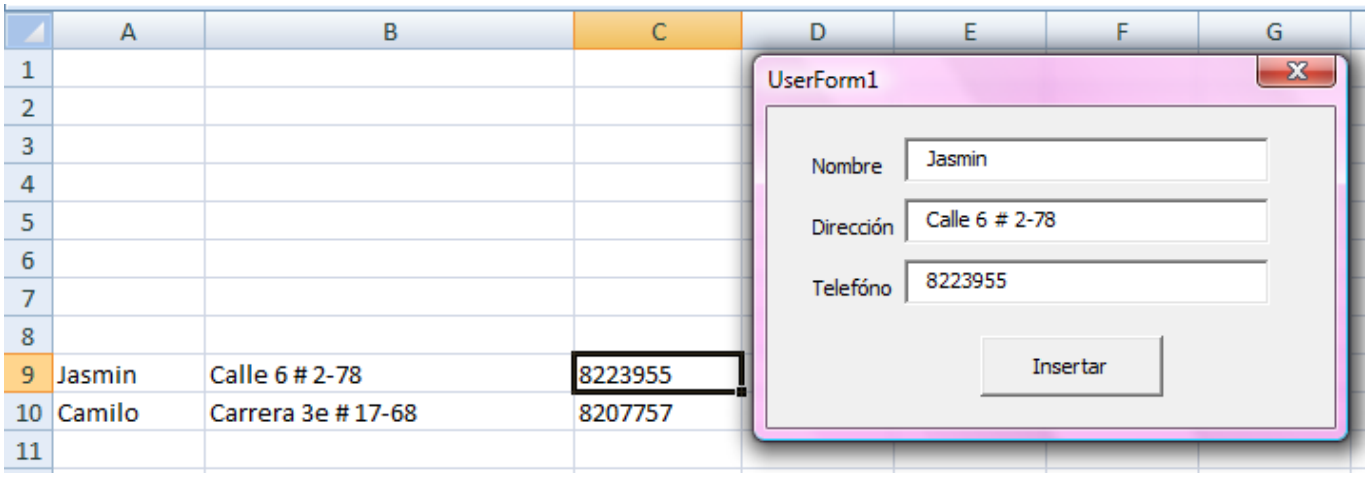
Continuemos con el registro observe como debe quedar:



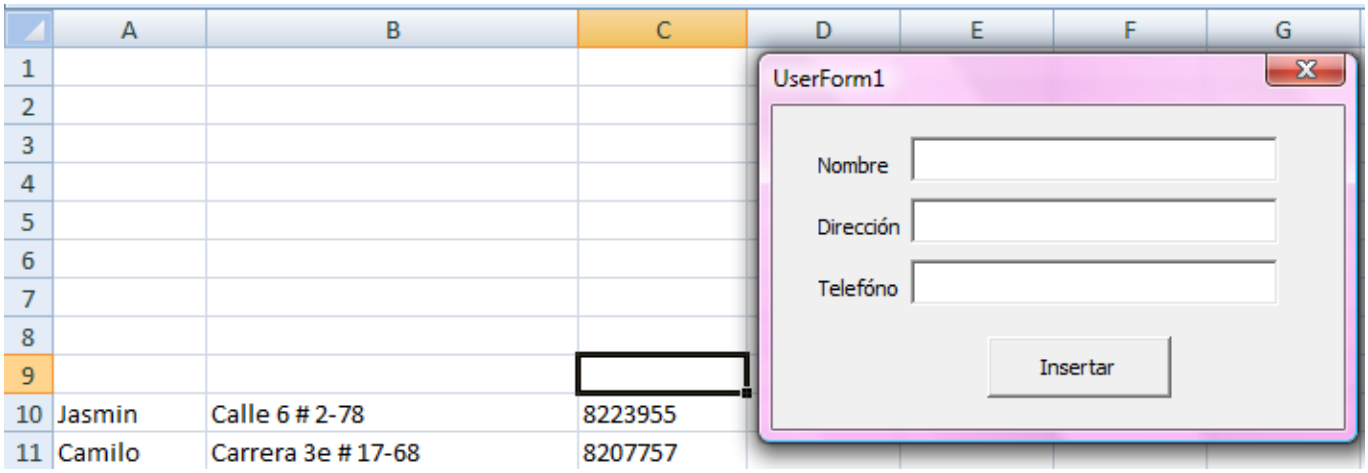
Damos clic en el botón **Insertar**, observe que los cuadros de texto del formulario quedan en blanco por si se quiere agregar otro registro y el registro que se digito queda guardado en la celda A10 de la hoja de Excel, observe:



Agreguemos un registro más:



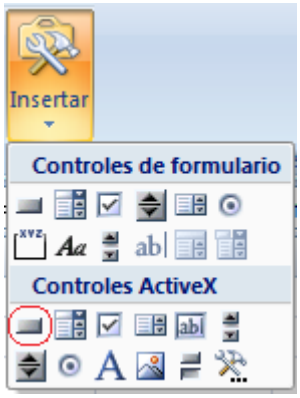
Damos clic en Insertar y observe como queda la lista ahora:



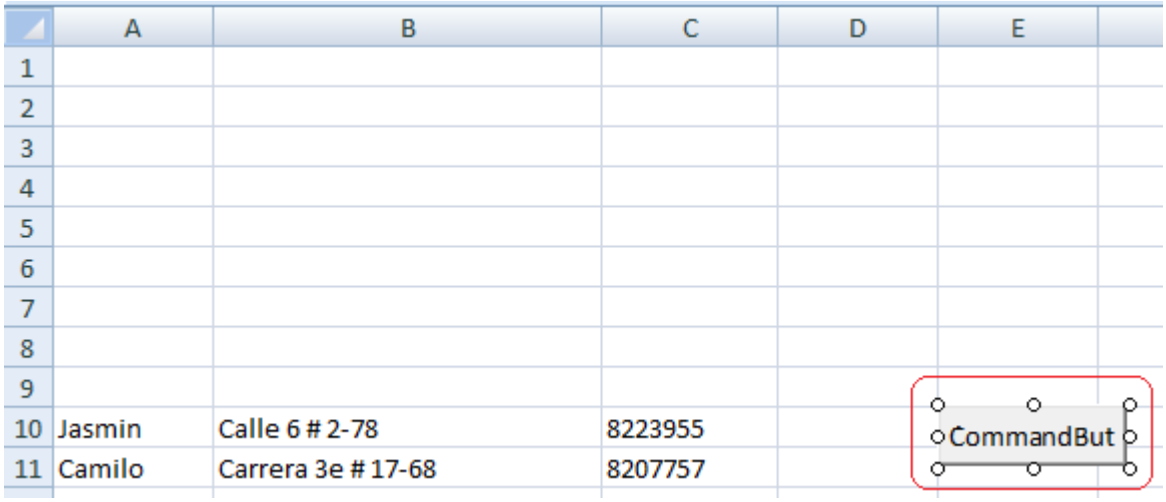
Cada vez que se agrega un nuevo registro este automáticamente inserta una fila por debajo de la celda que se escribe y los anteriores registros insertados bajan una posición.

Asignar el Formulario a un Botón desde Excel

Para cargar el formulario desde la hoja de Excel, en la vista Programador, en la opción Insertar, en Controles ActiveX seleccionamos el botón de comando, observe:



Damos clic en la hoja para ubicarlo donde queremos, observe:



Para programarlo damos doble clic en el botón y nos aparece el editor de visual Basic, observe:

```
Private Sub CommandButton1_Click()

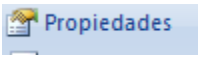
End Sub
```

En medio digitamos las siguientes líneas de código, observe:

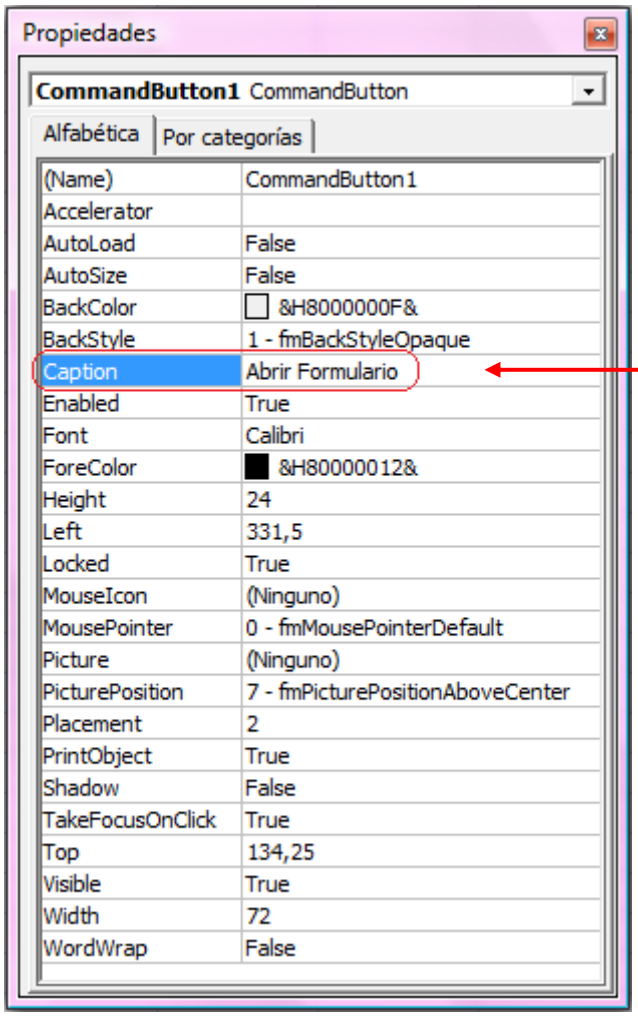
```
Private Sub CommandButton1_Click()
Load UserForm1
UserForm1.Show
End Sub
```

Load UserForm1 Carga el formulario
UserForm1.Show Hace visible el formulario lo muestra

Guardamos los cambios en el editor y regresamos a Excel.

Para cambiarle el nombre al botón, lo seleccionamos y damos clic en la opción 

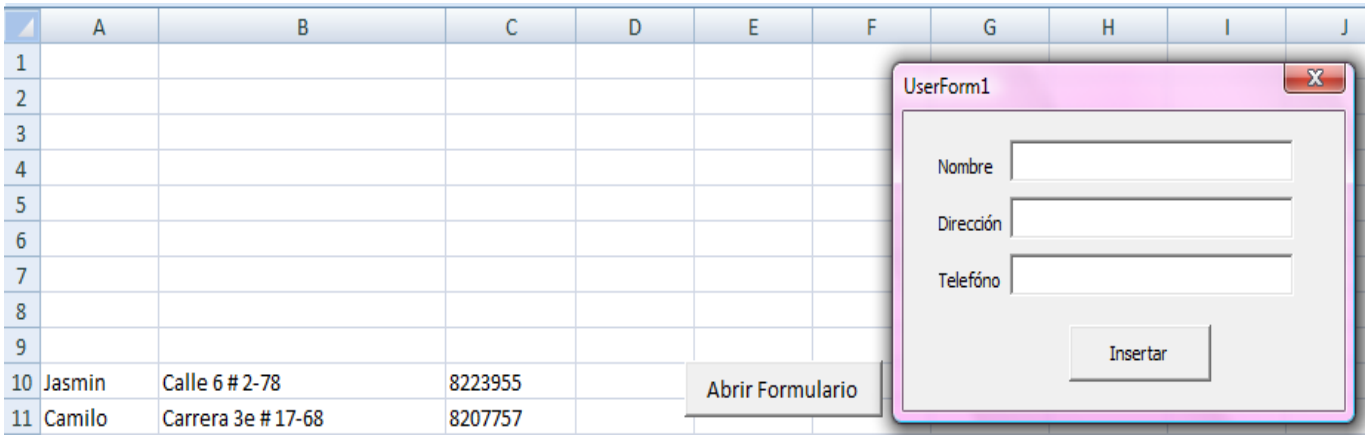
y nos aparece la siguiente ventana:



Nos Ubicamos al frente de la opción **Caption** y escribimos el nombre que deseamos ponerle al botón para nuestro caso será **Abrir Formulario** y cerramos la ventana.

Para activar el botón damos clic en la opción 

Damos clic en el botón Abrir Formulario y observe que ya nos muestra el formulario:



Otros Controles de Formulario

En Este nuevo formulario vamos a hacer uso de otros controles de formulario como los son: Cuadros combinados, Cuadro de lista, Casillas de Verificación, Botones de opción.

