



Todo ADO.NET Teoria

Lenguajes De Última Generación (Universidad Abierta Interamericana)

Todo ADO.NET

Compilación de
ejemplos prácticos

Material para Lenguajes de
Última Generación

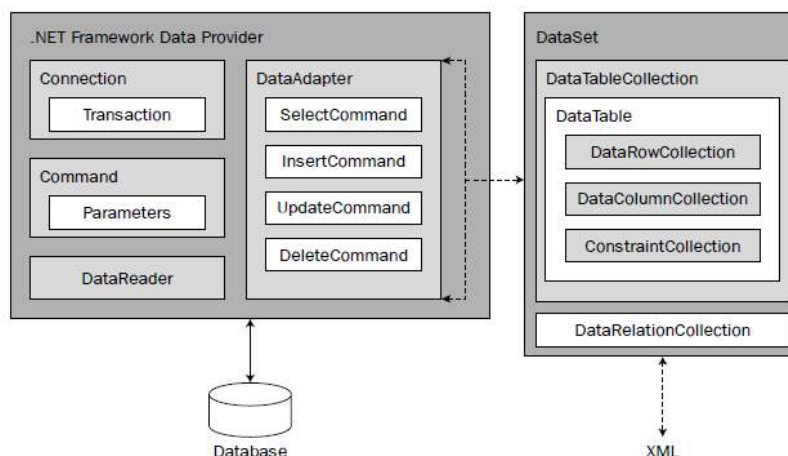
Contenido

.....	2
Espacios de nombres y clases en ADO .NET	2
Uso del command	5
Las clases DataReader	7
Conjuntos de datos y enlace (Data Binding)	9
Las clases DataAdapter	11
Navegación y edición de registros en modo desconectado	12
Data Binding. Enlace de datos a controles	13
Tipos de Data Binding	13
Elementos integrantes en un proceso de Data Binding	13
Ejercicios Prácticos: Enunciados	15
Uso de Conecction, Command a modo texto, Datareader y Grilla	17
Use Connection, Command con parámetros, DataReader y Grilla	21
Uso Connection, Command con parámetros, Transacción, DataReader y Grilla	25
Uso de Connection, DataAdapter, CommandBuilder, DataTable con clave primaria, Grilla...32	
Uso de Connection, Varios DataAdapter relacionados, Procedimientos Almacenados, CommandBuilder, DataTable, Grilla	37
Uso de Connection, Varios DataAdapter relacionados, Procedimientos Almacenados, CommandBuilder, DataTable, Grilla , XML	48
Modelo 3 Capas	54
CAPA UI	54
Clase ClienteVista	54
Bibliografía :	65

ADO.NET

ADO.NET Components

To better support the disconnected model, the ADO.NET components separate data access from data manipulation. This is accomplished via two main components, the *DataSet* and the *.NET Data Provider*. Figure 11-1 illustrates the concept of separating data access from data manipulation.



Espacios de nombres y clases en ADO .NET

En el presente apartado vamos a enumerar brevemente los principales elementos que forman parte del API de ADO .NET.

Primero vamos a comentar los distintos espacios de nombres que constituyen la tecnología ADO .NET:

- **System.Data**. Clases genéricas de datos de ADO .NET. Integra la gran mayoría de clases que habilitan el acceso a los datos de la arquitectura .NET.
- **System.Data.SqlClient**. Clases del proveedor de datos de SQL Server. Permite el acceso a proveedores SQL Server en su versión 7.0 y superior.
- **System.Data.OleDb**. Clases del proveedor de datos de OleDb. Permite el acceso a proveedores .NET que trabajan directamente contra controladores basados en los ActiveX de Microsoft.
- **System.Data.SqlTypes**. Definición de los tipos de datos de SQL Server. Proporciona la encapsulación en clases de todos los tipos de datos nativos de SQL Server y sus funciones de manejo de errores, ajuste y conversión de tipos, etc.
- **System.Data.Common**. Clases base, reutilizables de ADO .NET. Proporciona la colección de clases necesarias para acceder a una fuente de datos (como por ejemplo una Base de Datos).
- **System.Data.Internal**. Integra el conjunto de clases internas de las que se componen los proveedores de datos.

