Desarrollo de aplicaciones distribuidas con Visual Studio 2013 y MS SQLServer 2014

BASADO EN EL PATRON MVC

Ms. Ing. Camilo E. Suárez Rebaza
UNIVERSIDAD NACIONAL DEL SANTA | ESCUELA DE INGENIERÍA DE SISTEMAS

Desarrollo de Aplicaciones distribuidas con Visual Studio 2013 y SQLServer

1 DESCRIPCIÓN DEL CASO

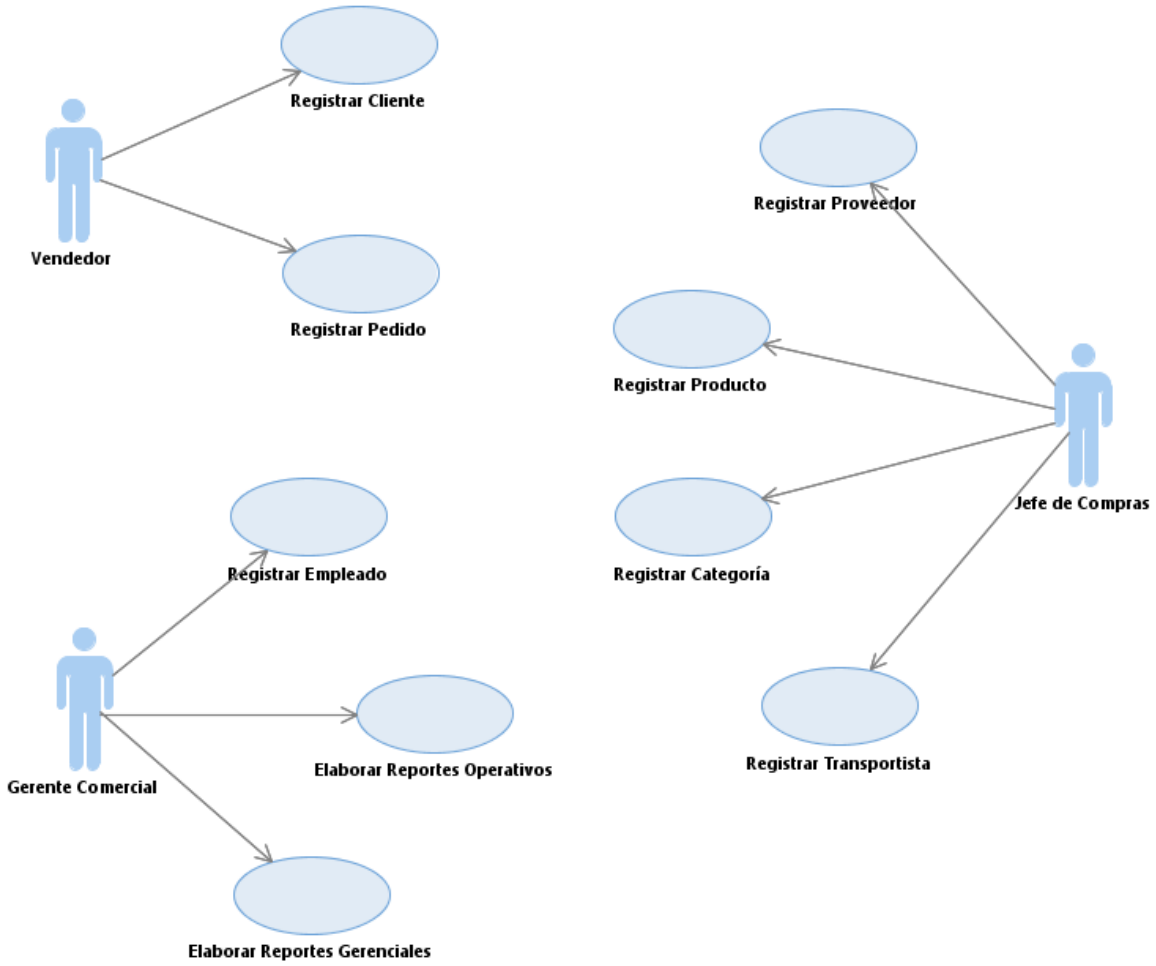
El Caso de estudio trata de la implementación de un sistema de gestion de pedidos basado sobre la base de datos Northwind, que viene a ser la base de datos de ejemplo de Microsoft SQL Server®.

El objetivo es desarrollar una aplicación que permita ingresar datos y procesar la información contenida en la mencionada base de datos, siguiendo el enfoque Modelo-Vista-Controlador; es decir las clases de la aplicación deberán estar agrupadas según las capas requeridas por el patron MVC.

La aplicación se desarrollará utilizando el Visual Studio 2013, concretamente el Visual Net, aprupando por proyestos las diferentes capas del patrón MVC: Logica de Presentación, Logica de Negocio, Lógica de acceso a Datos y Lógica de entidades.

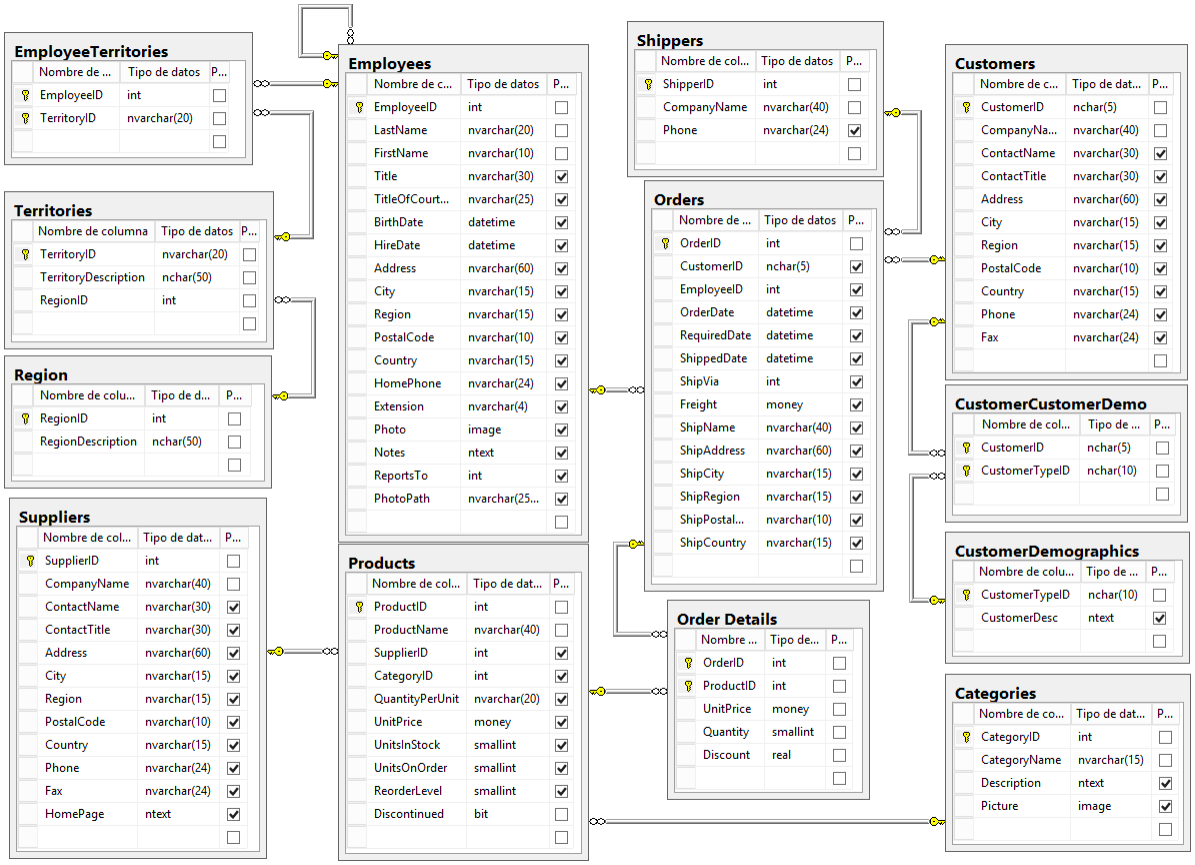
2 MODELADO UML DEL CASO

EL diagrama de casos de uso inicial es como sigue:



3 MODELO DE BASE DE DATOS

El caso utiliza la base de datos de ejemplo “Northwind” de MS SQL Server, la cual tiene la siguiente estructura:

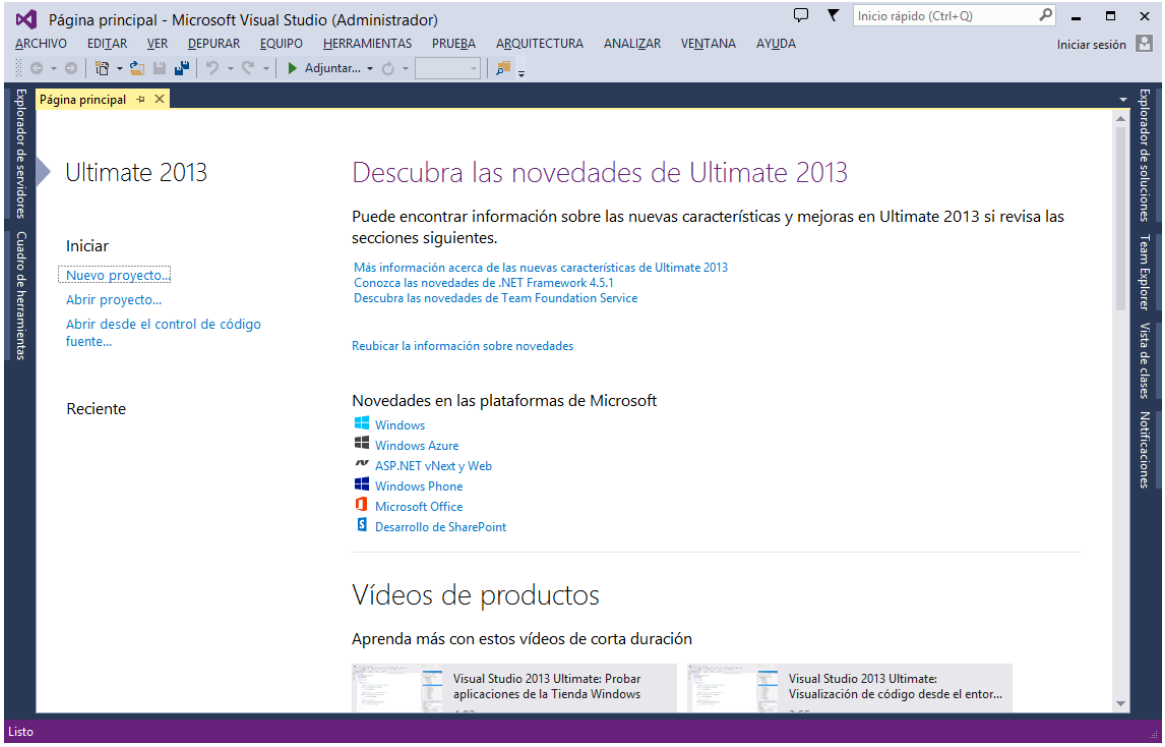


Se ingresarán datos y procesaran informes y reportes para las siguientes tablas:

Caso de Uso	Tablas
Registrar Cliente	Customers
Registrar Pedido	Orders, Order Details
Registrar Empleado	Employees
Registrar Proveedor	Suppliers
Registrar Producto	Products
Registrar Categoría	Categories
Registrar Transportista	Shippers
Elaborar Reportes Operativos	Varias
Elaborar Reportes Gerenciales	Varias

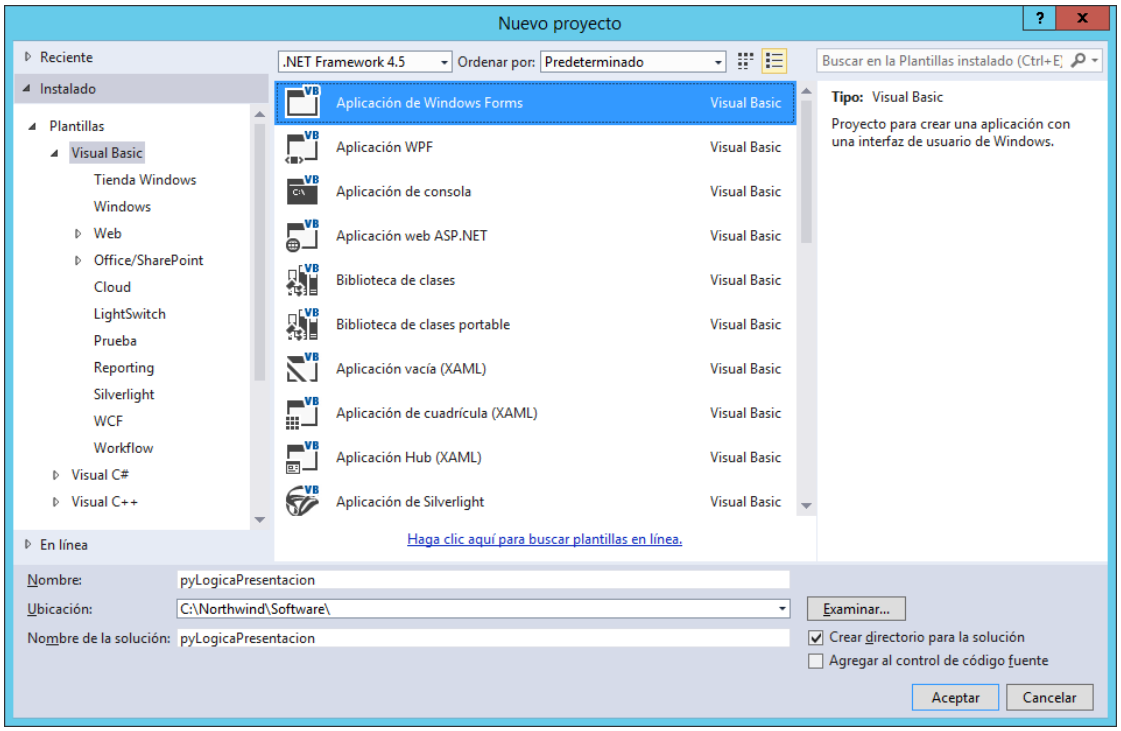
4 CREACIÓN DE LA SOLUCIÓN EN VISUAL STUDIO

La Solución de Visual Studio para el caso se compone de 4 proyectos diferentes e interrelacionados, iniciamos el Visual Studio 2013, y obtenemos:

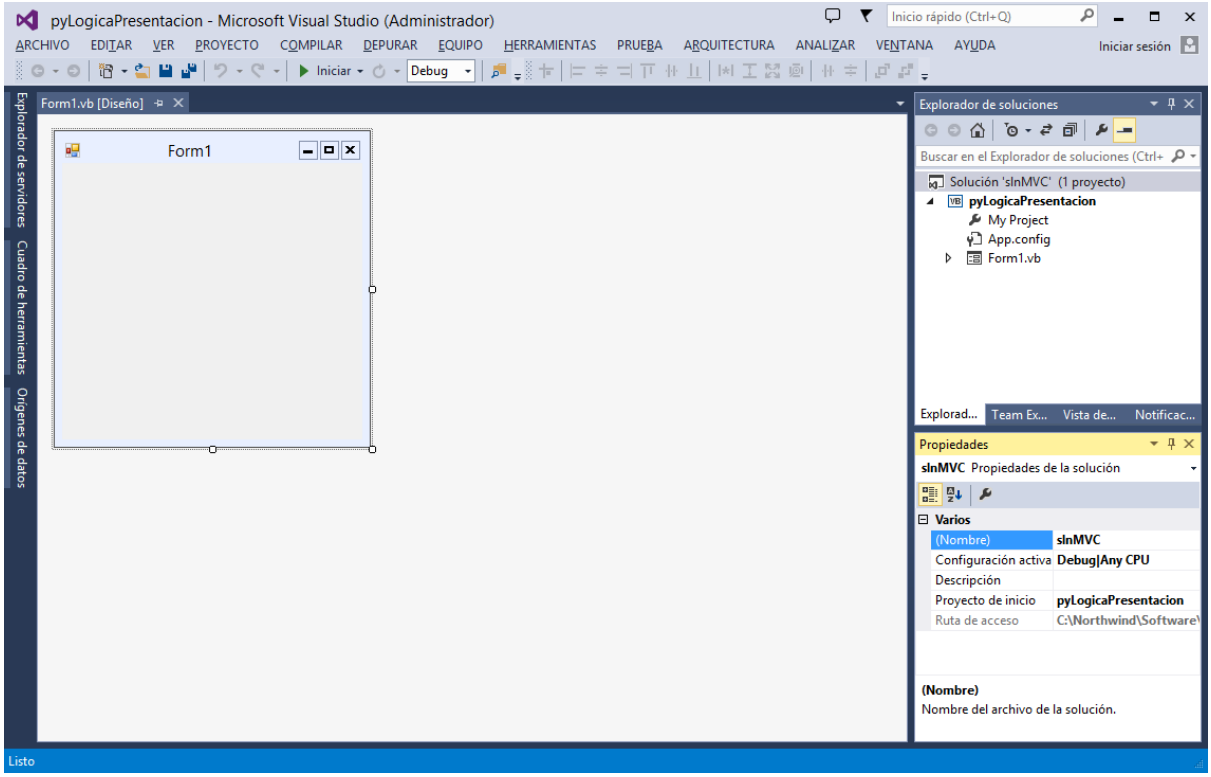


4.1 CAPA DE LÓGICA DE PRESENTACIÓN

La solución empieza con la creación el proyecto para los formularios de la lógica de presentación, hacemos clic en “Nuevo Proyecto” (Windows Forms) y configuramos como sigue:



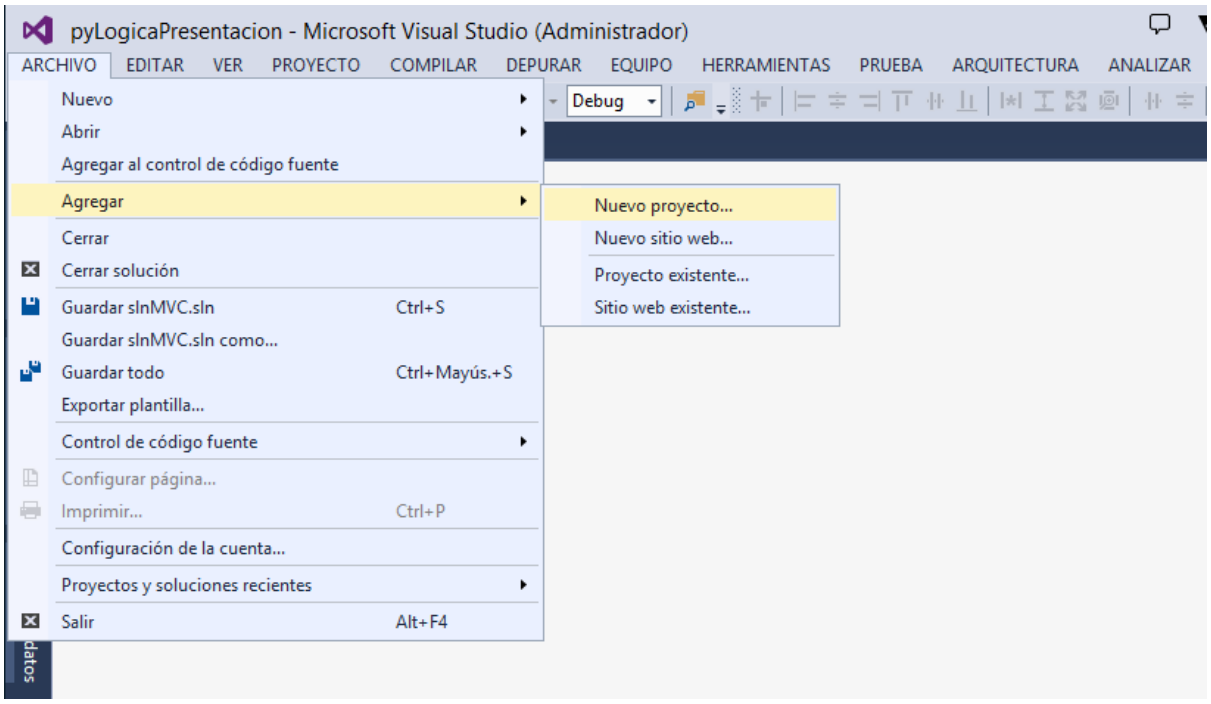
Clic en aceptar y tenemos nuestro primer proyecto creado. Cambiamos el nombre de la solución de la siguiente manera (La solución por defecto asume el nombre del primer proyecto):



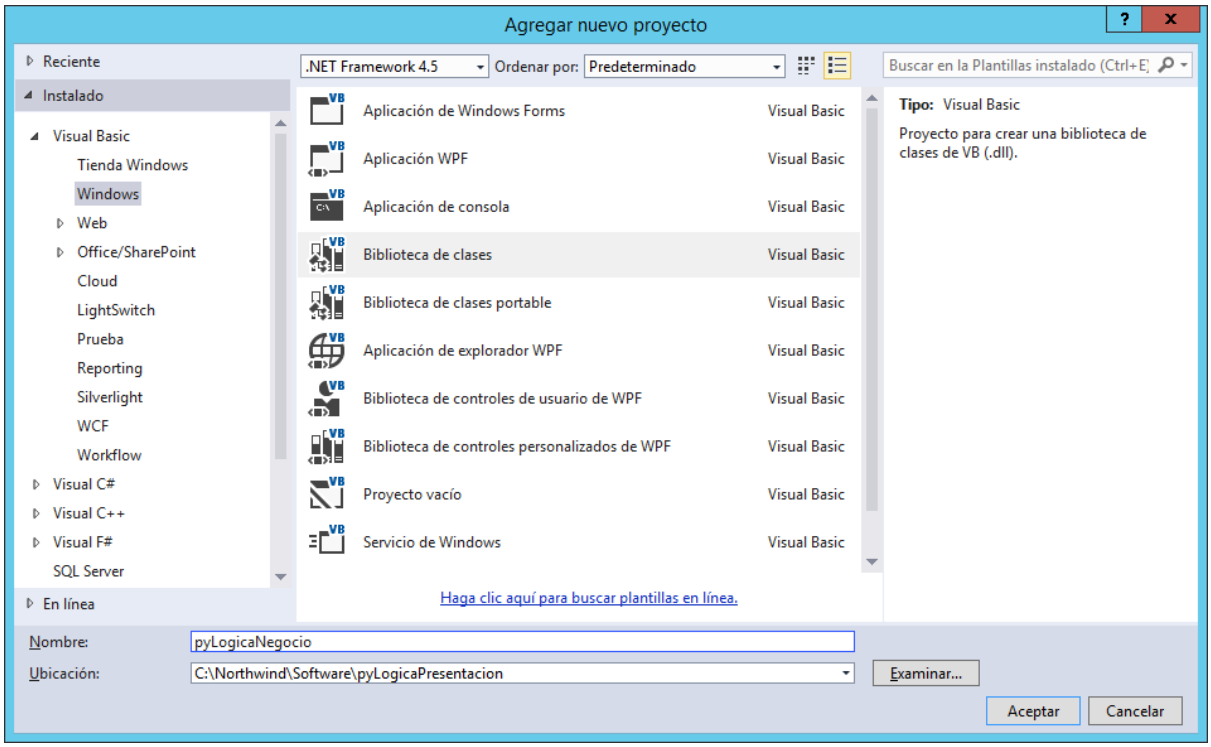
Luego seguimos agregando los demás proyectos, uno por cada capa.

4.2 Capa de Lógica de Negocio

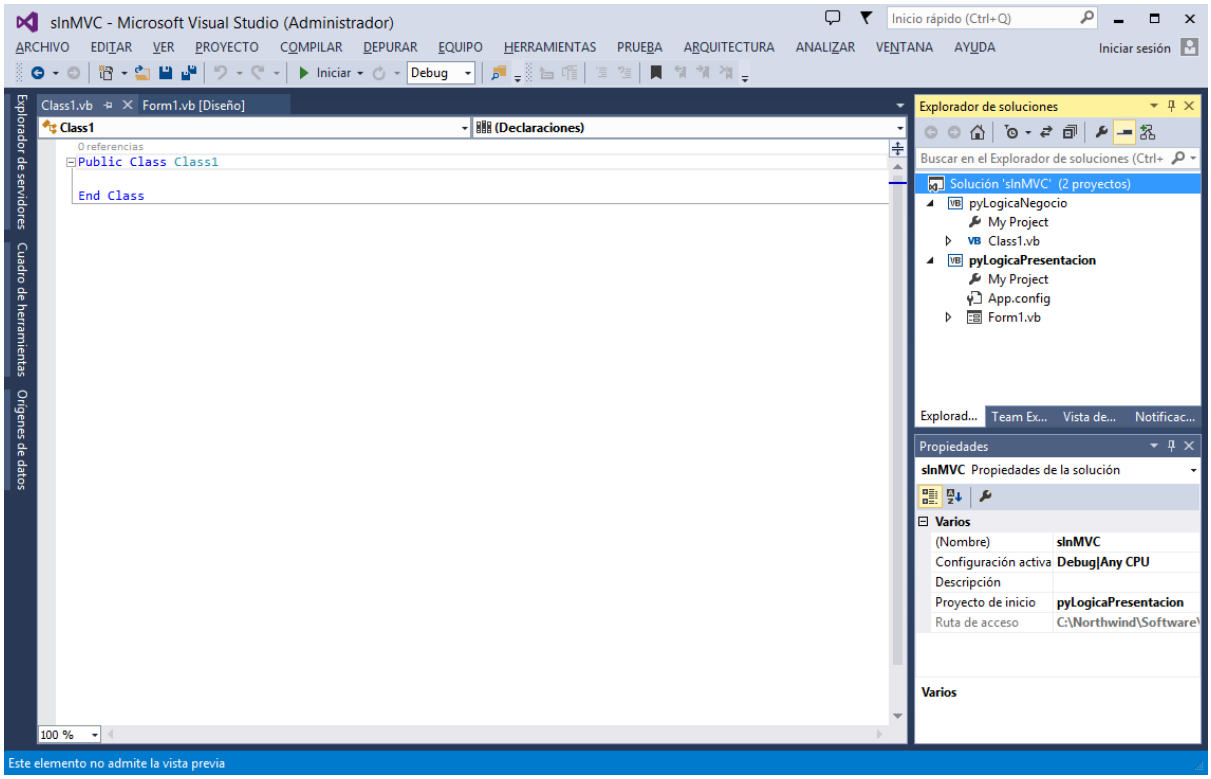
Para agregar el proyecto correspondiente a ésta capa hacemos clic en ***"Menú Archivo > Agregar > Nuevo Proyecto"***:



Escogemos “Biblioteca de Clases” y configuramos como se muestra a continuación:



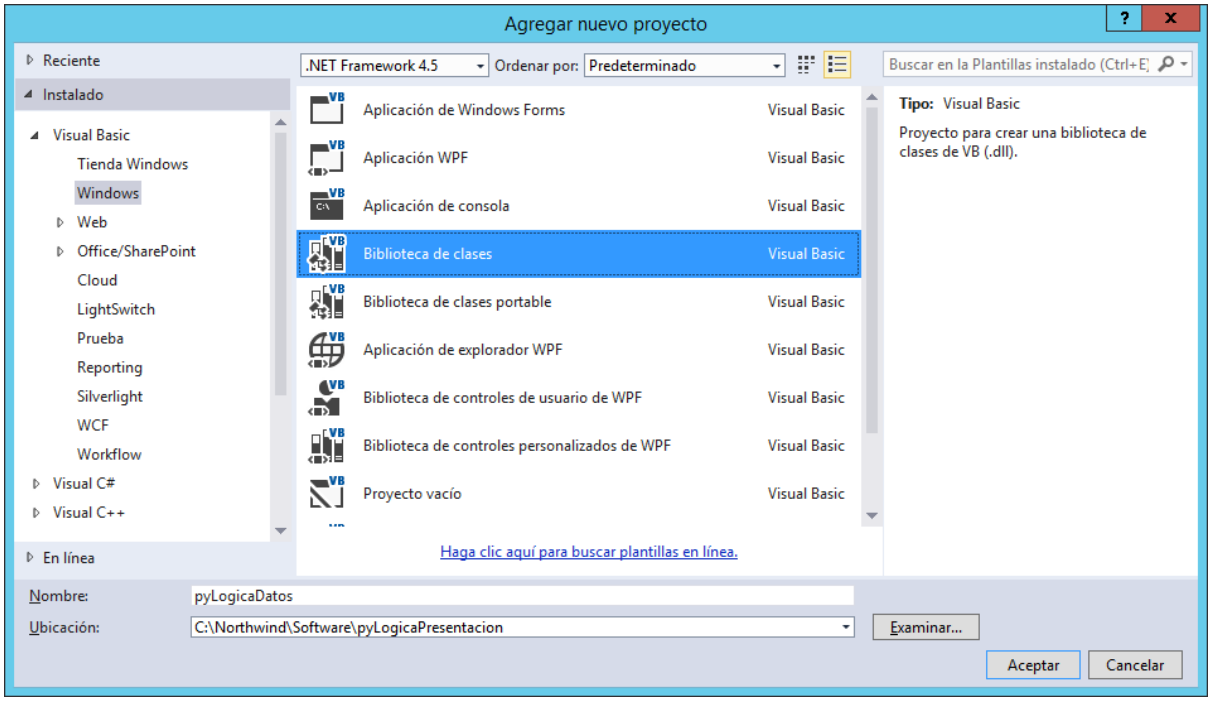
Clic en <Aceptar> y tendremos nuestro segundo proyecto creado:



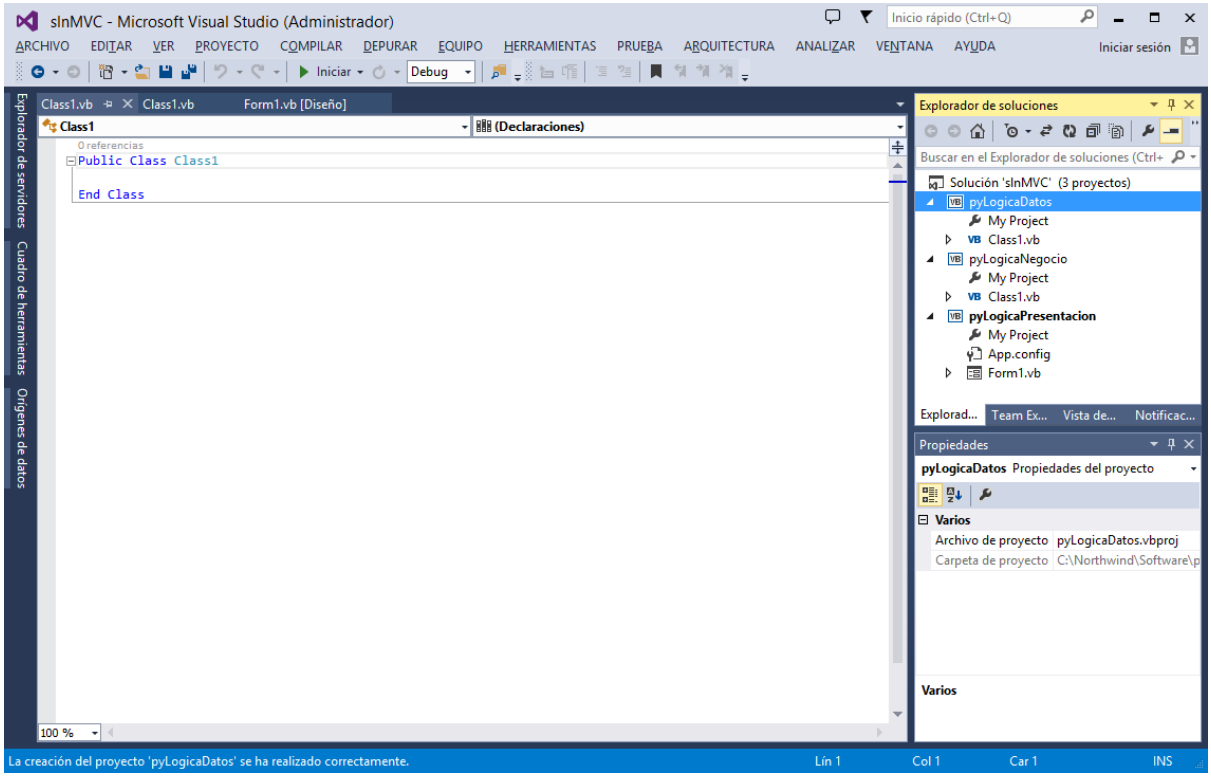
Continuamos con los proyectos para las demás capas lógicas.