Primero iniciemos creando los controles que vimos en el ejemplo anterior, observe como ordenarlos:



Son 4 etiquetas y 3 Cuadros de texto

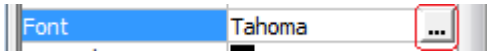
Ahora vamos a insertar un Marco , observe como debe quedar:



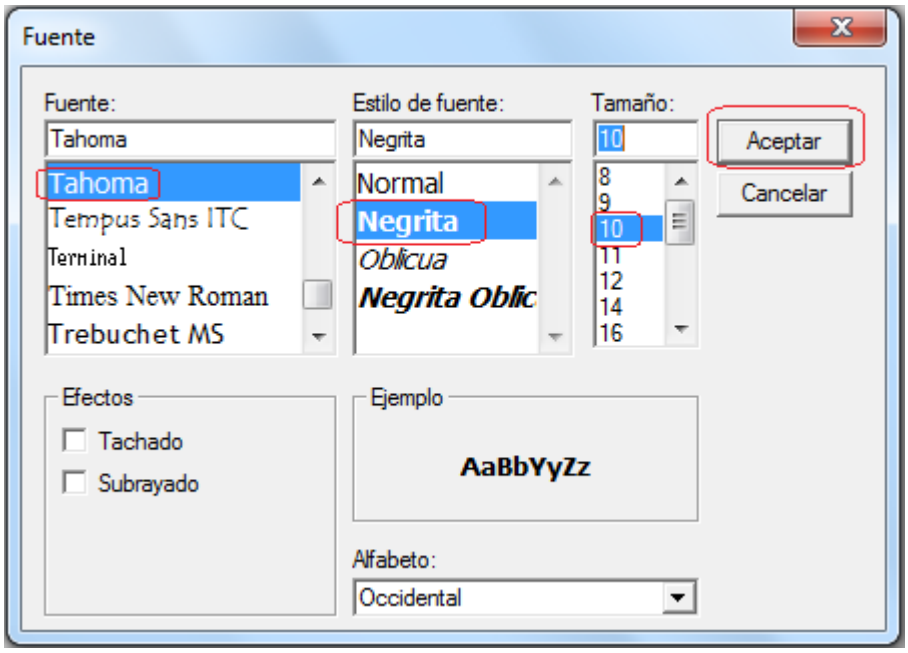
Para cambiarle el nombre al marco, lo seleccionamos y a la izquierda en la Propiedades, buscamos la opción **Caption** y al lado digitamos el texto que queremos aparezca, observe:



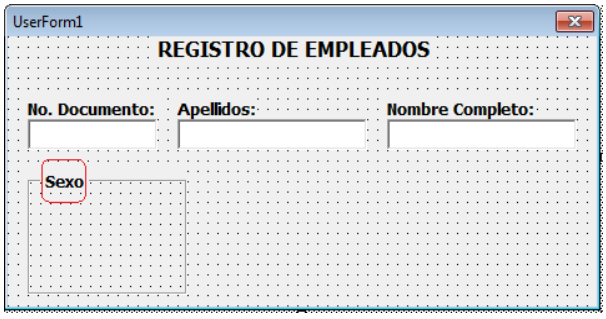
Para ponerlo en negrita y tamaño 10 hacemos lo siguiente, en las propiedades buscamos la opción **Font *Tahoma*** y damos clic en los puntos que aparecen, observe:



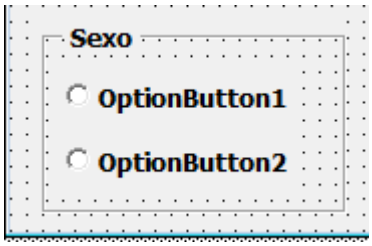
En la ventana que se muestra a continuación, podemos cambiar el tipo de Fuente, el estilo y el Tamaño, observe:



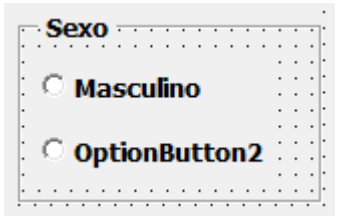
Escogemos fuente Tahoma, Estilo Negrita y Tamaño 10, damos clic en Aceptar y observe como queda:



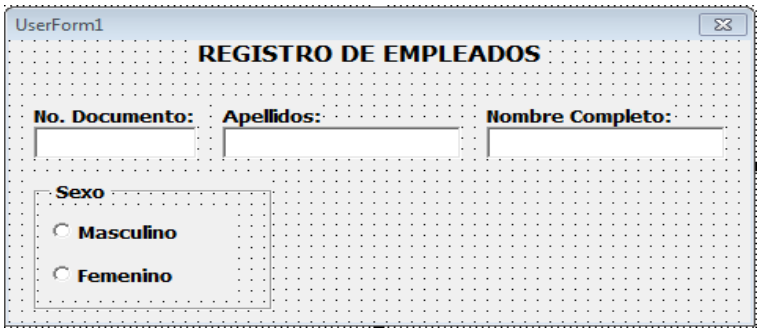
Ahora dentro del marco vamos a agregar dos Botones de opción , observe:



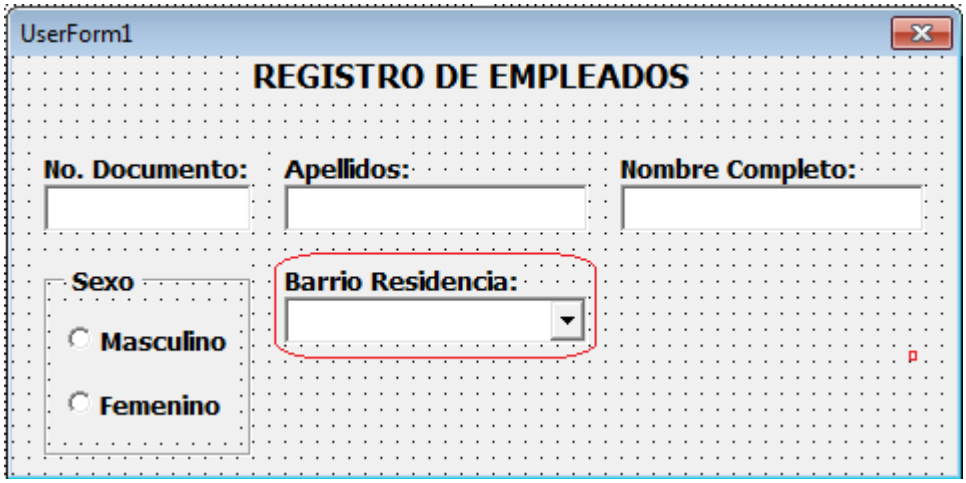
Para cambiarles el nombre, seleccionamos el botón, en las propiedades nos ubicamos en Caption y al lado digitamos el texto que queramos aparezca, observe:



De igual forma hacemos con el otro botón y lo configuramos así:



Ahora agregaremos un cuadro combinado , ordenándolo de la siguiente forma ,con su etiqueta, observe:



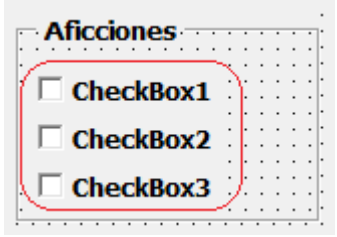
Ahora agregaremos un cuadro de lista , ordenándolo de la siguiente forma ,con su etiqueta, observe:



Insertamos un nuevo marco y lo configuramos de esta forma:



Luego le agregamos al marco 3 casillas de verificación, de esta forma:



Luego les ponemos los siguientes nombres, observe:



Luego seleccionamos el formulario y en la propiedad Caption le digitamos el nombre Empleados al formulario, Observe:



Por último agregamos dos botones de comando y los configuramos de esta forma:



Damos clic en el botón Ejecutar  para ver el formulario en funcionamiento, recuerde que no está programado, observe:



Cerramos el formulario para volver al editor de Visual Basic nuevamente y programar los controles:

Empezamos programando los textbox, similar en el ejemplo1, para ello damos doble clic en el textbox 1, observe:

```
Private Sub TextBox1_Change()

End Sub
```

Le digitamos el siguiente código:

```
Private Sub TextBox1_Change()
Range("A2").Select
ActiveCell.FormulaR1C1 = TextBox1
End Sub
```

De igual forma para el textbox 2 y textbox 3, observe:

```
Private Sub TextBox2_Change()
Range("B2").Select
ActiveCell.FormulaR1C1 = TextBox2
End Sub
Private Sub TextBox3_Change()
Range("C2").Select
ActiveCell.FormulaR1C1 = TextBox3
End Sub
```

NOTA: Recuerden que lo que hace estas líneas de código es ir a la hoja activa de Excel, seleccionar la celda asignada y escribir lo que contienen los textbox.

Para los botones de opción debemos validarlo de acuerdo a la opción que escojamos, para ello vamos a programarlos utilizando la condición IF, observe:

```
Private Sub OptionButton1_Click()
If OptionButton1 = True Then
Range("D3").Select
ActiveCell.FormulaR1C1 = "Masculino"
End If
If OptionButton2 = True Then
Range("D3").Select
ActiveCell.FormulaR1C1 = "Femenino"
End If
End Sub
Private Sub OptionButton2_Click()
If OptionButton1 = True Then
Range("D3").Select
ActiveCell.FormulaR1C1 = "Masculino"
End If
If OptionButton2 = True Then
Range("D3").Select
ActiveCell.FormulaR1C1 = "Femenino"
End If
End Sub
```

Vamos a evaluar los botones de opción con dos If

```
If OptionButton1 = True Then
Range("D3").Select
ActiveCell.FormulaR1C1 = "Masculino"
End If
```

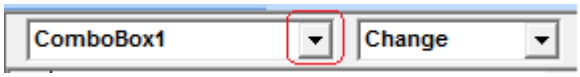
La primer línea indica si se selecciona el OptionButton1 que corresponde a la opción Masculino, sea igual a True es decir verdadero, este es el valor que toma un OptionButton al darle clic, si se cumple entonces que en la celda D3 escriba la palabra Masculino.

De igual forma funciona para el OptionButton2, que al seleccionarlo en la celda D3 pondría la palabra Femenino:

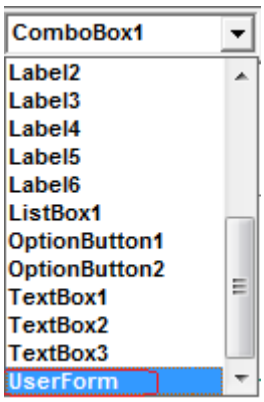
```
If OptionButton2 = True Then
Range("D3").Select
ActiveCell.FormulaR1C1 = "Femenino"
End If
```

Al ComboBox, primero hay que agregarle los valores, es decir las opciones que se pueden seleccionar al darle clic, para ello, debemos asignarlos en el momento en que el formulario sea activado, para ello se usa la propiedad actívate del userform1, veamos como activarla:

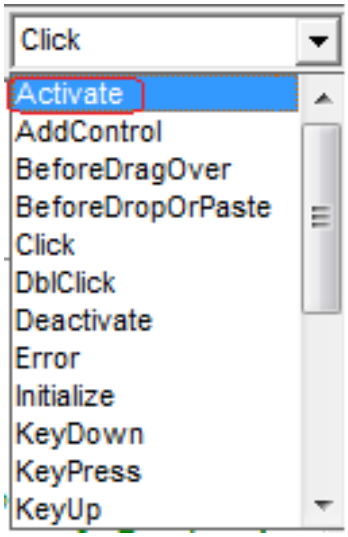
En la hoja donde se programan los controles den clic en este botón que esta al inicio



Luego seleccionar UserForm, observe:



Y en la opción de al lado escogen, Activate, observen:



Nos genera este código:

```
Private Sub UserForm_Activate()

End Sub
```

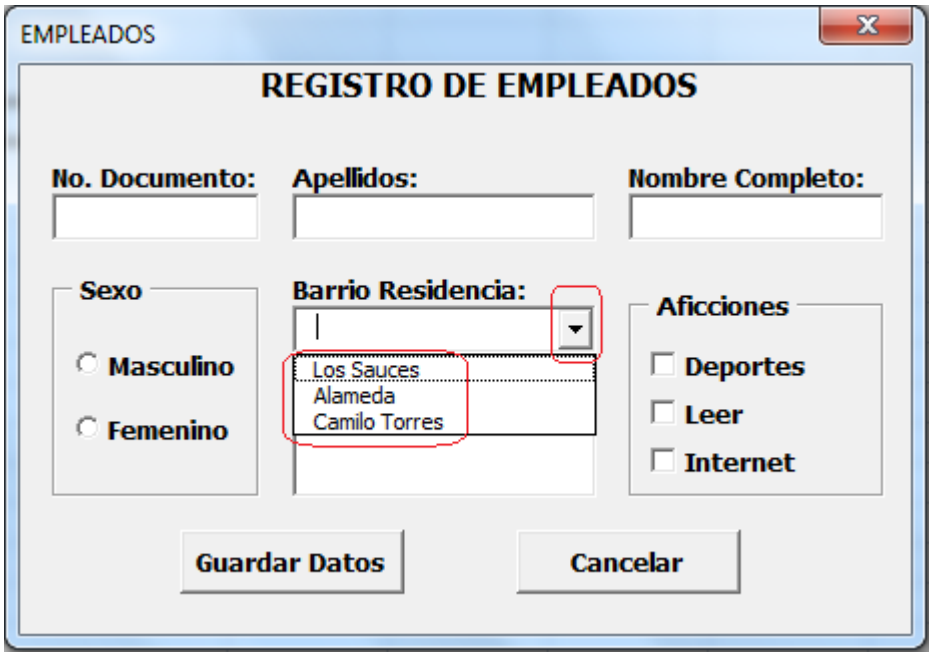
Ahora le escribimos el código que permite asignarle valores al ComboBox, observen como:

```
Private Sub UserForm_Activate()
ComboBox1.AddItem "Los Sauces"
ComboBox1.AddItem "Alameda"
ComboBox1.AddItem "Camilo Torres"
End Sub
```

ComboBox1.AddItem "Los Sauces"

Quiere decir que al cuadro combinado 1, agréguele la opción Los Sauces.

Si ejecutamos el formulario y hacemos clic en el cuadro combinado, podemos ver como aparecen las opciones, observen:



Con solo ejecutar el formulario los valores del cuadro combinado aparecen automáticamente, esto es lo que permite el código

```
Private Sub UserForm_Activate()
ComboBox1.AddItem "Los Sauces"
ComboBox1.AddItem "Alameda"
ComboBox1.AddItem "Camilo Torres"
End Sub
```

Ahora para que el ComboBox agregue en la Hoja el valor que se selecciona se programa de la siguiente, observe:

```
Private Sub ComboBox1_Change()
Range("E3").Select
ActiveCell.FormulaR1C1 = ComboBox1
End Sub
```

Para el ListBox (lista de valores) funciona similar al ComboBox, los valores que se desean mostrar también se deben agregar al activarse el formulario para ello también deben añadirse en el UserForm_Activate similar al ComboBox, observe:

```
Private Sub UserForm_Activate()
ComboBox1.AddItem "Los Sauces"
ComboBox1.AddItem "Alameda"
ComboBox1.AddItem "Camilo Torres"

ListBox1.AddItem "Estrato 1"
ListBox1.AddItem "Estrato 2"
ListBox1.AddItem "Estrato 3"
End Sub
```

Lo único que cambia es el nombre del control al cual se le van a asignar los valores.

Si ejecutamos el formulario podemos ver como aparecen las opciones del ListBox, en la opción Estrato, observen:



Ahora para que el ListBox agregue en la Hoja el valor que se selecciona se programa de la siguiente, observe:

```
Private Sub ListBox1_Click()
Range("F3").Select
ActiveCell.FormulaR1C1 = ListBox1
End Sub
```

Por último vamos a programar las Aficiones, en este caso los CheckBox o casillas de verificación, las cuales pueden seleccionarse varias opciones o al menos 1, para ello vamos hacer varias rutinas IF anidadas para verificar las opciones que se estén seleccionando, veamos:

Damos doble clic en el CheckBox 1 y nos aparece este código:

```
Private Sub CheckBox1_Click()

End Sub
```

Ahora digitaremos el código, veamos:

Primero definimos una variable llamada cadena de tipo String es decir que en esa variable vamos a guardar texto, la palabra Dim indica que vamos a definir una variable.