- . • **DataSet**. Almacén de datos por excelencia en ADO .NET. Representa una base de datos desconectada del proveedor de datos. Almacena tablas y sus relaciones.
- . • **DataTable**. Un contenedor de datos. Estructurado como un conjunto de filas (DataRow) y columnas (DataColumn).
- . • **DataRow**. Registro que almacena n valores. Representación en ADO .NET de una fila/tupla de una tabla de la base de datos.
- . • **DataColumn**. Contiene la definición de una columna. Metadatos y datos asociados a su dominio.
- . • **DataRelation**. Enlace entre dos o más columnas iguales de dos o mas tablas.
- . • **Constraint**. Reglas de validación de las columnas de una tabla.
- . • **DataColumnMapping**. Vínculo lógico existente entre una columna de un objeto del DataSet y la columna física de la tabla de la base de datos.
- . • **DataTableMapping**. Vínculo lógico existente entre una tabla del DataSet y la tabla física de la base de datos.

Dentro del espacio de nombres System.Data encontramos las siguientes clases compartidas, que constituyen el eje central de ADO .NET.

Además de estas clases, existe otro grupo de clases consistente en las clases específicas de un proveedor de datos. Estas clases conforman los aspectos particulares de un fabricante de proveedores de datos .NET. Tienen una sintaxis con el formato XXXClase, donde “XXX” es un prefijo que determina el tipo de plataforma de conexión a datos. Se definen en dos espacios de nombre: System.Data.SqlClient y System.Data.OleDb.

En la Tabla se ofrece una descripción de las clases que podemos encontrar en estos espacios de nombre. Clase	Descripción
SqlCommand OleDbCommand	Clases que representan un comando SQL contra un sistema gestor de datos.
SqlConnection OleDbConnection	Clase que representa la etapa de conexión a un proveedor de datos. Encapsula la seguridad, pooling de conexiones, etc.
SqlCommandBuilder OleDbCommandBuilder	Generador de comandos SQL de inserción, modificación y borrado desde una consulta SQL de selección de datos.
SqlDataReader OleDbDataReader	Un lector de datos de sólo avance, conectado a la base de datos.

En los ejemplos con datos que vamos a realizar, se ha utilizado SQL Server 2000 como servidor de datos, y fundamentalmente, la base de datos Northwind.

El primer paso obligado en un acceso a datos consiste en establecer una conexión con un almacén de datos. Esto lo vamos a conseguir gracias a las clases Connection de ADO .NET, que nos permitirán conectarnos a un origen de datos (ya sea una base de datos o no) , al igual que en ADO clásico empleábamos el objeto Connection.

En ADO se podía ejecutar directamente una sentencia contra el almacén de datos, o bien abrir un conjunto de registros (Recordset), pero en ADO .NET no vamos a realizar esta operación con este tipo de objetos.

Debemos recordar que existen dos implementaciones para algunos de los objetos de ADO .NET, cada uno específico del origen de datos con el que nos vamos a conectar. Esto ocurre con el objeto Connection, que tiene dos versiones, una como proveedor de datos de SQL Server, a través de la clase System.Data.SqlClient.SqlConnection, y otra como proveedor de datos OLEDB, a través de la clase System.Data.OleDb.OleDbConnection.

Por norma general, del objeto Connection utilizaremos los métodos Open() y Close(), para abrir y cerrar conexiones respectivamente, con el almacén de datos adecuado. Aunque tenemos el recolector de basura que gestiona de forma automática los recursos y objetos que no son utilizados, es recomendable cerrar las conexiones de forma explícita utilizando el método Close().