4.3 CAPA DE LÓGICA DE ACCESO A DATOS

Nuevamente agregamos un "Nuevo Proyecto" (Biblioteca de Clases), este proyecto está encargado de la conexión a la base de datos y de gestionar las inserciones, actualizaciones, eliminaciones y consultas que requiera nuestra aplicación:



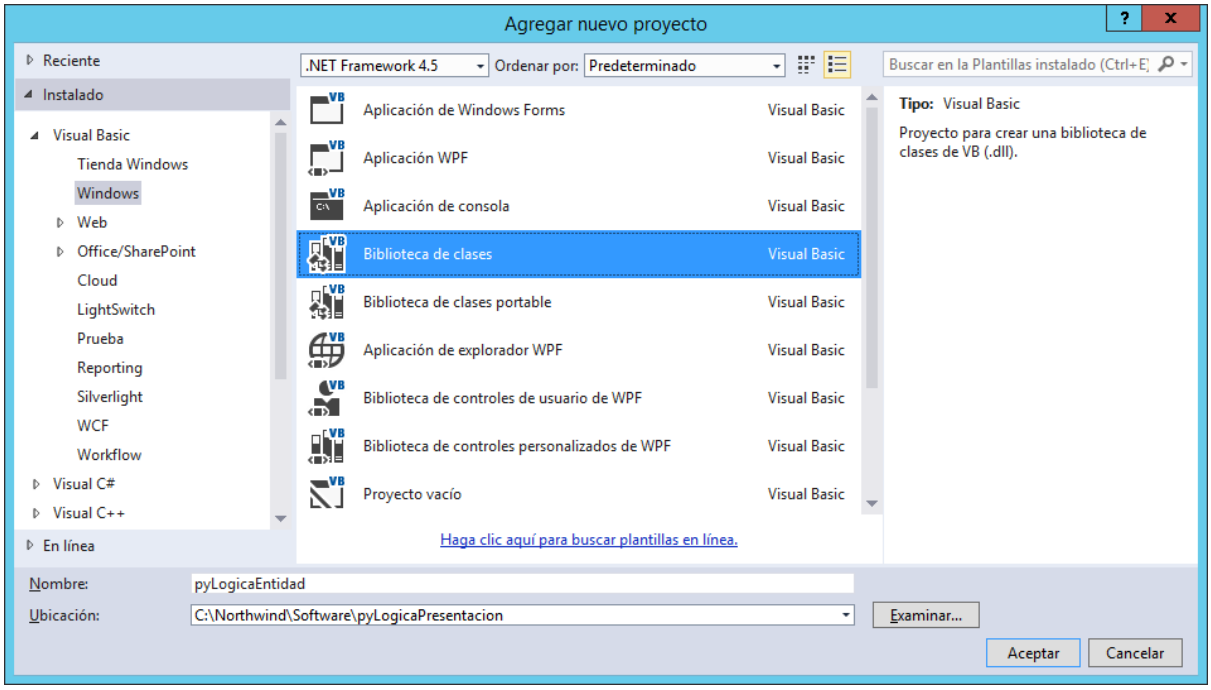
Clic en aceptar y tendremos:



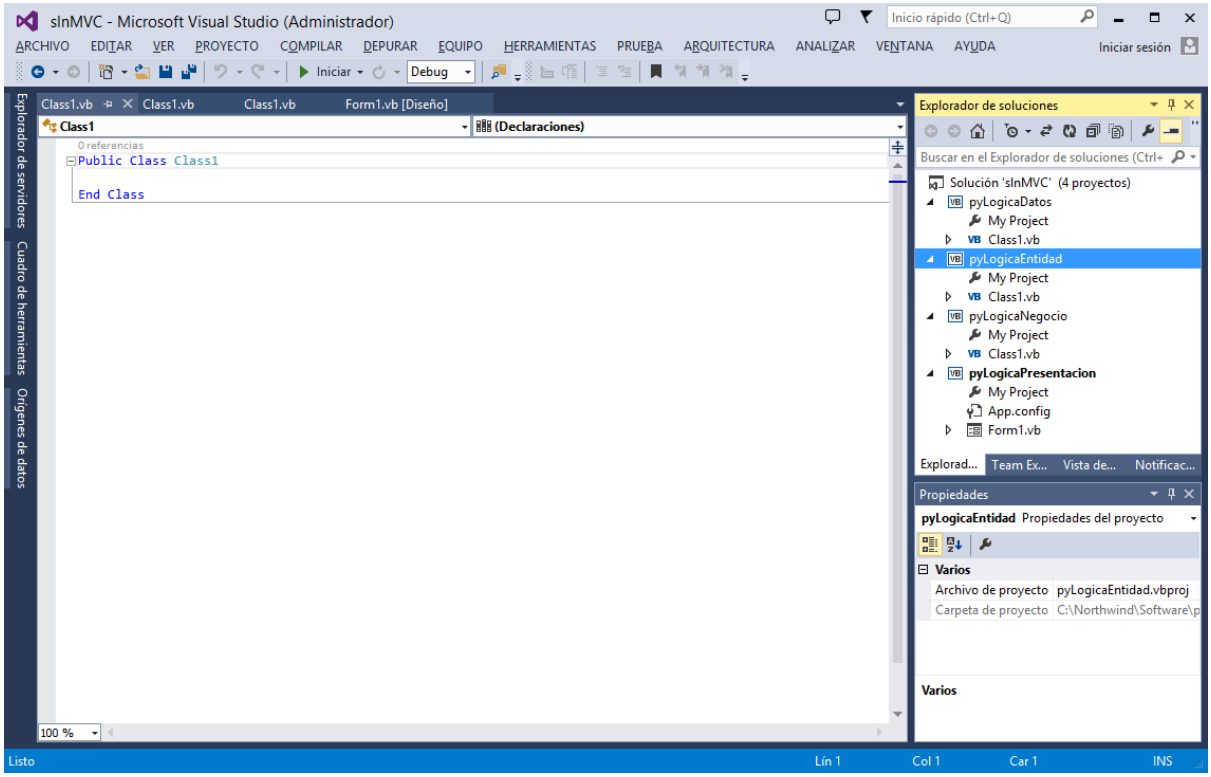
Y ya tenemos casi finalizada la estructura de nuestra solución.

4.4 CAPA DE LÓGICA DE ENTIDAD

Finalmente, creamos un proyecto más, esta vez para la Capa de Entidad, ésta capa es la encargada única y exclusivamente de la representación de los datos, es decir contiene clases que van a "mapear" con las tablas de la base de datos, así mismo contendrá las clases que "mapeen" con las vistas necesarias para los reportes de la solución. Clic en ***"Archivo > Agregar > Nuevo Proyecto"*** (Biblioteca de Clases) y configuramos de la siguiente manera:



Clic en aceptar y obtenemos:

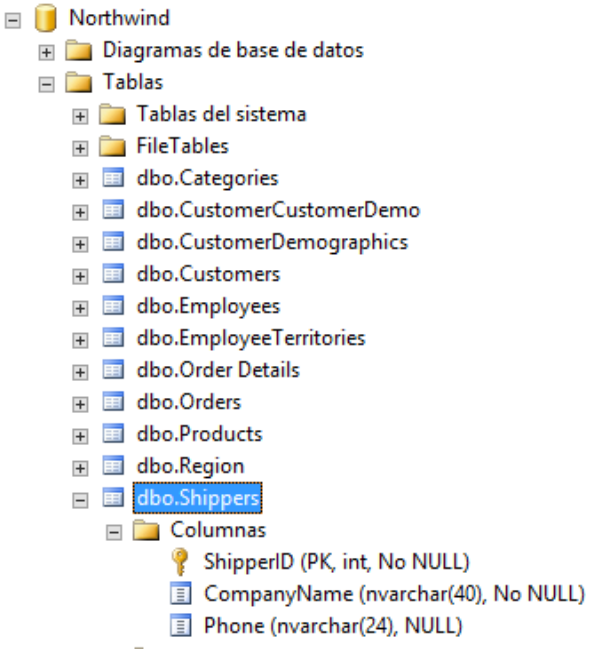


Guardamos los proyectos creados.

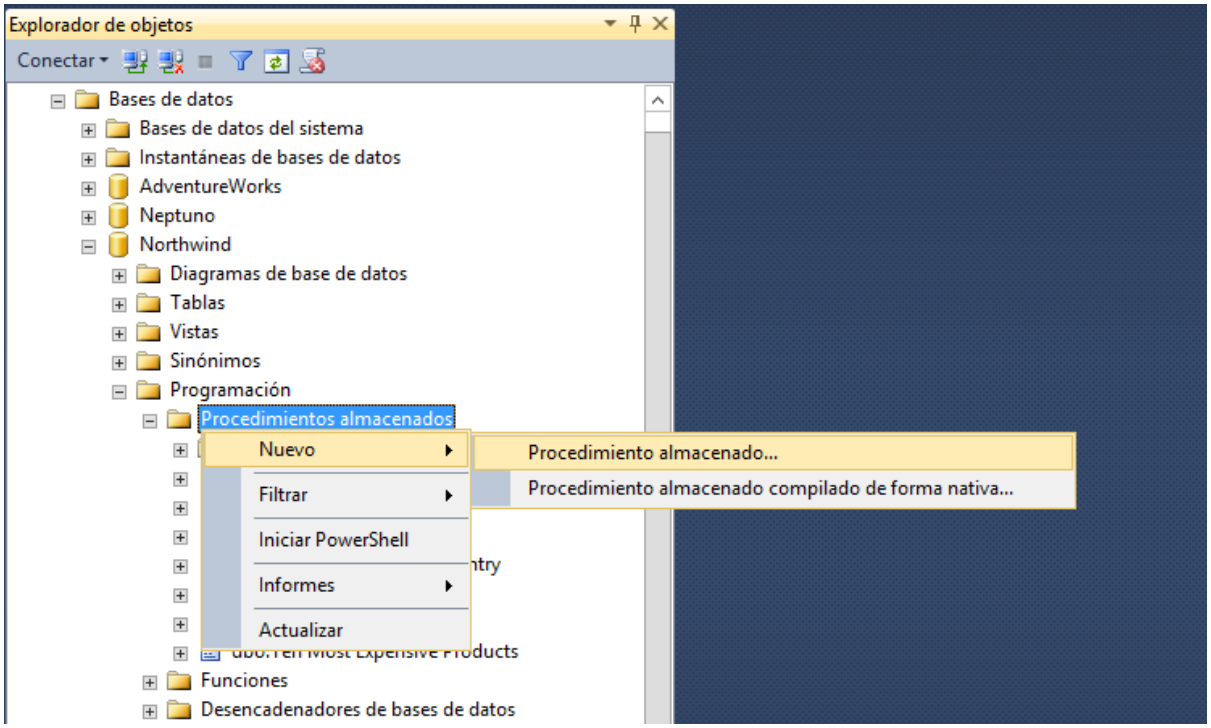
5 IMPLEMENTACIÓN DE LOS CASOS DE USO

5.1 REGISTRAR TRANSPORTISTA (SHIPPERS)

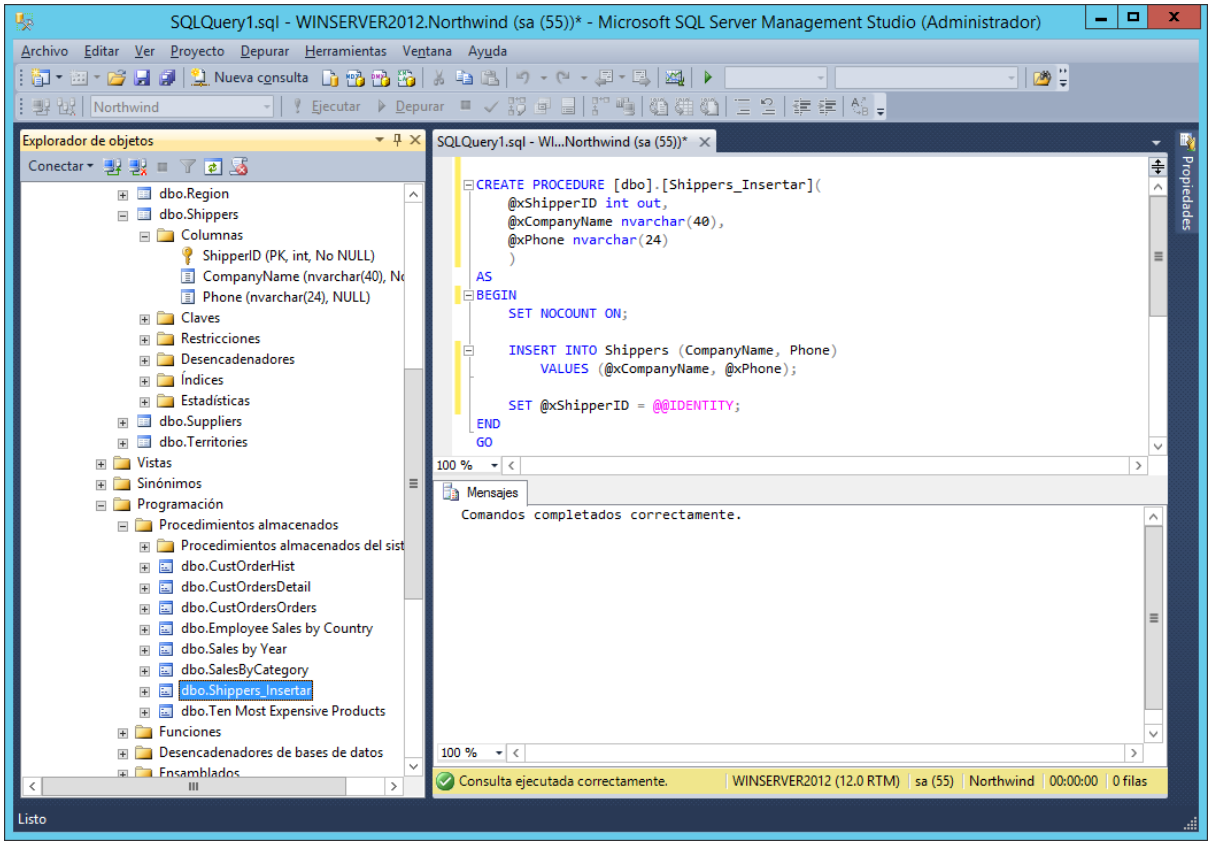
De acuerdo con la base de datos, la tabla "Shippers" contiene tres campos:



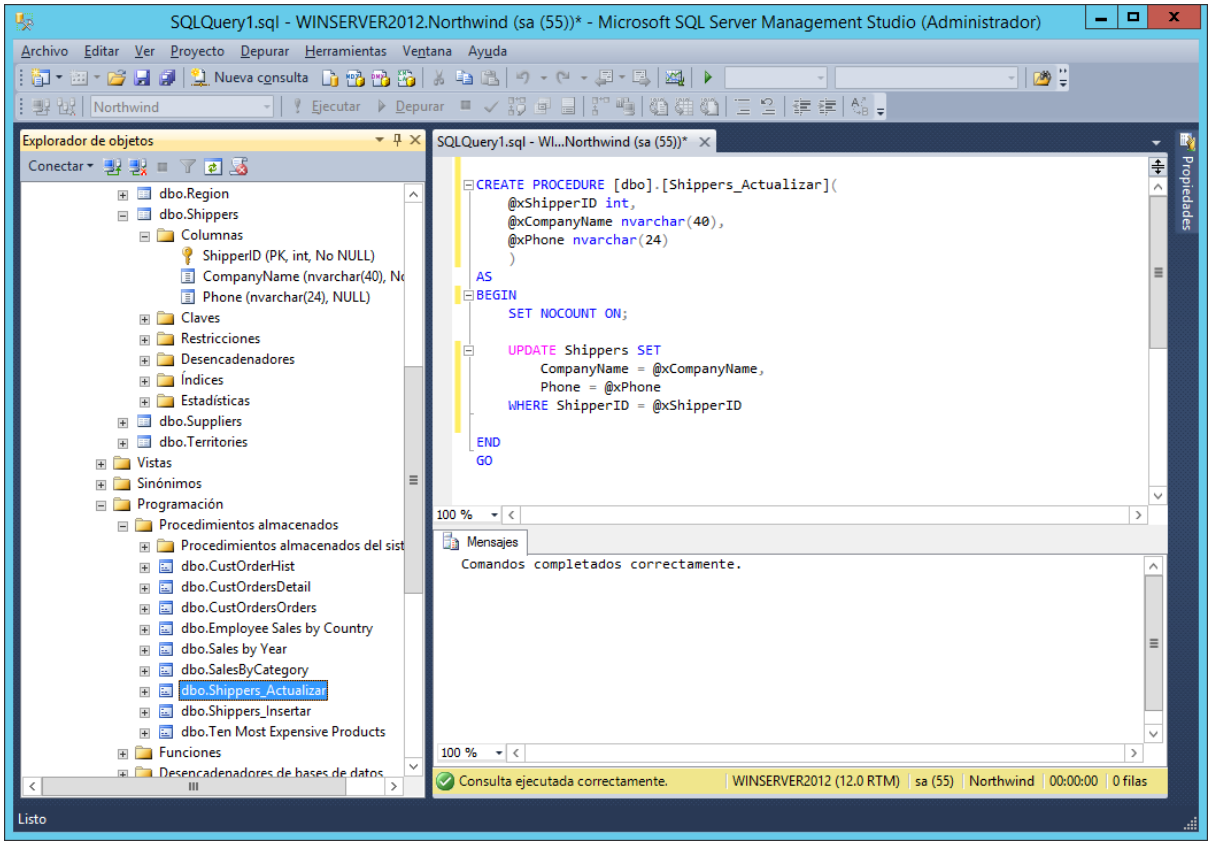
Paso 1: Creamos el procedimiento almacenado para agregar un nuevo Transportista (Shipper), para ello nos dirigimos a la consola de administración de MS SQLServer 2014 y:



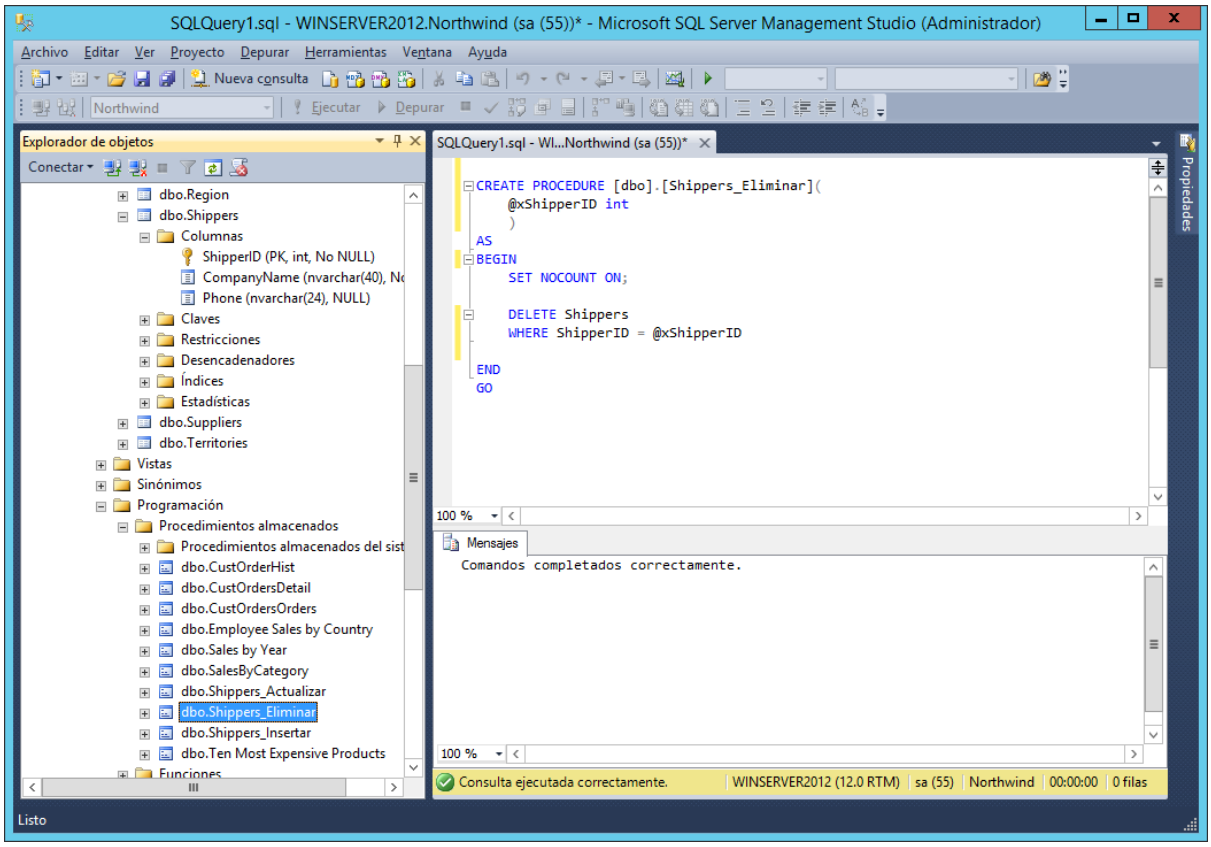
Clic en "Procedimiento almacenado" y codificamos como sigue:



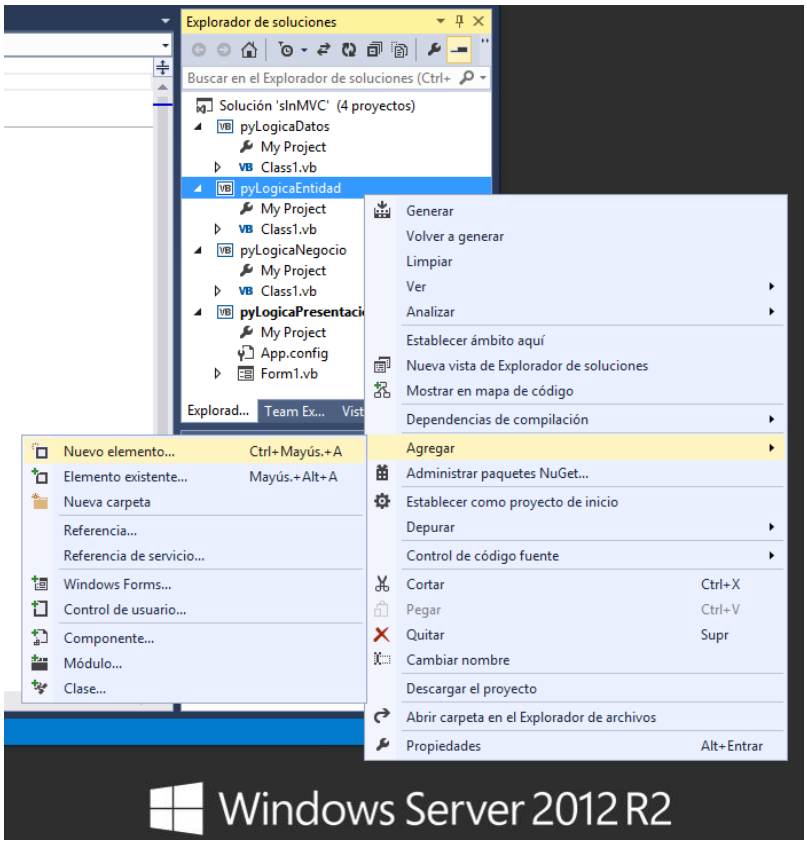
Ejecutamos con <F5> y el procedimiento queda agregado, lo mismo para el procedimiento de actualización:



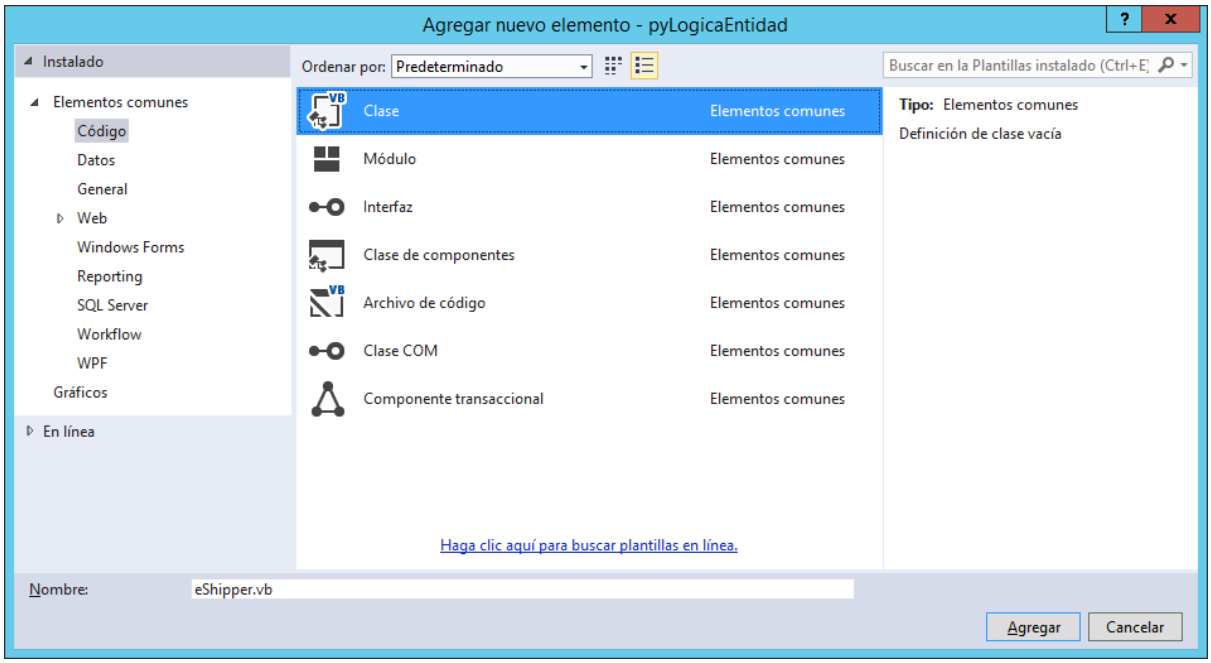
Y por último el procedimiento de eliminación:



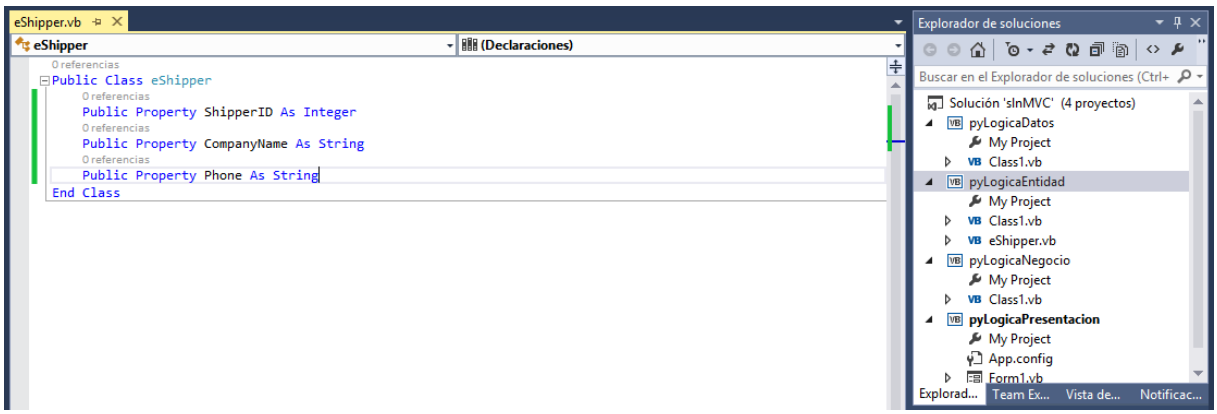
Paso 2: Debemos agregar una clase al proyecto de lógica de entidad para representar dicha estructura:



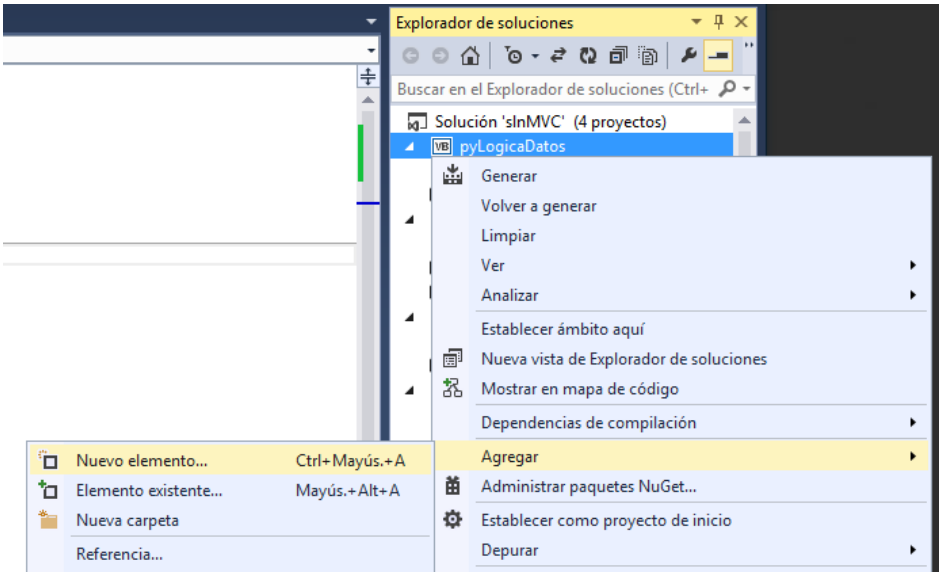
Clic en “Nuevo elemento” y seleccionamos “Clase”, le damos el nombre “eShipper”:



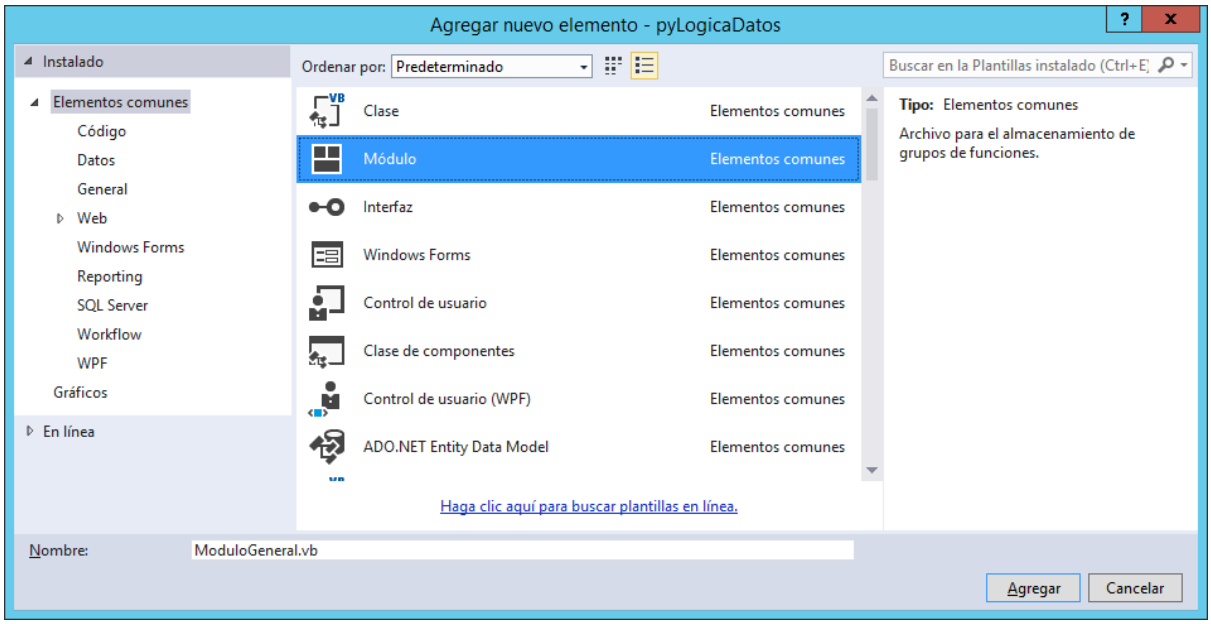
Clic en <Agregar> y codificamos como sigue:



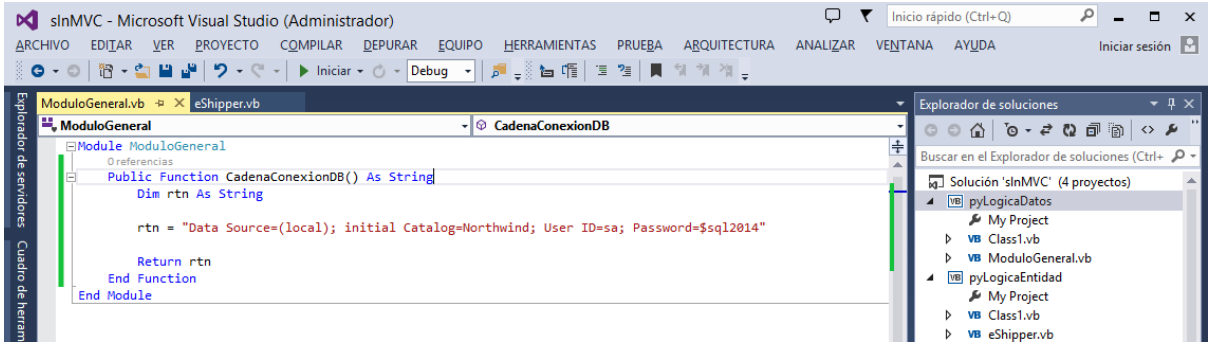
Paso 2, agregamos un módulo correspondiente a la cadena de conexión a la base de datos:



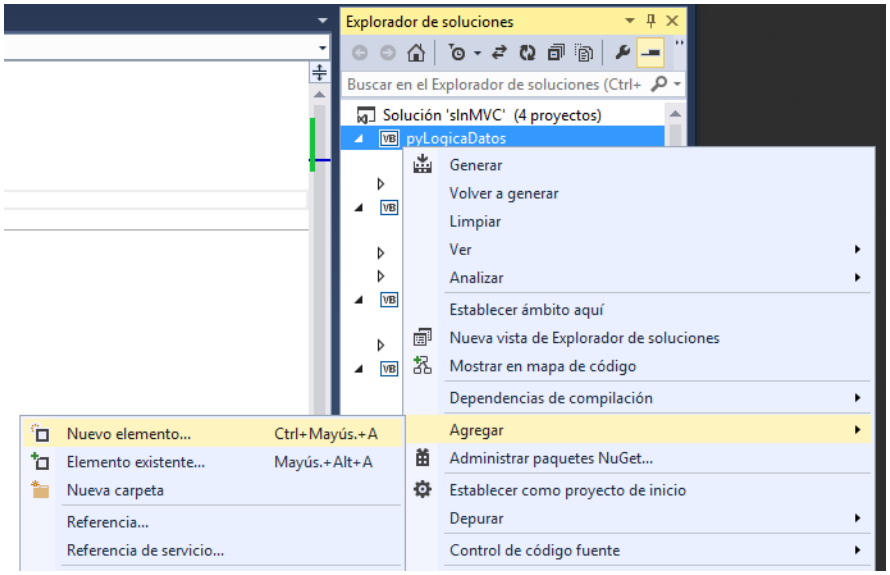
Clic en “Nuevo elemento” y seleccionamos “modulo” nombrándolo como sigue:



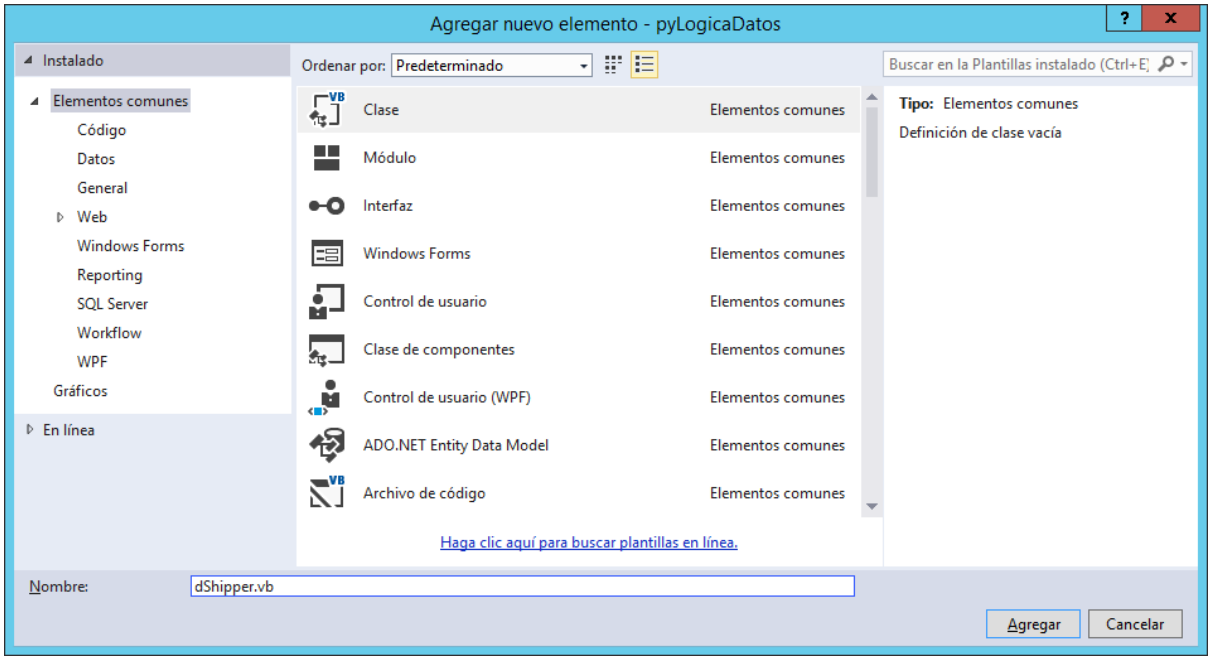
Clic en agregar y codificamos como sigue:



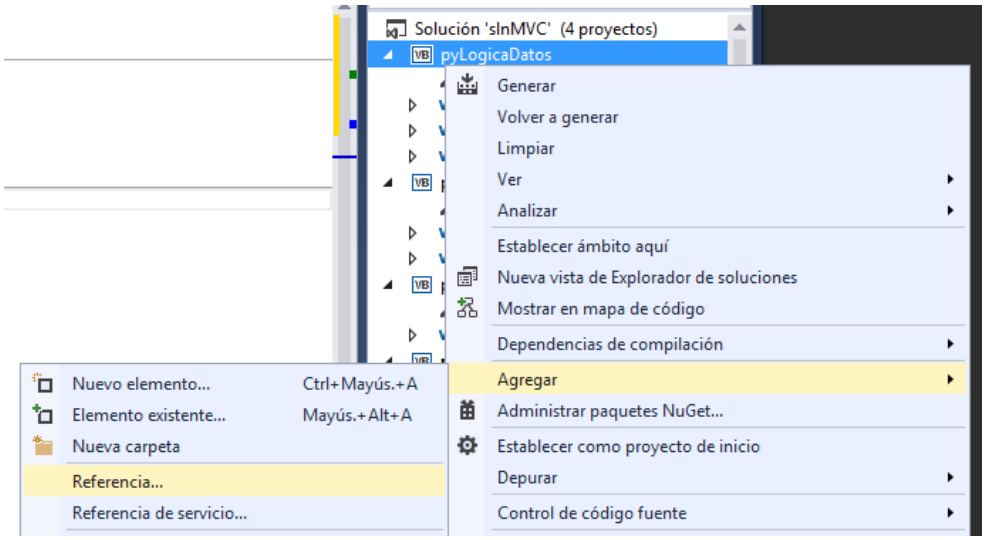
Paso 3, Agregamos la clase correspondiente en la capa de acceso a datos:

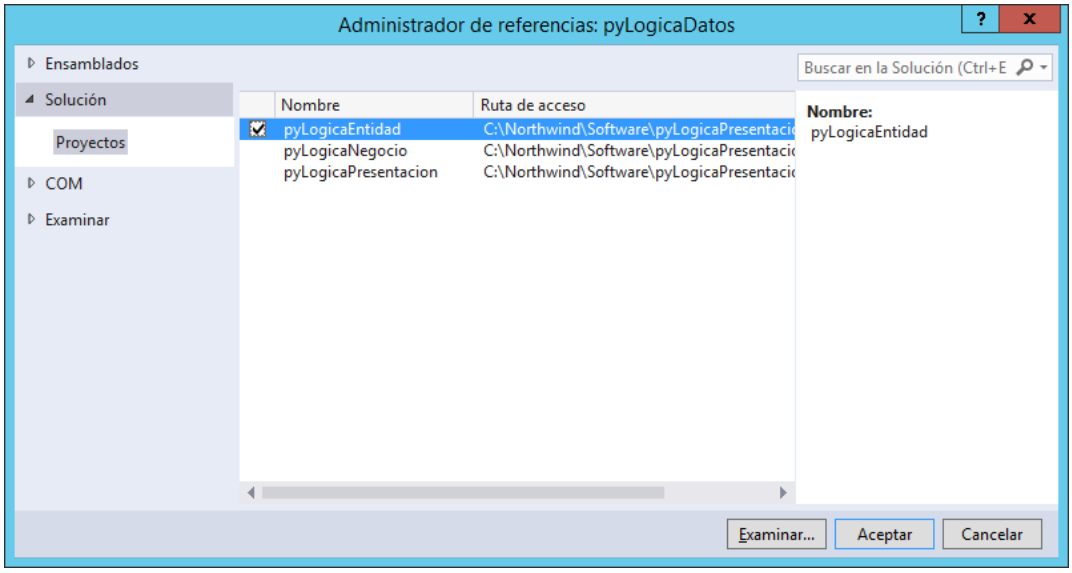


Clic en “Nuevo Elemento” y escogemos “Clase”, la nombramos “dShipper”:



Antes de seguir codificando debemos agregar la referencia a la capa de entidad:





Clic en Aceptar y listo, ya podemos importar las Clases de la Capa de Logica de Entidad a la Capa de Acceso a Datos; procediendo a codificar la clase “dShipper” como se muestra:

Inserción:

```
Imports System
Imports System.Data.SqlTypes
Imports System.Data.SqlClient
Imports pyLogicaEntidad

2 referencias
Public Class dShipper
    0 referencias
    Public Sub Insertar(ByRef x As eShipper)
        Dim CN As SqlConnection
        Dim CMD As SqlCommand
        Dim oParameter As SqlParameter

        CN = New SqlClient.SqlConnection(CadenaConexionDB())
        CN.Open()
        CMD = New SqlCommand                          'utilizamos una variable tipo comando para acceder a l
        CMD.CommandType = CommandType.StoredProcedure  'especificamos que abriremos un procedimiento almacena
        CMD.Connection = CN                           'especificamos la conexion
        CMD.CommandText = "Shippers_Insertar"          'especificamos el nombre del procedimiento

        oParameter = CMD.Parameters.Add("@xShipperID", SqlDbType.Int)
        oParameter.Direction = ParameterDirection.Output

        oParameter = CMD.Parameters.Add("@xCompanyName", SqlDbType.NVarChar, 40)
        oParameter.Value = x.CompanyName

        oParameter = CMD.Parameters.Add("@xPhone", SqlDbType.NVarChar, 24)
        oParameter.Value = x.Phone

        CMD.ExecuteNonQuery()
        x.ShipperID = CMD.Parameters(0).Value
        CN.Close()
        CN.Dispose()
        CMD.Dispose()
    End Sub
End Class
```

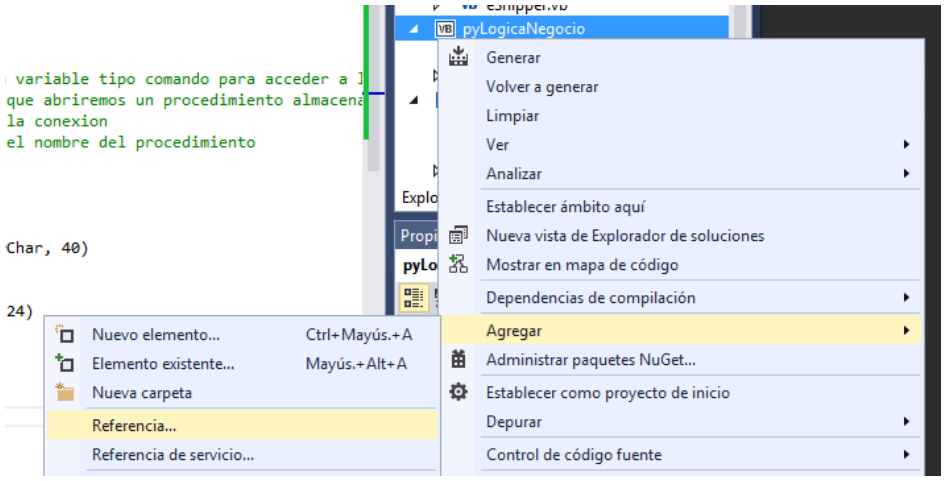
Actualización:

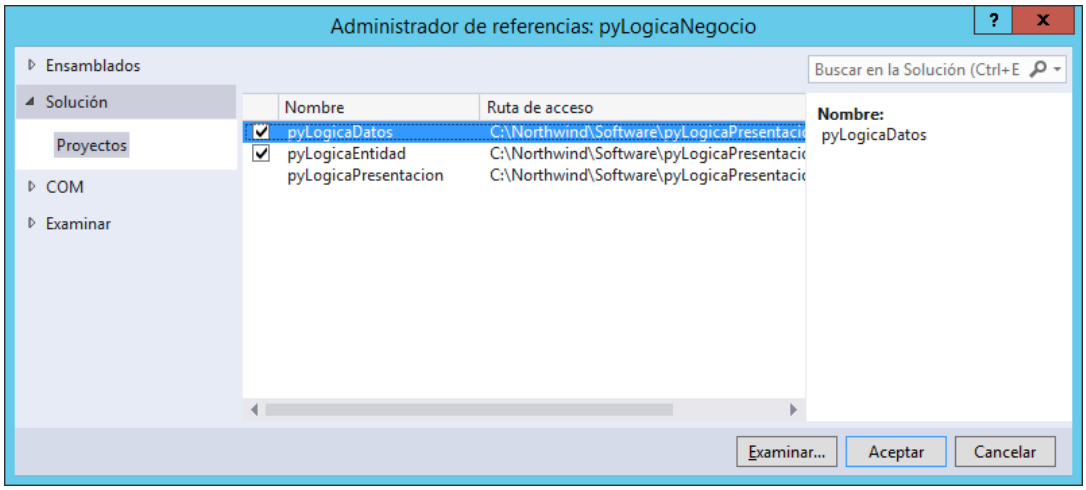
```
Public Class dShipper
    0 referencias
    Public Sub Insertar(ByRef x As eShipper) ...
    0 referencias
    Public Sub Actualizar(ByRef x As eShipper)
        Dim CN As SqlConnection
        Dim CMD As SqlCommand
        Dim oParameter As SqlParameter

        CN = New SqlClient.SqlConnection(CadenaConexionDB())
        CN.Open()
        CMD = New SqlCommand                          'utilizamos una variable tipo comando para acceder a l
        CMD.CommandType = CommandType.StoredProcedure  'especificamos que abriremos un procedimiento almacena
        CMD.Connection = CN                           'especificamos la conexion
        CMD.CommandText = "Shippers_Actualizar"          'especificamos el nombre del procedimiento

        oParameter = CMD.Parameters.Add("@xShipperID", SqlDbType.Int)
        oParameter.Value = x.ShipperID

        oParameter = CMD.Parameters.Add("@xCompanyName", SqlDbType.NVarChar, 40)
        oParameter.Value = x.CompanyName

        oParameter = CMD.Parameters.Add("@xPhone", SqlDbType.NVarChar, 24)
        oParameter.Value = x.Phone

        CMD.ExecuteNonQuery()

        CN.Close()
        CN.Dispose()
        CMD.Dispose()
    End Sub
End Class
```

Eliminación

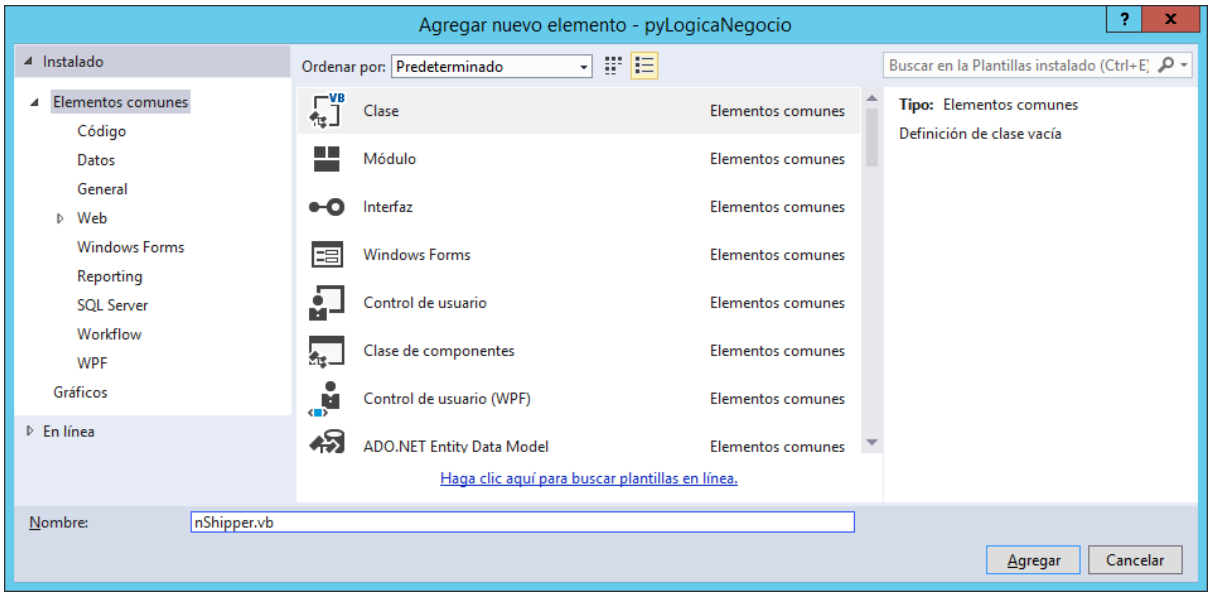
```
Imports System.Data.SqlClient
Imports pyLogicaEntidad

2 referencias
Public Class dShipper
    0 referencias
    Public Sub Insertar(ByRef x As eShipper) ...
    0 referencias
    Public Sub Actualizar(ByRef x As eShipper) ...
    0 referencias
    Public Sub Eliminar(ByRef x As eShipper)
        Dim CN As SqlConnection
        Dim CMD As SqlCommand
        Dim oParameter As SqlParameter

        CN = New SqlClient.SqlConnection(CadenaConexionDB())
        CN.Open()
        CMD = New SqlCommand                          'utilizamos una variable tipo comando para acceder a l
        CMD.CommandType = CommandType.StoredProcedure  'especificamos que abriremos un procedimiento almacena
        CMD.Connection = CN                            'especificamos la conexion
        CMD.CommandText = "Shippers_Actualizar"          'especificamos el nombre del procedimiento

        oParameter = CMD.Parameters.Add("@xShipperID", SqlDbType.Int)
        oParameter.Value = x.ShipperID

        CMD.ExecuteNonQuery()
        CN.Close()
        CN.Dispose()
        CMD.Dispose()
        CMD = Nothing
        CN = Nothing
    End Sub
End Class
```

Paso 4, agregamos la referencia para que la Capa de Negocio pueda llamar a las clases de la capa de datos como a las de entidad:

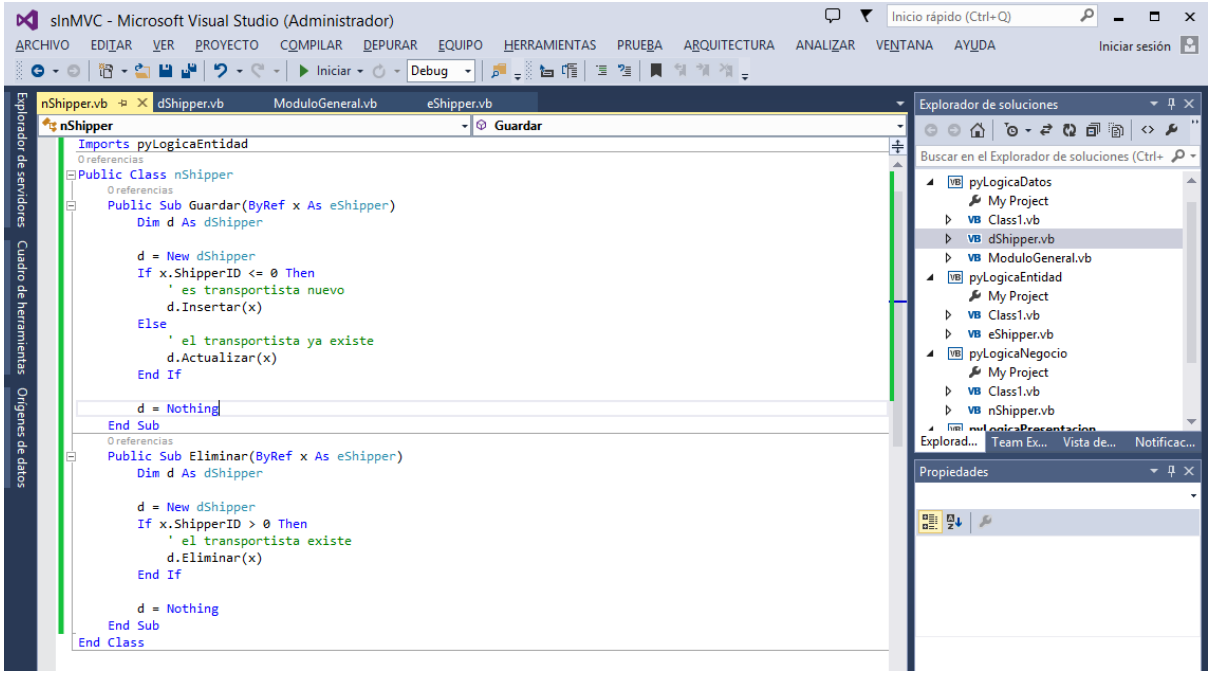




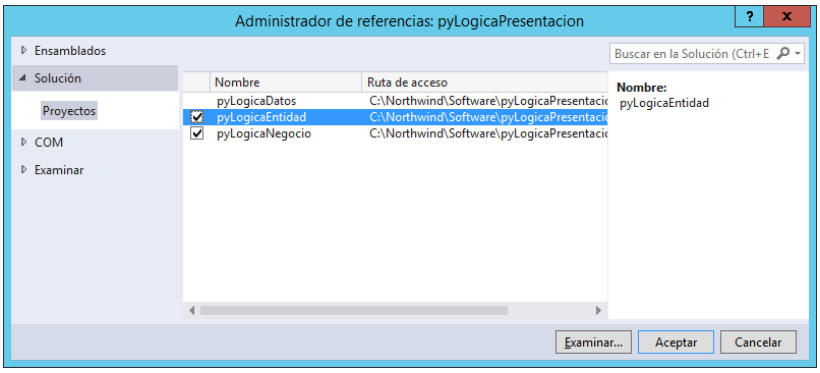
Paso 5, agregamos la clase en la capa de Logica de Negocio, tal como se muestra:



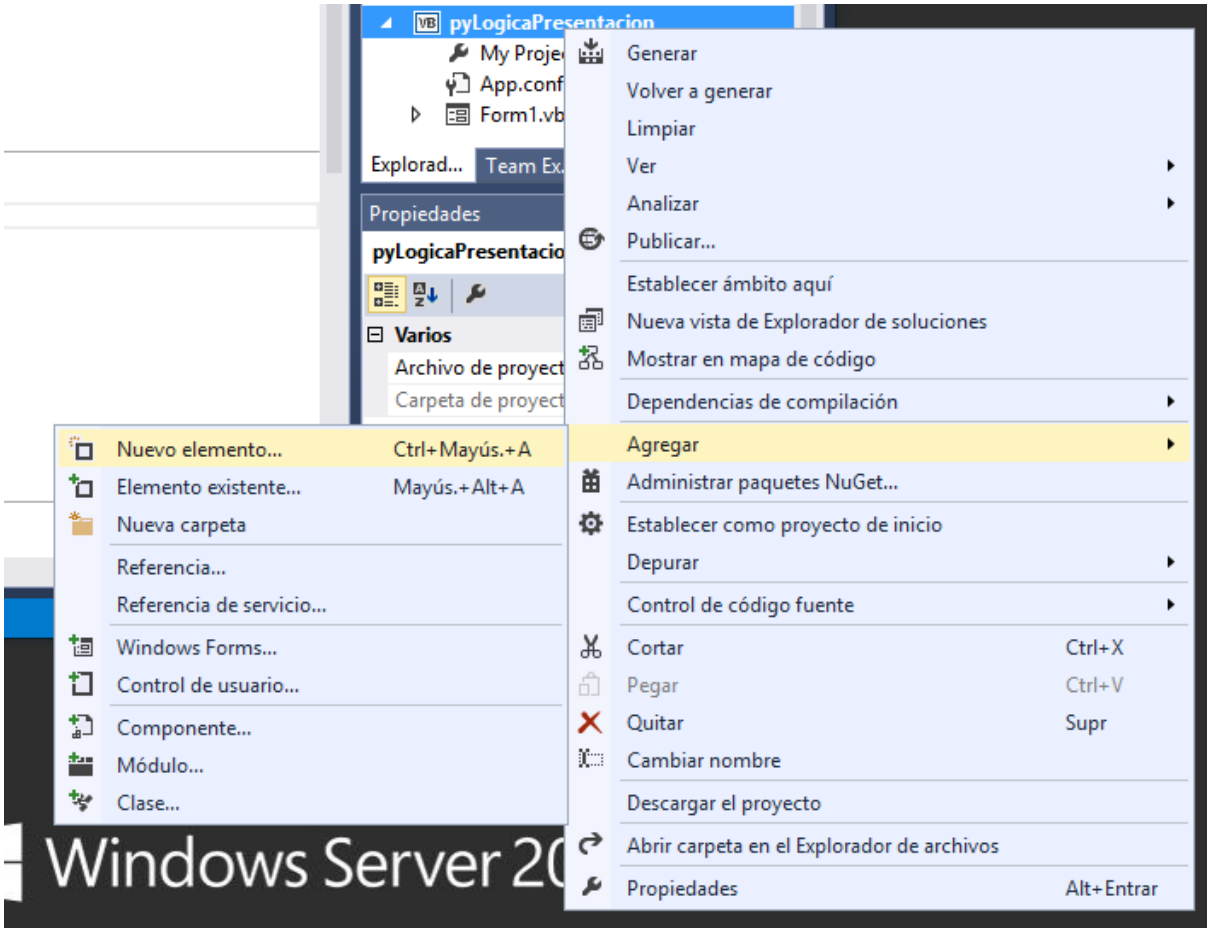
Funciones de Guardar y Eliminar:



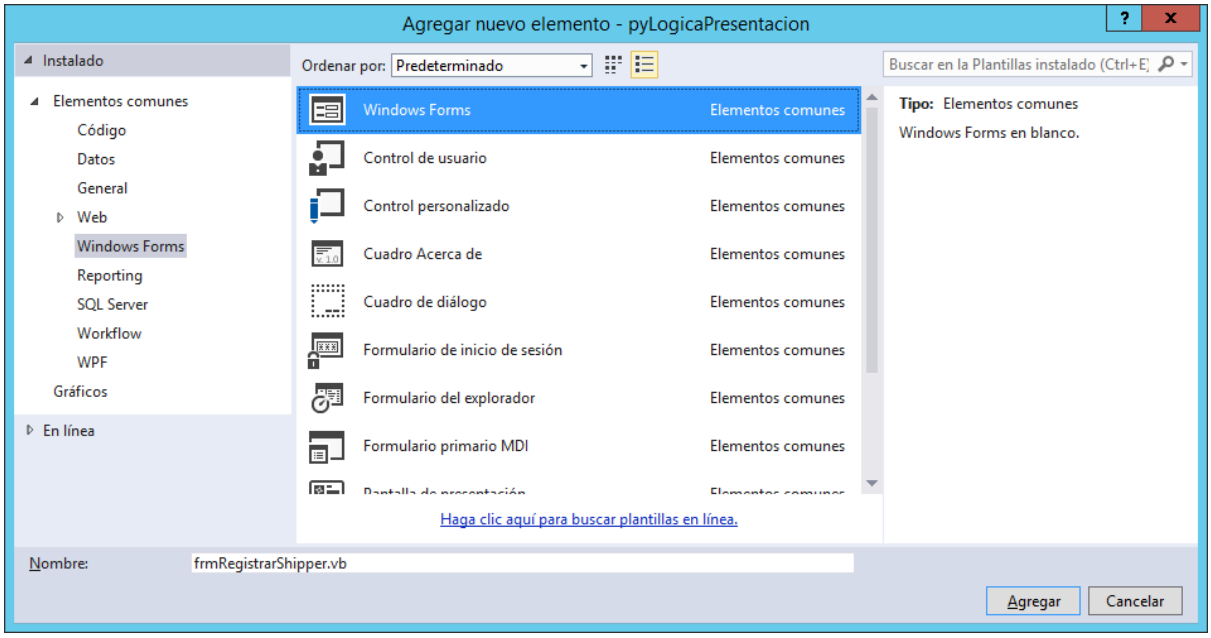
Paso 6, Agregar las referencias a la lógica de Presentación:



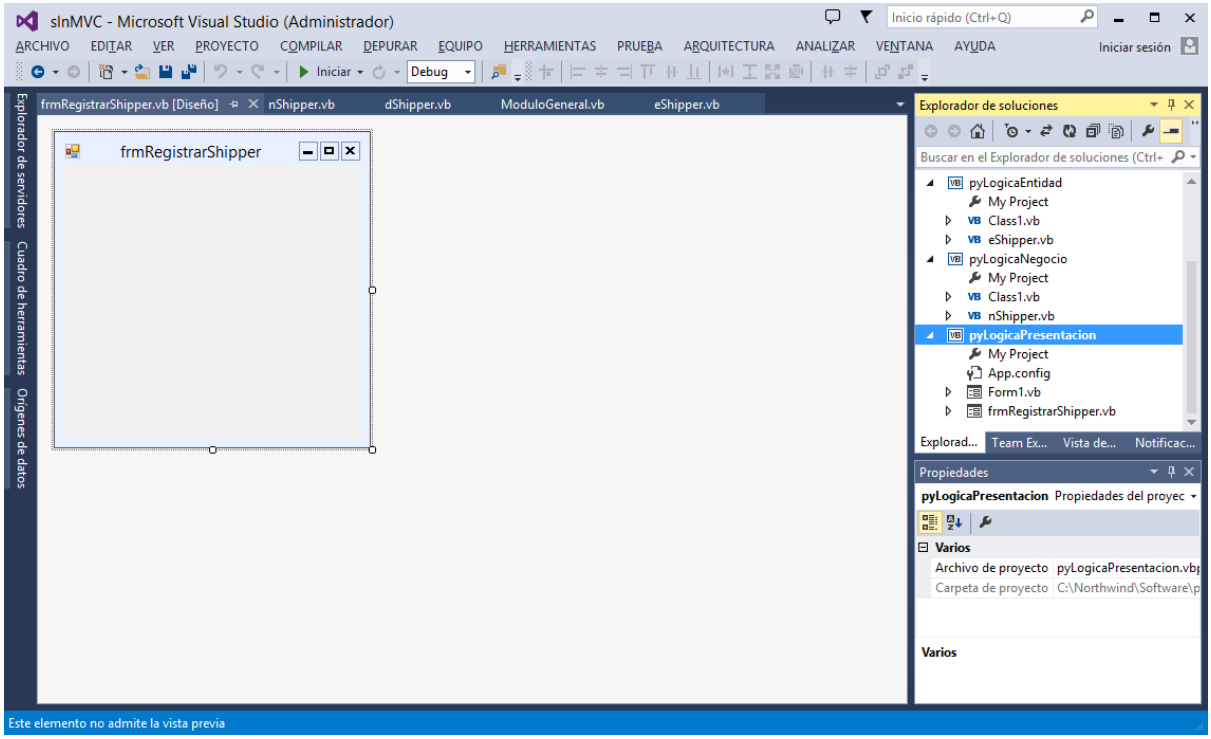
Paso 7, Diseñar y Codificar el Fomulario de ingreso de datos:



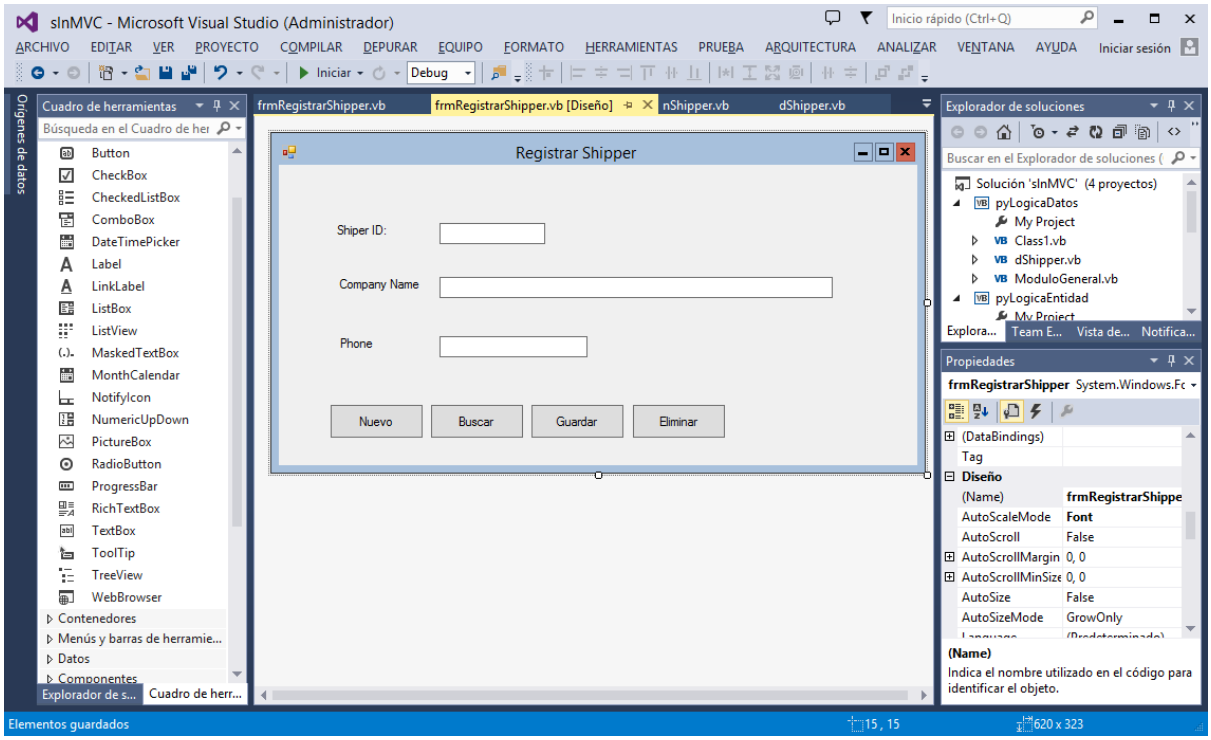
Escogemos “Windows Forms” y llenamos:



Clic en agregar y se muestra el diseño del formulario:



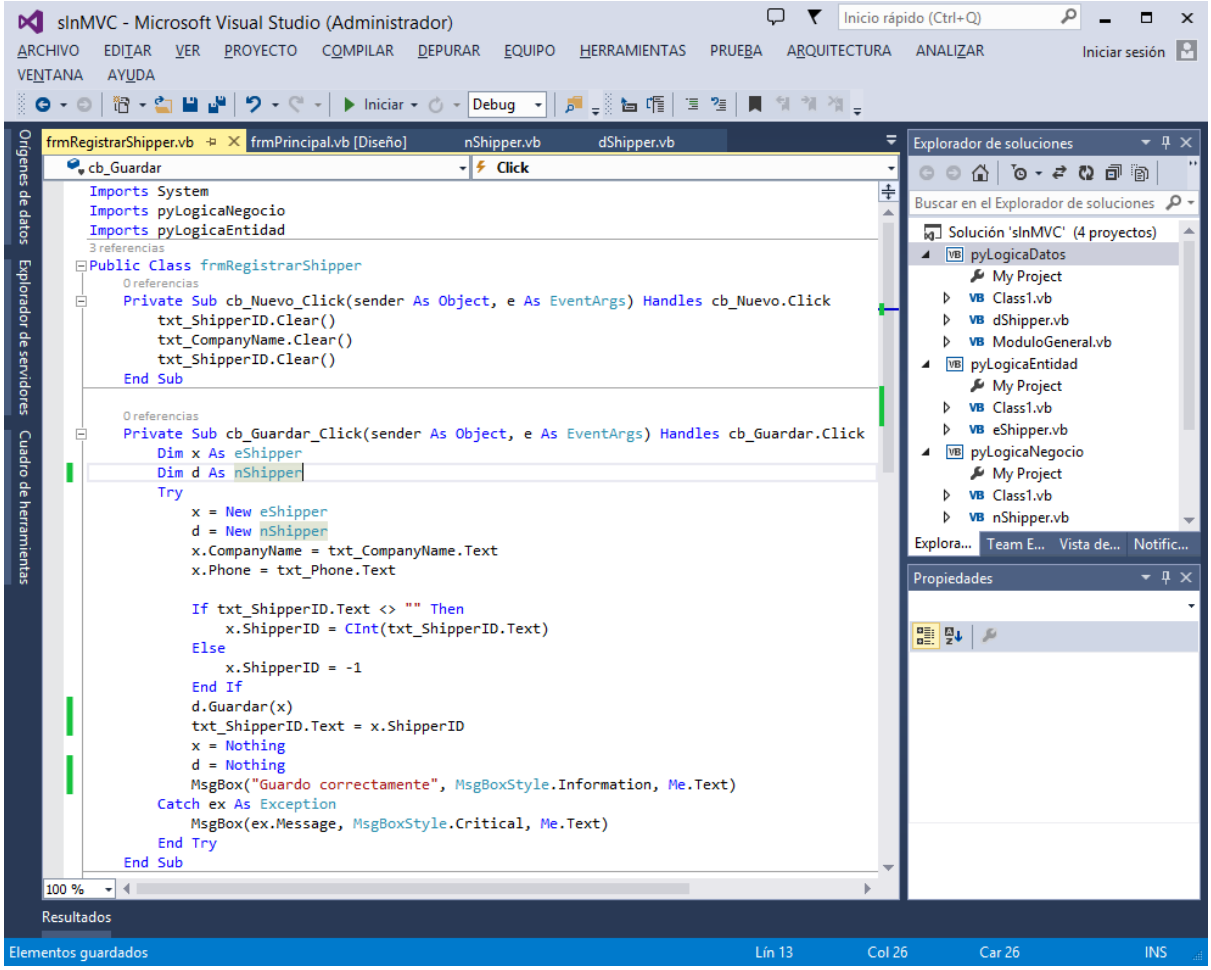
Agregamos los siguientes controles:

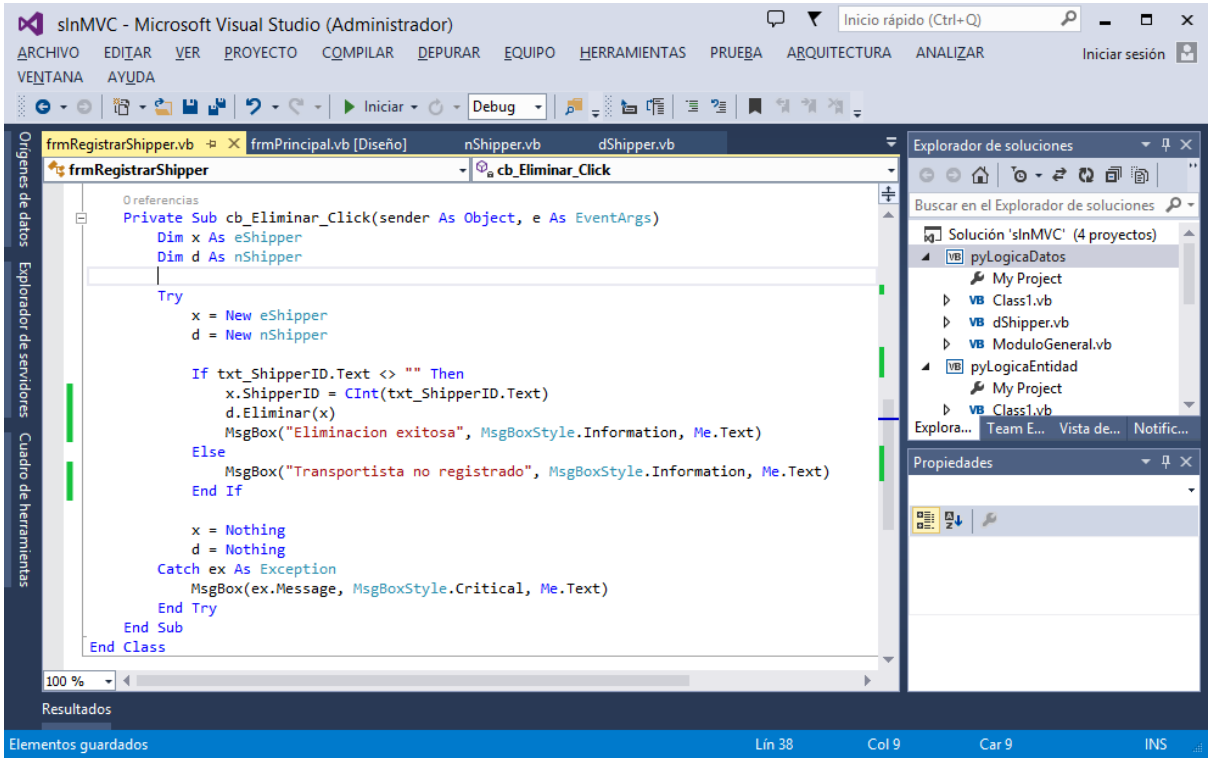


Desde arriba hacia abajo los controles son: txt_ShipperID, txt_CompanyName, txt_Phone, cb_Nuevo, cb_Buscar, cb_Guardar, cb_Eliminar.