Luego a la variable cadena la inicializamos vacía es decir que no contenga nada, Observe como queda el código:

```
Private Sub CheckBox1_Click()
Dim cadena As String
cadena = ""
End Sub
```

Ahora vamos a evaluar el CheckBox1, que pasa cuando se ejecute el formulario y se dé clic en él:

```
Private Sub CheckBox1_Click()
Dim cadena As String
cadena = ""

If CheckBox1 = True Then
    If Len(cadena) = 0 Then
        cadena = ""
    End If
    cadena = cadena + "Deportes"
End If

End Sub
```

If CheckBox1 = True Then

Esta línea indica Si el CheckBox1 es verdadero, es decir cuando se dé clic en él, que hace

If Len(cadena) = 0 Then cadena = ""	Len() es una función para evaluar la longitud de un texto
End If	significa el final del segundo if que esta anidado.

Estas líneas indican si la longitud de la variable cadena es igual a cero o vacio que haga cadena igual a vacio, es decir que la deje en blanco, en este caso cuando el formulario se ejecute y no sé de clic en ninguna opción que deje la celda vacía.

cadena = cadena + "Deportes"	Significa que a la variable cadena se le asigne la misma variable pero agregándole la palabra deportes
End If	significa el final del primer if.

Para evaluar los CheckBox2 y CheckBox3 el código es el mismo, lo único que cambia es cuando se le asigna la palabra leer e Internet de los dos checkbox.

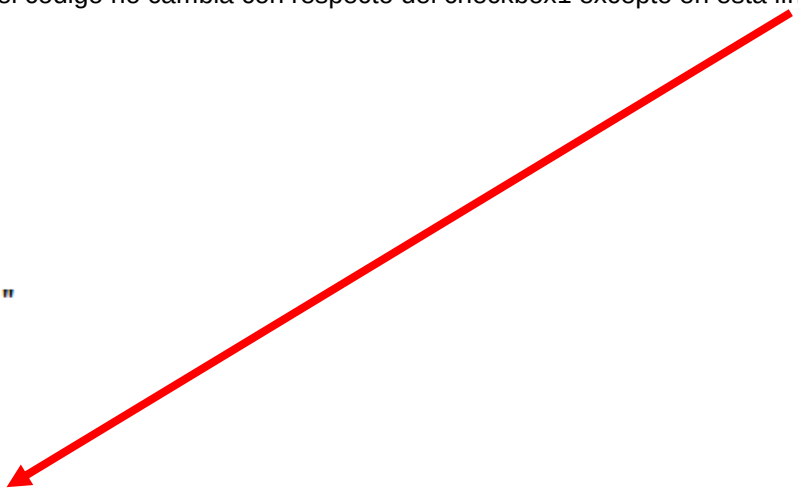
Veamos para el checkbox2, observen que el código no cambia con respecto del checkbox1 excepto en esta línea:

```
Private Sub CheckBox1_Click()
Dim cadena As String
cadena = ""

If CheckBox1 = True Then
    If Len(cadena) = 0 Then
        cadena = ""
    End If
    cadena = cadena + "Deportes"
End If

If CheckBox2 = True Then
    If Len(cadena) = 0 Then
        cadena = ""
    End If
    cadena = cadena + "," + "Leer"
End If

End Sub
```



En la línea indicada se le asigna a la variable cadena si ha seleccionado el checkbox1, en los if anteriores, también se le agrega la palabra Leer en el caso de que haya escogido las dos opciones, es decir si se ha dado clic tanto en el CheckBox1 y el CheckBox2, estos signos `+ "," +` quieren decir que una o concatene las dos palabras.

Para el checkBox3 es el mismo código salvo en la misma línea del CheckBox2, observe:

```
Private Sub CheckBox1_Click()
Dim cadena As String
cadena = ""

If CheckBox1 = True Then
    If Len(cadena) = 0 Then
        cadena = ""
    End If
    cadena = cadena + "Deportes"
End If

If CheckBox2 = True Then
    If Len(cadena) = 0 Then
        cadena = ""
    End If
    cadena = cadena + "," + "Leer"
End If

If CheckBox3 = True Then
    If Len(cadena) = 0 Then
        cadena = ""
    End If
    cadena = cadena + "," + "Internet"
End If

End Sub
```

Por último evaluamos la longitud de la variable cadena nuevamente:

```
Private Sub CheckBox1_Click()
Dim cadena As String
cadena = ""

If CheckBox1 = True Then
    If Len(cadena) = 0 Then
        cadena = ""
    End If
    cadena = cadena + "Deportes"
End If

If CheckBox2 = True Then
    If Len(cadena) = 0 Then
        cadena = ""
    End If
    cadena = cadena + "," + "Leer"
End If

If CheckBox3 = True Then
    If Len(cadena) = 0 Then
        cadena = ""
    End If
    cadena = cadena + "," + "Internet"
End If

If Len(cadena) <> 0 Then
Range("G3").Select
ActiveCell.FormulaR1C1 = cadena
End If
End Sub
```

Aquí evaluamos si la cadena no esta vacía, entonces que en la celda G3 se digite o se agregue el contenido de la variable cadena, que puede ser que se hayan seleccionado 1, 2 o las tres opciones.

Fin del checkbox1

Ahora lo que debemos hacer es copiar y pegar todo el código en los CkeckBox 2 y CkeckBox3. Observen:

```
Private Sub CheckBox2_Click()
Dim cadena As String
cadena = ""
If CheckBox1 = True Then
    If Len(cadena) = 0 Then
        cadena = ""
    End If
    cadena = cadena + "Deportes"
End If

If CheckBox2 = True Then
    If Len(cadena) = 0 Then
        cadena = ""
    End If
    cadena = cadena + "," + "Leer"
End If

If CheckBox3 = True Then
    If Len(cadena) = 0 Then
        cadena = ""
    End If
    cadena = cadena + "," + "Internet"
End If
If Len(cadena) <> 0 Then
Range("G3").Select
ActiveCell.FormulaR1C1 = cadena
End If
End Sub
```

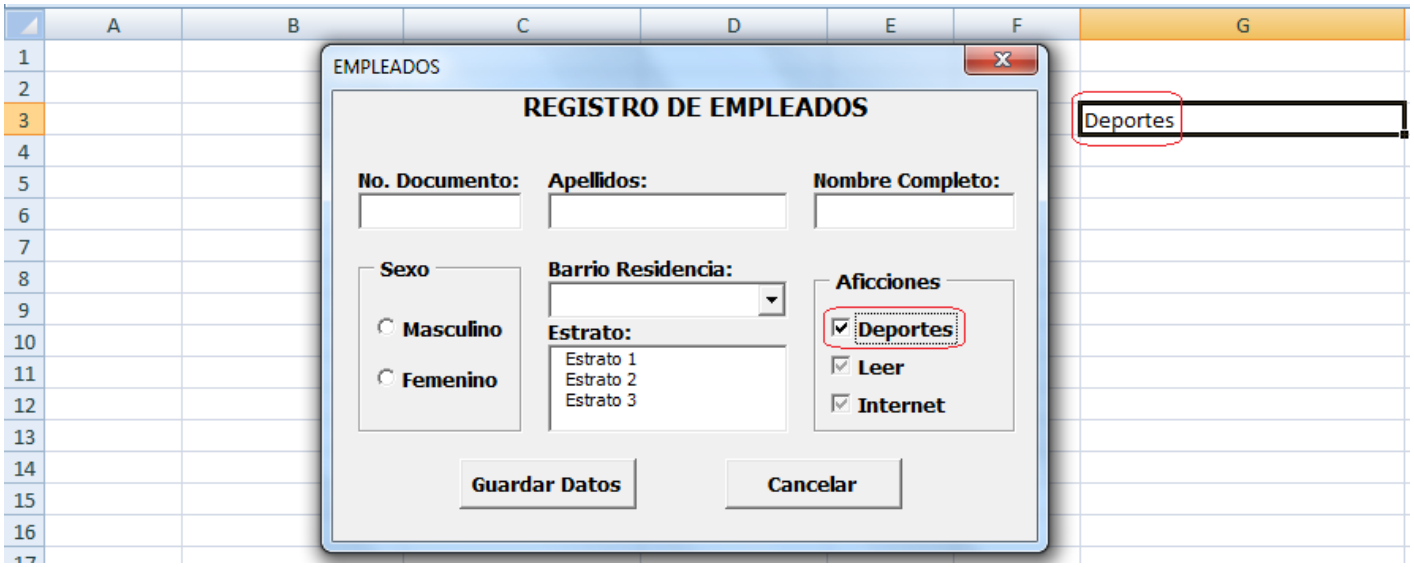
```
Private Sub CheckBox3_Click()
Dim cadena As String
cadena = ""
If CheckBox1 = True Then
    If Len(cadena) = 0 Then
        cadena = ""
    End If
    cadena = cadena + "Deportes"
End If

If CheckBox2 = True Then
    If Len(cadena) = 0 Then
        cadena = ""
    End If
    cadena = cadena + "," + "Leer"
End If

If CheckBox3 = True Then
    If Len(cadena) = 0 Then
        cadena = ""
    End If
    cadena = cadena + "," + "Internet"
End If
If Len(cadena) <> 0 Then
Range("G3").Select
ActiveCell.FormulaR1C1 = cadena
End If
End Sub
```

Listo ejecutemos el formulario y veamos como es el funcionamiento de los CheckBox:

Al dar clic en la opción Deportes que corresponde al CheckBox1, observen que automáticamente aparece la palabra en la celda G3:



Ahora si se da clic en la opción Leer que corresponde al CheckBox2, observen que automáticamente se agrega la palabra separada de una coma:



Ocurre lo mismo si se da clic en la opción Internet que corresponde al CheckBox3, observen:



Ahora en el caso de que se desea quitar una de las opciones solo habría que desactivar o dar nuevamente clic en la opción, por ejemplo desactivemos la opción Leer, observen que automáticamente la palabra se borra de la celda G3, pero se mantienen las otras dos opciones escogidas:



Cerremos el formulario y ahora programemos los dos botones, Guardar Datos y Cancelar:

Demos doble clic en el botón Guardar Datos, observe el código que aparece:

```
Private Sub CommandButton1_Click()

End Sub
```

Digitamos el siguiente código que ya saben que hace cuando se creó el ejemplo1 del primer formulario:

Selection.EntireRow.Insert lo que hace es insertar los datos escritos y crea una nueva fila para agregar otro nuevo
Empty lo que hace es limpiar o dejar en blanco los controles.
TextBox1.SetFocus lo que hace es luego de insertar los datos digitamos y poner nuevamente las celdas y controles en blanco, ubica el cursor en el primer textbox.

```
Private Sub CommandButton1_Click()
Selection.EntireRow.Insert

TextBox1 = Empty
TextBox2 = Empty
TextBox3 = Empty
OptionButton1 = Empty
OptionButton2 = Empty
ComboBox1 = Empty
ListBox1 = Empty
CheckBox1 = Empty
CheckBox2 = Empty
CheckBox3 = Empty

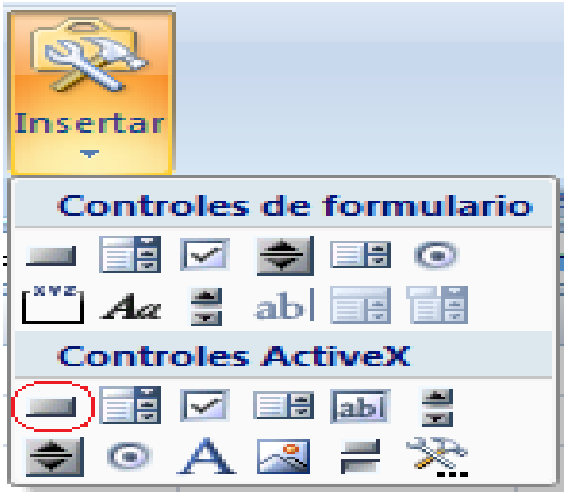
TextBox1.SetFocus
End Sub
```

Este es el código para el botón Cancelar el cual deja los controles en Blanco:

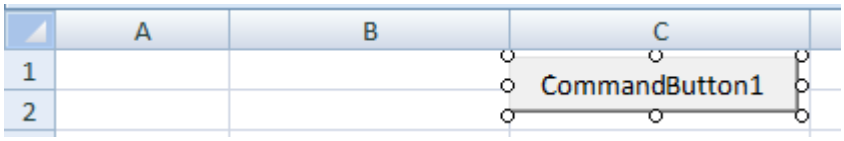
```
Private Sub CommandButton2_Click()
TextBox1 = ""
TextBox2 = ""
TextBox3 = ""
ComboBox1 = ""
Range("D3").Select
ActiveCell.FormulaR1C1 = ""
Range("F3").Select
ActiveCell.FormulaR1C1 = ""
Range("G3").Select
ActiveCell.FormulaR1C1 = ""
End Sub
```

Vayan a la Hoja de Excel y Borren todo, dejen la hoja en blanco.

Ahora desde la hoja de Excel creemos un botón para ejecutar el formulario, en la vista Programador, en la opción Insertar, en Controles ActiveX seleccionamos el botón de comando, observe:



Luego forme el botón y ubíquelo en una posición de la Hoja, Observe:



Para programarlo damos doble clic en el botón y nos aparece el editor de visual Basic, observe:

```
Private Sub CommandButton1_Click()

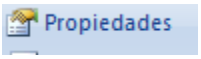
End Sub
```

En medio digitamos las siguientes líneas de código, observe:

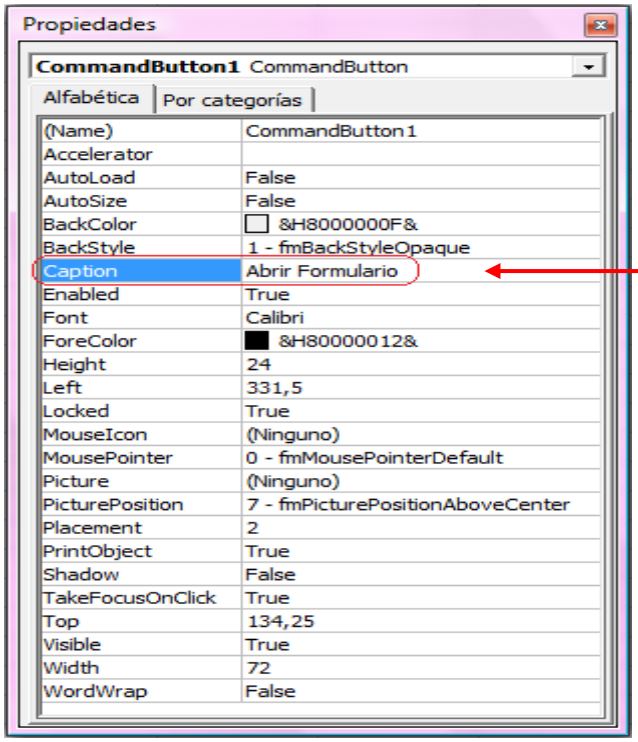
```
Private Sub CommandButton1_Click()
Load UserForm1
UserForm1.Show
End Sub
```

Load UserForm1 Carga el formulario
UserForm1.Show Hace visible el formulario lo muestra

Guardamos los cambios en el editor y regresamos a Excel.