Las conexiones se abrirán de forma explícita utilizando el método Open(), pero también se puede hacer de forma implícita utilizando un objeto DataAdapter, esta posibilidad la veremos más adelante. Cuando ejecutamos el método Open() sobre un objeto Connection (SqlConnection o OleDbConnection), se abrirá la conexión que se ha indicado en su propiedadConnectionString, es decir, esta propiedad indicará la cadena de conexión que se va a utilizar para establecer la conexión con el almacén de datos correspondiente. El método Open() no posee parámetros.

El constructor de la clase Connection (al decir clase Connection de forma genérica nos estamos refiriendo en conjunto a las clases SqlConnection y OleDbConnection de ADO .NET) se encuentra sobrecargado, y en una de sus versiones recibe como parámetro una cadena que será la cadena de conexión que se aplique a su propiedad ConnectionString.

Si hacemos uso de la clase SqlConnection, en la cadena de conexión no podremos especificar una DSN de ODBC, ya que la conexión se va a realizar en este caso directamente con SQL Server. Y si utilizamos la clase OleDbConnection debemos especificar el proveedor OLEDB que se va a utilizar para establecer la conexión, una excepción es el proveedor OLEDB para ODBC (MSDASQL), que no puede ser utilizado, ya que el proveedor OLEDB de .NET no soporta el proveedor de ODBC, en este caso deberemos realizar la conexión utilizando el proveedor adecuado al almacén de datos. Los proveedores OLEDB que son compatibles con ADO .NET son:

- SQLOLEDB: Microsoft OLE DB Provider for SQL Server.
- MSDAORA: Microsoft OLE DB Provider for Oracle.
- Microsoft.Jet.OLEDB.4.0: OLE DB Provider for Microsoft Jet.

Uso del command

Establecida una conexión con un almacén de datos, la siguiente operación lógica consiste en enviarle sentencias para realizar los distintos tipos de operaciones que habitualmente realizamos con los datos. Las clases Command de ADO .NET serán las usaremos para realizar tales operaciones.

SqlCommand y OleDbCommand, son muy similares al objeto Command existente en ADO. El objeto Command nos va a permitir ejecutar una sentencia SQL o un procedimiento almacenado sobre la fuente de datos a la que estamos accediendo.

A través de un objeto Command también podremos obtener un conjunto de resultados del almacén de datos. En este caso, los resultados se pasarán a otros objetos de ADO .NET, como DataReader o DataAdapter; estos dos objetos los comentaremos más adelante.

Un objeto Command lo vamos a crear a partir de una conexión ya existente, y va a contener una sentencia SQL para ejecutar sobre la conexión establecida con el origen de datos.

Entre las propiedades que ofrecen los objetos SqlCommand y OleDbCommand, caben destacar las siguientes.

- **CommandText.** Contiene una cadena de texto que va a indicar la sentencia SQL o procedimiento almacenado que se va a ejecutar sobre el origen de los datos.

- **CommandTimeout.** Tiempo de espera en segundos que se va a aplicar a la ejecución de un objeto Command. Su valor por defecto es de 30 segundos.

- **CommandType.** Indica el tipo de comando que se va a ejecutar contra el almacén de datos, es decir, indica como se debe interpretar el valor de la propiedad CommandText. Puede tener los siguientes valores: StoredProcedure, para indicar que se trata de un procedimiento almacenado; TableDirect se trata de obtener una tabla por su nombre (únicamente aplicable al objeto OleDbCommand); y Text que indica que es una sentencia SQL. EL valor por defecto es Text.

- **Connection.** Devuelve el objeto SqlConnection o OleDbConnection utilizado para ejecutar el objeto Command correspondiente.

- **Parameters.** Colección de parámetros que se pueden utilizar para ejecutar el objeto Command, esta colección se utiliza cuando deseamos ejecutar sentencias SQL que hacen uso de parámetros, esta propiedad devuelve un objeto de la clase SqlParameterCollection o un objeto de la clase OleDbParameterCollection. Estas colecciones contendrán objetos de la clase SqlParameter y OleDbParameter, respectivamente, para representar a cada uno de los parámetros utilizados. Estos parámetros también son utilizados para ejecutar procedimientos almacenados.