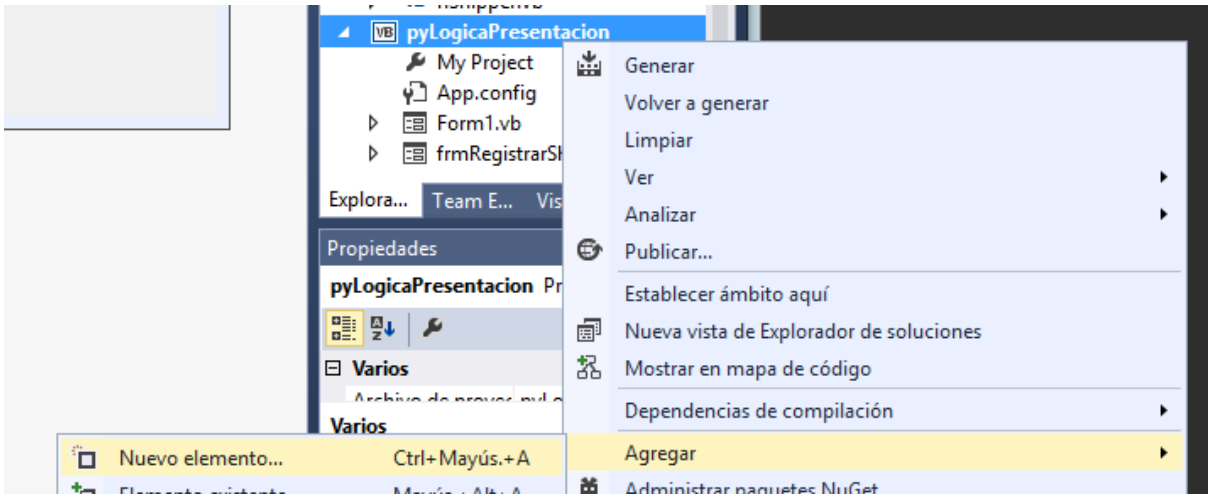
Para txt_ShipperID Establecemos su propiedad Enabled = false.

Paso 8, Procedemos a codificar los diferentes botones de comando.

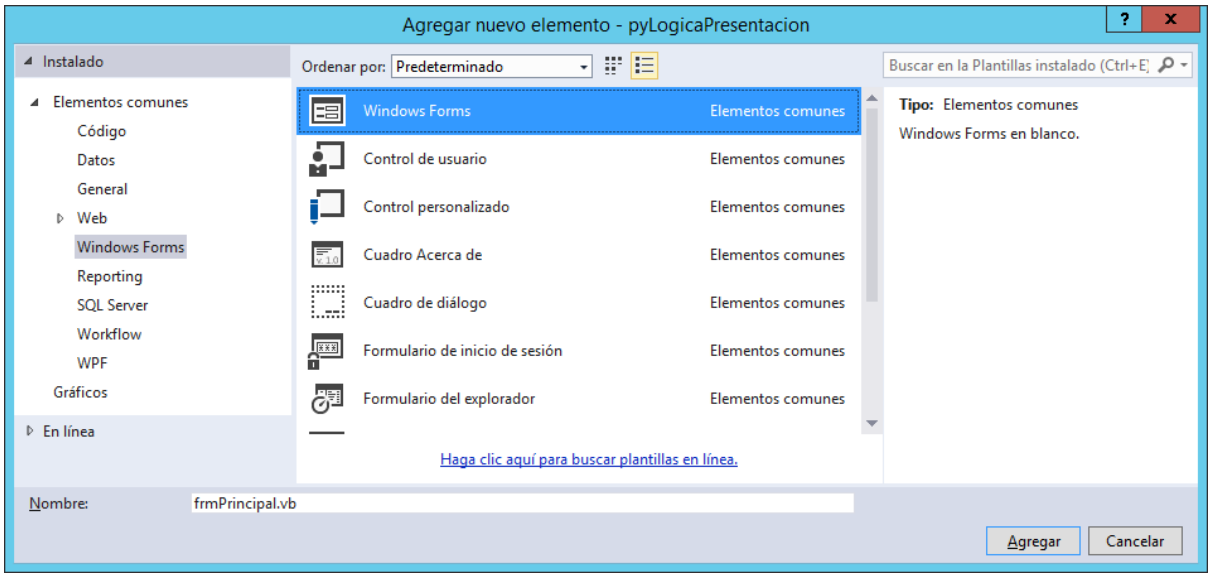




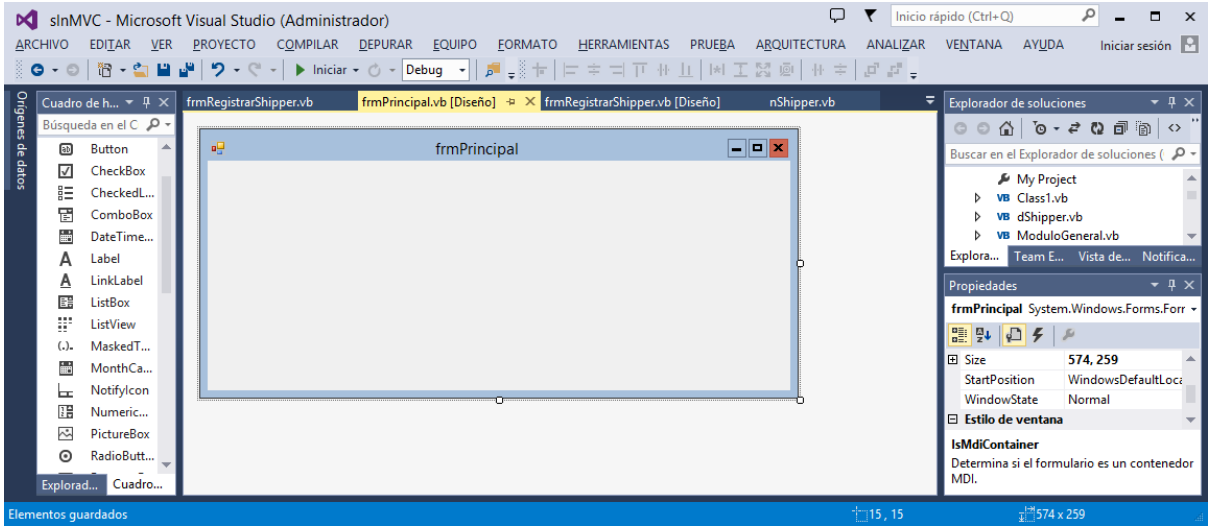
Paso 9, creamos el formulario MDI para la aplicación:



Seleccionamos “Windows Forms” y nombramos como sigue:



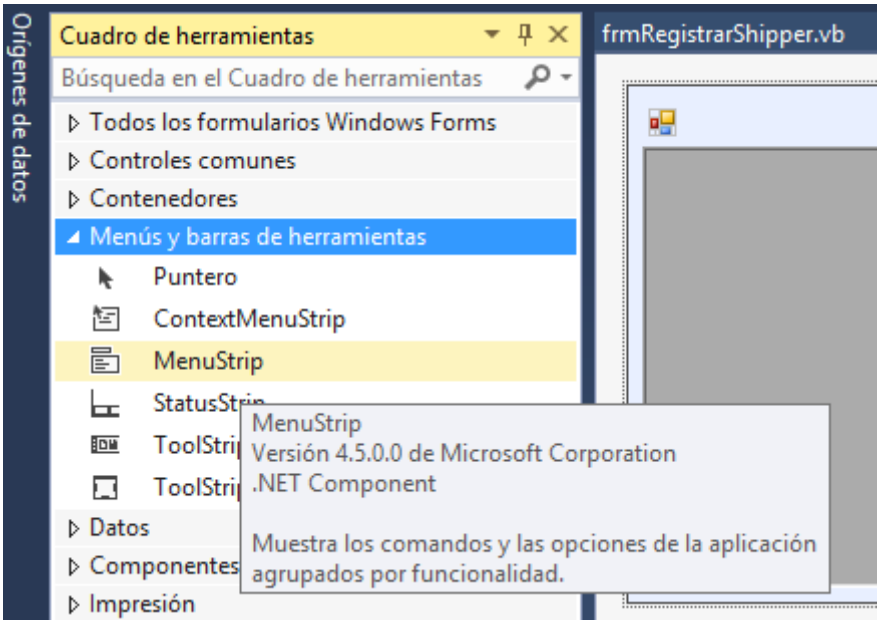
Clic en aceptar y tenemos:

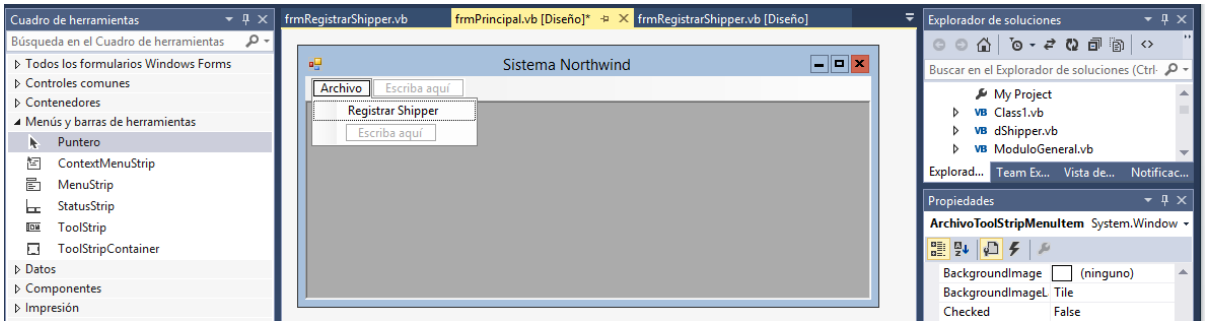


Establecemos las siguientes propiedades:

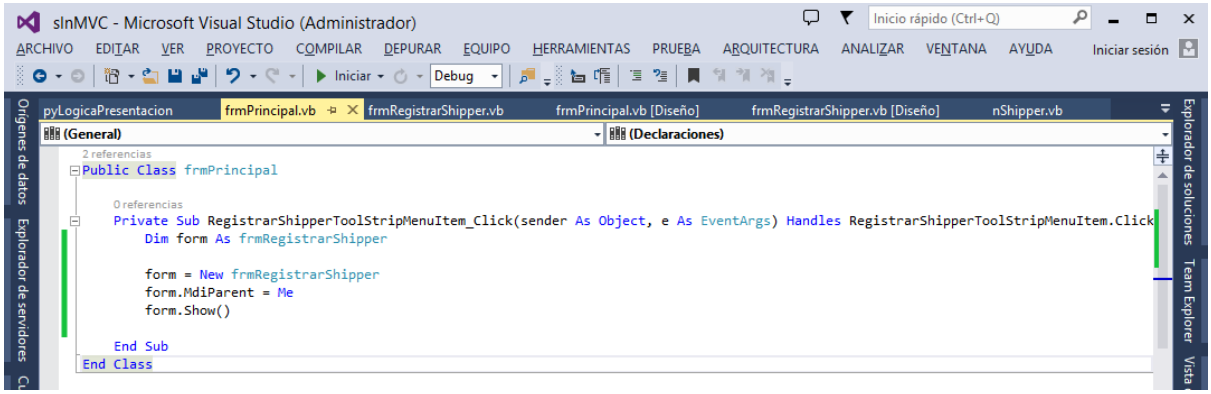
Propiedad	Valor
isMDIContainer	True
WindowState	Maximized
Text	Sistema Northwind

Agregamos un menú:

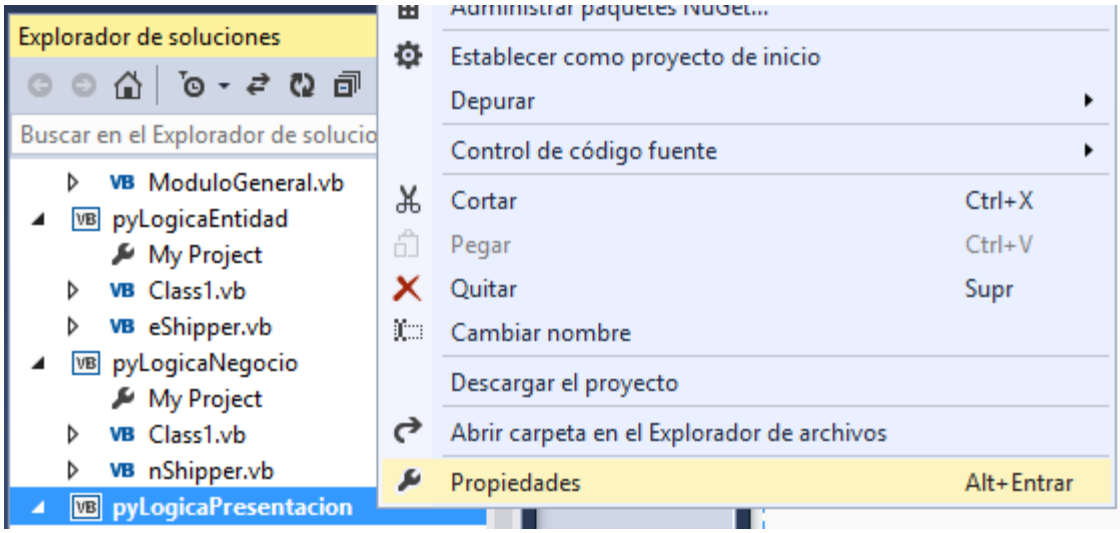




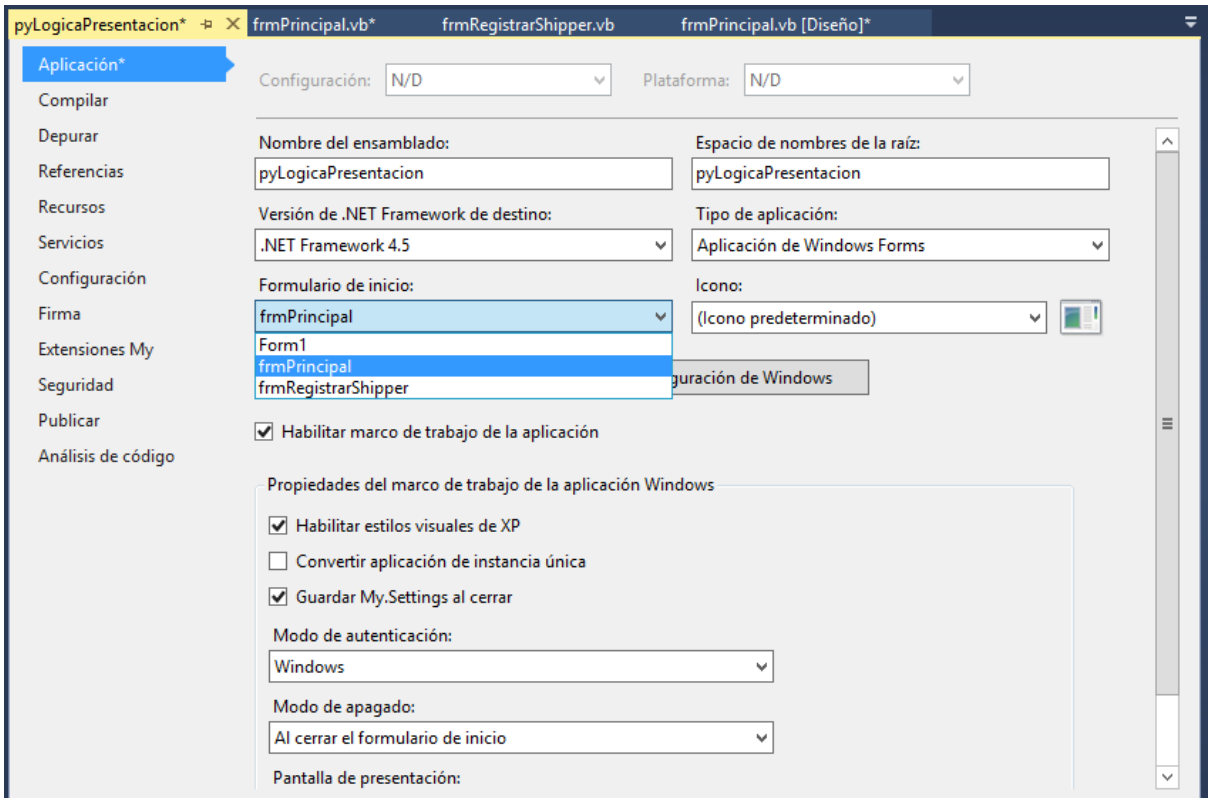
Luego codificamos dicho menú como sigue:



Establecemos el formulario de inicio del proyecto como sigue:



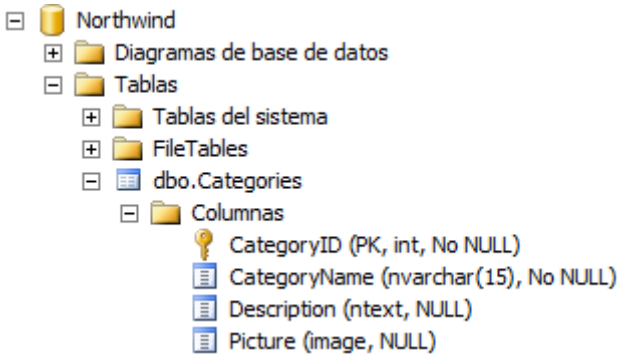
Y configuramos como sigue:



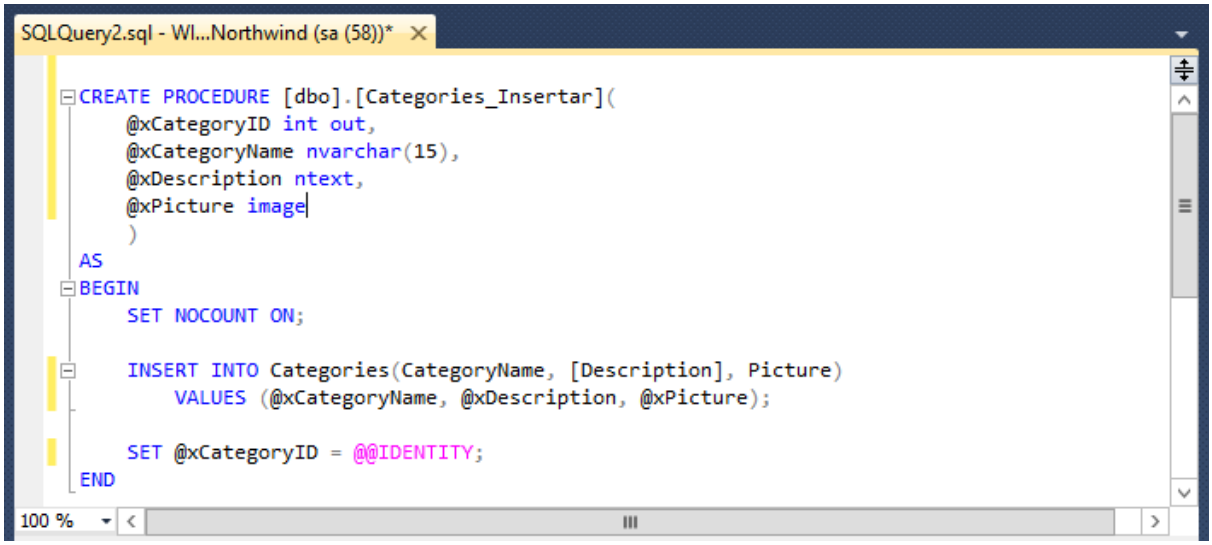
5.2 REGISTRAR CATEGORÍA (CATEGORIES)

5.2.1 Crear Procedimientos almacenados de Inserción, Actualización y Eliminación

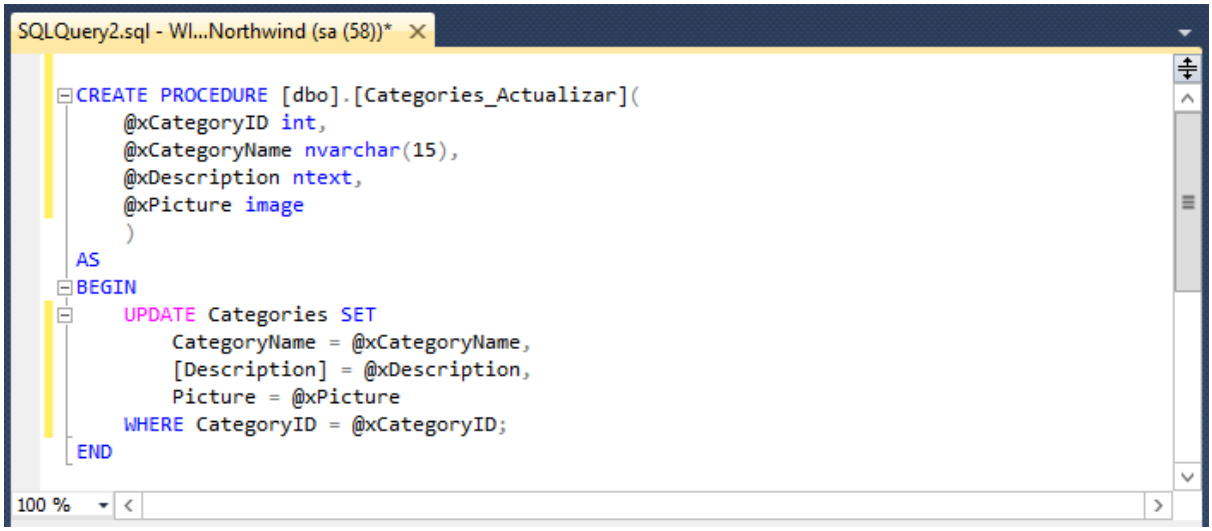
La tabla "Categories" tiene los siguientes campos y tipos de datos:



Procedimiento Almacenado para inserción:



Procedimiento Almacenado para actualizar:



Procedimiento Almacenado para eliminar:

```sql
CREATE PROCEDURE [dbo].[Categories_Eliminar](
    @xCategoryID int
    )
AS
BEGIN
    DELETE FROM Categories
    WHERE CategoryID = @xCategoryID;
END
```

Procedimiento almacenado para buscar por nombre de categoría:

```sql
CREATE PROCEDURE [dbo].[Categories_Buscar](
    @xCategoryID int out,
    @xCategoryName nvarchar(15),
    @xDescription nvarchar(1000) out,
    @xPicture varbinary(max) out
    )
AS
BEGIN
    SELECT @xCategoryID = CategoryID,
        @xDescription = CAST([Description] as nvarchar(1000)),
        @xPicture = CAST(Picture as varbinary(max))
    FROM Categories
    WHERE CategoryName = @xCategoryName;
END
```

5.2.2 Crear Clase Entidad

La clase entidad se crea, en el proyecto correspondiente a la lógica de entidad:

```vb
Imports System.IO
Imports System.Drawing

16 referencias
Public Class eCategory
    10 referencias
    Public Property CategoryID As Integer
    4 referencias
    Public Property CategoryName As String
    3 referencias
    Public Property Description As String
    4 referencias
    Public Property Picture As Byte()
    2 referencias
    Public Property PictureJPG As Image
        Set(value As Image)
            Dim ms As New MemoryStream()
            value.Save(ms, System.Drawing.Imaging.ImageFormat.Jpeg)
            Picture = ms.GetBuffer()
        End Set
        Get
            Dim ms As New MemoryStream(Picture)
            Return Image.FromStream(ms)
        End Get
    End Property
End Class
```

5.2.3 Crear clase de acceso a datos

dShipper.vb | dCategory.vb* | eCategory.vb | eShipper.vb

dCategory | (Declaraciones)

```
Imports System
Imports System.Data.SqlTypes
Imports System.Data.SqlClient
Imports pyLogicaEntidad
0 referencias
Public Class dCategory
    0 referencias
    Public Sub Eliminar(ByRef x As eCategory)
        Dim CN As SqlConnection
        Dim CMD As SqlCommand
        Dim oParameter As SqlParameter

        CN = New SqlClient.SqlConnection(CadenaConexionDB())
        CN.Open()
        CMD = New SqlCommand                            'utilizamos una variable tipo coma
        CMD.CommandType = CommandType.StoredProcedure   'especificamos que abriremos un pr
        CMD.Connection = CN                             'especificamos la conexion
        CMD.CommandText = "Categories_Eliminar"         'especificamos el nombre del proce

        oParameter = CMD.Parameters.Add("@xCategoryID", SqlDbType.Int)
        oParameter.Value = x.CategoryID
        CMD.ExecuteNonQuery()

        CN.Close()
        CN.Dispose()
        CMD.Dispose()
    End Sub
```

dShipper.vb | dCategory.vb | eCategory.vb | eShipper.vb

dCategory | Insertar

```
    0 referencias
    Public Sub Insertar(ByRef x As eCategory)
        Dim CN As SqlConnection
        Dim CMD As SqlCommand
        Dim oParameter As SqlParameter

        CN = New SqlClient.SqlConnection(CadenaConexionDB())
        CN.Open()
        CMD = New SqlCommand                            'utilizamos una variable tipo coma
        CMD.CommandType = CommandType.StoredProcedure   'especificamos que abriremos un pr
        CMD.Connection = CN                             'especificamos la conexion
        CMD.CommandText = "Categories_Insertar"         'especificamos el nombre del proce

        oParameter = CMD.Parameters.Add("@xCategoryID", SqlDbType.Int)
        oParameter.Direction = ParameterDirection.Output
        oParameter = CMD.Parameters.Add("@xCategoryName", SqlDbType.NVarChar, 15)
        oParameter.Value = x.CategoryName
        oParameter = CMD.Parameters.Add("@xDescription", SqlDbType.NText)
        oParameter.Value = x.Description
        oParameter = CMD.Parameters.Add("@xPicture", SqlDbType.Image)
        oParameter.Value = x.Picture
        CMD.ExecuteNonQuery()
        x.CategoryID = CMD.Parameters(0).Value

        CN.Close()
        CN.Dispose()
        CMD.Dispose()
    End Sub
```

dShipper.vb | dCategory.vb | eCategory.vb | eShipper.vb

dCategory | Insertar

```vb
0 referencias
Public Sub Actualizar(ByRef x As eCategory)
    Dim CN As SqlConnection
    Dim CMD As SqlCommand
    Dim oParameter As SqlParameter

    CN = New SqlClient.SqlConnection(CadenaConexionDB())
    CN.Open()
    CMD = New SqlCommand                            'utilizamos una variable tipo coma
    CMD.CommandType = CommandType.StoredProcedure   'especificamos que abriremos un pr
    CMD.Connection = CN                             'especificamos la conexion
    CMD.CommandText = "Categories_Actualizar"         'especificamos el nombre del pro

    oParameter = CMD.Parameters.Add("@xCategoryID", SqlDbType.Int)
    oParameter.Value = x.CategoryID

    oParameter = CMD.Parameters.Add("@xCategoryName", SqlDbType.NVarChar, 15)
    oParameter.Value = x.CategoryName

    oParameter = CMD.Parameters.Add("@xDescription", SqlDbType.NText)
    oParameter.Value = x.Description

    oParameter = CMD.Parameters.Add("@xPicture", SqlDbType.Image)
    oParameter.Value = x.Picture

    CMD.ExecuteNonQuery()
    CN.Close()
    CN.Dispose()
    CMD.Dispose()
End Sub
```

dShipper.vb | dCategory.vb | eCategory.vb | eShipper.vb

dCategory | Listar

```vb
0 referencias
Public Function Listar() As List(Of eCategory)
    Dim CN As SqlConnection
    Dim CMD As SqlCommand
    Dim rtn As List(Of eCategory)
    Dim Reg As SqlDataReader
    Dim x As eCategory

    CN = New SqlClient.SqlConnection(CadenaConexionDB())
    CN.Open()
    CMD = New SqlCommand                'utilizamos una variable tipo comando para acc
    CMD.CommandType = CommandType.Text  'especificamos que usaremos una consulta
    CMD.Connection = CN                 'especificamos la conexion
    CMD.CommandText = "Select CategoryID, CategoryName from Categories Order By 2"
    Reg = CMD.ExecuteReader(CommandBehavior.CloseConnection)
    rtn = New List(Of eCategory)
    While (Reg.Read())
        x = New eCategory()
        x.CategoryID = Reg.GetInt32(0)
        x.CategoryName = Reg.GetString(1)
        rtn.Add(x)
    End While

    Reg.Close()
    CN.Dispose()
    CMD.Dispose()
    Return (rtn)
End Function
```