Para cambiarle el nombre al botón, lo seleccionamos y damos clic en la opción 

y nos aparece la siguiente ventana:



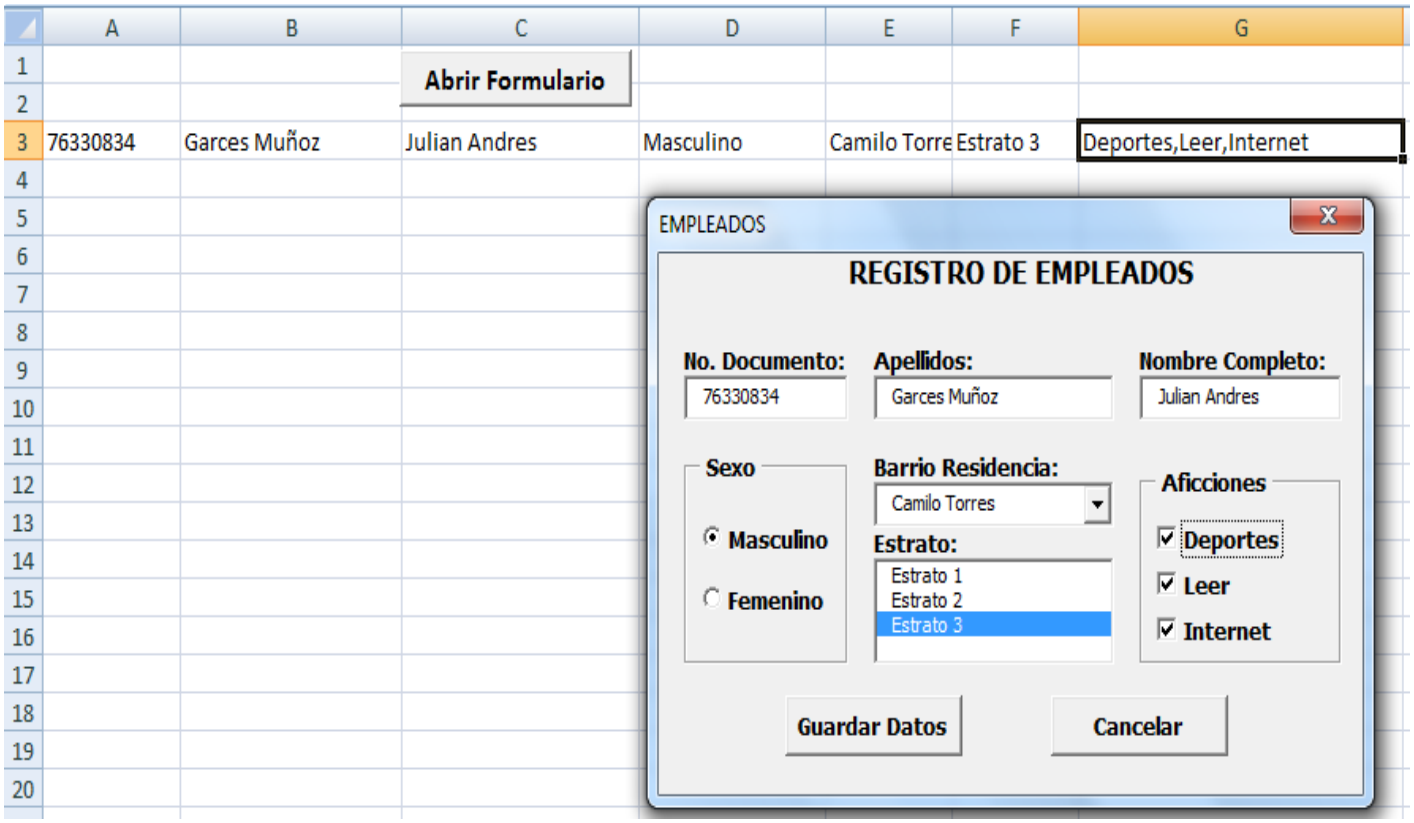
Nos Ubicamos al frente de la opción **Caption** y escribimos el nombre que deseamos ponerle al botón para nuestro caso será **Abrir Formulario** y cerramos la ventana.

Para activar el botón damos clic en la opción 

Damos clic en el botón Abrir Formulario y observe que ya nos muestra el formulario:

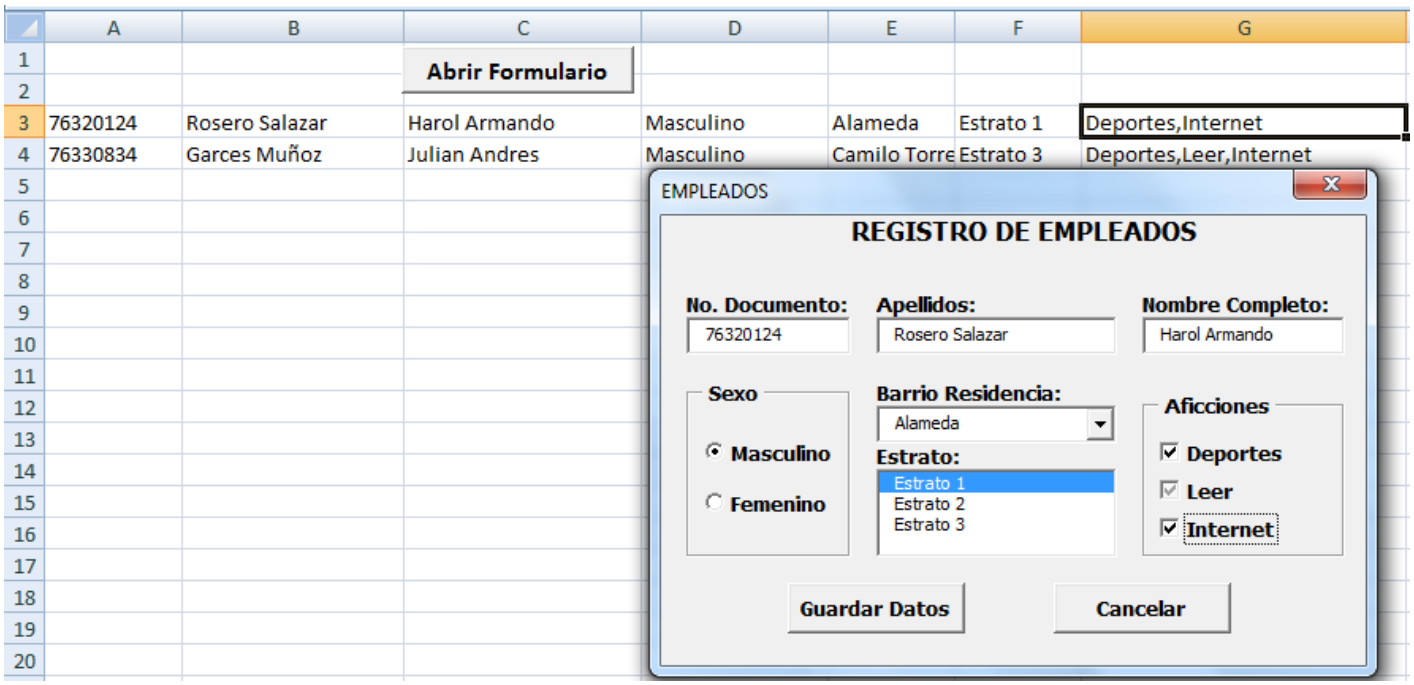


Ahora empecemos a insertar datos, observen que al ir digitando los valores automáticamente se van agregando a las celdas asignadas:



Damos clic en el Botón Guardar datos para agregar nuevos datos:

Miremos otro registro, observe que el anterior ha bajado una fila lo que indica que ya ha sido almacenado y los nuevos valores aparecen por encima, listos para guardar:



Bueno Aprendices con este Material damos por culminado los documentos del curso que espero les hayan servido de apoyo en sus labores cotidianas y laborales, los invito a realizar con entusiasmo el último Taller del curso.

Desarrollo de contenidos

1. Modelo ER a modelo relacional

1.1 Transformación MER a modelo relacional

Uno de los pasos iniciales para la construcción de una base de datos es el modelado de ellos por medio de un Modelo Entidad Relación (MER), modelo que permite por medio de entidades, atributos y relaciones acercarse a una representación preliminar del diseño de una base de datos.
Luego de terminar un MER debe entrarse en el diseño de un modelo lógico, el cual es llamado modelo relacional. La forma como se logra desarrollar una base de datos relacional es a través del modelado y la metodología de diseño de una base de datos puede estar enfocada en los siguientes 3 pasos:

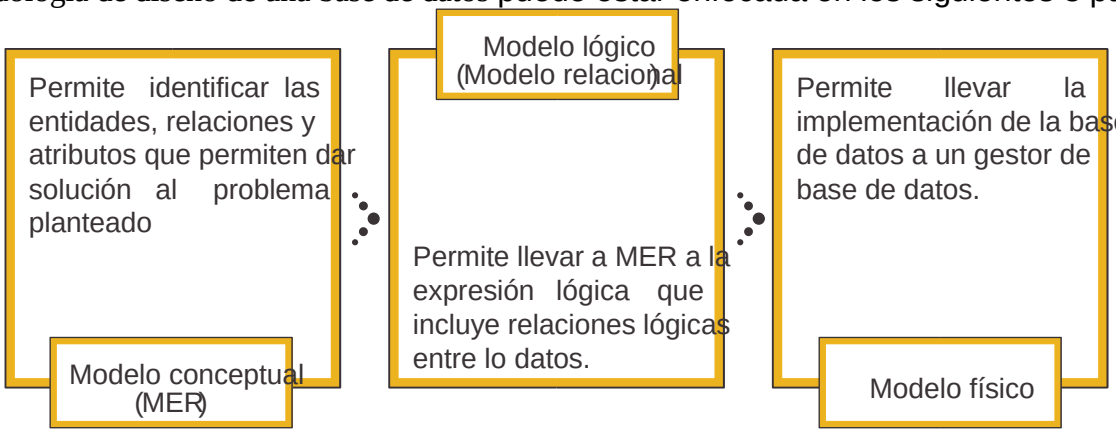


Figura 1. Pasos para la conversión del modelo ER a Relacional

En estos momentos se abordará el proceso de la construcción de un modelo relacional a partir de un MER.

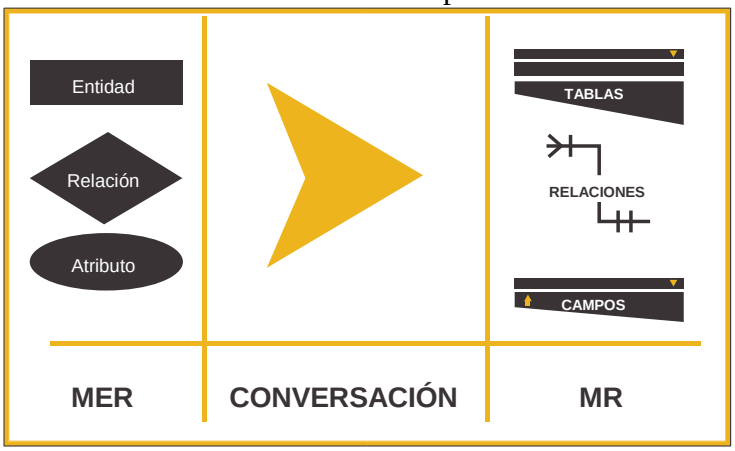


Figura 2. Modelo de conversión MER a MR

1.2 Reglas generales de transformación

La transformación de un Modelo Entidad Relación a Relacional, se puede expresar mediante un esquema relacional o un modelo relacional.

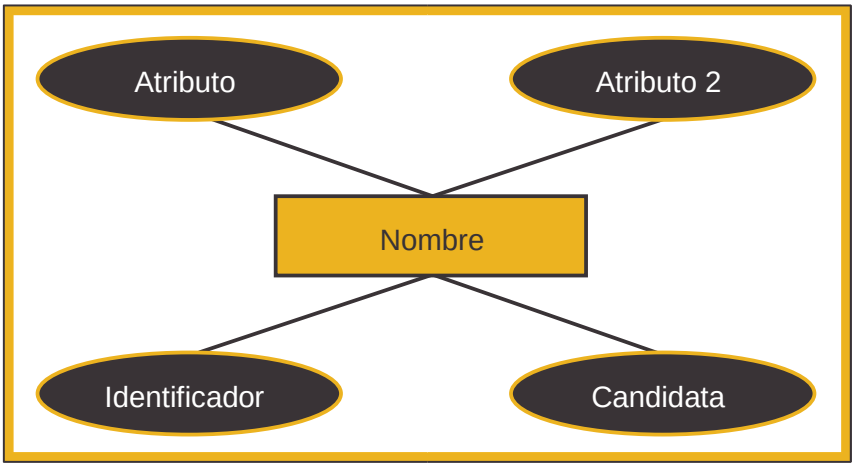


Figura 3. Esquema relacional

El esquema relacional consta de palabras y relaciones mediante flechas que indican la asociación de las entidades; el modelo relacional es un dibujo que representa las tablas, atributos, llaves primarias y llaves foráneas según la transformación realizada.
A continuación se explica detalladamente la transformación de un MER según las reglas para las entidades y se expresa en esquema relacional y modelo relacional.
NombreTabla(Identificador, Atributo1, Atributo2, Candidata)

1.2.1 Transformación para las entidades y atributos

- Entidades. Las entidades pasan a ser tablas.
- Atributos. Los atributos pasan a ser columnas o atributos de la tabla.
- Identificadores principales. Pasan a ser claves primarias.
- Identificadores candidatos. Pasan a ser claves candidatas.