Una vez vistas algunas de las propiedades de las clases SqlCommand y OleDbCommand, vamos a pasar a comentar brevemente los principales métodos de estas clases.

- **CreateParameter.** Crea un parámetro para el que después podremos definir una serie de características específicas como pueden ser el tipo de dato, su valor, tamaño, etc. Devolverá un objeto de la clase SqlParameter u OleDbParameter.

- **ExecuteNonQuery.** Ejecuta la sentencia SQL definida en la propiedad ComandText contra la conexión definida en la propiedad Connection. La sentencia a ejecutar debe ser de un tipo que no devuelva un conjunto de registros, por ejemplo Update, Delete o Insert. Este método

devuelve un entero indicando el número de filas que se han visto afectadas por la ejecución del objeto Command.

- **ExecuteReader.** Ejecuta la sentencia SQL definida en la propiedad ComandText contra la conexión definida en la propiedad Connection. En este caso, la sentencia sí devolverá un conjunto de registros. El resultado de la ejecución de este será un objeto de la clase SqlDataReader/OleDbDataReader, que nos va a permitir leer y recorrer los resultados devueltos por la ejecución del objeto Command correspondiente.

- **ExecuteScalar.** Este método se utiliza cuando deseamos obtener la primera columna de la primera fila del conjunto de registros, el resto de datos no se tendrán en cuenta. La utilización de este método tiene sentido cuando estamos ejecutando una sentencia SQL del tipo Select Count(*). Este método devuelve un objeto de la clase genérica Object.

- **Prepare.** Este método construye una versión compilada del objeto Command dentro del almacén de datos.

A continuación mostraremos algunos ejemplos de uso de objetos Command.

El Código fuente 561 ilustra la inserción de un registro utilizando un objeto SqlCommand. En primer lugar utilizamos un constructor de la clase, que recibe como parámetro la sentencia a ejecutar y la conexión. Como vamos a ejecutar una sentencia que no produce un conjunto de resultados, emplearemos el método ExecuteNonQuery(). Observe también el lector en este ejemplo, que la conexión sólo permanece abierta en el momento de ejecutar el comando; esta es la técnica recomendable que debemos utilizar para todas las operaciones con datos: mantener abierta la conexión el menor tiempo posible.

Ejemplo:

```
' crear conexión Dim oConexion As New
SqlConnection() oConexion.ConnectionString =
"Server=(local);" & _
"Database=Gestion;uid=sa;pwd=;"

' crear sentencia SQL Dim sSQL As String sSQL =
"INSERT INTO Clientes (IDCliente,Nombre,FIngreso) " & _
"VALUES(10,'Alfredo','18/7/2002')"
```

```
' crear comando Dim oComando As
New SqlCommand(sSQL, oConexion)

Dim iResultado As Integer oConexion.Open() ' abrir
conexión iResultado = oComando.ExecuteNonQuery() '
ejecutar comando oConexion.Close() ' cerrar conexión

MessageBox.Show("Registros añadidos:" & iResultado)
```


Las clases DataReader

Un objeto DataReader permite la navegación hacia delante y de sólo lectura, de los registros devueltos por una consulta. Es lo más parecido al objeto Recordset de ADO de tipo read only/forward only.

A diferencia del resto de objetos, que siguen un esquema desconectado de manipulación de datos, los DataReader permanecen conectados durante todo el tiempo que realizan el recorrido por los registros que contienen. Las clases que implementan este tipo de objeto son SqlDataReader y OleDbDataReader.

Para obtener un DataReader, ejecutaremos el método ExecuteReader() de un objeto Command basado en una consulta SQL o procedimiento almacenado.

A continuación vamos a pasar a describir las principales propiedades de las clases SqlDataReader y OleDbDataReader.

- . • **FieldCount.** Devuelve el número de columnas (campos) presentes en el fila (registro) actual.
- . • **IsClosed.** Devolverá los valores True o False, para indicar si el objeto DataReader está cerrado o no.
- . • **Item.** Devuelve en formato nativo, el valor de la columna cuyo nombre le indicamos como índice en forma de cadena de texto.