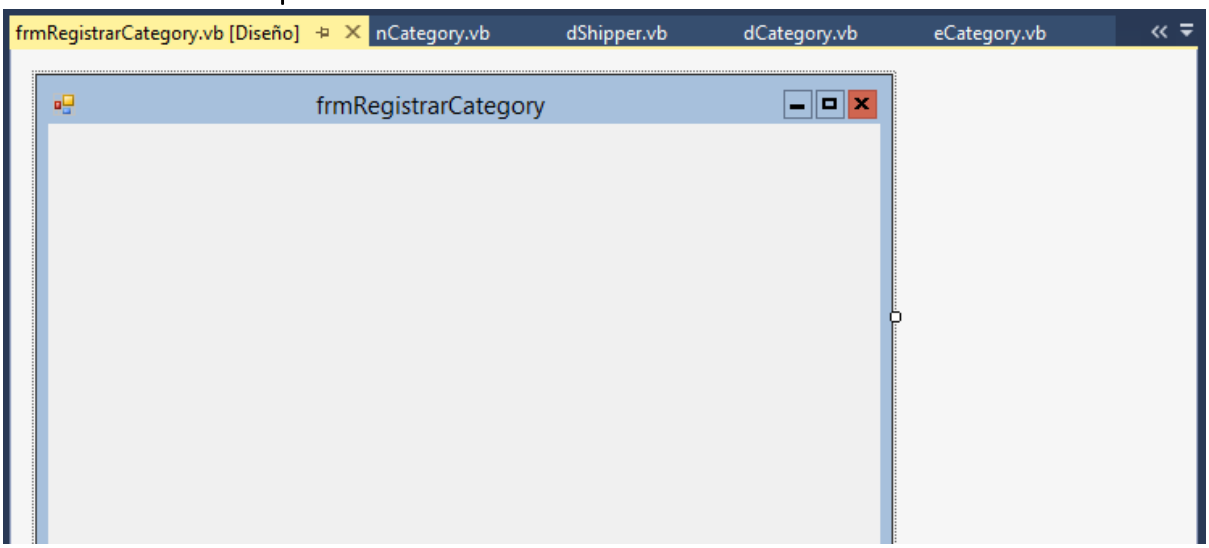
```
1 referencia
Public Sub Buscar(ByRef x As eCategory)
    Dim CN As SqlConnection
    Dim CMD As SqlCommand
    Dim oParameter As SqlParameter

    CN = New SqlClient.SqlConnection(CadenaConexionDB())
    CN.Open()
    CMD = New SqlCommand                          'utilizamos una variable tipo comando para acceder a los procedi
    CMD.CommandType = CommandType.StoredProcedure  'especificamos que abriremos un procedimiento almacenado
    CMD.Connection = CN                           'especificamos la conexion
    CMD.CommandText = "Categories_Buscar"        'especificamos el nombre del procedimiento

    oParameter = CMD.Parameters.Add("@xCategoryID", SqlDbType.Int)
    oParameter.Direction = ParameterDirection.Output

    oParameter = CMD.Parameters.Add("@xCategoryName", SqlDbType.NVarChar, 15)
    oParameter.Value = x.CategoryName

    oParameter = CMD.Parameters.Add("@xDescription", SqlDbType.NVarChar, 1000)
    oParameter.Direction = ParameterDirection.Output

    oParameter = CMD.Parameters.Add("@xPicture", SqlDbType.VarBinary, 512002)
    'imagenes de hasta 500KB + 2 bytes de control internos
    oParameter.Direction = ParameterDirection.Output

    CMD.ExecuteNonQuery()
    x.CategoryID = IIf(IsDBNull(CMD.Parameters("@xCategoryID").Value), -1, CMD.Parameters("@xCategoryID").Value)
    x.Description = IIf(IsDBNull(CMD.Parameters("@xDescription").Value), "", CMD.Parameters("@xDescription").Value)
    If Not IsDBNull(CMD.Parameters("@xPicture").Value) Then
        x.Picture = CMD.Parameters("@xPicture").Value
    End If

    CN.Close()
    CN.Dispose()
    CMD.Dispose()
End Sub
End Class
```

5.2.4 Crear clase de negocio

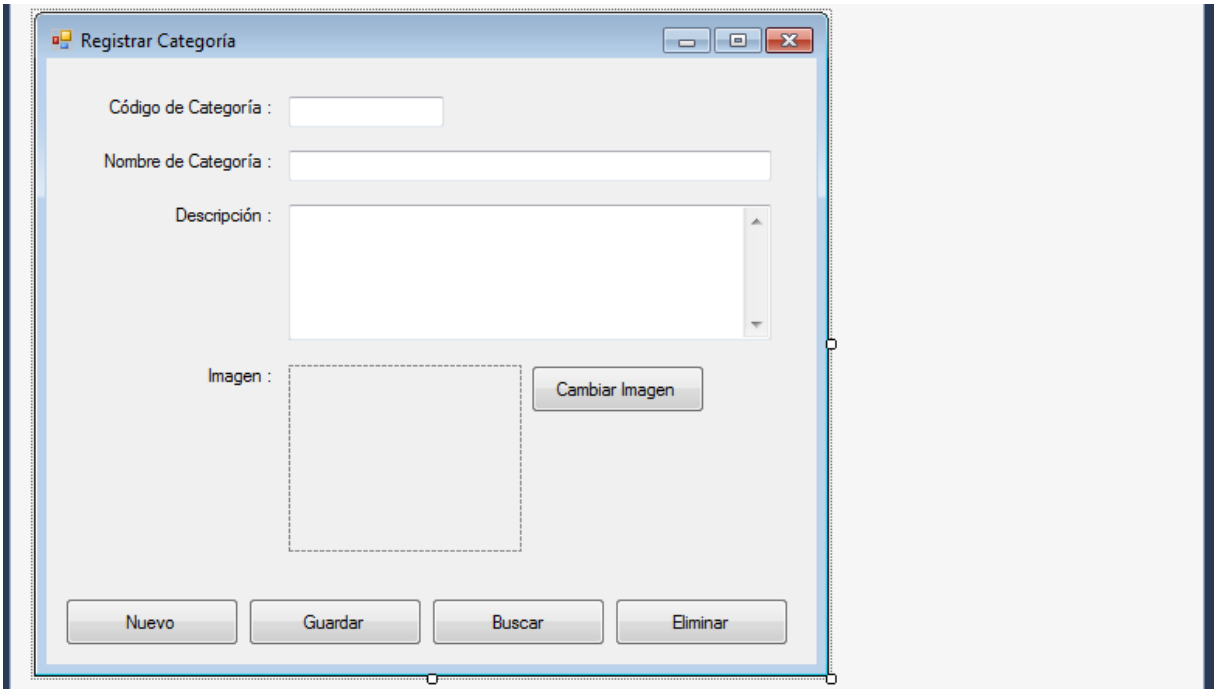
```
nCategory.vb   dShipper.vb   dCategory.vb   eCategory.vb   eShipper.vb   nShipper.vb
nCategory                                   Eliminar

Imports System
Imports pyLogicaDatos
Imports pyLogicaEntidad
0 referencias
Public Class nCategory
    0 referencias
    Public Sub Guardar(ByRef x As eCategory)
        Dim d As dCategory

        d = New dCategory
        If x.CategoryID <= 0 Then    ' es Categoria nueva
            d.Insertar(x)
        Else                         ' la categoria ya existe
            d.Actualizar(x)
        End If

        d = Nothing
    End Sub
    0 referencias
    Public Sub Eliminar(ByRef x As eCategory)
        Dim d As dCategory

        d = New dCategory
        If x.CategoryID > 0 Then     ' La categoria existe
            d.Eliminar(x)
        End If

        d = Nothing
    End Sub
```

```
    0 referencias
    Public Function Listar() As List(Of eCategory)
        Dim d As dCategory
        Dim rtn As List(Of eCategory)

        d = New dCategory()
        rtn = d.Listar()
        d = Nothing
        Return rtn
    End Function
    0 referencias
    Public Sub Buscar(ByRef x As eCategory)
        Dim d As dCategory

        d = New dCategory
        d.Buscar(x)
        d = Nothing
    End Sub
End Class
```

5.2.5 Crear formulario de presentación



Se agregan los siguientes controles:



Los controles de ingreso de datos tienen los siguientes nombres de arriba hacia abajo: txt_CategoryID, txt_CategoryName, txt_Descripcion, pb_Imagen, cb_CambiarImagen, cb_Nuevo, cb_Guardar, cb_Buscar, cb_Eliminar. Adicionalmente se ha establecido la propiedad **text** del formulario a ***"Registrar Categoría"*** y la propiedad **Enabled** de txt_CategoryID a ***False***.

La programación es como sigue:

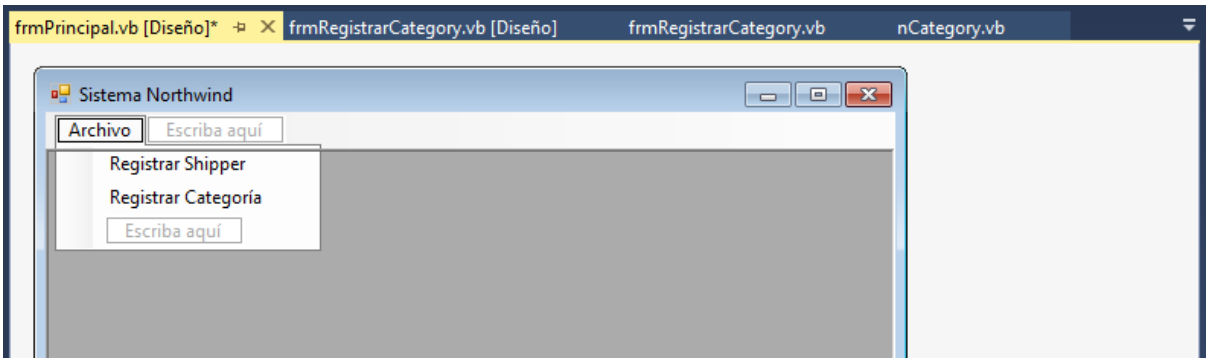
```
frmRegistrarCategory.vb [Diseño]   frmRegistrarCategory.vb   nCategory.vb   dShipper.vb
cb_Guardar                                                    Click

Imports pyLogicaEntidad
Imports pyLogicaNegocio
Imports System.IO
1 referencia
Public Class frmRegistrarCategory

    0 referencias
    Private Sub cb_Nuevo_Click(sender As Object, e As EventArgs) Handles cb_Nuevo.Click
        txt_CategoryID.Clear()
        txt_CategoryName.Clear()
        txt_Descripcion.Clear()
        pb_Imagen.ResetText()
    End Sub

    0 referencias
    Private Sub cb_CambiarImagen_Click(sender As Object, e As EventArgs) Handles cb_CambiarImagen
        Dim file As OpenFileDialog = New OpenFileDialog()

        file.Filter = "Archivo JPG|*.jpg"
        If (file.ShowDialog() = DialogResult.OK) Then
            pb_Imagen.Image = Image.FromFile(file.FileName)
        End If
    End Sub
```

```
frmRegistrarCategory.vb   nCategory.vb   dShipper.vb   eCategory.vb   dCategory.vb   eShipper.vb
cb_Guardar                                                    Click

    0 referencias
    Private Sub cb_Guardar_Click(sender As Object, e As EventArgs) Handles cb_Guardar.Click
        Dim n As nCategory
        Dim x As eCategory

        Try
            n = New nCategory()
            x = New eCategory()
            x.CategoryName = txt_CategoryName.Text
            x.Description = txt_Descripcion.Text
            x.PictureJPG = pb_Imagen.Image  'internamente se procesa y asigna valor a x.Picture

            If txt_CategoryID.Text <> "" Then
                x.CategoryID = CInt(txt_CategoryID.Text)
            Else
                x.CategoryID = -1
            End If
            n.Guardar(x)
            MsgBox("Se guardó correctamente", MsgBoxStyle.Information, Me.Text)
            txt_CategoryID.Text = x.CategoryID.ToString()

        Catch ex As Exception
            MsgBox("Error: " & ex.Message, MsgBoxStyle.Critical, Me.Text)

        Finally
            n = Nothing
            x = Nothing
        End Try
    End Sub
```

```vb
frmRegistrarCategory.vb   nCategory.vb   dShipper.vb   eCategory.vb   dCategory.vb   eShipper.vb
cb_Eliminar                                                                    Click
    0 referencias
    Private Sub cb_Eliminar_Click(sender As Object, e As EventArgs) Handles cb_Eliminar.Click
        Dim n As nCategory
        Dim x As eCategory

        Try
            If txt_CategoryID.Text <> "" Then
                n = New nCategory
                x = New eCategory
                x.CategoryID = CInt(txt_CategoryID.Text)
                n.Eliminar(x)
            End If
            MsgBox("Se eliminó correctamente", MsgBoxStyle.Information, Me.Text)

        Catch ex As Exception
            MsgBox("Error: " & ex.Message, MsgBoxStyle.Critical, Me.Text)

        Finally
            n = Nothing
            x = Nothing
        End Try
    End Sub
```

```vb
    Private Sub cb_Buscar_Click(sender As Object, e As EventArgs) Handles cb_Buscar.Click
        Dim n As nCategory
        Dim x As eCategory
        Try
            If txt_CategoryName.Text <> "" Then
                n = New nCategory
                x = New eCategory
                x.CategoryName = txt_CategoryName.Text.Trim()
                n.Buscar(x)
                If x.CategoryID >= 1 Then
                    MsgBox("Categoría encontrada", MsgBoxStyle.Information, Me.Text)
                    txt_CategoryID.Text = x.CategoryID.ToString()
                    txt_Descripcion.Text = x.Description
                    pb_Imagen.Image = x.PictureJPG
                Else
                    MsgBox("Categoría no encontrada", MsgBoxStyle.Information, Me.Text)
                End If
            End If

        Catch ex As Exception
            MsgBox("Error: " & ex.Message, MsgBoxStyle.Critical, Me.Text)

        Finally
            n = Nothing
            x = Nothing
        End Try

    End Sub
End Class
```

Añadimos una opción a la ventana principal:



Y codificamos como sigue:

frmPrincipal.vb | frmPrincipal.vb [Diseño] | frmRegistrarCategory.vb [Diseño] | frmRegistrarCategory.vb

RegistrarCategoríaToolStripMenuItem | Click

```
2 referencias
Public Class frmPrincipal

    0 referencias
    Private Sub RegistrarShipperToolStripMenuItem_Click(sender As Object, e As EventArgs) Handles Regist
        Dim form As frmRegistrarShipper

        form = New frmRegistrarShipper
        form.MdiParent = Me
        form.Show()

    End Sub

    0 referencias
    Private Sub RegistrarCategoríaToolStripMenuItem_Click(sender As Object, e As EventArgs) Handles Regi
        Dim form As frmRegistrarCategory

        form = New frmRegistrarCategory
        form.MdiParent = Me
        form.Show()
    End Sub
End Class
```

5.3 REGISTRAR PROVEEDOR (SUPPLIERS)

5.3.1 Crear Procedimientos almacenados de Inserción, Actualización y Eliminación

La tabla "Suppliers" tiene la siguiente estructura:

- dbo.Suppliers
  - Columnas
    - SupplierID (PK, int, No NULL)
    - CompanyName (nvarchar(40), No NULL)
    - ContactName (nvarchar(30), NULL)
    - ContactTitle (nvarchar(30), NULL)
    - Address (nvarchar(60), NULL)
    - City (nvarchar(15), NULL)
    - Region (nvarchar(15), NULL)
    - PostalCode (nvarchar(10), NULL)
    - Country (nvarchar(15), NULL)
    - Phone (nvarchar(24), NULL)
    - Fax (nvarchar(24), NULL)
    - HomePage (ntext, NULL)

Procedimiento almacenado para inserción:

SQLQuery1.sql - Wl...Northwind (sa (53))*

```sql
CREATE PROCEDURE [dbo].[Suppliers_Insertar](
    @xSupplierID int out,
    @xCompanyName nvarchar(40),
    @xContactName nvarchar(30),
    @xContactTitle nvarchar(30),
    @xAddress nvarchar(60),
    @xCity nvarchar(15),
    @xRegion nvarchar(15),
    @xPostalCode nvarchar(10),
    @xCountry nvarchar(15),
    @xPhone nvarchar(24),
    @xFax nvarchar(24),
    @xHomePage ntext
    )
AS
BEGIN
    INSERT INTO Suppliers (CompanyName, ContactName, ContactTitle, "Address",
        City, Region, PostalCode, Country, Phone, Fax, HomePage)
        VALUES
        (@xCompanyName, @xContactName, @xContactTitle, @xAddress,
        @xCity, @xRegion, @xPostalCode, @xCountry, @xPhone, @xFax, @xHomePage);

    SET @xSupplierID = @@IDENTITY;
END
```

100 %

Procedimiento almacenado para actualización:

```sql
CREATE PROCEDURE [dbo].[Suppliers_Actualizar](
    @xSupplierID int,
    @xCompanyName nvarchar(40),
    @xContactName nvarchar(30),
    @xContactTitle nvarchar(30),
    @xAddress nvarchar(60),
    @xCity nvarchar(15),
    @xRegion nvarchar(15),
    @xPostalCode nvarchar(10),
    @xCountry nvarchar(15),
    @xPhone nvarchar(24),
    @xFax nvarchar(24),
    @xHomePage ntext
    )
AS
BEGIN
    UPDATE Suppliers SET
        CompanyName = @xCompanyName,
        ContactName = @xContactName,
        ContactTitle = @xContactTitle,
        [Address] = @xAddress,
        City = @xCity,
        Region = @xRegion,
        PostalCode = @xPostalCode,
        Country = @xCountry,
        Phone = @xPhone,
        Fax = @xFax,
        HomePage = @xHomePage
    WHERE SupplierID = @xSupplierID;
END
```

Procedimiento almacenado de eliminación:

```sql
CREATE PROCEDURE [dbo].[Suppliers_Eliminar](
    @xSupplierID int
    )
AS
BEGIN
    DELETE FROM Suppliers
    WHERE SupplierID = @xSupplierID;
END
```

Procedimiento almacenado para buscar por nombre:

SQLQuery1.sql - Wl...Northwind (sa (54))*

```sql
CREATE PROCEDURE [dbo].[Suppliers_Buscar](
    @xSupplierID int out,
    @xCompanyName nvarchar(40),
    @xContactName nvarchar(30) out,
    @xContactTitle nvarchar(30) out,
    @xAddress nvarchar(60) out,
    @xCity nvarchar(15) out,
    @xRegion nvarchar(15) out,
    @xPostalCode nvarchar(10) out,
    @xCountry nvarchar(15) out,
    @xPhone nvarchar(24) out,
    @xFax nvarchar(24) out,
    @xHomePage nvarchar(200) out
    )
AS
BEGIN
    SELECT @xSupplierID = SupplierID,
        @xCompanyName = CompanyName,
        @xContactName = ContactName,
        @xContactTitle = ContactTitle,
        @xAddress = address,
        @xCity = City,
        @xRegion = Region,
        @xPostalCode = PostalCode,
        @xCountry = Country,
        @xFax = Fax,
        @xHomePage = CAST([HomePage] as nvarchar(200))
    FROM Suppliers
    WHERE CompanyName = @xCompanyName;
END
```

5.3.2 Crear Clase Entidad

eSupplier.vb | eProduct.vb | nCategory.vb | dCategory.vb | eCategory.vb

(General) | (Declaraciones)

```vb
0 referencias
Public Class eSupplier
    0 referencias
    Public Property SupplierID As Integer
    0 referencias
    Public Property CompanyName As String
    0 referencias
    Public Property ContactName As String
    0 referencias
    Public Property ContactTitle As String
    0 referencias
    Public Property Address As String
    0 referencias
    Public Property City As String
    0 referencias
    Public Property Region As String
    0 referencias
    Public Property PostalCode As String
    0 referencias
    Public Property Country As String
    0 referencias
    Public Property Phone As String
    0 referencias
    Public Property Fax As String
    0 referencias
    Public Property HomePage As String
End Class
```

5.3.3 Crear clase de acceso a datos

dSupplier.vb | eSupplier.vb | eProduct.vb | nCategory.vb | dCategory.vb | eCategory.vb

dSupplier | (Declaraciones)

```vb
Imports System
Imports System.Data.SqlClient
Imports pyLogicaEntidad
0 referencias
Public Class dSupplier
    0 referencias
    Public Sub Insertar(ByRef x As eSupplier)
        Dim CN As SqlConnection
        Dim CMD As SqlCommand

        CN = New SqlClient.SqlConnection(CadenaConexionDB())
        CN.Open()
        CMD = New SqlCommand                            'utilizamos una variable tipo comando para acce
        CMD.CommandType = CommandType.StoredProcedure   'especificamos que abriremos un procedimiento a
        CMD.Connection = CN                             'especificamos la conexion
        CMD.CommandText = "Suppliers_Insertar"         'especificamos el nombre del procedimiento

        CMD.Parameters.Add("@xSupplierID", SqlDbType.Int).Direction = ParameterDirection.Output
        CMD.Parameters.Add("@xCompanyName", SqlDbType.NVarChar, 40).Value = x.CompanyName
        CMD.Parameters.Add("@xContactName", SqlDbType.NVarChar, 30).Value = x.ContactName
        CMD.Parameters.Add("@xContactTitle", SqlDbType.NVarChar, 30).Value = x.ContactTitle
        CMD.Parameters.Add("@xAddress", SqlDbType.NVarChar, 60).Value = x.Address
        CMD.Parameters.Add("@xCity", SqlDbType.NVarChar, 15).Value = x.City
        CMD.Parameters.Add("@xRegion", SqlDbType.NVarChar, 15).Value = x.Region
        CMD.Parameters.Add("@xPostalCode", SqlDbType.NVarChar, 10).Value = x.PostalCode
        CMD.Parameters.Add("@xCountry", SqlDbType.NVarChar, 15).Value = x.Country
        CMD.Parameters.Add("@xPhone", SqlDbType.NVarChar, 24).Value = x.Phone
        CMD.Parameters.Add("@xFax", SqlDbType.NVarChar, 24).Value = x.Fax
        CMD.Parameters.Add("@xHomePage", SqlDbType.NText).Value = x.HomePage

        CMD.ExecuteNonQuery()
        x.SupplierID = CMD.Parameters(0).Value

        CN.Close()
        CN.Dispose()
        CMD.Dispose()
    End Sub
```

```vb
    0 referencias
    Public Sub Eliminar(ByRef x As eSupplier)
        Dim CN As SqlConnection
        Dim CMD As SqlCommand

        CN = New SqlClient.SqlConnection(CadenaConexionDB())
        CN.Open()
        CMD = New SqlCommand                            'utilizamos una variable tipo comando para acce
        CMD.CommandType = CommandType.StoredProcedure   'especificamos que abriremos un procedimiento a
        CMD.Connection = CN                             'especificamos la conexion
        CMD.CommandText = "Suppliers_Eliminar"         'especificamos el nombre del procedimiento

        CMD.Parameters.Add("@xSupplierID", SqlDbType.Int).Value = x.SupplierID
        CMD.ExecuteNonQuery()

        CN.Close()
        CN.Dispose()
        CMD.Dispose()
    End Sub
```

```
1 referencia
Public Sub Actualizar(ByRef x As eSupplier)
    Dim CN As SqlConnection
    Dim CMD As SqlCommand

    CN = New SqlClient.SqlConnection(CadenaConexionDB())
    CN.Open()
    CMD = New SqlCommand                            'utilizamos una variable tipo comando para acce
    CMD.CommandType = CommandType.StoredProcedure   'especificamos que abriremos un procedimiento a
    CMD.Connection = CN                             'especificamos la conexion
    CMD.CommandText = "Suppliers_Actualizar"         'especificamos el nombre del procedimiento

    CMD.Parameters.Add("@xSupplierID", SqlDbType.Int).Value = x.SupplierID
    CMD.Parameters.Add("@xCompanyName", SqlDbType.NVarChar, 40).Value = x.CompanyName
    CMD.Parameters.Add("@xContactName", SqlDbType.NVarChar, 30).Value = x.ContactName
    CMD.Parameters.Add("@xContactTitle", SqlDbType.NVarChar, 30).Value = x.ContactTitle
    CMD.Parameters.Add("@xAddress", SqlDbType.NVarChar, 60).Value = x.Address
    CMD.Parameters.Add("@xCity", SqlDbType.NVarChar, 15).Value = x.City
    CMD.Parameters.Add("@xRegion", SqlDbType.NVarChar, 15).Value = x.Region
    CMD.Parameters.Add("@xPostalCode", SqlDbType.NVarChar, 10).Value = x.PostalCode
    CMD.Parameters.Add("@xCountry", SqlDbType.NVarChar, 15).Value = x.Country
    CMD.Parameters.Add("@xPhone", SqlDbType.NVarChar, 24).Value = x.Phone
    CMD.Parameters.Add("@xFax", SqlDbType.NVarChar, 24).Value = x.Fax
    CMD.Parameters.Add("@xHomePage", SqlDbType.NText).Value = x.HomePage

    CMD.ExecuteNonQuery()
    CN.Close()
    CN.Dispose()
    CMD.Dispose()
End Sub
```

```
0 referencias
Public Function Listar() As List(Of eSupplier)
    Dim CN As SqlConnection
    Dim CMD As SqlCommand
    Dim rtn As List(Of eSupplier)
    Dim Reg As SqlDataReader
    Dim x As eSupplier

    CN = New SqlClient.SqlConnection(CadenaConexionDB())
    CN.Open()
    CMD = New SqlCommand                'utilizamos una variable tipo comando para acceder a los pr
    CMD.CommandType = CommandType.Text  'especificamos que usaremos una consulta
    CMD.Connection = CN                 'especificamos la conexion
    CMD.CommandText = "Select CategoryID, CategoryName from Categories Order By 2"
    Reg = CMD.ExecuteReader(CommandBehavior.CloseConnection)
    rtn = New List(Of eSupplier)
    While (Reg.Read())
        x = New eSupplier()
        x.SupplierID = Reg.GetInt32(0)
        x.CompanyName = Reg.GetString(1)
        x.ContactName = Reg.GetString(2)
        x.ContactTitle = Reg.GetString(3)
        x.Address = Reg.GetString(4)
        x.City = Reg.GetString(5)
        x.Region = Reg.GetString(6)
        x.PostalCode = Reg.GetString(7)
        x.Country = Reg.GetString(8)
        x.Phone = Reg.GetString(9)
        x.Fax = Reg.GetString(10)
        x.HomePage = Reg.GetString(11)
        rtn.Add(x)
    End While

    Reg.Close()
    CN.Dispose()
    CMD.Dispose()
    Return (rtn)
End Function
```

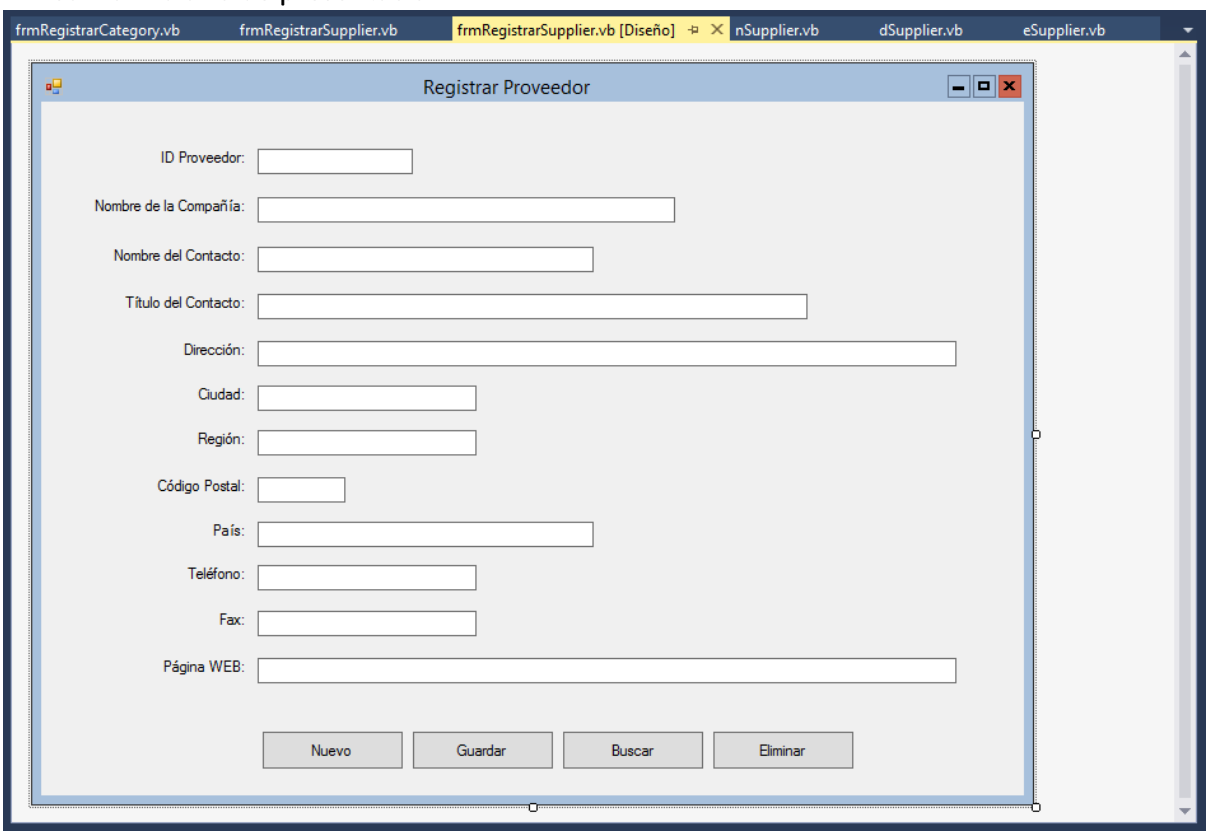
```
0 referencias
Public Sub Buscar(ByRef x As eSupplier)
    Dim CN As SqlConnection
    Dim CMD As SqlCommand

    CN = New SqlClient.SqlConnection(CadenaConexionDB())
    CN.Open()
    CMD = New SqlCommand                          'utilizamos una variable tipo comando para acceder a los procedimient
    CMD.CommandType = CommandType.StoredProcedure  'especificamos que abriremos un procedimiento almacenado
    CMD.Connection = CN                           'especificamos la conexion
    CMD.CommandText = "Suppliers_Buscar"          'especificamos el nombre del procedimiento

    CMD.Parameters.Add("@xSupplierID", SqlDbType.Int).Value = x.CompanyName
    CMD.Parameters.Add("@xCompanyName", SqlDbType.NVarChar, 40).Direction = ParameterDirection.Output
    CMD.Parameters.Add("@xContactName", SqlDbType.NVarChar, 30).Direction = ParameterDirection.Output
    CMD.Parameters.Add("@xContactTitle", SqlDbType.NVarChar, 30).Direction = ParameterDirection.Output
    CMD.Parameters.Add("@xAddress", SqlDbType.NVarChar, 60).Direction = ParameterDirection.Output
    CMD.Parameters.Add("@xCity", SqlDbType.NVarChar, 15).Direction = ParameterDirection.Output
    CMD.Parameters.Add("@xRegion", SqlDbType.NVarChar, 15).Direction = ParameterDirection.Output
    CMD.Parameters.Add("@xPostalCode", SqlDbType.NVarChar, 10).Direction = ParameterDirection.Output
    CMD.Parameters.Add("@xCountry", SqlDbType.NVarChar, 15).Direction = ParameterDirection.Output
    CMD.Parameters.Add("@xPhone", SqlDbType.NVarChar, 24).Direction = ParameterDirection.Output
    CMD.Parameters.Add("@xFax", SqlDbType.NVarChar, 24).Direction = ParameterDirection.Output
    CMD.Parameters.Add("@xHomePage", SqlDbType.NText).Direction = ParameterDirection.Output

    CMD.ExecuteNonQuery()
    x.CompanyName = IIf(IsDBNull(CMD.Parameters("@xCompanyName").Value), "", CMD.Parameters("@xCompanyName").Value
    x.ContactName = IIf(IsDBNull(CMD.Parameters("@xContactName").Value), "", CMD.Parameters("@xContactName").Value)
    x.ContactTitle = IIf(IsDBNull(CMD.Parameters("@xContactTitle").Value), "", CMD.Parameters("@xContactTitle").Value)
    x.Address = IIf(IsDBNull(CMD.Parameters("@xAddress").Value), "", CMD.Parameters("@xAddress").Value)
    x.City = IIf(IsDBNull(CMD.Parameters("@xCity").Value), "", CMD.Parameters("@xCity").Value)
    x.Region = IIf(IsDBNull(CMD.Parameters("@xRegion").Value), "", CMD.Parameters("@xRegion").Value)
    x.PostalCode = IIf(IsDBNull(CMD.Parameters("@xPostalCode").Value), "", CMD.Parameters("@xPostalCode").Value)
    x.Country = IIf(IsDBNull(CMD.Parameters("@xCountry").Value), "", CMD.Parameters("@xCountry").Value)
    x.Fax = IIf(IsDBNull(CMD.Parameters("@xFax").Value), "", CMD.Parameters("@xFax").Value)
    x.HomePage = IIf(IsDBNull(CMD.Parameters("@xHomePage").Value), "", CMD.Parameters("@xHomePage").Value)

    CN.Close()
    CN.Dispose()
    CMD.Dispose()
End Sub
End Class
```

5.3.4 Crear clase de negocio

```
dCategory.vb   nSupplier.vb   dSupplier.vb   eCategory.vb   eSupplier.vb
nSupplier                                    Eliminar

Imports System
Imports pyLogicaDatos
Imports pyLogicaEntidad
0 referencias
Public Class nSupplier
    0 referencias
    Public Sub Guardar(ByRef x As eSupplier)
        Dim d As dSupplier

        d = New dSupplier()
        If x.SupplierID <= 0 Then    ' es Proveedor nuevo
            d.Insertar(x)
        Else                         ' el Proveedor ya existe
            d.Actualizar(x)
        End If

        d = Nothing
    End Sub
    0 referencias
    Public Sub Eliminar(ByRef x As eSupplier)
        Dim d As dSupplier

        d = New dSupplier()
        If x.SupplierID > 0 Then     ' el Proveedor existe
            d.Eliminar(x)
        End If

        d = Nothing
    End Sub
```

```
    0 referencias
    Public Function Listar() As List(Of eSupplier)
        Dim d As dSupplier
        Dim rtn As List(Of eSupplier)

        d = New dSupplier()
        rtn = d.Listar()
        d = Nothing
        Return rtn
    End Function
    0 referencias
    Public Sub Buscar(ByRef x As eSupplier)
        Dim d As dSupplier

        d = New dSupplier()
        d.Buscar(x)
        d = Nothing
    End Sub
End Class
```

5.3.5 Crear formulario de presentación



Los nombres de los controles están de acuerdo a los nombres de los campos de la tabla en la base de datos y de arriba hacia abajo son: txt_SupplierID, txt_CompanyName, txt_ContactName, txt_ContactTitle, txt_Address, txt_City, txt_Region, txt_City, txt_PostalCode, txt_Country, txt_Phone, txt_Fax, txt_HomePage.

Los botones de comando tienen los siguientes nombres: cb_Nuevo, cb_Guardar, cb_Buscar, cb_Eliminar.