La siguiente imagen ilustra los anteriores elementos:

Campo Nulo

EMP_ID	EMP_NOMBRE	EMP_CARGO	EMP_JEFE	DEP_ID
1000	LOPEZ	PRESIDENTE		10
1100	BENAVIDEZ	GERENTE	1000	30
1200	CORDOBA	GERENTE	1000	10
1300	JIMENEZ	GERENTE	1000	20
2100	MARTINEZ	VENDEDOR	1100	30
2200	ARTEAGA	VENDEDOR	1100	30
2300	TOLEDO	VENDEDOR	1100	30
2400	JARAMILLO	OFICINISTA		30
			1100	30
2500	ZAMBRANO	VENDEDOR	1100	30

| Filo / Tupla

Celda / Campo

Llave Primaria

Columna

Atributo

Foranea PK

FK

Figura 4. Transformación para las entidades y atributos

Por ejemplo:

Si se tiene la siguiente entidad Aprendiz se convierte entonces:

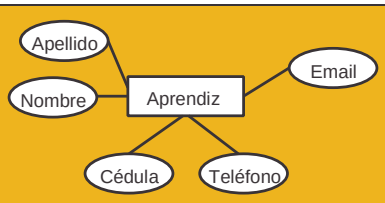
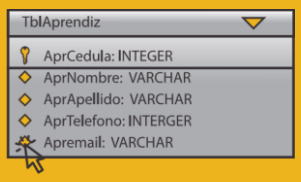
MER	
Esquema Relacional	TblAprendiz(AprCedula,AprNombre,AprApellido, AprTelefono,AprEmail) En este esquema la llave primaria se identifica porque se encuentra subrayado el atributo que será llave primaria.
Modelo Relacional	 *La tabla se llama Tb/Aprendiz y la llave primaria se identifica porque el nombre puede estar: subrayado, tener al frente el estereotipo <<pk>> o una llave en color amarillo.*

Figura 5. Transformación para las entidades y atributos

NOTA: *para garantizar unicidad de estilos se debe recordar que debe tener una estandarización para nombrar las tablas y los atributos. Por ejemplo para este caso la entidad aprendiz se convierte en la tabla aprendiz y es llamada como TblAprendiz, el estilo de nombrado es iniciar con las tres primeras letras Tbl para indicar que es tabla, seguido del nombre de la entidad e iniciando con la primera letra en mayúscula.*

Para el atributo Apellido, en el modelo relacional se convierte en AprApellido indicando Apr que viene de la tabla Aprendiz y seguido del nombre del atributo cada uno iniciando con Mayúscula la primera letra.

1.2.2 Transformación para las relaciones

Relación muchos a muchos: esta relación se distingue en un MER porque su cardinalidad entre las dos entidades esta designada por (M..M) o (N..M).

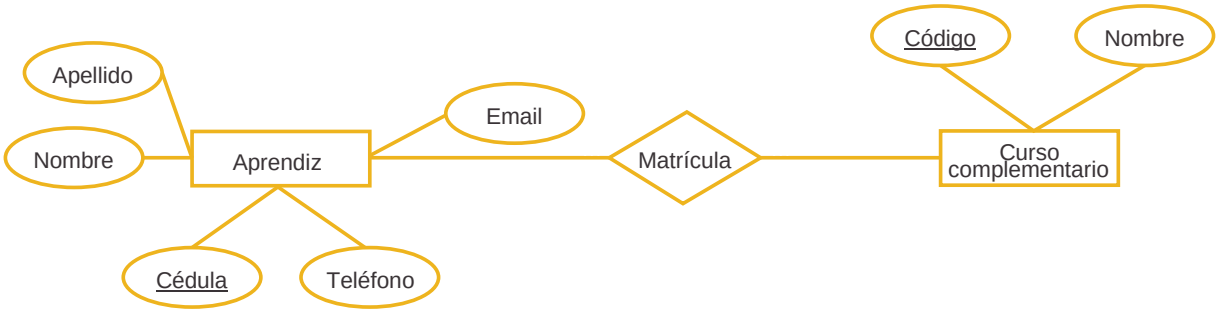


Figura 6. Relación muchos a muchos

- **Transformación relación muchos a muchos**: cuando se tiene una relación de este tipo se debe crear una nueva relación (tabla) la relación se transforma en una tabla cuyos atributos son: los atributos de la relación y las claves de las entidades relacionadas (que pasarán a ser claves externas). La clave de la tabla la forman todas las claves externas.

- **Transformación a esquema relacional**: cuando se hace la transformación a esquema relacional se debe tener un esquema de nombrado para las tablas que se generaran, por ejemplo para este caso la entidad aprendiz la tabla generada se denomina TblAprendiz, anteponiéndole tres letras Tbl que indican que es una tabla; de la misma manera para los atributos de esta entidad se le antepone las tres letras Apr para indicar que son atributos de la tabla Aprendiz. Así se continuara el estilo de nombrado para todos los casos de estudio y ejemplos de este material.

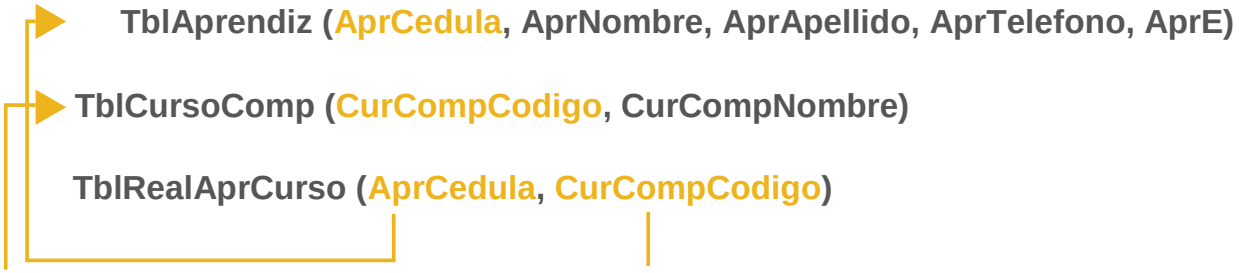


Figura 7. Esquema relacional

- **Transformación a modelo relacional**: para realizar las transformaciones a modelo relacional se puede utilizar herramientas case, para estos ejemplos se ha utilizado el software Día, el uso de ésta última se detallará en los ejercicios que acompañan el laboratorio correspondiente.

Se visualiza la nueva tabla donde las llaves primarias son a la vez llaves foráneas, nótese que la cardinalidad de la relación está representada por los rombos en blanco (1) o sombreados (M).

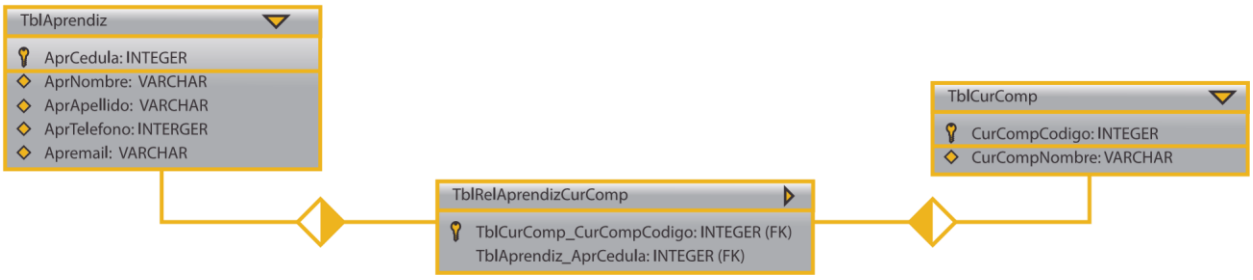
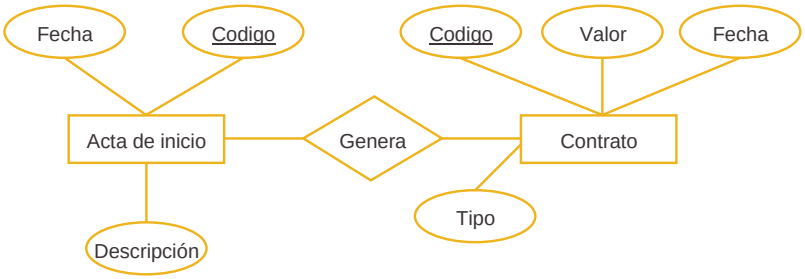


Figura 8. Transformación a modelo relacional

La nueva tabla creada es llamada TblRelAprendizCurComp, que tiene incluida las dos llaves primarias de la tabla TblAprendiz y TblCurComp. Estas llaves en la nueva tabla son llamadas llaves foráneas y son representadas por un <fk> en el modelo.

El jueves se le pregunta al Experto si decidió sacar la siguiente información o se le pasó en tenerla en cuenta:



Nota: *las llaves primarias pasada a la nueva tabla se llaman llaves foráneas. 3.2.2 Relación 1 a 1 (1..1): esta relación se distingue en un MER porque su cardinalidad entre las dos entidades esta designada por (1..1).* Ejemplo:

Transformación Relación 1 a 1 (1..1): cuando se tiene una relación de este tipo se envía la llave primaria de una tabla a la otra tabla que participa de la relación, se envía la llave primaria que se considere sea más relevante para enviar a la otra tabla.

Esquema relacional

TblContrato (ConCodigo,ConValor, ConFecha, ConTipo)

TblActaInicio (ActCodigo,ActFecha, ActDescripción, TblContratoConCodigo)

Para este ejemplo específico se ha pasado la llave primaria del contrato (ConCodigo) a la tabla ActaInicio.

Transformación a modelo relacional



En la tabla TblActaInicio se incluye la clave primaria de la tabla TblContrato (ConCodigo) quedando como llave foránea.

Relación uno a muchos: esta relación se distingue en un MER porque su cardinalidad entre las dos entidades esta designada por (1..M).

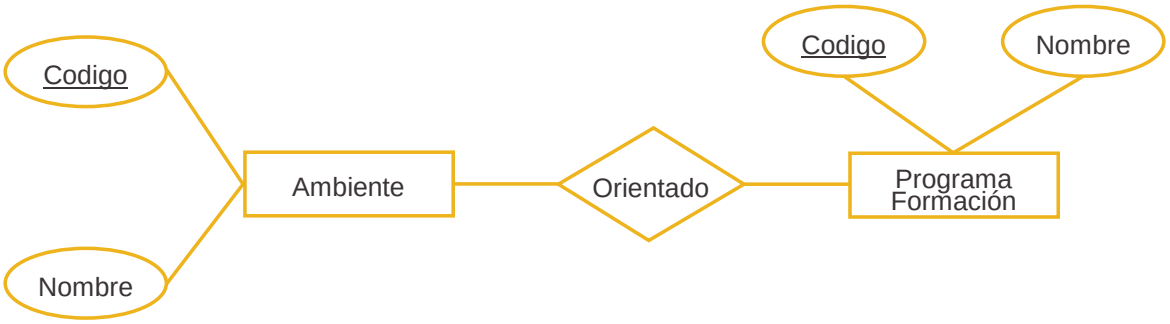


Figura 9. Relación uno a muchos

- **Transformación relación uno a muchos**: cuando se tiene una relación de este tipo se debe colocar o enviar la llave primaria de la tabla que tiene en cardinalidad 1 a la tabla que tiene cardinalidad M.

Transformación a esquema relacional:

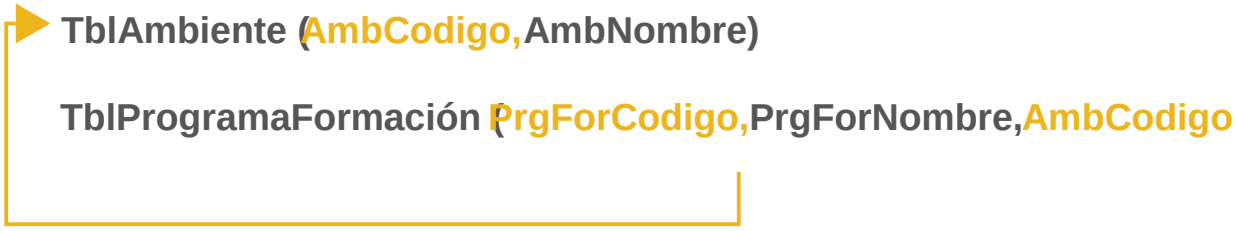


Figura 10. Esquema relacional uno a muchos

Visualizar que La tabla TblAmbiente tiene como cardinalidad 1, por la tanto la clave primaria de esta tabla es pasada a la tabla TblProgramaFormacion.

- Transformación a modelo relacional: Notar que la clave de la tabla TblAmbiente es pasada a la Tabla TblProgramaFormacion la cual tiene su estereotipo <fk>.

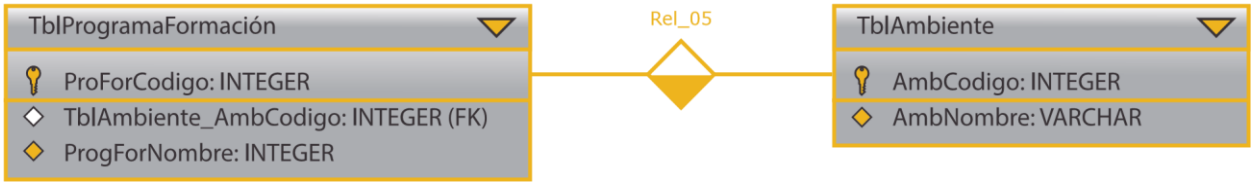


Figura 11. Esquema relacional uno a muchos

2. Transformacion MER-extendido a relacional

A continuación se presenta un ejemplo de transformación de modelo entidad relación a Relacional, que aplica las reglas descritas. Para este ejemplo se toma una situación problemática y su modelo conceptual, para lo cual se presenta la descripción del problema, el MER y el modelo relacional resultante.

Descripción del problema.

En el SENA Regional Cauca del Centro de Teleinformática y Producción industrial el coordinador requiere un sistema de información que permita gestionar los horarios de los instructores pertenecientes a un centro de formación que orientan formación profesional integral, tanto de planta como de contrato, tenga en cuenta que un instructor de contrato puede tener su relación de contrato por horas o asignación mensual.

Para lo cual es necesario conocer: un instructor orienta formación profesional (clase) en un programa de formación (el programa de formación se refiere a las tecnologías o tecnólogos que ofrece el SENA), o en un curso complementario (los cuales son cursos de duración de 2 a 3 meses y solicitados por empresas del sector para capacitar a sus empleados).

Ambos: programa de formación y complementarios son identificados por un código.

Cuando un aprendiz se matricula a un programa de formación o curso complementario tiene que durante toda su etapa lectiva (clase), cursar competencias laborales.
Una competencia podría asemejarse como una asignatura que un aprendiz debe cursar, pero para este caso siempre manejaremos el término competencia.
Asignación de competencias: para asignar una competencia a un instructor es necesario que esté legalmente contratado por el centro o que tenga resolución de nombramiento y según su especialidad se le asignan las competencias que va a orientar.
Ambientes de formación: es el sitio específico (salón) donde se orientaran las competencias asignadas. Cada ambiente tiene una codificación asignada.

Reglas del negocio: un instructor de contrato puede iniciar labores una vez tenga el acta de inicio.

Tener en cuenta que un instructor puede laborar de lunes a viernes de 7:00 a 10:00 p.m. de la noche, sábados y domingos de 8:00 a 6:00 p.m.

Un instructor puede orientar diferentes competencias en diferentes programas de formación y cursos complementarios.
Un instructor no puede orientar una competencia en diferentes ambientes.
Dos o más instructores pueden orientar una o más competencias en un mismo ambiente.
Usuarios: los usuarios que interactúan con la base de datos.

Coordinador académico.
Instructores del SENA centro de Teleinformática y PI Regional Cauca.
Informes por Usuario: los informes necesarios para mostrar a los usuarios identificados son:

- Visualizar el horario de todos los instructores de planta.
- Visualizar el horario de todos los instructores de contrato.
- Visualizar el horario de un solo instructor.
- Visualizar el horario de todos los ambientes.
- Visualizar el horario de un solo ambiente.
- Visualizar el horario de todos los programas de formación.

Modelo entidad relación

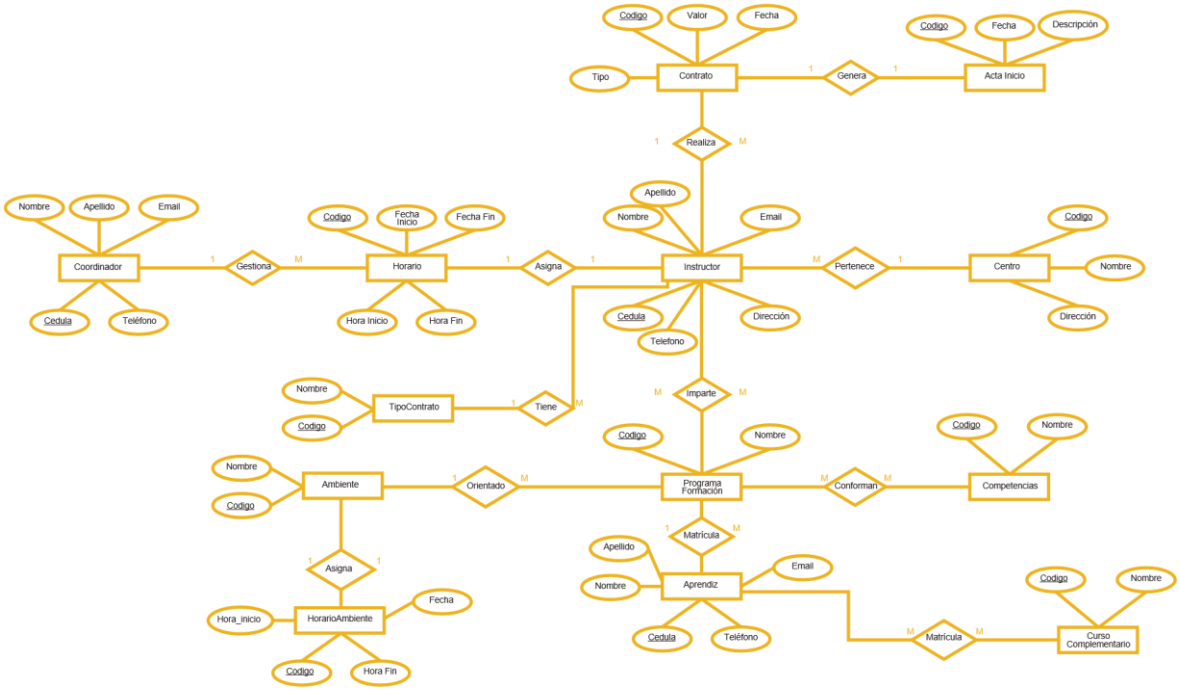


Figura 12. Modelo entidad relación

Modelo relacional

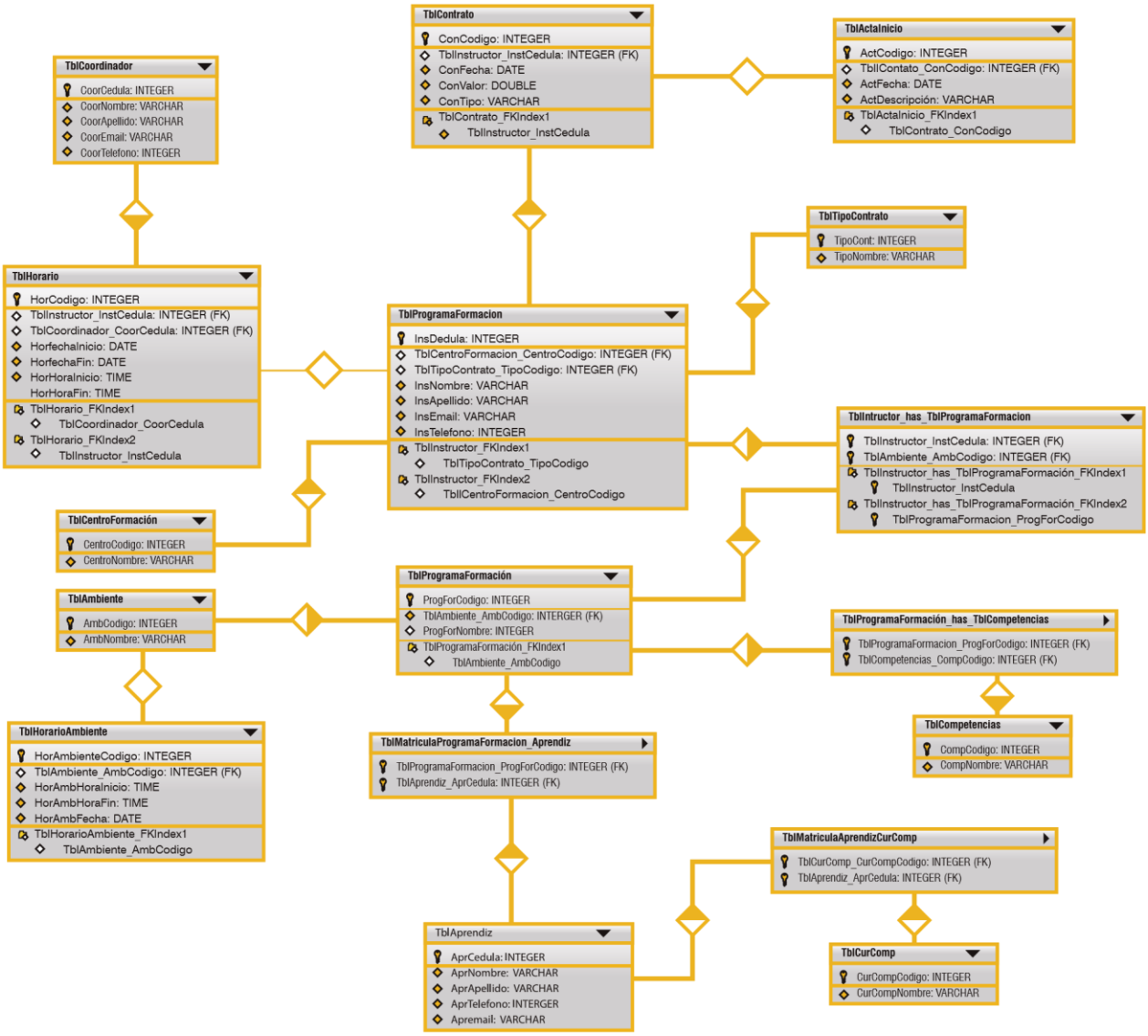


Figura 13. Modelo relacional

3. Construccion transformacion MER-extendido a relacional

El MER-Extendido ha sido creado para modelar otros tipos de relaciones que no contempla el MER tradicional, los cuales son la generalización y especialización, recursivas, inclusividad y exculsividad. A continuación se muestra como se transforma cada una de estas relaciones a un modelo relacional.

3.1 Relación de especialización y generalización

Recordar que en un MER-Extendido la generalización puede identificarse como solapamiento, exclusividad, parcialidad o totalidad, en estas relaciones se encuentran entidad superclase o supertipos y entidades subclase o subtipos.

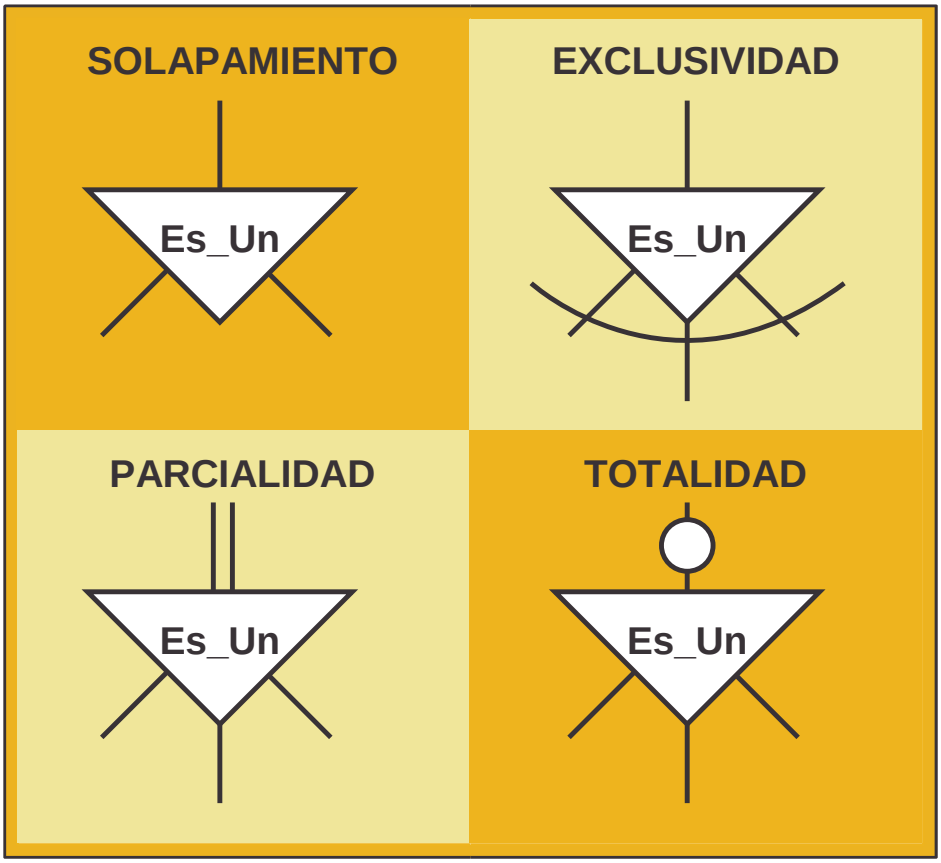


Figura 14. Relación de generalización de un MER extendido

Existen cuatro opciones para la transformación de estos tipos de relaciones.
1. Si es solapada y parcial: crear una relación (tabla) para la superclase, con sus atributos correspondientes y una relación (tabla) para cada subclase con sus atributos incluyendo la clave primaria de la superclase.
Generalización: esta representación es solapada y parcial, es decir una persona puede ser profesor y estudiante a la vez y no es obligatorio que asuma un subtipo.

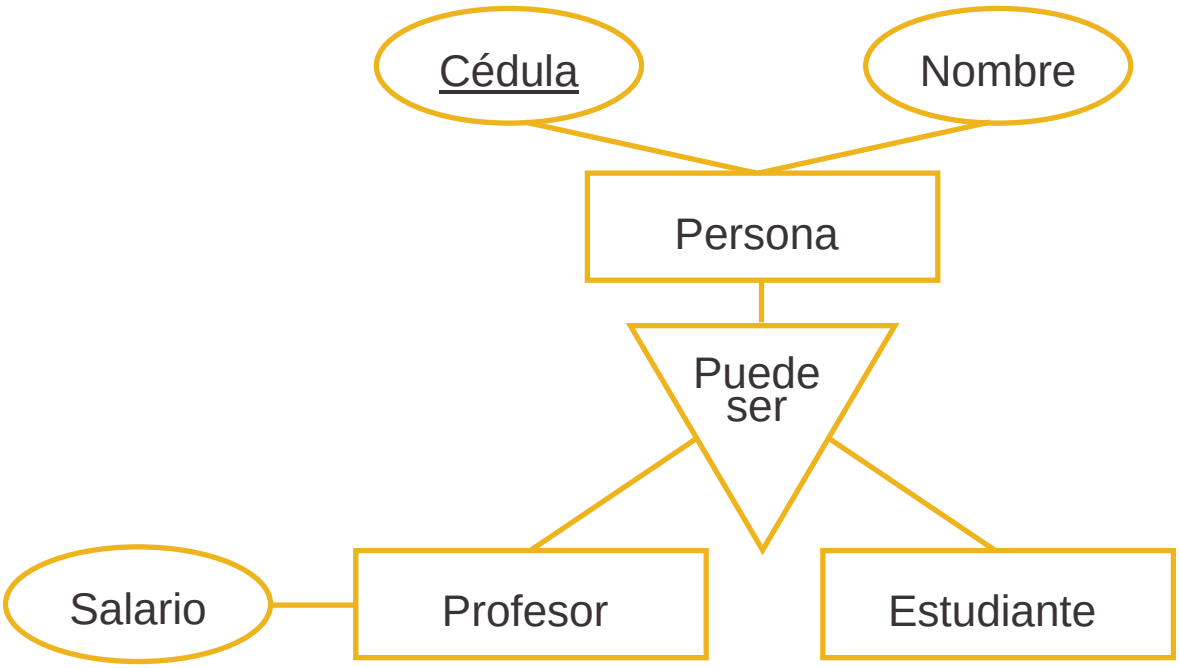


Figura 15. Generalización

Transformación a esquema relacional.