La Programación de la ventana es como sigue:

frmRegistrarCategory.vb | frmRegistrarSupplier.vb | frmRegistrarSupplier.vb [Diseño] | nSupplier.vb

cb_Nuevo | Click

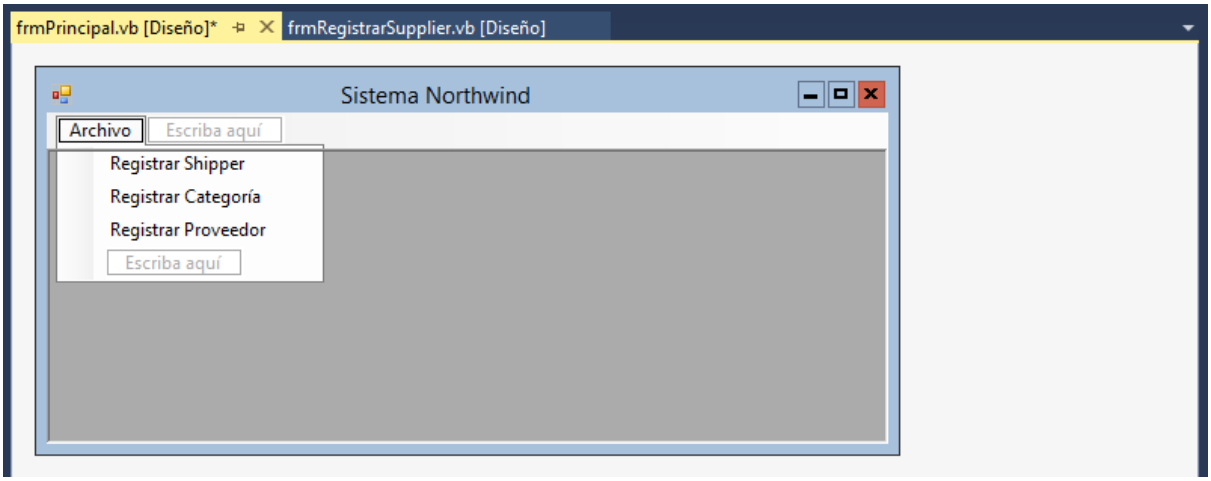
```
1 referencia
Public Class frmRegistrarSupplier

    0 referencias
    Private Sub cb_Nuevo_Click(sender As Object, e As EventArgs) Handles cb_Nuevo.Click
        txt_SupplierID.Clear()
        txt_CompanyName.Clear()
        txt_ContactName.Clear()
        txt_ContactTitle.Clear()
        txt_Address.Clear()
        txt_City.Clear()
        txt_Region.Clear()
        txt_PostalCode.Clear()
        txt_Country.Clear()
        txt_Phone.Clear()
        txt_Fax.Clear()
        txt_HomePage.Clear()
    End Sub
```

```
    0 referencias
    Private Sub cb_Guardar_Click(sender As Object, e As EventArgs) Handles cb_Guardar.Click
        Dim n As nSupplier
        Dim x As eSupplier

        Try
            n = New nSupplier()
            x = New eSupplier()
            x.CompanyName = txt_CompanyName.Text
            x.ContactName = txt_ContactName.Text
            x.ContactTitle = txt_ContactTitle.Text
            x.Address = txt_Address.Text
            x.City = txt_City.Text
            x.Region = txt_Region.Text
            x.PostalCode = txt_PostalCode.Text
            x.Country = txt_Country.Text
            x.Phone = txt_Phone.Text
            x.Fax = txt_Fax.Text
            x.HomePage = txt_HomePage.Text

            If txt_SupplierID.Text <> "" Then
                x.SupplierID = CInt(txt_SupplierID.Text)
            Else
                x.SupplierID = -1
            End If
            n.Guardar(x)
            MsgBox("Se guardó correctamente", MsgBoxStyle.Information, Me.Text)
            txt_SupplierID.Text = x.SupplierID.ToString()

        Catch ex As Exception
            MsgBox("Error: " & ex.Message, MsgBoxStyle.Critical, Me.Text)

        Finally
            n = Nothing
            x = Nothing
        End Try

    End Sub
```

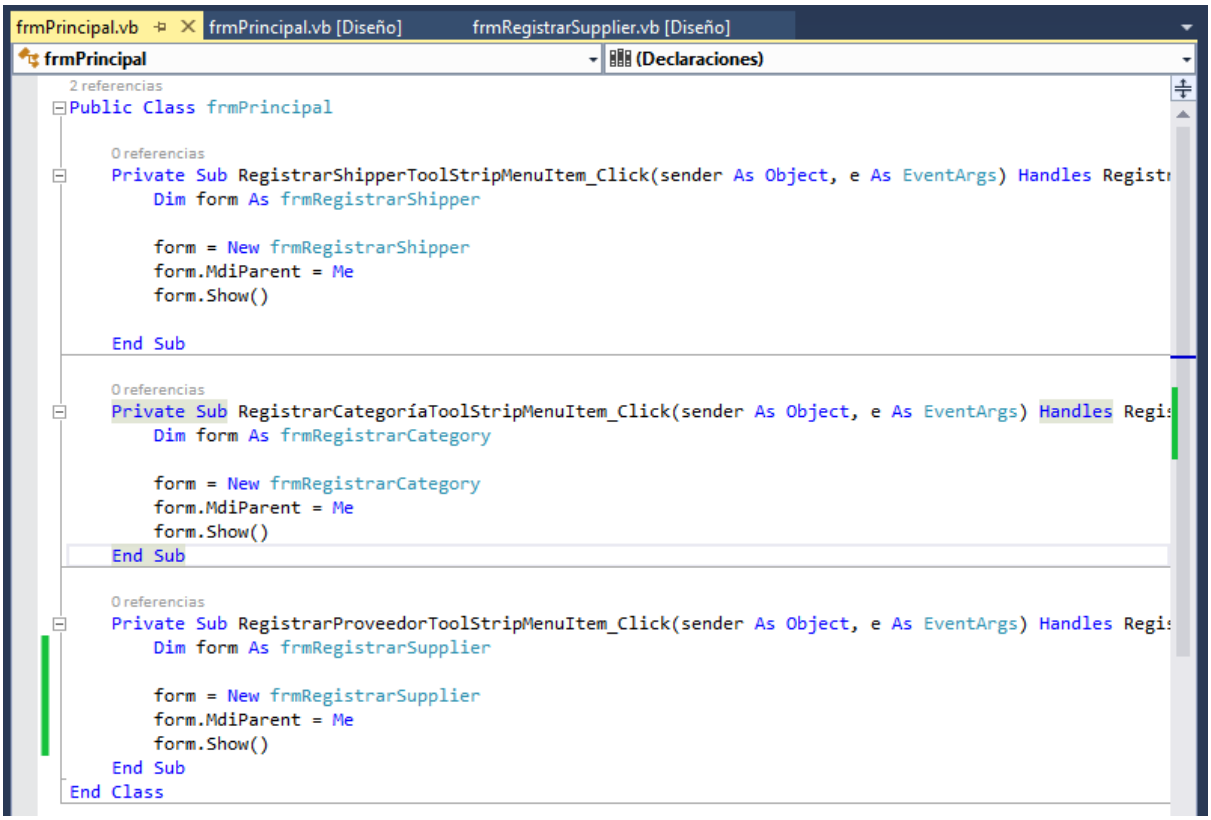
```
0 referencias
Private Sub cb_Buscar_Click(sender As Object, e As EventArgs) Handles cb_Buscar.Click
    Dim n As nSupplier
    Dim x As eSupplier
    Try
        If txt_CompanyName.Text <> "" Then
            n = New nSupplier
            x = New eSupplier
            x.CompanyName = txt_CompanyName.Text.Trim()
            n.Buscar(x)
            If x.SupplierID >= 1 Then
                MsgBox("Categoría encontrada", MsgBoxStyle.Information, Me.Text)
                txt_CompanyName.Text = x.CompanyName.ToString()
                txt_ContactName.Text = x.ContactName
                txt_ContactTitle.Text = x.ContactTitle
                txt_Address.Text = x.Address
                txt_City.Text = x.City
                txt_Region.Text = x.Region
                txt_PostalCode.Text = x.PostalCode
                txt_Country.Text = x.Country
                txt_Phone.Text = x.Phone
                txt_Fax.Text = x.Fax
                txt_HomePage.Text = x.HomePage
            Else
                MsgBox("Proveedor no encontrado!", MsgBoxStyle.Information, Me.Text)
            End If
        End If

    Catch ex As Exception
        MsgBox("Error: " & ex.Message, MsgBoxStyle.Critical, Me.Text)

    Finally
        n = Nothing
        x = Nothing
    End Try

End Sub
```

```
0 referencias
Private Sub cb_Eliminar_Click(sender As Object, e As EventArgs) Handles cb_Eliminar.Click
    Dim n As nSupplier
    Dim x As eSupplier

    Try
        If txt_SupplierID.Text <> "" Then
            n = New nSupplier
            x = New eSupplier
            x.SupplierID = CInt(txt_SupplierID.Text)
            n.Eliminar(x)
        End If
        MsgBox("Se eliminó correctamente", MsgBoxStyle.Information, Me.Text)

    Catch ex As Exception
        MsgBox("Error: " & ex.Message, MsgBoxStyle.Critical, Me.Text)

    Finally
        n = Nothing
        x = Nothing
    End Try
End Sub
End Class
```

Agregamos una opción al menú principal:



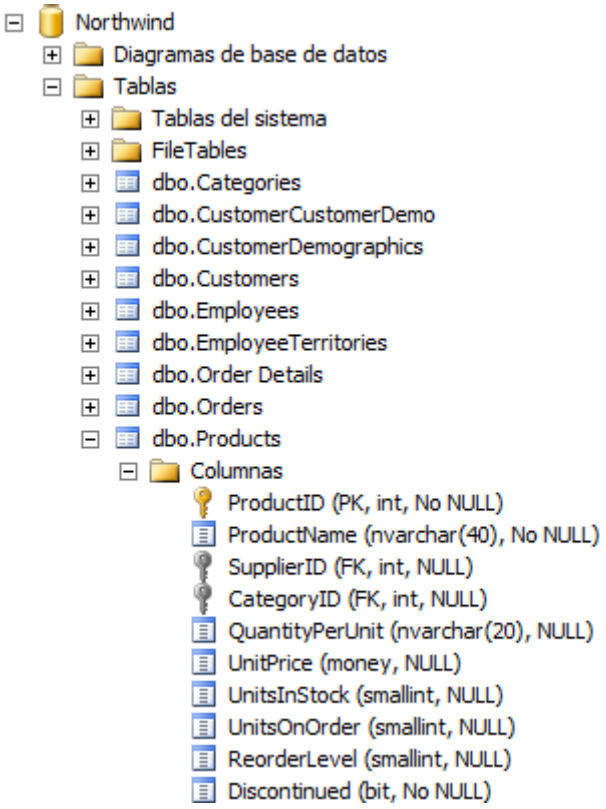
Y añadimos el siguiente código:



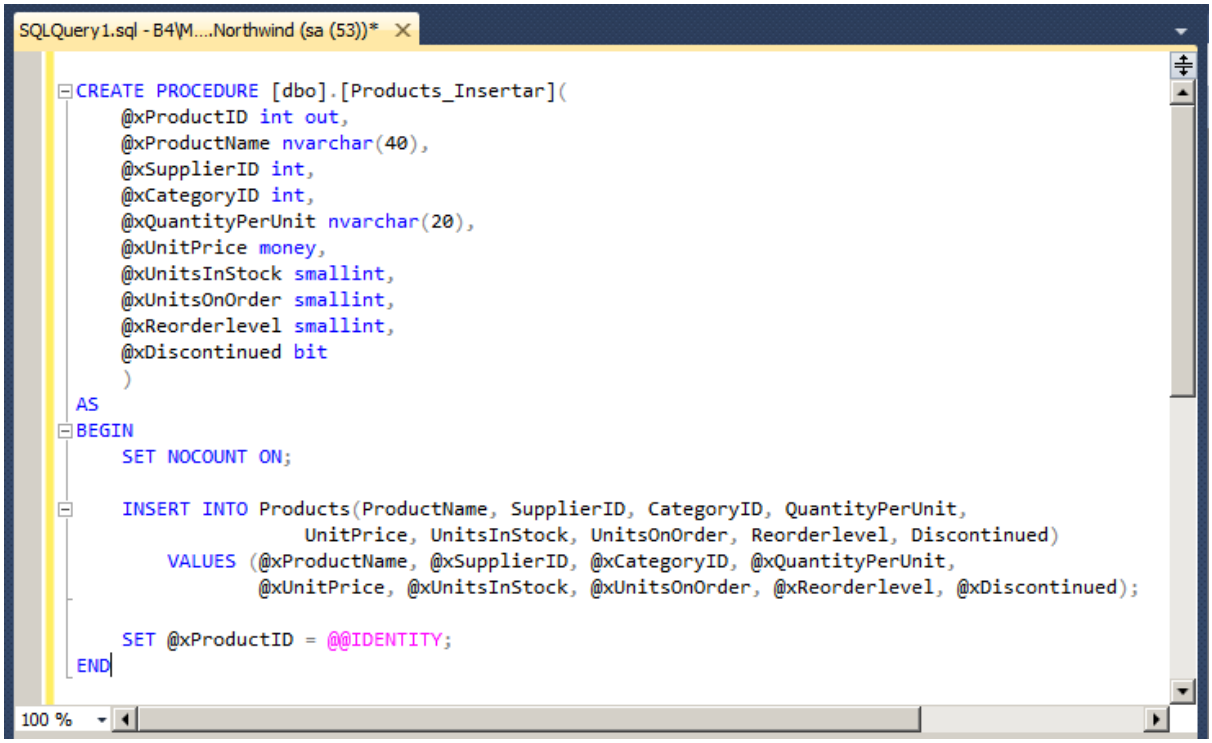
5.4 REGISTRAR PRODUCTO (PRODUCTS)

5.4.1 Crear Procedimientos almacenados de Inserción, Actualización y Eliminación

La tabla "Products" tiene la siguiente estructura:



Procedimiento almacenado para inserción:




```sql
CREATE PROCEDURE [dbo].[Products_Insertar](
    @xProductID int out,
    @xProductName nvarchar(40),
    @xSupplierID int,
    @xCategoryID int,
    @xQuantityPerUnit nvarchar(20),
    @xUnitPrice money,
    @xUnitsInStock smallint,
    @xUnitsOnOrder smallint,
    @xReorderlevel smallint,
    @xDiscontinued bit
    )
AS
BEGIN
    SET NOCOUNT ON;

    INSERT INTO Products(ProductName, SupplierID, CategoryID, QuantityPerUnit,
                    UnitPrice, UnitsInStock, UnitsOnOrder, Reorderlevel, Discontinued)
        VALUES (@xProductName, @xSupplierID, @xCategoryID, @xQuantityPerUnit,
                @xUnitPrice, @xUnitsInStock, @xUnitsOnOrder, @xReorderlevel, @xDiscontinued);

    SET @xProductID = @@IDENTITY;
END
```

Procedimiento almacenado para actualización:

SQLQuery1.sql - B4\M....Northwind (sa (53))*

```sql
CREATE PROCEDURE [dbo].[Products_Actualizar](
    @xProductID int,
    @xProductName nvarchar(40),
    @xSupplierID int,
    @xCategoryID int,
    @xQuantityPerUnit nvarchar(20),
    @xUnitPrice money,
    @xUnitsInStock smallint,
    @xUnitsOnOrder smallint,
    @xReorderlevel smallint,
    @xDiscontinued bit
    )
AS
BEGIN
    UPDATE Products SET
        ProductName = @xProductName, SupplierID = @xSupplierID, CategoryID = @xCategoryID,
        QuantityPerUnit = @xQuantityPerUnit, @xUnitPrice = UnitPrice,
        UnitsInStock = @xUnitsInStock, UnitsOnOrder = @xUnitsOnOrder,
        Reorderlevel = @xReorderlevel, Discontinued = @xDiscontinued
    WHERE ProductID = @xProductID
END
```

Procedimiento almacenado de eliminación:

SQLQuery1.sql - B4\M....Northwind (sa (53))*

```sql
CREATE PROCEDURE [dbo].[Products_Eliminar](
    @xProductID int
    )
AS
BEGIN
    DELETE FROM Products
    WHERE ProductID = @xProductID
END
```

parámetro @xProductID int

SQLQuery1.sql - Wl...Northwind (sa (53))*

```sql
CREATE PROCEDURE [dbo].[Products_Buscar](
    @xProductID int out,
    @xProductName nvarchar(40),
    @xSupplierID int out,
    @xCategoryID int out,
    @xQuantityPerUnit nvarchar(20) out,
    @xUnitPrice money out,
    @xUnitsInStock smallint out,
    @xUnitsOnOrder smallint out,
    @xReorderlevel smallint out,
    @xDiscontinued bit out
    )
AS
BEGIN
    SELECT @xProductID = ProductID, @xSupplierID = SupplierID,
        @xCategoryID = CategoryID, @xQuantityPerUnit = QuantityPerUnit,
        @xUnitPrice = UnitPrice, @xUnitsInStock = UnitsInStock,
        @xUnitsOnOrder = UnitsOnOrder, @xReorderlevel = ReorderLevel,
        @xDiscontinued = Discontinued
    FROM Products
    WHERE ProductName = @xProductName
END
```

5.4.2 Crear Clase Entidad

```
eProduct.vb   nCategory.vb   eCategory.vb
eProduct                    (Declaraciones)
0 referencias
Public Class eProduct
    0 referencias
    Public Property ProductID As Integer
    0 referencias
    Public Property ProductName As String
    0 referencias
    Public Property SupplierID As Integer
    0 referencias
    Public Property CategoryID As Integer
    0 referencias
    Public Property QuantityPerUnit As String
    0 referencias
    Public Property UnitPrice As Decimal
    0 referencias
    Public Property UnitsInStock As Integer
    0 referencias
    Public Property UnitsOnOrder As Integer
    0 referencias
    Public Property ReorderLevel As Integer
    0 referencias
    Public Property Discontinued As Boolean
End Class
```

5.4.3 Crear clase de acceso a datos

```
nProduct.vb   dProduct.vb   frmRegistrarCategory.vb [Diseño]   frmRegistrarProduct.vb [Diseño]
dProduct                    Eliminar
Imports pyLogicaEntidad
Imports System.Data.SqlClient
8 referencias
Public Class dProduct
    1 referencia
    Public Sub Eliminar(ByRef x As eProduct)
        Dim CN As SqlConnection
        Dim CMD As SqlCommand
        Dim oParameter As SqlParameter

        CN = New SqlClient.SqlConnection(CadenaConexionDB())
        CN.Open()
        CMD = New SqlCommand                          'utilizamos una variable tipo coman
        CMD.CommandType = CommandType.StoredProcedure 'especificamos que abriremos un pro
        CMD.Connection = CN                           'especificamos la conexion
        CMD.CommandText = "Products_Eliminar"         'especificamos el nombre del proced

        oParameter = CMD.Parameters.Add("@xProductID", SqlDbType.Int)
        oParameter.Value = x.ProductID
        CMD.ExecuteNonQuery()

        CN.Close()
        CN.Dispose()
        CMD.Dispose()
    End Sub
```

dProduct.vb | frmRegistrarCategory.vb [Diseño] | frmRegistrarProduct.vb [Diseño] | frmRegistrarProduct.vb

dProduct | Eliminar

```
0 referencias
Public Sub Insertar(ByRef x As eProduct)
    Dim CN As SqlConnection
    Dim CMD As SqlCommand

    CN = New SqlClient.SqlConnection(CadenaConexionDB())
    CN.Open()
    CMD = New SqlCommand                            'utilizamos una variable tipo comando para acce
    CMD.CommandType = CommandType.StoredProcedure   'especificamos que abriremos un procedimiento a
    CMD.Connection = CN                             'especificamos la conexion
    CMD.CommandText = "Products_Insertar"           'especificamos el nombre del procedimiento

    CMD.Parameters.Add("@xProductID", SqlDbType.Int).Direction = ParameterDirection.Output
    CMD.Parameters.Add("@xProductName", SqlDbType.NVarChar, 40).Value = x.ProductName
    CMD.Parameters.Add("@xSupplierID", SqlDbType.Int).Value = x.SupplierID
    CMD.Parameters.Add("@xCategoryID", SqlDbType.Int).Value = x.CategoryID
    CMD.Parameters.Add("@xQuantityPerUnit", SqlDbType.NVarChar, 20).Value = x.QuantityPerUnit
    CMD.Parameters.Add("@xUnitPrice", SqlDbType.Money).Value = x.UnitPrice
    CMD.Parameters.Add("@xUnitsInStock", SqlDbType.SmallInt).Value = x.UnitsInStock
    CMD.Parameters.Add("@xUnitsOnOrder", SqlDbType.SmallInt).Value = x.UnitsOnOrder
    CMD.Parameters.Add("@xReorderLevel", SqlDbType.SmallInt).Value = x.UnitPrice
    CMD.Parameters.Add("@xDiscontinued", SqlDbType.Bit).Value = x.Discontinued
    CMD.ExecuteNonQuery()
    x.CategoryID = CMD.Parameters(0).Value

    CN.Close()
    CN.Dispose()
    CMD.Dispose()
End Sub
```

dProduct.vb | frmRegistrarCategory.vb [Diseño] | frmRegistrarProduct.vb [Diseño] | frmRegistrarProduct.vb

dProduct | Actualizar

```
0 referencias
Public Sub Actualizar(ByRef x As eProduct)
    Dim CN As SqlConnection
    Dim CMD As SqlCommand

    CN = New SqlClient.SqlConnection(CadenaConexionDB())
    CN.Open()
    CMD = New SqlCommand                            'utilizamos una variable tipo comando para acce
    CMD.CommandType = CommandType.StoredProcedure   'especificamos que abriremos un procedimiento a
    CMD.Connection = CN                             'especificamos la conexion
    CMD.CommandText = "Products_Actualizar"         'especificamos el nombre del procedimiento

    CMD.Parameters.Add("@xProductID", SqlDbType.Int).Value = x.ProductID
    CMD.Parameters.Add("@xProductName", SqlDbType.NVarChar, 40).Value = x.ProductName
    CMD.Parameters.Add("@xSupplierID", SqlDbType.Int).Value = x.SupplierID
    CMD.Parameters.Add("@xCategoryID", SqlDbType.Int).Value = x.CategoryID
    CMD.Parameters.Add("@xQuantityPerUnit", SqlDbType.NVarChar, 20).Value = x.QuantityPerUnit
    CMD.Parameters.Add("@xUnitPrice", SqlDbType.Money).Value = x.UnitPrice
    CMD.Parameters.Add("@xUnitsInStock", SqlDbType.SmallInt).Value = x.UnitsInStock
    CMD.Parameters.Add("@xUnitsOnOrder", SqlDbType.SmallInt).Value = x.UnitsOnOrder
    CMD.Parameters.Add("@xReorderLevel", SqlDbType.SmallInt).Value = x.UnitPrice
    CMD.Parameters.Add("@xDiscontinued", SqlDbType.Bit).Value = x.Discontinued

    CMD.ExecuteNonQuery()
    CN.Close()
    CN.Dispose()
    CMD.Dispose()
End Sub
```

dProduct.vb | frmRegistrarCategory.vb [Diseño] | frmRegistrarProduct.vb [Diseño]

dProduct | Listar

```vb
0 referencias
Public Function Listar() As List(Of eProduct)
    Dim CN As SqlConnection
    Dim CMD As SqlCommand
    Dim rtn As List(Of eProduct)
    Dim Reg As SqlDataReader
    Dim x As eProduct

    CN = New SqlClient.SqlConnection(CadenaConexionDB())
    CN.Open()
    CMD = New SqlCommand                    'utilizamos una variable tipo comando para acceder
    CMD.CommandType = CommandType.Text      'especificamos que usaremos una consulta
    CMD.Connection = CN                     'especificamos la conexion
    CMD.CommandText = "Select ProductID, ProductName from Products Order By ProductName"
    Reg = CMD.ExecuteReader(CommandBehavior.CloseConnection)
    rtn = New List(Of eProduct)
    While (Reg.Read())
        x = New eProduct()
        x.ProductID = Reg.GetInt32(0)
        x.ProductName = Reg.GetString(1)
        rtn.Add(x)
    End While

    Reg.Close()
    CN.Dispose()
    CMD.Dispose()
    Return (rtn)
End Function
```

dProduct.vb | frmRegistrarCategory.vb [Diseño] | frmRegistrarProduct.vb [Diseño] | frmRegistrarProduct.vb | frmRegistrarSupplier.vb [Diseño]

dProduct | Buscar

```vb
0 referencias
Public Sub Buscar(ByRef x As eProduct)
    Dim CN As SqlConnection
    Dim CMD As SqlCommand

    CN = New SqlClient.SqlConnection(CadenaConexionDB())
    CN.Open()
    CMD = New SqlCommand                        'utilizamos una variable tipo comando para acceder a los procedimientos alma
    CMD.CommandType = CommandType.StoredProcedure   'especificamos que abriremos un procedimiento almacenado
    CMD.Connection = CN                         'especificamos la conexion
    CMD.CommandText = "Products_Buscar"         'especificamos el nombre del procedimiento

    CMD.Parameters.Add("@xProductID", SqlDbType.Int).Direction = ParameterDirection.Output
    CMD.Parameters.Add("@xProductName", SqlDbType.NVarChar, 40).Value = x.ProductName
    CMD.Parameters.Add("@xSupplierID", SqlDbType.Int).Direction = ParameterDirection.Output
    CMD.Parameters.Add("@xCategoryID", SqlDbType.Int).Direction = ParameterDirection.Output
    CMD.Parameters.Add("@xQuantityPerUnit", SqlDbType.NVarChar, 20).Direction = ParameterDirection.Output
    CMD.Parameters.Add("@xUnitPrice", SqlDbType.Money).Direction = ParameterDirection.Output
    CMD.Parameters.Add("@xUnitsInStock", SqlDbType.SmallInt).Direction = ParameterDirection.Output
    CMD.Parameters.Add("@xUnitsOnOrder", SqlDbType.SmallInt).Direction = ParameterDirection.Output
    CMD.Parameters.Add("@xReorderLevel", SqlDbType.SmallInt).Direction = ParameterDirection.Output
    CMD.Parameters.Add("@xDiscontinued", SqlDbType.Bit).Direction = ParameterDirection.Output

    CMD.ExecuteNonQuery()
    x.ProductID = IIf(IsDBNull(CMD.Parameters("@xProductID").Value), -1, CMD.Parameters("@xProductID").Value)
    x.SupplierID = IIf(IsDBNull(CMD.Parameters("@xSupplierID").Value), -1, CMD.Parameters("@xSupplierID").Value)
    x.CategoryID = IIf(IsDBNull(CMD.Parameters("@xCategoryID").Value), -1, CMD.Parameters("@xCategoryID").Value)
    x.QuantityPerUnit = IIf(IsDBNull(CMD.Parameters("@xQuantityPerUnit").Value), "", CMD.Parameters("@xQuantityPerUnit").Value)
    x.UnitPrice = IIf(IsDBNull(CMD.Parameters("@xUnitPrice").Value), -1, CMD.Parameters("@xUnitPrice").Value)
    x.UnitsInStock = IIf(IsDBNull(CMD.Parameters("@xUnitsInStock").Value), -1, CMD.Parameters("@xUnitsInStock").Value)
    x.UnitsOnOrder = IIf(IsDBNull(CMD.Parameters("@xUnitsOnOrder").Value), -1, CMD.Parameters("@xUnitsOnOrder").Value)
    x.ReorderLevel = IIf(IsDBNull(CMD.Parameters("@xReorderLevel").Value), -1, CMD.Parameters("@xReorderLevel").Value)
    x.Discontinued = IIf(IsDBNull(CMD.Parameters("@xDiscontinued").Value), -1, CMD.Parameters("@xDiscontinued").Value)

    CN.Close()
    CN.Dispose()
    CMD.Dispose()
End Sub
End Class
```

5.4.4 Crear clase de negocio

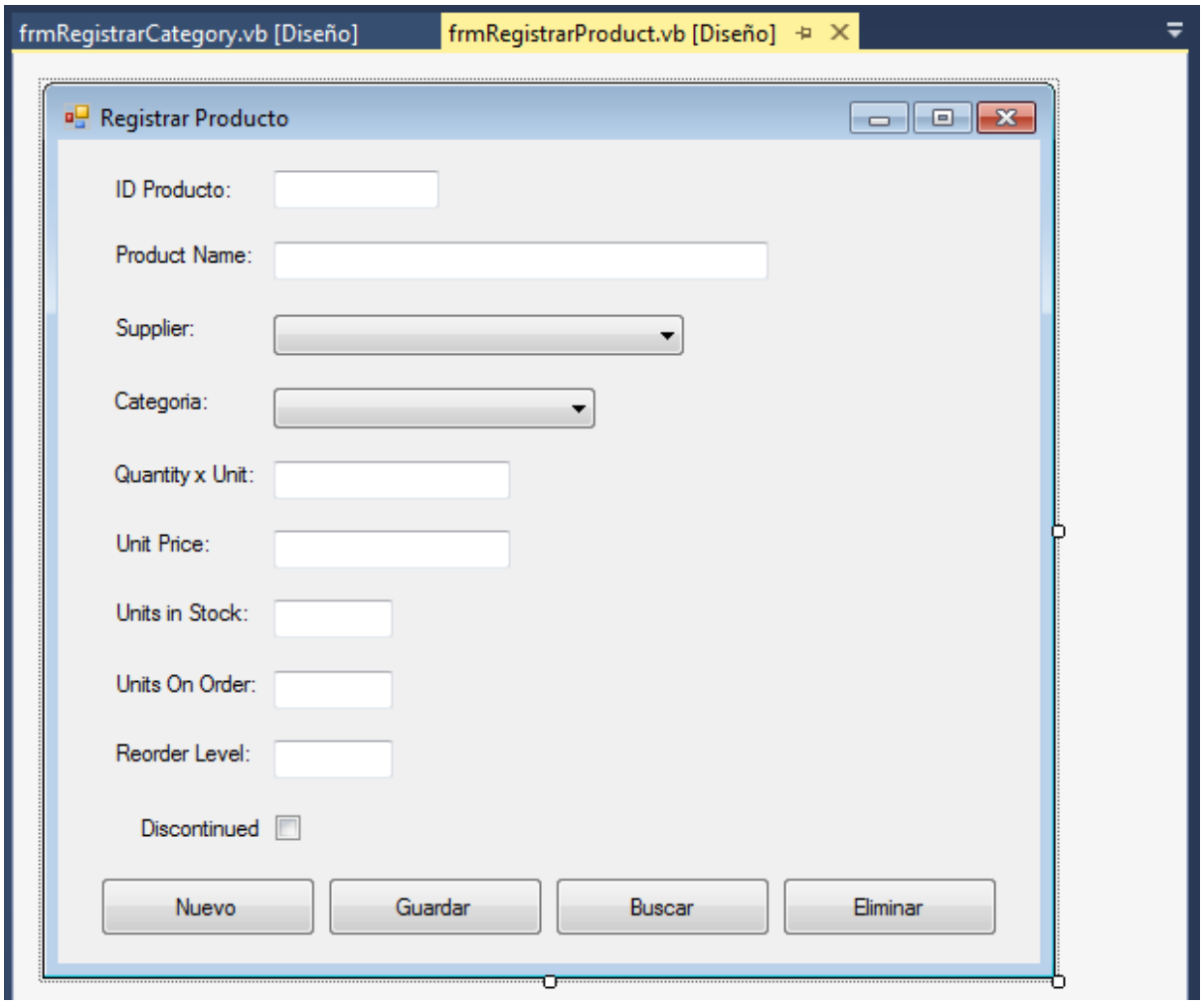
nProduct.vb | dProduct.vb | frmRegistrarCategory.vb [Diseño] | frmRegistrarProduct.vb [Diseño]

nProduct | Guardar

```vb
Imports pyLogicaEntidad
Imports pyLogicaDatos
0 referencias
Public Class nProduct
    0 referencias
    Public Sub Guardar(ByRef x As eProduct)
        Dim d As dProduct

        d = New dProduct()
        If x.ProductID <= 0 Then    ' es Producto nuevo
            d.Insertar(x)
        Else                        ' el Producto ya existe
            d.Actualizar(x)
        End If

        d = Nothing
    End Sub

    0 referencias
    Public Sub Eliminar(ByRef x As eProduct)
        Dim d As dProduct

        d = New dProduct()
        If x.ProductID > 0 Then     ' el Proveedor existe
            d.Eliminar(x)
        End If

        d = Nothing
    End Sub

    0 referencias
    Public Function Listar() As List(Of eProduct)
        Dim d As dProduct
        Dim rtn As List(Of eProduct)

        d = New dProduct()
        rtn = d.Listar()
        d = Nothing
        Return rtn
    End Function
```

```vb
    0 referencias
    Public Sub Buscar(ByRef x As eProduct)
        Dim d As dProduct

        d = New dProduct()
        d.Buscar(x)
        d = Nothing
    End Sub
End Class
```

5.4.5 Crear formulario de presentación



Los nombres de los controles de arriba hacia abajo son: txt_ProductID, txt_ProductName, cbo_SupplierID, cbo_ProductID, txt_QuantityxUnit, txt_UnitPrice, txt_UnitsInStock, txt_UnitsOnOrder, txt_ReorderLevel, check_Discontinued, cb_Nuevo, cb_Guardar, cb_Buscar, cb_Eliminar.