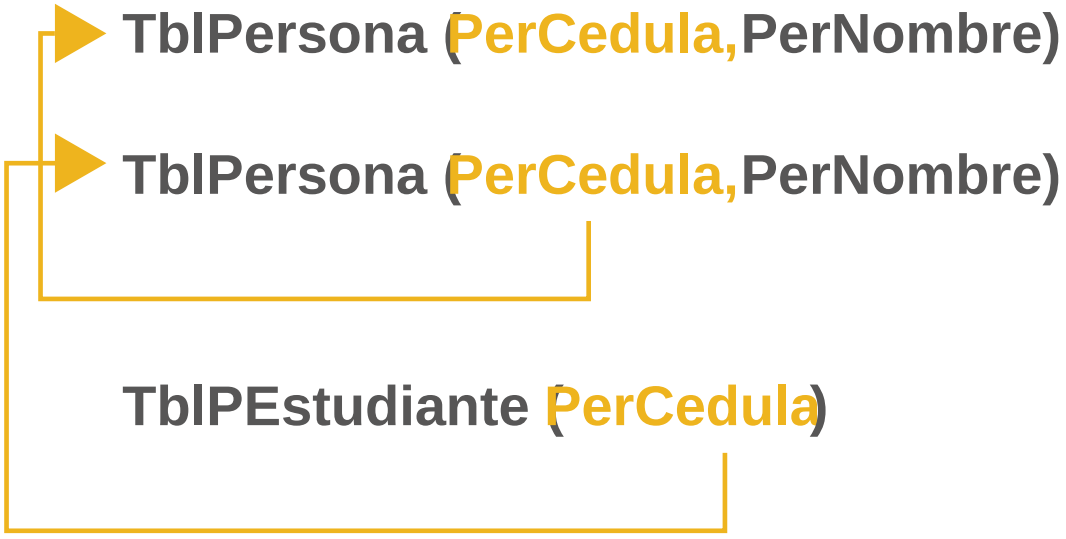


Figura 16. Transformación a esquema relacional

Transformación modelo relacional

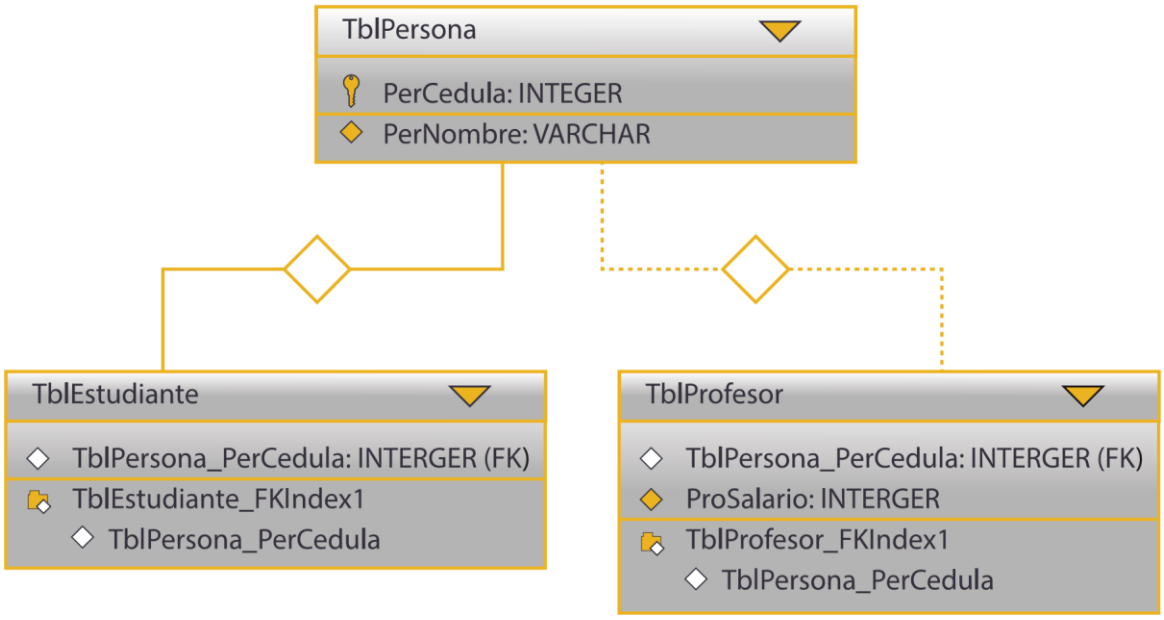


Figura 17. Transformación a modelo relacional

3.2 Relación recursiva

Las relaciones recursivas se tratan de la misma forma que las otras, sólo que hay que imaginar que la tabla se divide en dos, una por cada rol. Teniendo en cuenta eso, la solución es idéntica a lo ya resuelto en los casos anteriores.

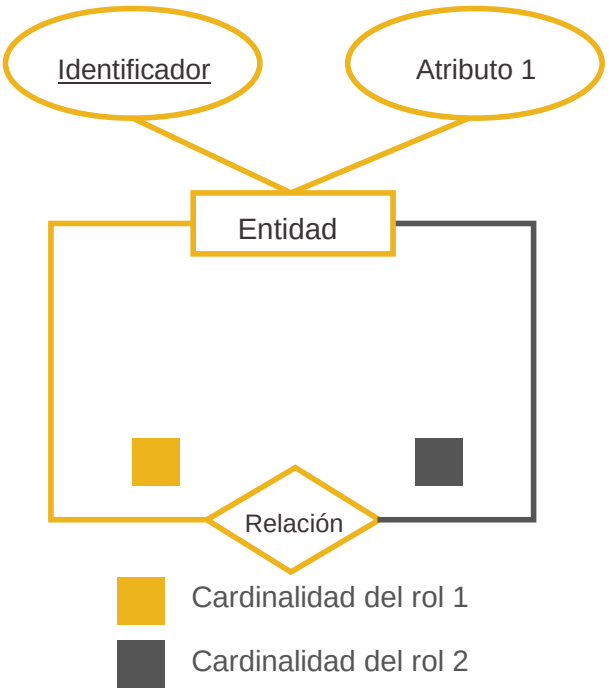
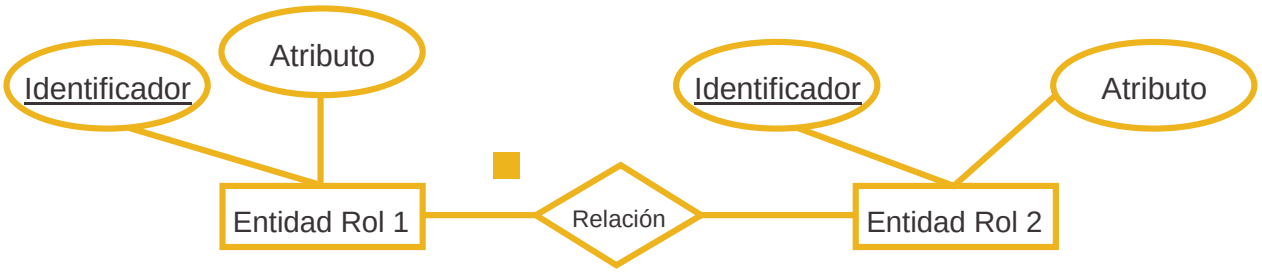


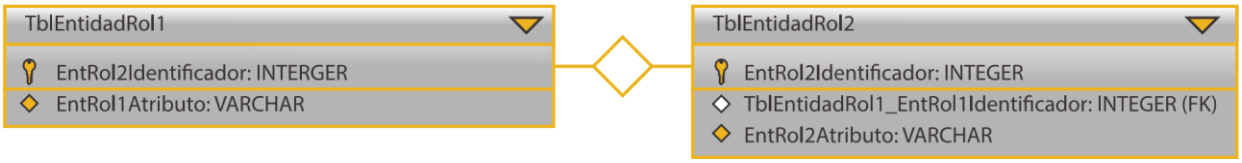
Figura 18. Relación recursiva

La forma de imaginarse las relaciones recursivas para resolverlas es:



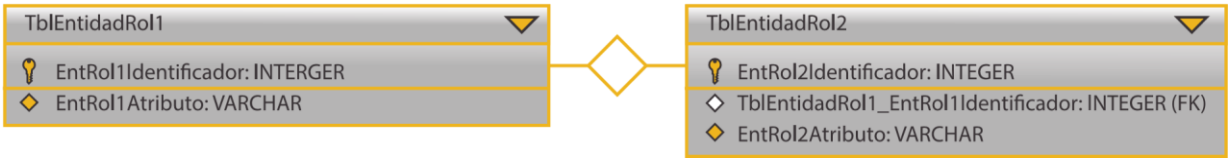
Donde se "duplica la entidad" y se conservan las cardinalidades, luego dependiendo de ésta, se aplica la regla de transformación correspondiente teniendo las siguientes posibilidades:

1 caso: si la cardinalidad de la relación es de 1 a 1.

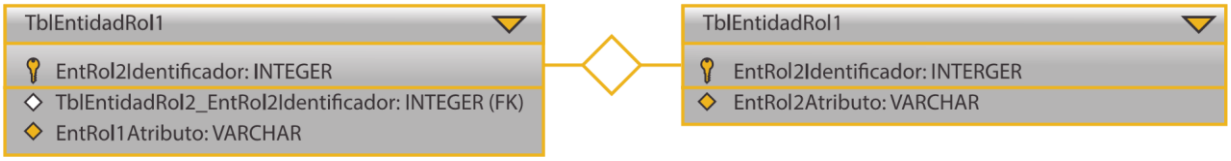


La entidad con Rol débil hereda el atributo de identificación de la otra entidad y pasa a ser su llave foránea

2 caso: si la cardinalidad de la relación es de 1 a M:

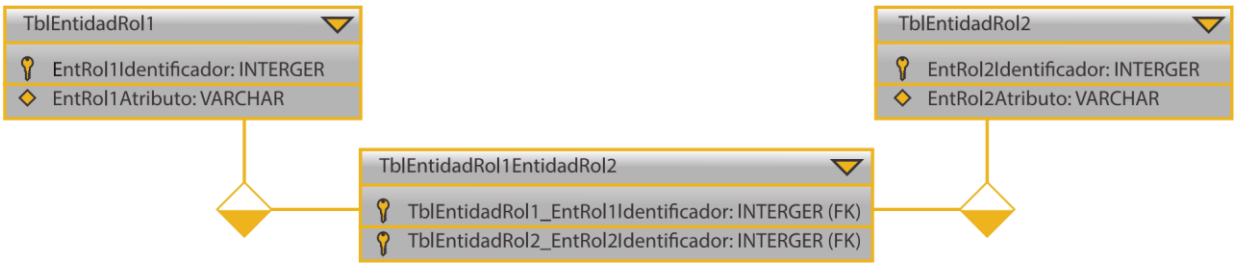


O de M a 1



En cualquiera de los casos, la entidad con relación muchos hereda el atributo de identificación de la otra y esta se convierte en la llave foránea

3 caso: si la cardinalidad de la relación es de M a M:



Se genera en este caso una tercera tabla que relaciona las dos entidades, está compuesta por los atributos de identificación que hereda de cada entidad, más los atributos propios de la relación (si existieran).

4. Normalizacion

Una vez finalizado el proceso de conversión, se pueden presentar ciertos inconvenientes tales como:

- Incapacidad de almacenar ciertos hechos.
- Redundancias y por tanto posibilidad de inconsistencias.
- Ambigüedades.
- Perdida de información.
- Perdida de ciertas restricciones de integridad que dan lugar a interdependencia de los datos.
- Existencia de valores nulos.
- Aparición de estados inválidos en los proceso de manipulación.

La normalización es el proceso mediante el cual se transforman datos complejos a un conjunto de estructuras de datos más pequeñas, que además de ser más simples y más estables, son más fáciles de mantener. También se puede entender la normalización como una serie de reglas que sirven para ayudar a los diseñadores de bases de datos a desarrollar un esquema que minimice los problemas de lógica. Cada regla está basada en la que le antecede.
La normalización también hace las cosas fáciles de entender. Los seres humanos tenemos la tendencia de simplificar las cosas al máximo. Lo hacemos con casi todo, desde los animales hasta con los automóviles. Vemos una imagen de gran tamaño y la hacemos más simple agrupando cosas similares juntas. Las guías que la normalización provee crean el marco de referencia para simplificar una estructura de datos compleja.
Otra ventaja de la normalización de base de datos es el consumo de espacio. Una base de datos normalizada ocupa menos espacio en disco que una no normalizada. Hay menos repetición de datos, lo que tiene como consecuencia un mucho menor uso de espacio en disco.



Una vez obtenido el esquema relacional resultante del esquema entidad/ relación que representa la base de datos, normalmente se tendrá una buena base de datos. Pero otras veces, debido a fallos en el diseño o a problemas indetectables, se tendrá un esquema que puede producir una base de datos que incorpore estos problemas:
Redundancia: se llama así a los datos que se repiten continua e innecesariamente por las tablas de las bases de datos. Cuando es excesiva es evidente que el diseño hay que revisarlo, es el primer síntoma de problemas y se detecta fácilmente.
Ambigüedades: datos que no clarifican suficientemente el registro al que representan. Los datos de cada registro podrían referirse a más de un registro o incluso puede ser imposible saber a qué ejemplar exactamente se están refiriendo. Es un problema muy grave y difícil de detectar.

Perdida restricciones de integridad: normalmente debido a dependencias funcionales. Más adelante se explica este problema. Se arreglan fácilmente siguiendo una serie de pasos concretos.
Anomalías en operaciones de modificación de datos: el hecho de que al insertar un solo elemento haya que repetir tuplas en una tabla para variar unos pocos datos. O que eliminar un elemento suponga eliminar varias tuplas necesariamente (por ejemplo que eliminar un cliente suponga borrar seis o siete filas de la tabla de clientes, sería un error muy grave y por lo tanto un diseño terrible).
El principio fundamental reside en que las tablas deben referirse a objetos o situaciones muy concretas, relacionados exactamente con elementos reconocibles por el sistema de información de forma inequívoca. Cada fila de una tabla representa inequívocamente un elemento reconocible en el sistema. Lo que ocurre es que conceptualmente es difícil agrupar esos elementos correctamente.
En cualquier caso la mayor parte de problemas se agravan si no se sigue un modelo conceptual y se decide crear directamente el esquema relacional. En ese caso, el diseño tiene una garantía casi asegurada de funcionar mal.
Cuando aparecen los problemas enumerados, entonces se les puede resolver usando reglas de normalización. Estas reglas suelen forzar la división de una tabla en dos o más tablas para arreglar ese problema.

4.1 Grado de normalización

Las formas normales corresponden a una teoría de normalización iniciada por el propio Coddy continuada por otros autores (entre los que destacan Boycey Fagin). Codd definió en 1970 la primera forma normal, desde ese momento aparecieron la segunda, tercera, la Boyce-Codd, la cuarta y la quinta forma normal.
Una tabla puede encontrarse en primera forma normal y no en segunda forma normal, pero no al contrario. Es decir los números altos de formas normales son más restrictivos (la quinta forma normal cumple todas las anteriores).

4.1.1 Primera forma normal 1N

Es una forma normal inherente al esquema relacional. Es decir toda tabla realmente relacional la cumple.
Se dice que una tabla se encuentra en primera forma normal si impide que un atributo de una tupla pueda tomar más de un valor. La tabla:

EMPLEADO		
identificación	nombre	departamento
12345	Pedro Murcia	Sistemas
24680	Maria Dolores Rosas	Contabilidad
12232	Juan José Muñoz	Gerencia Recursos Humanos

Lo que se ve, es una tabla, pero no una tabla relacional (lo que en terminología de bases de datos relacionales se llama relación). No cumple la primera forma normal.
Se podría mejorar para que cumpla la primera forma normal así:

EMPLEADO		
identificación	nombre	departamento
12345	Pedro Murcia	Sistemas
24680	María Dolores Rosas	Contabilidad
12232	Juan José Muñoz	Gerencia

12232	Juan José Muñoz	Recursos Humanos

Asignación de competencias:

4.1.2 Segunda forma normal 2N

Si una tabla está en primera forma normal y además cada atributo que no sea clave, depende de forma funcional completa respecto de cualquiera de las claves. Toda la clave principal debe hacer dependientes al resto de atributos, si hay atributos que depende sólo de parte de la clave, entonces esa parte de la clave y esos atributos formarán otra tabla.
Para asignar una competencia a un instructor es necesario que esté legalmente contratado por el centro o que tenga resolución de nombramiento y según su especialidad se le asignan las competencias que va a orientar.

Alumno				
identificacion	codCurso	nombre	apellido	nota
11	34	Sofia	Loren	5
11	25	Sofía	Loren	4
12	34	Marcos	Pinto	3.5
13	25	Mireya	Ruano	4.5
13	34	Mireya	Ruano	5

Suponiendo que la identificación y el código del curso formen una clave principal para esta tabla, sólo la nota tiene dependencia funcional completa. El nombre y el apellido dependen de forma completa de la identificación. La tabla no es 2FN, para convertirla se debe convertir esa tabla en dos tablas así:

Alumno		
identificacion	nombre	apellido
11	Sofia	Loren
11	Sofía	Loren
12	Marcos	Pinto
13	Mireya	Ruano
13	Mireya	Ruano
NotasCurso		
identificacion	codCurso	nota
11	34	4
11	25	4
12	34	3.5
13	25	4.5

13	34	5

4.1.3 Tercera forma normal 3N

Ocurre cuando una tabla está en 2FN y además ningún atributo que no sea clave depende transitivamente de las claves de la tabla. Es decir no ocurre cuando algún atributo depende funcionalmente de atributos que no son clave.

Ejemplo:

ALUMNOS				
identificación	nombre	apellido	codCiudad	ciudad
31	Pedro	Picapiedra	100	Cali
32	Mónica	Galindo	100	Cali
33	Amparo	Grisales	101	Popayán
34	Roberto	Gómez	102	Neiva
35	Blanca	Moreno	102	Neiva

En la tabla anterior la ciudad depende funcionalmente del código de la ciudad, lo que hace que no esté en 3FN. La manera de solucionar el problema sería dividiendo la tabla en dos así:

ALUMNOS			
identificación	nombre	apellido	codCiudad
31	Pedro	Picapiedra	100
32	Mónica	Galindo	100
33	Amparo	Grisales	101

En la primera tabla creamos la tabla Alumnos con los atributos propios de él y solo se agrega el atributo de codCiudad para saber a qué ciudad pertenece.

ALUMNOS

CI JDAD	
codCiudad	Ciudad
100	Cali
100	Cali
101	Popayán
102	Neiva
102	Neiva

La segunda tabla corresponde a la ciudad con sus atributos.

4.1.4 Forma normal Boyce-Cood(FNBC o BCFN)

Ocurre si una tabla está en tercera forma normal y además todo determinante es una clave candidata.

Ejemplo:

EMPRESA		
Empleado	Departamento	Responsable
Pedro	Dirección	Juan
Mari	Ventas	Luis
Jose	Mercadeo	Erika
Rosa	Ventas	Roció
Blanca	Gerencia	Andrés
Julian	Ventas	Mireya

La cuestión es que un empleado puede trabajar en varios departamentos. En cada departamento hay varios responsables, pero cada empleado sólo tiene asignado uno. El detalle importante que no se ha tenido en cuenta, es que el o la responsable sólo puede ser responsable en un departamento. Este detalle último produce una dependencia funcional ya que:

Por lo tanto se encuentra un determinante que no es clave candidata. No está por tanto en FNBC. En este caso la redundancia ocurre por mala selección de clave. La redundancia del departamento es completamente evitable. La solución sería:

PERSONAL	
Empleado	Responsable
Pedro	Juan
Mari	Luis
Jose	Erika
Rosa	Roció
Blanca	Andrés
Julian	Mireya
RESPONSABLES	
Responsable	Departamento
Juan	Dirección
Luis	Ventas
Erika	Mercadeo
Roció	Ventas
Andrés	Gerencia
Mireya	Ventas

En las formas de Boyce-Codd hay que tener cuidado al descomponer ya que se podría perder información por una mala descomposición.

4.1.5 Cuarta forma normal (4FN) dependencias multievaluadas.

Para el resto de formas normales (las diseñadas por Fagin, mucho más complejas), es importante definir este tipo de dependencia, que es distinta de las funcionales. Si las funcionales eran la base de la segunda y tercera forma normal (y de la de Boyce-Codd), éstas son la base de la cuarta forma normal.

Una dependencia multivaluada de X sobre Y (es decir X->>Y), siendo X e Y atributos de la misma tabla, ocurre cuando Y tiene un conjunto de valores bien definidos sobre cualquier valor de X. Es decir, dado X sabremos los posibles valores que puede tomar Y.
Se refiere a posibles valores (en plural) y se trata de que los valores de ese atributo siempre son los mismos según el valor de un atributo y no del otro.

Ejemplo:

Cód go Curso	nstructor	Mater a
11	Marta	1
11	Marta	2
11	Jaime	1
11	Jaime	2
12	Marta	1
12	Marta	2
12	Marta	3

La tabla cursos, instructores y materiales del curso. La tabla está en FNBC ya que no hay dependencias transitivas y todos los atributos son clave sin dependencia funcional hacia ellos. Sin embargo hay redundancia. Los materiales se van a repetir para cualquier instructor dando cualquier curso, ya que los instructores van a utilizar todos los materiales del curso (de no ser así no habría ninguna redundancia).
Los materiales del curso dependen de forma multivaluada del curso y no del instructor en una dependencia multivaluada (no hay dependencia funcional ya que los posibles valores son varios). Para el Código curso e Instructor podemos saber los materiales; pero se sabe por el curso y no por el instructor.

Cuarta forma normal

Ocurre esta forma normal cuando una tabla está en forma normal de BoyceCodd y toda dependencia multivaluada no trivial es una dependencia funcional. Son triviales aquellas dependencias multivaluadas en las que el conjunto formado por el determinante y el implicado no forman la clave primaria de la tabla y además el implicado no forma parte del determinante: es decir si X->>Y y además Y ¢ X y X,Y no es la clave de la tabla, tenemos una dependencia multivaluada no trivial (como ocurre en el ejemplo anterior).

Codigo Curso	Material
11	1
11	2
12	1
12	2
12	3

Codigo Curso	Instructor
11	Marta

11	Jaime
12	Marta

4.1.6 Quinta forma normal (5FN) dependencia de join o de reunión

Una proyección de una tabla es la tabla resultante de tomar un subconjunto de los atributos de una. Es decir una tabla formada por unas cuantas columnas de la tabla original.
La operación JOIN procedente también del álgebra relacional, consiste en formar una tabla con la unión de dos tablas. La tabla resultante estará formada por la combinación de todas las columnas y filas de ambas, excepto las columnas y filas repetidas.
Se dice que se tiene una tabla con dependencia de unión (o de tipo JOIN) si se puede obtener esa tabla como resultado de combinar mediante la operación JOIN varias proyecciones de la misma.
Quinta forma normal o forma normal de proyección-unión: ocurre cuando una tabla está en 4FN y cada dependencia de unión (JOIN) en ella es implicada por las claves candidatas.

Esta forma normal es la más compleja y polémica de todas. Es polémica porque no es claro que al pasar a 5FN mejore la base de datos, premisa definida por **Fagin**. Es raro encontrarse este tipo de problemas cuando la normalización llega a 4FN. Se deben a restricciones semánticas especiales aplicadas sobre la tabla.
Ejemplo:

Proveedor	Material	Proyecto
1	1	2
1	2	1
2	1	1
1	1	1

Indican códigos de material suministrado por un proveedor y utilizado en un determinado proyecto. Así vista la tabla, no permite ninguna proyección en la que no perdamos datos. Pero si ocurre una restricción especial como por ejemplo: cuando un proveedor nos ha suministrado alguna vez un determinado material, si ese material aparece en otro proyecto, haremos que el proveedor anterior nos suministre también ese material para el proyecto.
Eso ocurre en los datos como el proveedor número 1 nos suministró el material número 1 para el proyecto 2 y en el proyecto 1 utilizamos el material 1, aparecerá la tupla proveedor 1, material 1 y proyecto 1. Si un nuevo proyecto necesitará el material 1, entonces habrá que pedirlo a los proveedores 1 y 2 (ya que en otros proyectos los hemos utilizado).
La dependencia de reunión que produce esta restricción es muy difícil de ver ya que es lejana. Para esa restricción esta proyección de tablas sería válida:

Proveedor	Material
1	1
1	2
2	1
Proveedor	**Proyecto**
1	2
1	1
2	1

Material	Proyecto
1	2
2	1
1	1

Esa descomposición no pierde valores en este caso, sabiendo que si el proveedor nos suministra un material podremos relacionarle con todos los proyectos que utilizan ese material. Una tabla no está en quinta forma normal si hay una descomposición de esa tabla que muestre la misma información que la original y esa descomposición no tenga como clave la clave original de la tabla.

Forma Normal de Dominio Clave (FNDC):

Se la conoce más con sus siglas en inglés DKNF. Se trata de una forma normal enunciada también por Faginen 1981 al darse cuenta de los problemas de redundancia que ocurrían con algunos dominios. En este caso no se basó en dependencias entre los datos, sino que se basó en restricciones de dominio y restricciones de clave.

Restricciones de dominio: se trata de la restricción que hace que un determinado atributo obtenga sólo ciertos valores, los que estén de acuerdo a la definición de un determinado dominio.

Restricción de clave: es la restricción que permite que un atributo o un conjunto de atributos forme una clave candidata.

Fagin dice que una tabla está en FNDC si toda restricción sobre la tabla es consecuencia lógica de aplicar las restricciones de dominio y clave sobre la misma. Fagin demostró que si esto ocurría la tabla incluso estaba en. Ejemplo:

Alumno	Nivel Logrado	Nota
Rosa	Muy alto y eficiente	9
María	Muy alto aunque sin trabajo constante	9
Juan	Alto con trabajo habitual	7
José	Medio aunque con trabajo	5

Observando los datos de la tabla se observa que cuando la nota es igual o superior a 9, en el nivel aparece la palabra alto, cuando es un 7 o un 8 medio, y un 5 o 6 sería medio. Es decir tenemos restricciones que no son ni de dominio ni de clave en esa tabla. Lo lógico sería:

A umno	N ve ograd o	Nota
Rosa	ficiente	9
María	rabajo constante	9
Juan	rabajo habitual	7
José	rabajo medio	5
N ve acadé m co	Nota mín ma	Nota máx ma

Rosa	9	10
María	7	8,99
Juan	5	6,99
José	0	4,99

No se pierde información al diseñar las tablas de esta forma y de hecho es más eficiente para la base de datos.

Glosario

Modelo Entidad Relación: modelo que consta de entidades, atributos y relaciones y permite modelar conceptualmente una base de datos relacional.
Modelo Entidad Relación Extendido: modelo que permite modelar nuevos tipos de relaciones, que no incluye el MER normal, tales como agregación, generalización, exclusión e inclusión.
Modelo Relacional: es un modelo que se obtiene de la tabla: es la representación de una entidad en un modelo relacional, la cual incluye atributos.

Campo: es la representación de un atributo en un modelo relacional el cual contiene su nombre, tipo de dato y descripción de si es clave primaria o llave foránea.

Clave Primaria: atributo único que no se repite, es identificado en un modelo relacional porque esta subrayado o tiene el estereotipo <pk> y en un esquema relacional se encuentra subrayado.
Clave Foránea: atributo que es pasado a una tabla, según la regla de transformación, se distingue en un modelo relacional porque tiene el estereotipo <fk> y en un esquema relacional porque despliega una flecha hacia la tabla de la cual proviene.

Relación: es una asociación que existe entre dos entidades, en un MER se distinguen tres tipos de relaciones: uno a uno (1..1), uno a muchos (1..M) y muchos a muchos (M..M).

Esquema Relacional: expresar el modelo relacional por medio un esquema de palabras y asociaciones entre ellas (flechas) que representan las relaciones.

1FN: abreviatura de Primera Forma Normal. 2FN: abreviatura de Segunda Forma Normal.

3FN: abreviatura de Tercera Forma Normal.

4FN: abreviatura de Cuarta Forma Normal. 5FN: abreviatura de Quinta Forma Normal.

FNBC: abreviatura de Froma Normal de BoyceCood.