Se han modificado los valores predeterminados de las siguientes propiedades:

Control	Propiedad	Nuevo Valor
txt_ProductID	Enabled	False
cbo_SupplierID	DropDownStyle	DropDownList
cbo_CategoryID	DropDownStyle	DropDownList
check_Discontinued	Text	“Discontinued”

El código del formulario es como sigue:

```
dProduct.vb        frmRegistrarProduct.vb [Diseño]        frmRegistrarProduct.vb
cb_Nuevo                                                  Click

Imports pyLogicaEntidad
Imports pyLogicaNegocio
1 referencia
Public Class frmRegistrarProduct
    1 referencia
    Private Sub CargarCategorias()
        Dim n As nCategory

        n = New nCategory()
        cbo_CategoryID.DataSource = n.Listar()
        cbo_CategoryID.DisplayMember = "CategoryName"
        cbo_CategoryID.ValueMember = "CategoryID"
    End Sub

    1 referencia
    Private Sub CargarProveedores()
        Dim n As nSupplier

        n = New nSupplier()
        cbo_SupplierID.DataSource = n.Listar()
        cbo_SupplierID.DisplayMember = "SupplierName"
        cbo_SupplierID.ValueMember = "SupplierID"
    End Sub

    0 referencias
    Private Sub frmRegistrarProduct_Load(sender As Object, e As EventAr
        Me.CargarCategorias()
        Me.CargarProveedores()
    End Sub

    0 referencias
    Private Sub cb_Nuevo_Click(sender As Object, e As EventArgs) Handle
        Me.txt_ProductID.Clear()
        Me.txt_ProductName.Clear()
        Me.cbo_SupplierID.SelectedIndex = -1
        Me.cbo_CategoryID.SelectedIndex = -1
        Me.txt_QuantityxUnit.Clear()
        Me.txt_UnitPrice.Clear()
        Me.txt_UnitsInStock.Clear()
        Me.txt_UnitsOnOrder.Clear()
        Me.txt_ReorderLevel.Clear()
        Me.check_Discontinued.Checked = False
    End Sub
```

```
0 referencias
Private Sub cb_Guardar_Click(sender As Object, e As EventArgs) Handles cb_Guardar.Click
    Dim n As nProduct
    Dim x As eProduct

    Try
        n = New nProduct()
        x = New eProduct()
        x.ProductName = txt_ProductName.Text
        x.SupplierID = cbo_SupplierID.SelectedValue
        x.CategoryID = cbo_CategoryID.SelectedValue
        x.QuantityPerUnit = txt_QuantityxUnit.Text
        x.UnitPrice = CDbl(txt_UnitPrice.Text)
        x.UnitsInStock = CInt(txt_UnitsInStock.Text)
        x.UnitsOnOrder = txt_UnitsOnOrder.Text
        x.ReorderLevel = check_Discontinued.Checked

        If txt_ProductID.Text <> "" Then
            x.ProductID = CInt(txt_ProductID.Text)
        Else
            x.ProductID = -1
        End If
        n.Guardar(x)
        MsgBox("Se guardó correctamente", MsgBoxStyle.Information, Me.Text)
        txt_ProductID.Text = x.ProductID.ToString()

    Catch ex As Exception
        MsgBox("Error: " & ex.Message, MsgBoxStyle.Critical, Me.Text)

    Finally
        n = Nothing
        x = Nothing
    End Try
End Sub
```

```
0 referencias
Private Sub cb_Buscar_Click(sender As Object, e As EventArgs) Handles cb_Buscar.Click
    Dim n As nProduct
    Dim x As eProduct
    Try
        If txt_ProductName.Text <> "" Then
            n = New nProduct
            x = New eProduct
            x.ProductName = txt_ProductName.Text.Trim()
            n.Buscar(x)
            If x.SupplierID >= 1 Then
                MsgBox("Categoría encontrada", MsgBoxStyle.Information, Me.Text)
                txt_ProductName.Text = x.ProductName.ToString()
                cbo_SupplierID.SelectedValue = x.SupplierID
                cbo_CategoryID.SelectedValue = x.CategoryID
                txt_QuantityxUnit.Text = x.QuantityPerUnit
                txt_UnitPrice.Text = x.UnitPrice
                txt_UnitsInStock.Text = x.UnitsInStock
                txt_UnitsOnOrder.Text = x.UnitsOnOrder
                txt_ReorderLevel.Text = x.ReorderLevel
                check_Discontinued.Checked = x.Discontinued
            Else
                MsgBox("Producto no encontrado!", MsgBoxStyle.Information, Me.Text)
            End If
        End If

    Catch ex As Exception
        MsgBox("Error: " & ex.Message, MsgBoxStyle.Critical, Me.Text)

    Finally
        n = Nothing
        x = Nothing
    End Try
End Sub
```

```
    0 referencias
    Private Sub cb_Eliminar_Click(sender As Object, e As EventArgs) Handles cb_Eliminar.Click
        Dim n As nProduct
        Dim x As eProduct

        Try
            If txt_ProductID.Text <> "" Then
                n = New nProduct
                x = New eProduct
                x.CategoryID = CInt(txt_ProductID.Text)
                n.Eliminar(x)
            End If
            MsgBox("Se eliminó correctamente", MsgBoxStyle.Information, Me.Text)

        Catch ex As Exception
            MsgBox("Error: " & ex.Message, MsgBoxStyle.Critical, Me.Text)

        Finally
            n = Nothing
            x = Nothing
        End Try
    End Sub
End Class
```

5.5 Reporte de Empleados

5.5.1 Crear procedimiento almacenado para el reporte

SQLQuery2.sql - UN...orthwind (sa (56)) | SQLQuery1.sql - UN...orthwind (sa (53))*

```sql
CREATE PROCEDURE DBO.Employees_ReporteEmpleados
AS
BEGIN

    SELECT  EmployeeID,
            LastName + ', ' + FirstName as Nombres,
            Title,
            DATEDIFF(year, hiredate, getdate()) as Antiguedad,
            "Address"
    FROM Employees
    ORDER by Nombres;

END
```

100 %

Resultados

```
Comandos completados correctamente.
```

100 %

5.5.2 Crear Clase Entidad para el Reporte

eRep_ReporteEmpleados.vb | dEmployee.vb | dCategory.vb | nCategory.vb

eRep_ReporteEmpleados | (Declaraciones)

```vb
5 referencias
Public Class eRep_ReporteEmpleados
    1 referencia
    Public Property EmployeeID As Integer
    1 referencia
    Public Property Nombres As String
    1 referencia
    Public Property Title As String
    1 referencia
    Public Property Antiguedad As Integer
    1 referencia
    Public Property Addreess As String
End Class
```

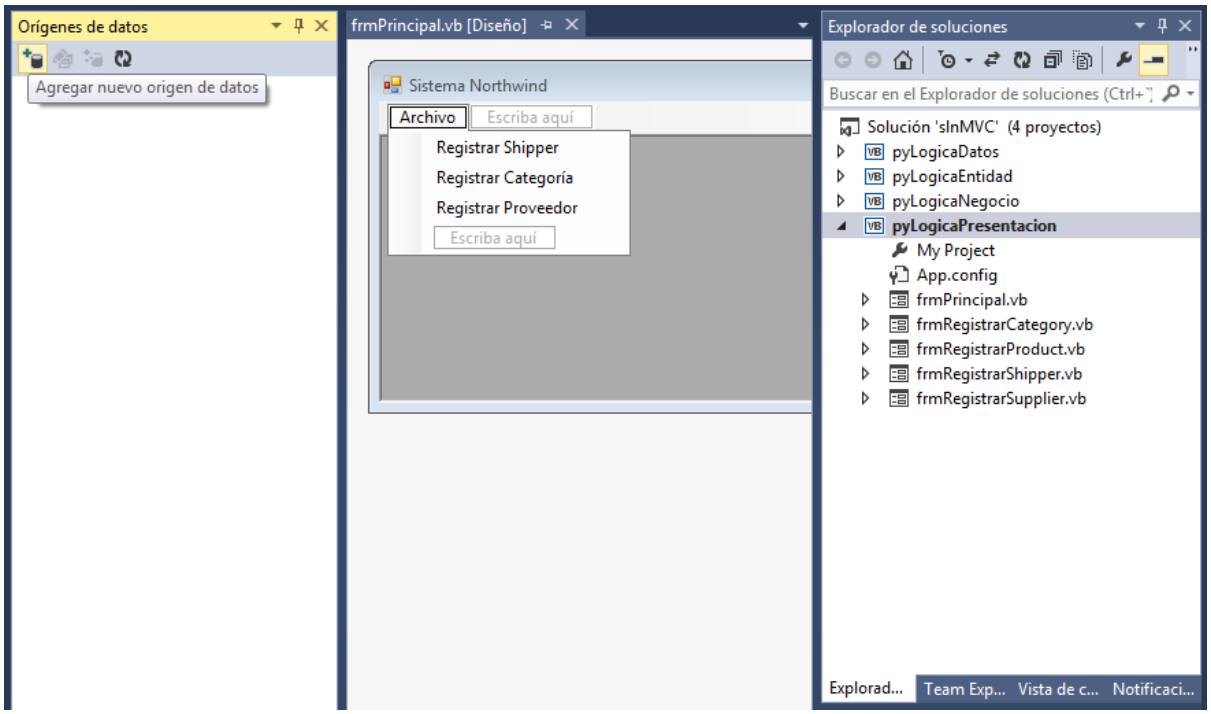
5.5.3 Agregar función a la clase “dEmployees”

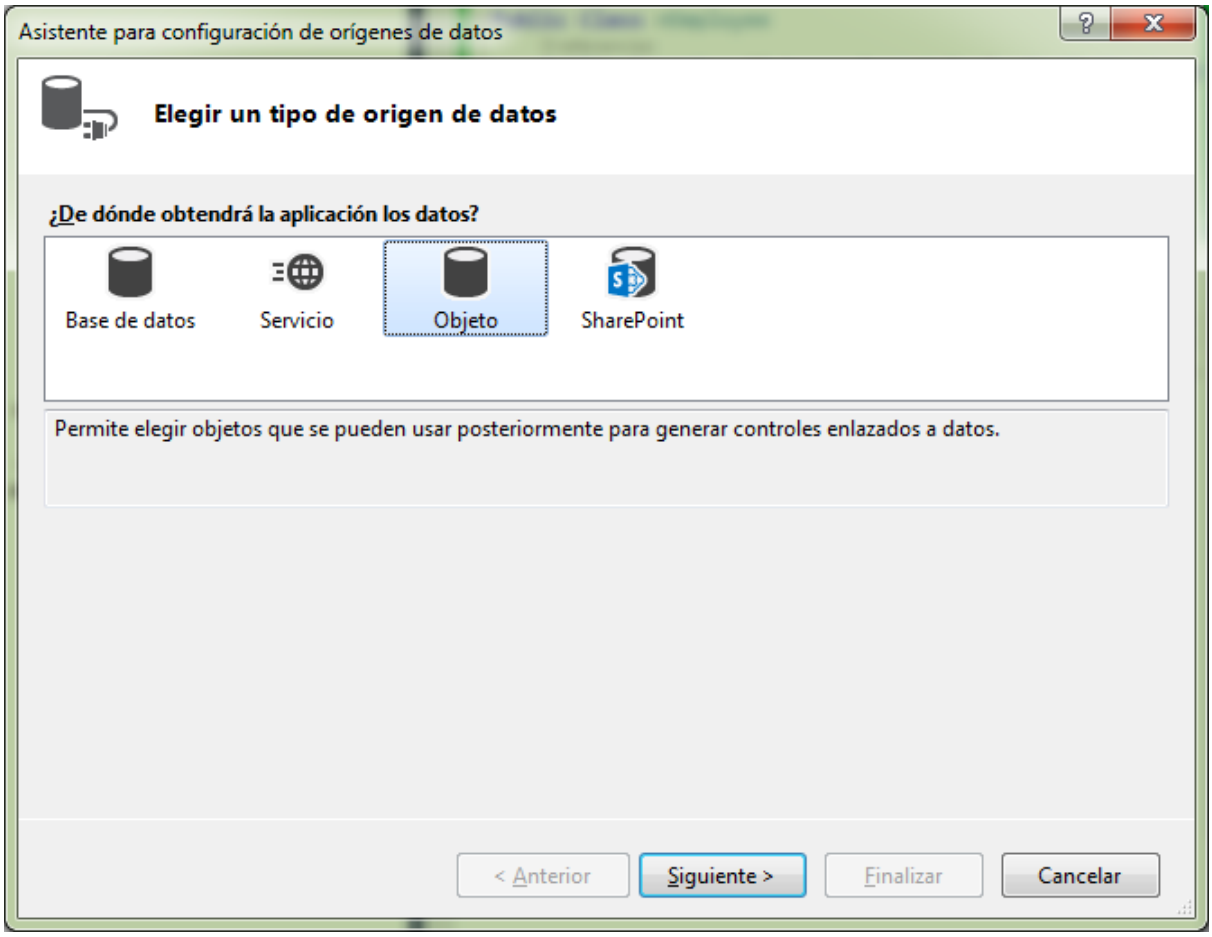
```
eRep_ReporteEmpleados.vb   dEmployee.vb   dCategory.vb   nCategory.vb
dEmployee                  Elaborar_ReporteEmpleados

Imports pyLogicaEntidad
Imports System.Data.SqlClient
Imports System.Data.SqlTypes
0 referencias
Public Class dEmployee
    0 referencias
    Public Function Elaborar_ReporteEmpleados() As List(Of eRep_ReporteEmpleados)
        Dim CN As SqlConnection
        Dim CMD As SqlCommand
        Dim rtn As List(Of eRep_ReporteEmpleados)
        Dim Reg As SqlDataReader
        Dim x As eRep_ReporteEmpleados

        CN = New SqlClient.SqlConnection(CadenaConexionDB())
        CN.Open()
        CMD = New SqlCommand                'utilizamos una variable tipo comando para acce
        CMD.CommandType = CommandType.StoredProcedure  'usaremos un Stored Procedure
        CMD.Connection = CN                 'especificamos la conexion
        CMD.CommandText = "Employees_ReporteEmpleados"
        Reg = CMD.ExecuteReader(CommandBehavior.CloseConnection)
        rtn = New List(Of eRep_ReporteEmpleados)
        While (Reg.Read())
            x = New eRep_ReporteEmpleados()
            x.EmployeeID = Reg.GetInt32(0)
            x.Nombres = Reg.GetString(1)
            x.Title = Reg.GetString(2)
            x.Antiguedad = Reg.GetInt32(3)
            x.Addreess = Reg.GetString(4)
            rtn.Add(x)
        End While

        Reg.Close()
        CN.Dispose()
        CMD.Dispose()
        Return (rtn)
    End Function
End Class
```

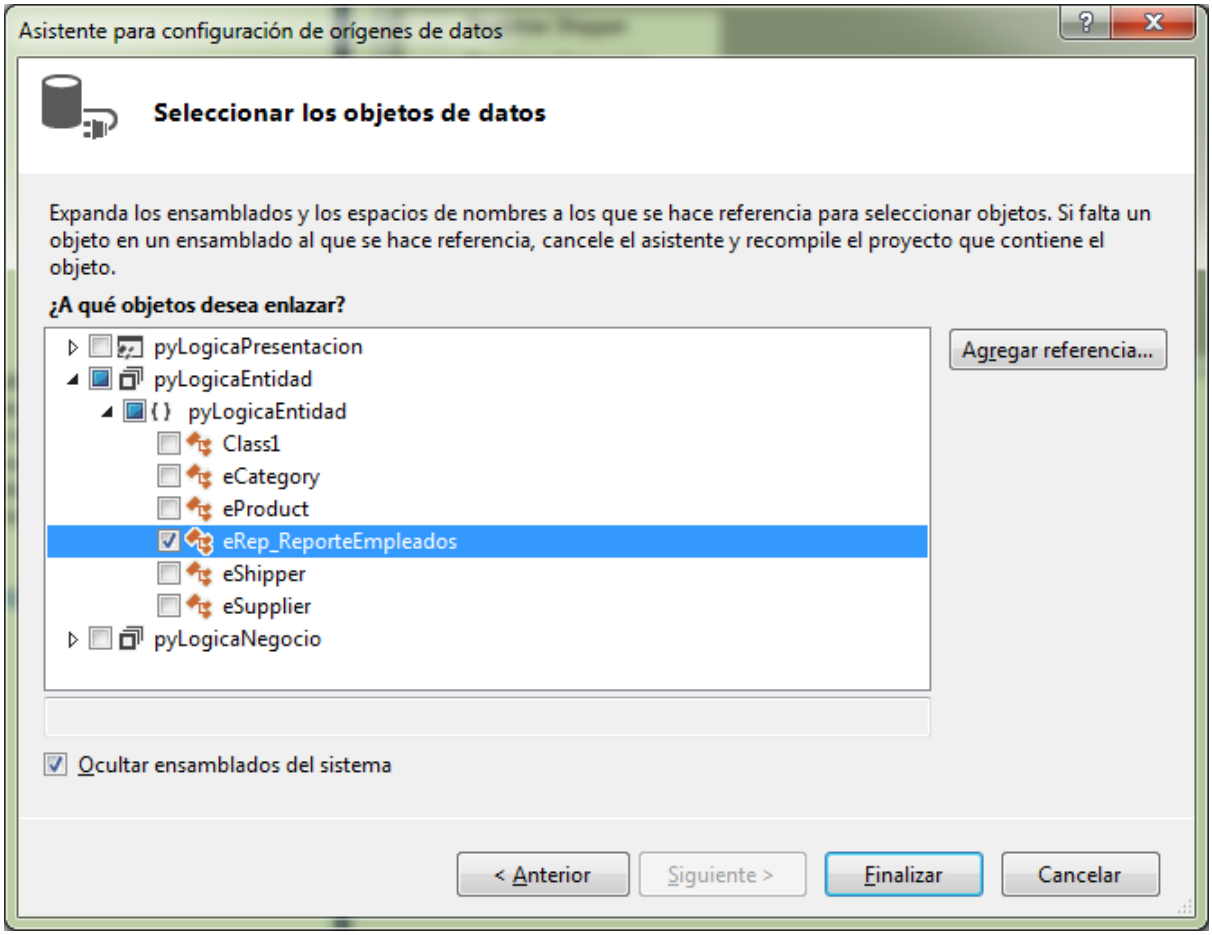
5.5.4 Agregar función a la clase “nEmployee”

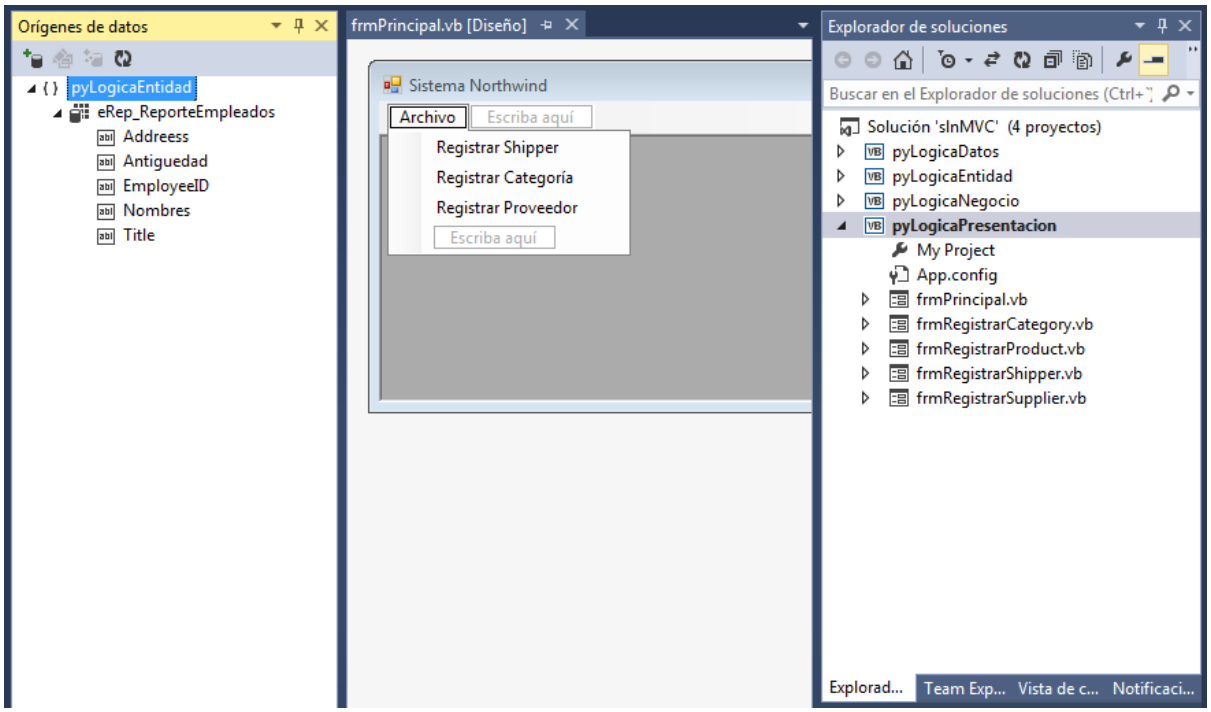
```
nEmployee.vb   dEmployee.vb
nEmployee                  (Declaraciones)

Imports pyLogicaEntidad
Imports pyLogicaDatos
0 referencias
Public Class nEmployee
    0 referencias
    Public Function Elaborar_ReporteEmpleados() As List(Of eRep_ReporteEmpleados)
        Dim d As dEmployee
        Dim rtn As List(Of eRep_ReporteEmpleados)

        d = New dEmployee()
        rtn = d.Elaborar_ReporteEmpleados()
        d = Nothing
        Return rtn
    End Function

End Class
```

5.5.5 Crear Origen de datos para el reporte

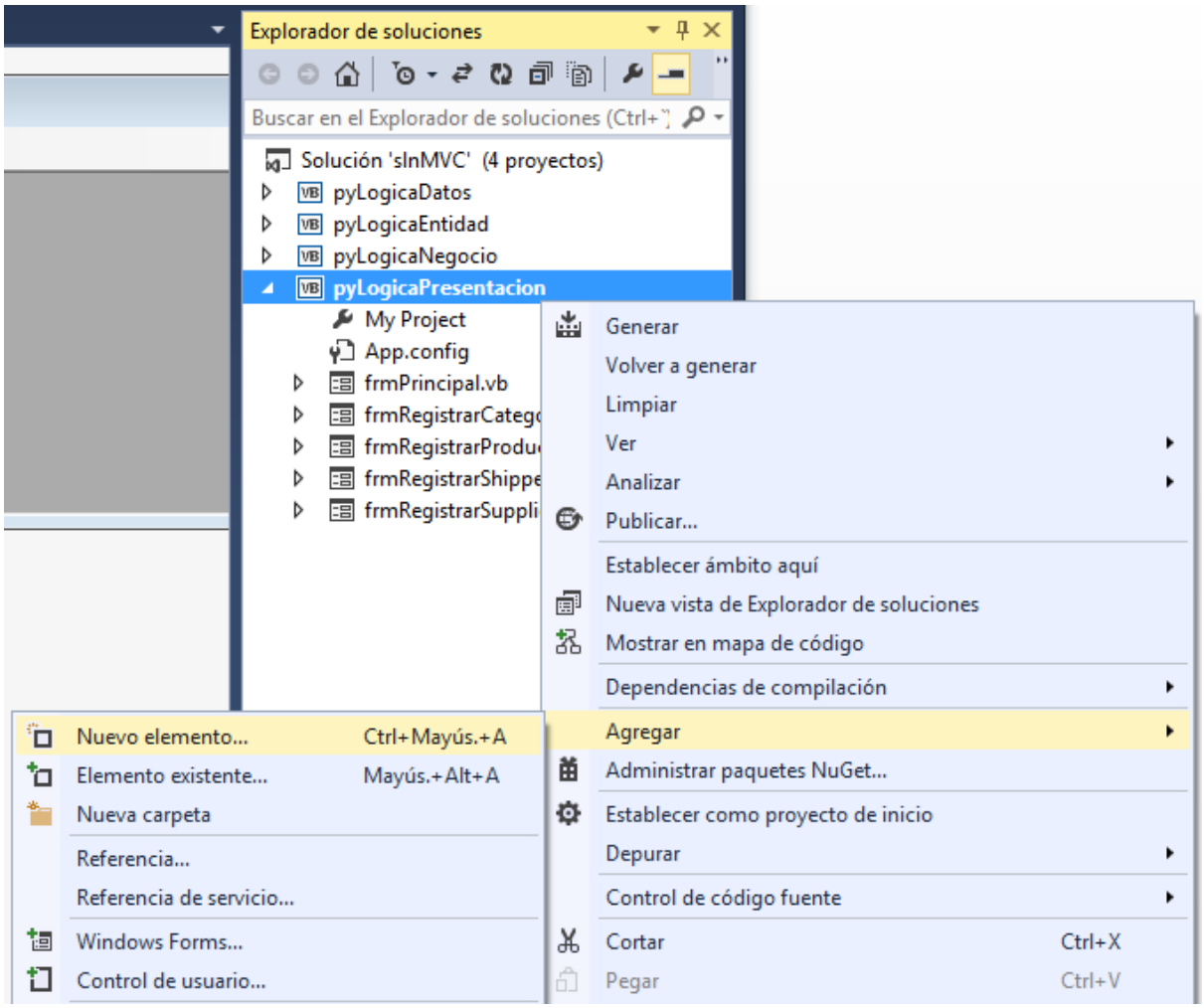


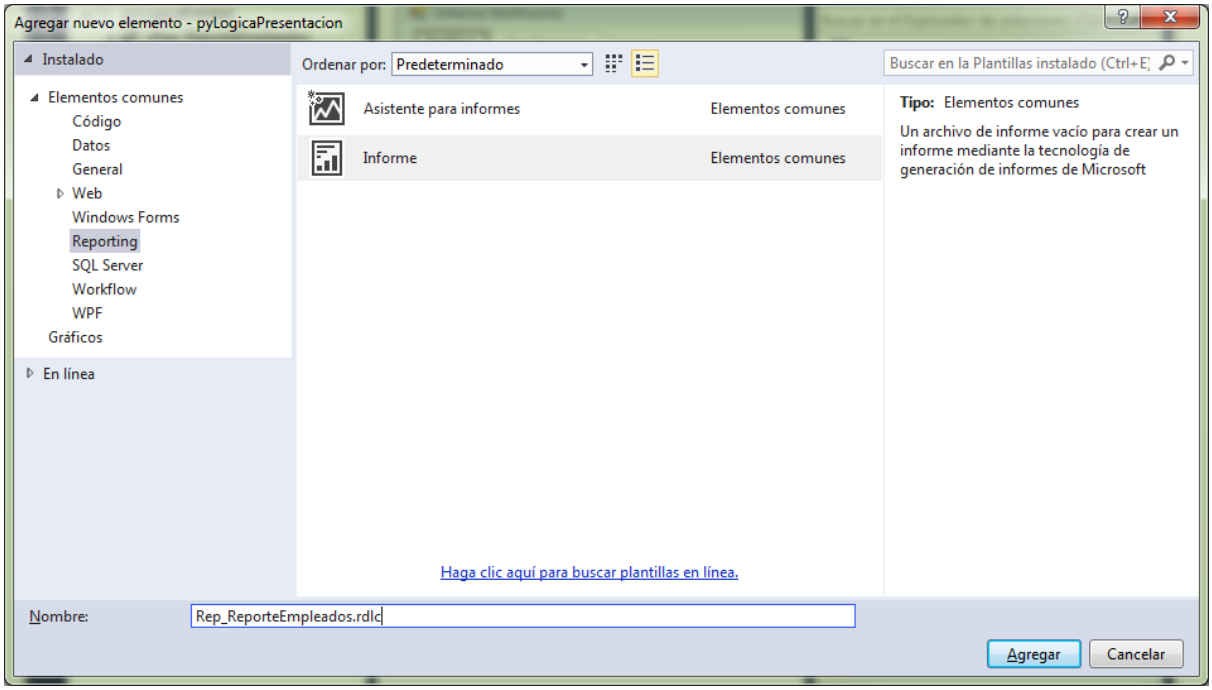


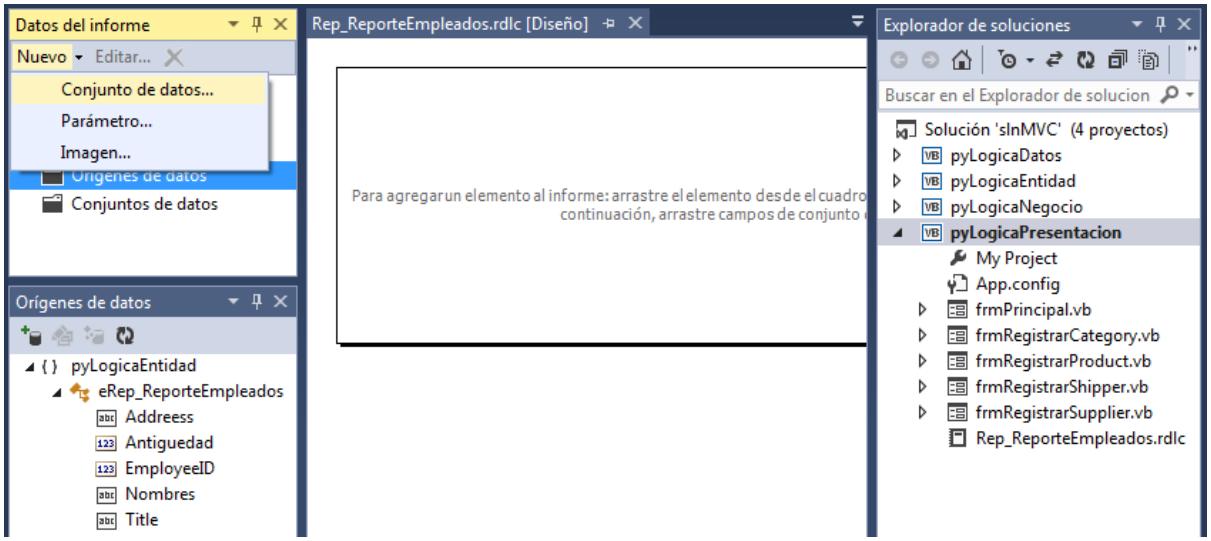

Asistente para configuración de orígenes de datos
Seleccionar los objetos de datos
Expanda los ensamblados y los espacios de nombres a los que se hace referencia para seleccionar objetos. Si falta un objeto en un ensamblado al que se hace referencia, cancele el asistente y recompile el proyecto que contiene el objeto.
¿A qué objetos desea enlazar?
pyLogicaPresentacion
pyLogicaEntidad
pyLogicaEntidad
Class1
eCategory
eProduct
eRep_ReporteEmpleados
eShipper
eSupplier
pyLogicaNegocio
Agregar referencia...
Ocultar ensamblados del sistema
< Anterior
Siguiente >
Finalizar
Cancelar

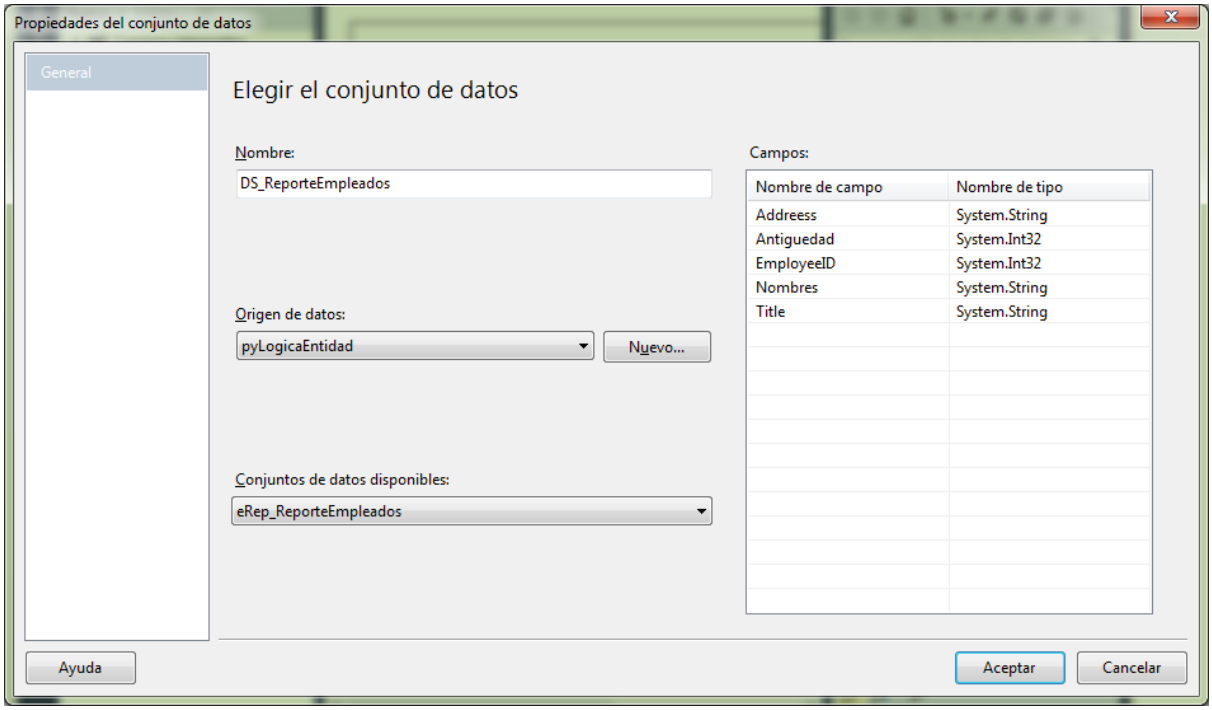

Orígenes de datos
pyLogicaEntidad
eRep_ReporteEmpleados
Addreess
Antiguedad
EmployeeID
Nombres
Title
frmPrincipal.vb [Diseño]
Sistema Northwind
Archivo
Escriba aquí
Registrar Shipper
Registrar Categoría
Registrar Proveedor
Escriba aquí
Explorador de soluciones
Buscar en el Explorador de soluciones (Ctrl+;)
Solución 'slnMVC' (4 proyectos)
pyLogicaDatos
pyLogicaEntidad
pyLogicaNegocio
pyLogicaPresentacion
My Project
App.config
frmPrincipal.vb
frmRegistrarCategory.vb
frmRegistrarProduct.vb
frmRegistrarShipper.vb
frmRegistrarSupplier.vb
Explorad...
Team Exp...
Vista de c...
Notificaci...

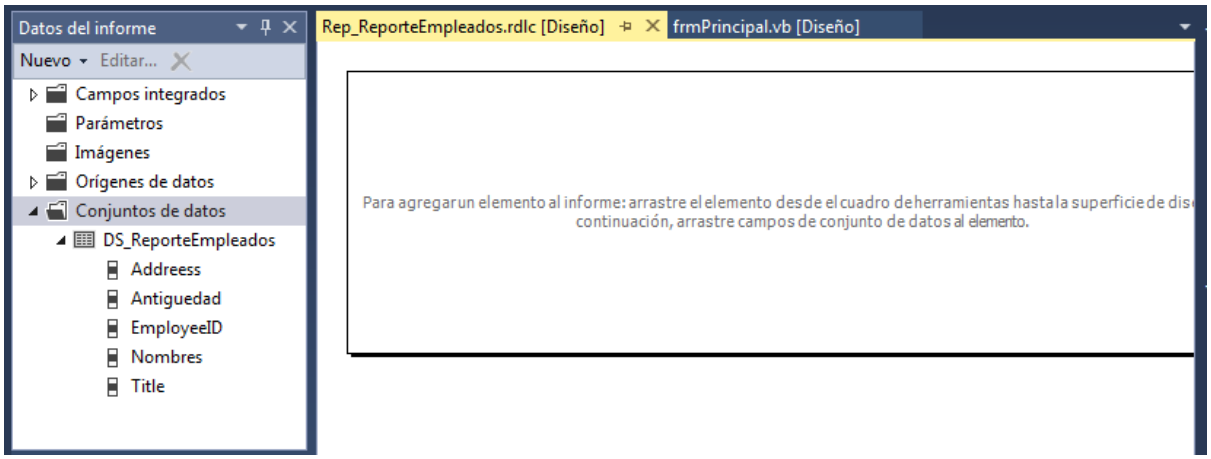
5.5.6 Crear Informe

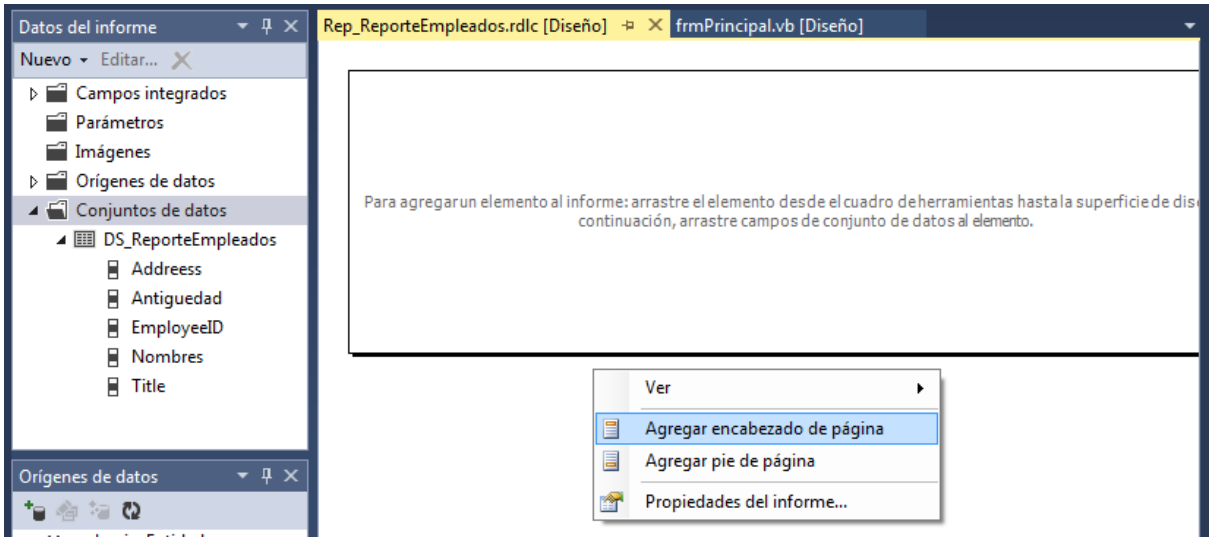


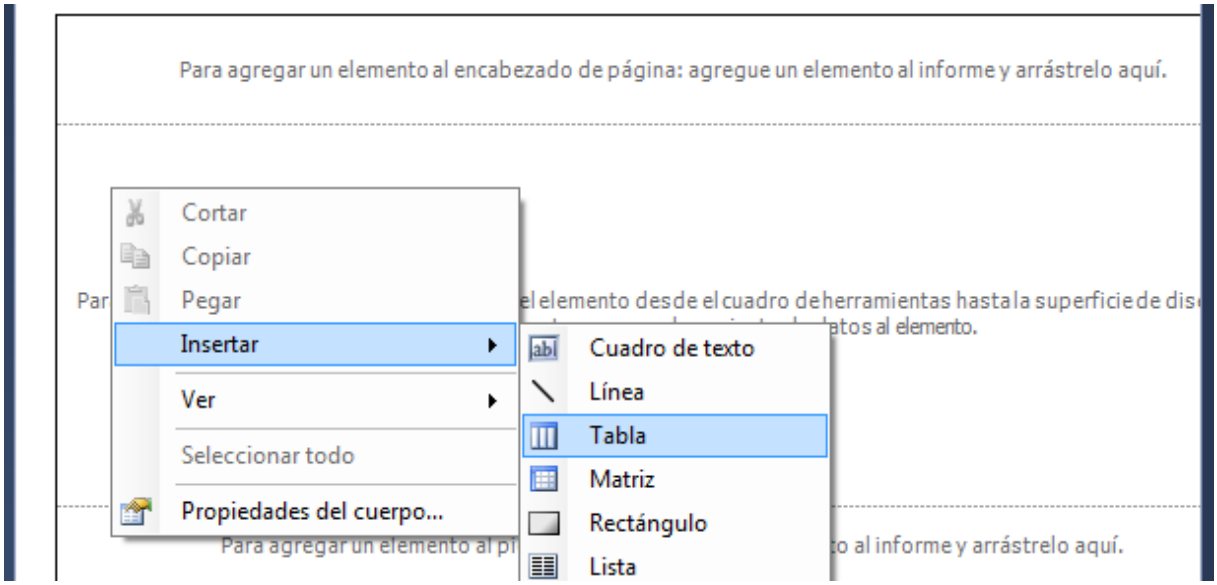


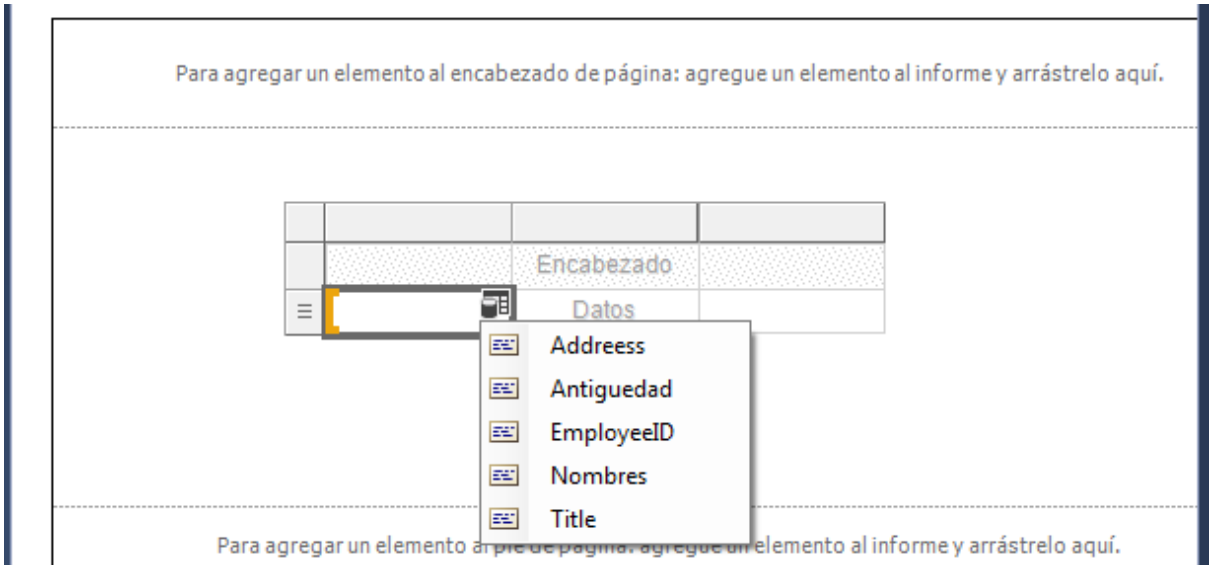

Datos del informe
Nuevo
Editar...
Conjunto de datos...
Parámetro...
Imagen...
Orígenes de datos
Conjuntos de datos
Rep_ReporteEmpleados.rdlc [Diseño]
Para agregar un elemento al informe: arrastre el elemento desde el cuadro
continuación, arrastre campos de conjunto
Explorador de soluciones
Buscar en el Explorador de solucion
Solución 'sInMVC' (4 proyectos)
pyLogicaDatos
pyLogicaEntidad
pyLogicaNegocio
pyLogicaPresentacion
My Project
App.config
frmPrincipal.vb
frmRegistrarCategory.vb
frmRegistrarProduct.vb
frmRegistrarShipper.vb
frmRegistrarSupplier.vb
Rep_ReporteEmpleados.rdlc
Orígenes de datos
pyLogicaEntidad
eRep_ReporteEmpleados
Addreess
Antiguedad
EmployeeID
Nombres
Title

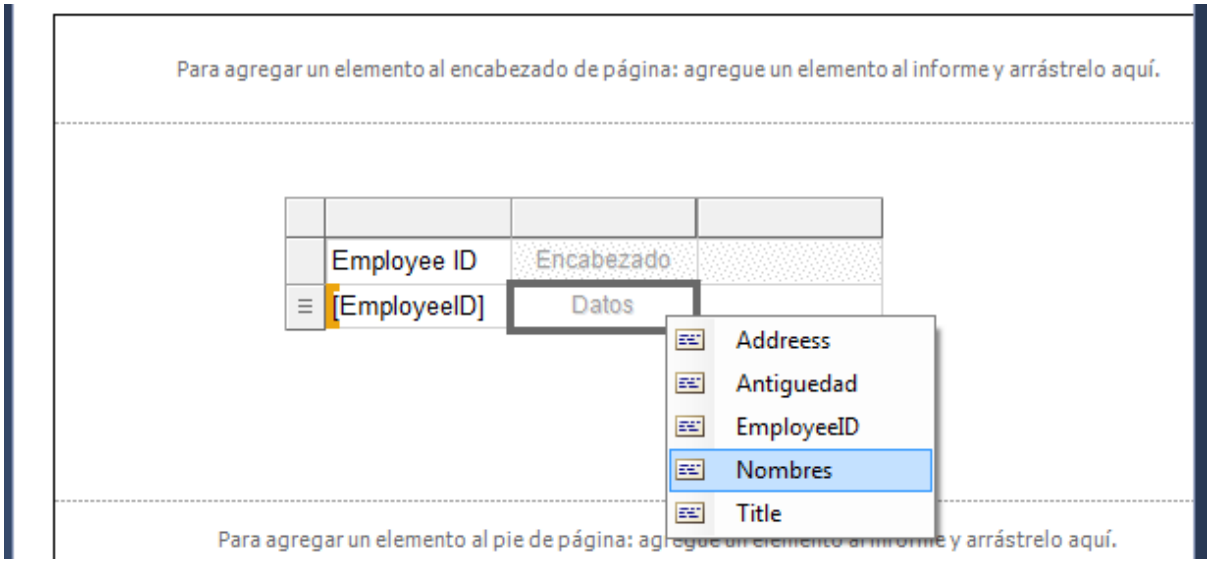

Propiedades del conjunto de datos
General
Elegir el conjunto de datos
Nombre:
DS_ReporteEmpleados
Origen de datos:
pyLogicaEntidad
Nuevo...
Conjuntos de datos disponibles:
eRep_ReporteEmpleados
Campos:
Nombre de campo
Nombre de tipo
Addreess
System.String
Antiguedad
System.Int32
EmployeeID
System.Int32
Nombres
System.String
Title
System.String
Ayuda
Aceptar
Cancelar


Datos del informe
Nuevo
Editar...
Campos integrados
Parámetros
Imágenes
Orígenes de datos
Conjuntos de datos
DS_ReporteEmpleados
Addreess
Antiguedad
EmployeeID
Nombres
Title
Rep_ReporteEmpleados.rdlc [Diseño]
frmPrincipal.vb [Diseño]
Para agregar un elemento al informe: arrastre el elemento desde el cuadro de herramientas hasta la superficie de dis
continuación, arrastre campos de conjunto de datos al elemento.

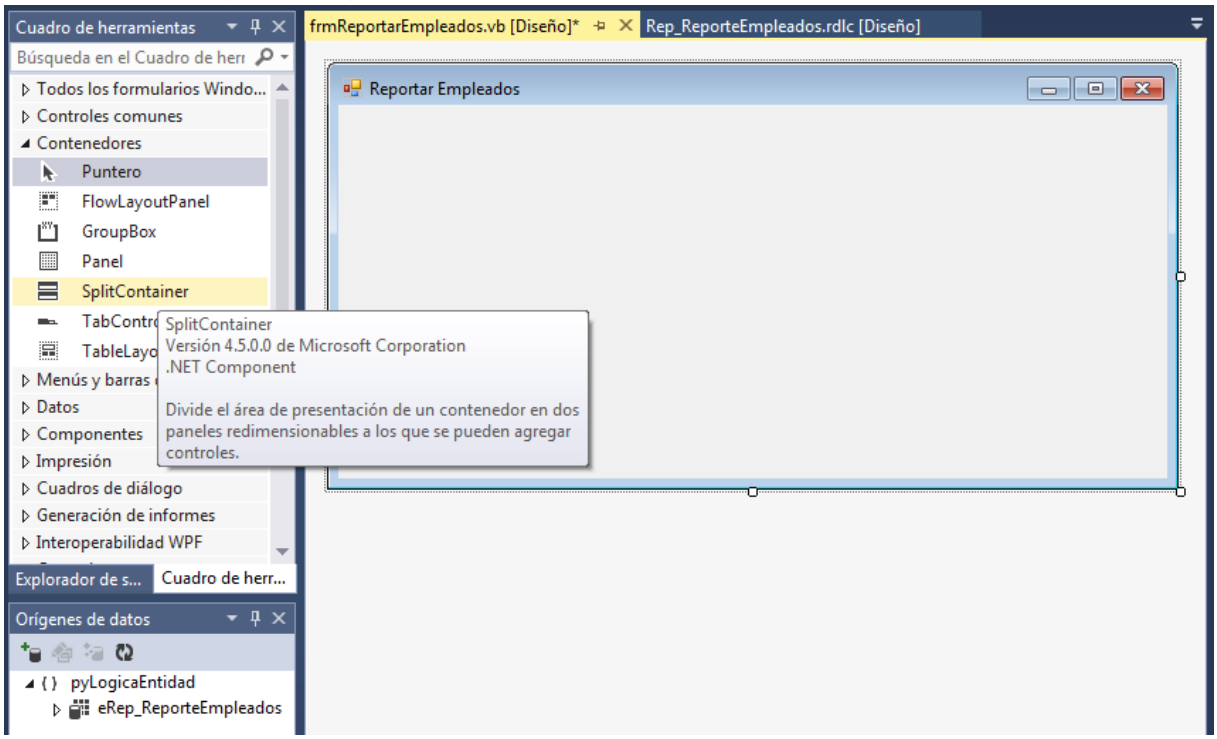

Datos del informe
Nuevo
Editar...
Campos integrados
Parámetros
Imágenes
Orígenes de datos
Conjuntos de datos
DS_ReporteEmpleados
Addreess
Antiguedad
EmployeeID
Nombres
Title
Orígenes de datos
Rep_ReporteEmpleados.rdlc [Diseño]
frmPrincipal.vb [Diseño]
Para agregar un elemento al informe: arrastre el elemento desde el cuadro de herramientas hasta la superficie de dis
continuación, arrastre campos de conjunto de datos al elemento.
Ver
Agregar encabezado de página
Agregar pie de página
Propiedades del informe...

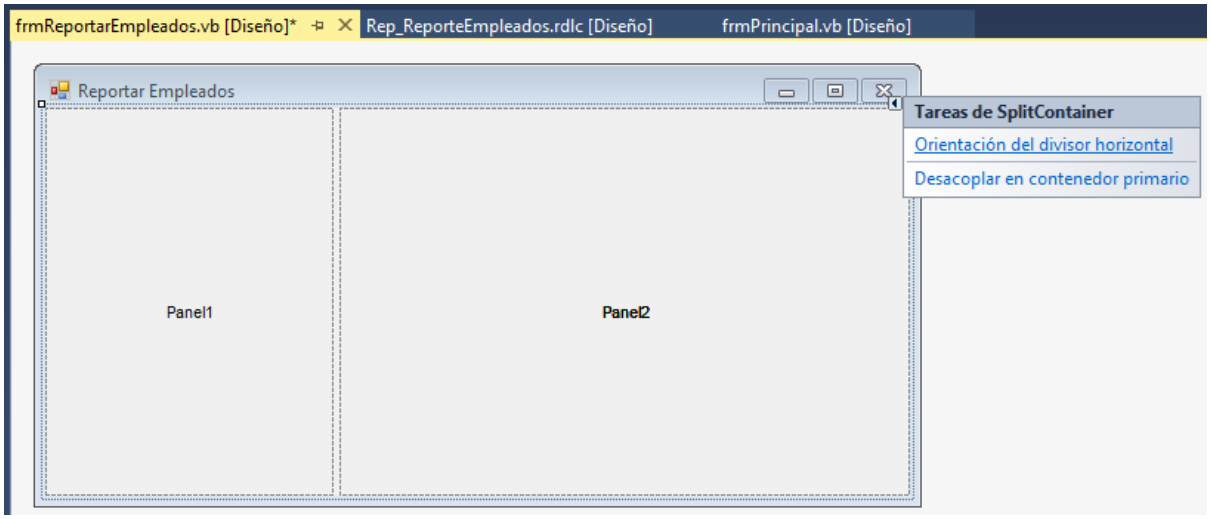

Para agregar un elemento al encabezado de página: agregue un elemento al informe y arrástrelo aquí.
Cortar
Copiar
Pegar
Insertar
Ver
Seleccionar todo
Propiedades del cuerpo...
Cuadro de texto
Línea
Tabla
Matriz
Rectángulo
Lista

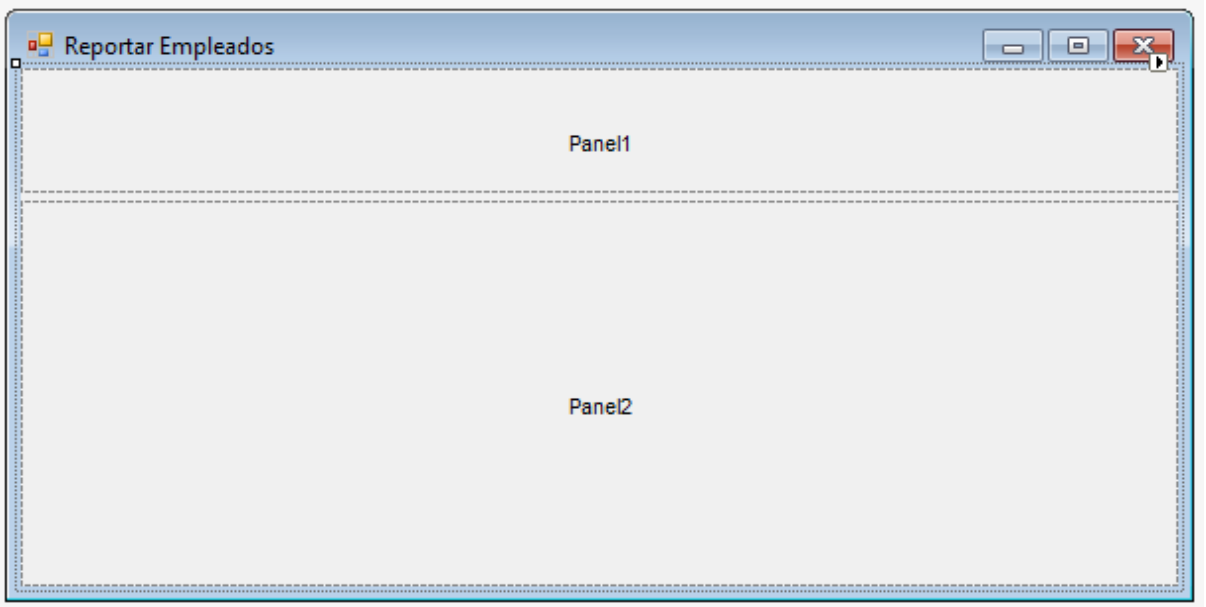

Para agregar un elemento al encabezado de página: agregue un elemento al informe y arrástrelo aquí.
Encabezado
Datos
Addreess
Antiguedad
EmployeeID
Nombres
Title
Para agregar un elemento al informe y arrástrelo aquí.

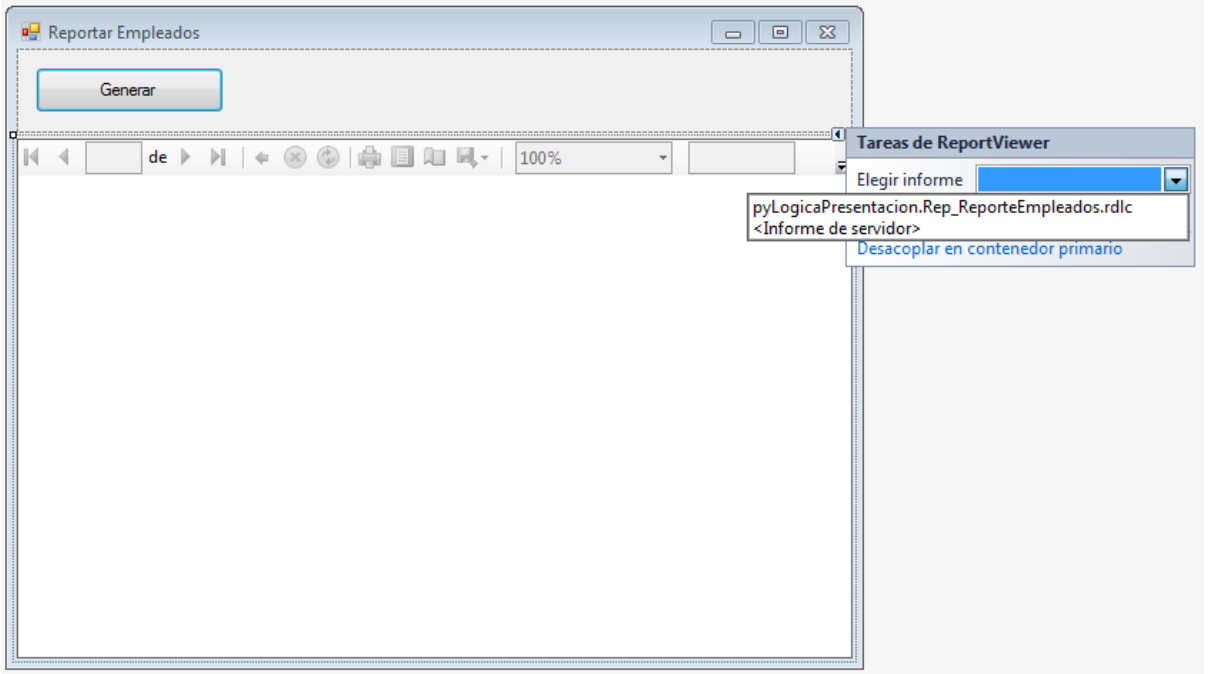


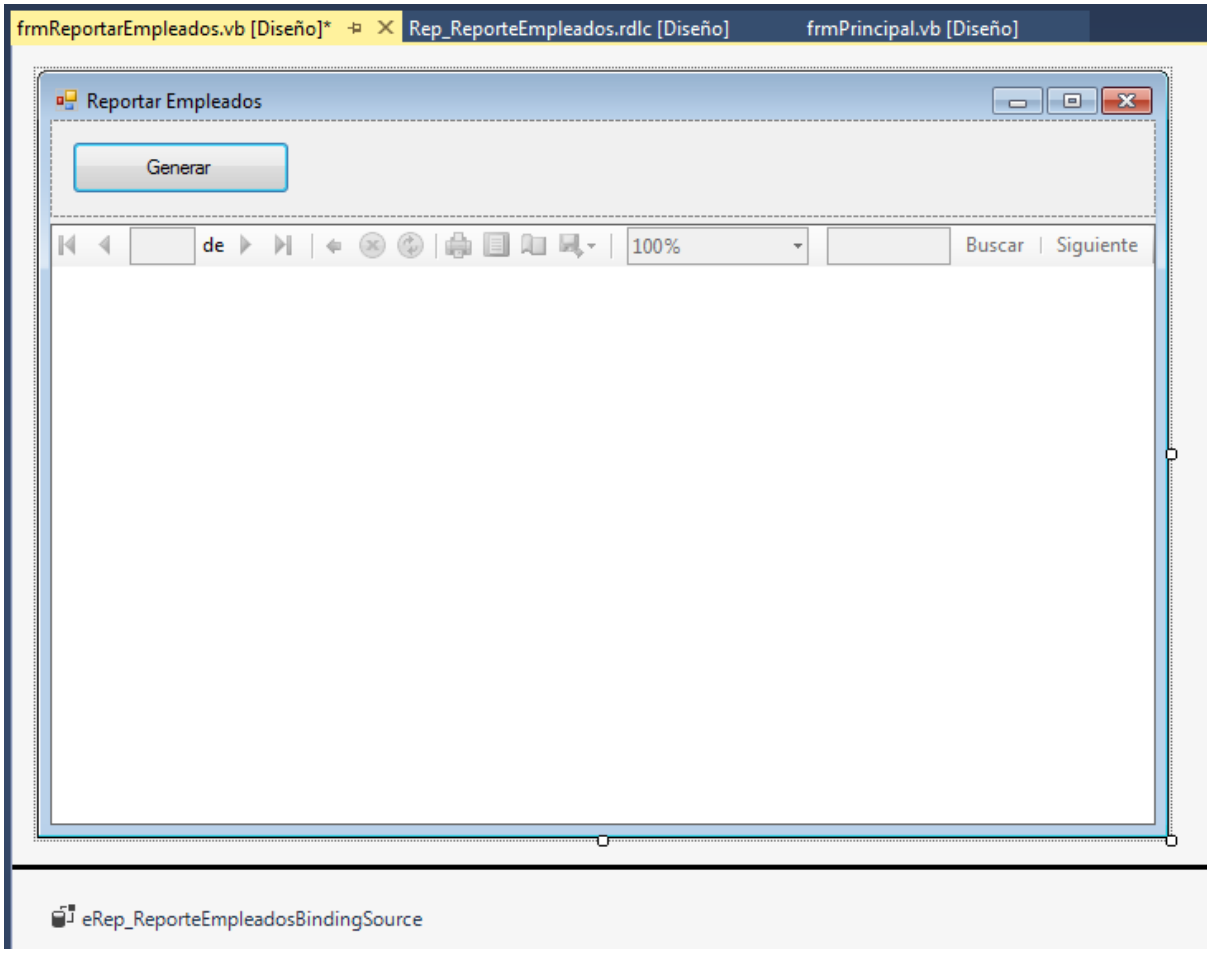
5.5.7 Crear Formulario para el reporte



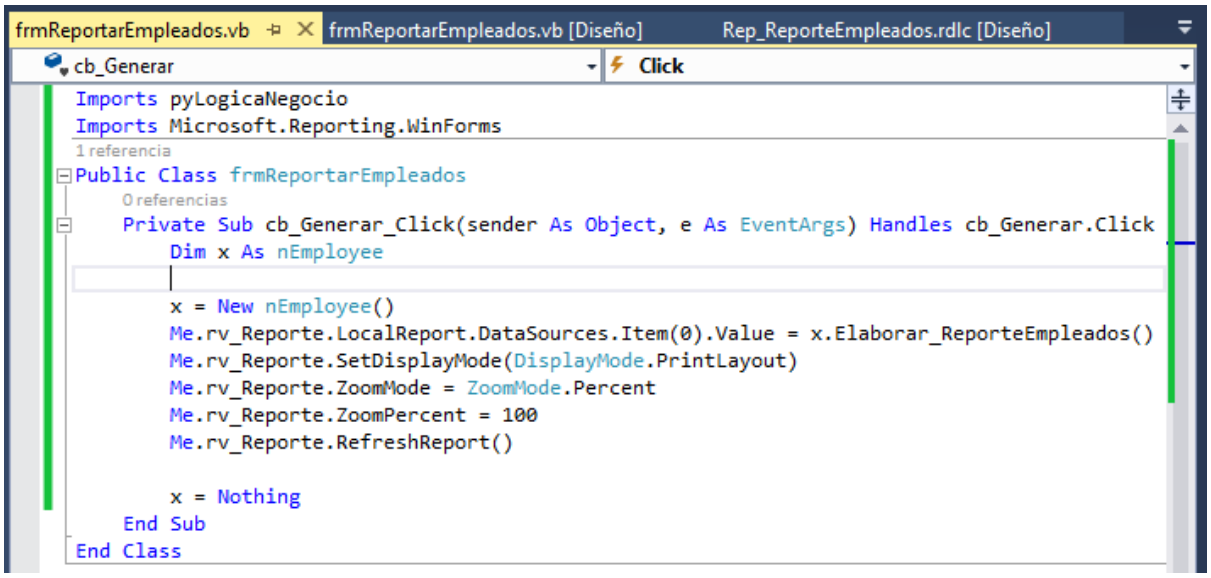

frmReportarEmpleados.vb [Diseño]*
Rep_ReporteEmpleados.rdlc [Diseño]
frmPrincipal.vb [Diseño]
Reportar Empleados
Tareas de SplitContainer
Orientación del divisor horizontal
Desacoplar en contenedor primario
Panel1
Panel2


Reportar Empleados
Panel1
Panel2


Reportar Empleados
Generar
de
100%
Tareas de ReportViewer
Elegir informe
pyLogicaPresentacion.Rep_ReporteEmpleados.rdlc
<Informe de servidor>
Desacoplar en contenedor primario

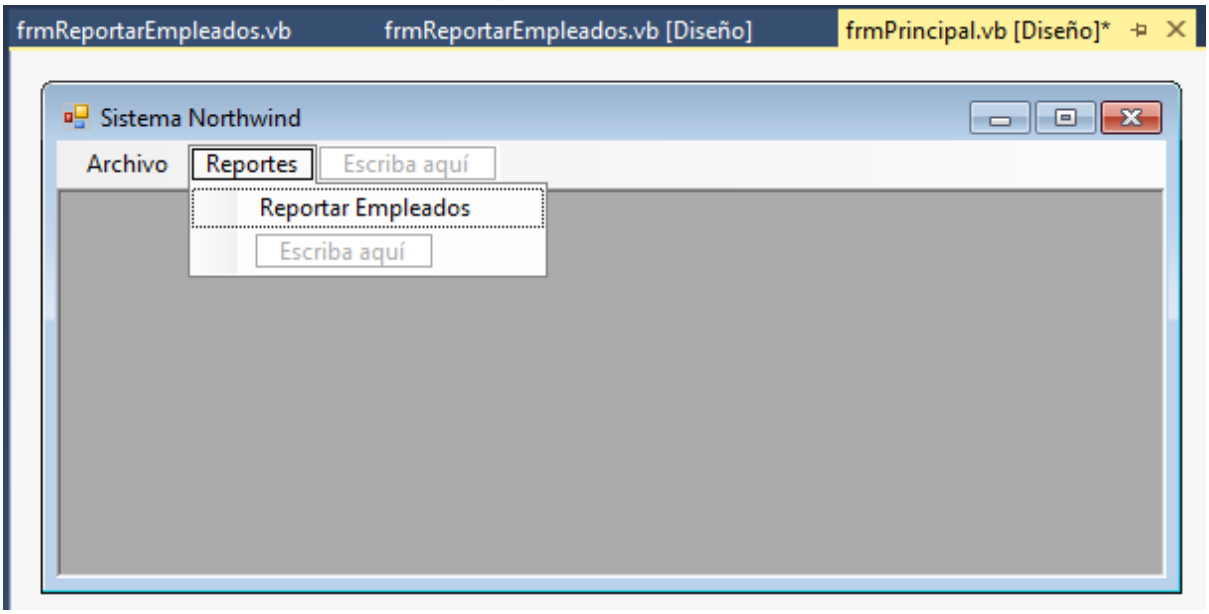


Código del Formulario:

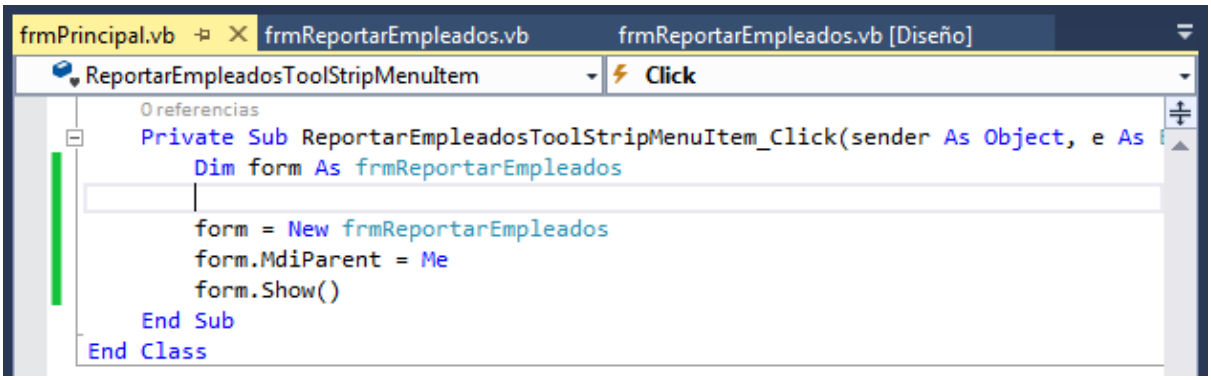


```
Imports pyLogicaNegocio
Imports Microsoft.Reporting.WinForms
1 referencia
Public Class frmReportarEmpleados
    0 referencias
    Private Sub cb_Generar_Click(sender As Object, e As EventArgs) Handles cb_Generar.Click
        Dim x As nEmployee

        x = New nEmployee()
        Me.rv_Reporte.LocalReport.DataSources.Item(0).Value = x.Elaborar_ReporteEmpleados()
        Me.rv_Reporte.SetDisplayMode(DisplayMode.PrintLayout)
        Me.rv_Reporte.ZoomMode = ZoomMode.Percent
        Me.rv_Reporte.ZoomPercent = 100
        Me.rv_Reporte.RefreshReport()

        x = Nothing
    End Sub
End Class
```

5.5.8 Modificar Menú del Formulario Principal



Código:



5.6 Reporte de Ventas Anuales por Categoría de Producto

5.6.1 Crear procedimiento almacenado para el reporte

5.6.2 Crear Clase Entidad para el Reporte

5.6.3 Agregar función a la clase “dCategory”

5.6.4 Agregar función a la clase “nCategory”

5.6.5 Crear Origen de datos para el reporte

5.6.6 Crear Informe

5.6.7 Crear Formulario para el reporte

5.6.8 Modificar Menú del Formulario Principal