Una vez vistas las propiedades, vamos a comentar los métodos más destacables.

- . • **Close().** Cierra el objeto DataReader liberando los recursos correspondientes.

. • **GetXXX().** El objeto DataReader presenta un conjunto de métodos que nos van a permitir obtener los valores de las columnas contenidas en el mismo en forma de un tipo de datos determinado, según el método GetXXX empleado. Existen diversos métodos GetXXX atendiendo al tipo de datos de la columna, algunos ejemplos son GetBoolean(), GetInt32(), GetString(), GetChar(), etc. Como parámetro a este método le debemos indicar el número de orden de la columna que deseamos recuperar.

. • **NextResult().** Desplaza el puntero actual al siguiente conjunto de registros, cuando la sentencia es un procedimiento almacenado de SQL o una sentencia SQL que devuelve más de un conjunto de registros, no debemos confundir este método con el método MoveNext() de ADO, ya que en este caso no nos movemos al siguiente registro, sino al siguiente conjunto de registros.

. • **Read().** Desplaza el cursor actual al siguiente registro permitiendo obtener los valores del mismo a través del objeto DataReader. Este método devolverá True si existen más registros dentro del objeto DataReader, False si hemos llegado al final del conjunto de registros. La posición por defecto del objeto DataReader en el momento inicial es antes del primer registro, por lo tanto para recorrer un objeto DataReader debemos comenzar con una llamada al método Read(), y así situarnos en el primer registro.

El proyecto PruDataReader (hacer clic aquí para acceder al ejemplo), contiene un formulario con algunos controles, que muestran el uso de objetos DataReader.

El botón Empleados crea a partir de un comando, un objeto DataReader que recorreremos para llenar un ListBox con los valores de una de las columnas de la tabla que internamente contiene el DataReader

Más adelante veremos su aplicación.

Conjuntos de datos y enlace (Data Binding)

DataSet pertenece al conjunto común de clases de ADO .NET, empleándose para todo tipo de proveedores, por lo que no existe una versión particular para SqlConnection u OleDb,. En la introducción que sobre ADO .NET realizamos en el anterior tema, hemos comentado algunos aspectos sobre esta clase.

Básicamente, un objeto DataSet va a ser similar a un objeto Recordset de ADO, pero más potente y complejo. Es el almacén de datos por excelencia en ADO .NET, representando una base de datos en memoria y desconectada del proveedor de datos, que contiene tablas y sus relaciones.

El objeto DataSet nos proporciona el mejor concepto sobre datos desconectados: una copia en el cliente de la arquitectura de la base de datos, basada en un esquema XML que la independiza del fabricante, proporcionando al desarrollador la libertad de trabajo independiente de la plataforma.

Cada tabla contenida dentro de un objeto DataSet se encuentra disponible a través de su propiedad Tables, que es una colección de objetos System.Data.DataTable. Cada objeto DataTable contiene una colección de objetos DataRow que representan las filas de la tabla. Y si seguimos con esta analogía tenemos que decir que cada objeto DataRow, es decir, cada fila, posee una colección de objetos DataColumn, que representan cada una de las columnas de la fila actual. Existen además, colecciones y objetos para representar las relaciones, claves y valores por defecto existentes dentro de un objeto DataSet.

Cada objeto DataTable dispone de una propiedad llamada DefaultView, que devuelve un objeto de la clase DataView, el cual nos ofrece una vista de los datos de la tabla para que podamos recorrer los datos, filtrarlos, ordenarlos, etc.

Para poder crear e inicializar las tablas del DataSet debemos hacer uso del objeto DataAdapter, que posee las dos versiones, es decir, el objeto SqlDataAdapter para SQL Server y OleDbDataAdapter genérico de OLE DB.

Al objeto DataAdapter le pasaremos como parámetro una cadena que representa la consulta que se va a ejecutar, y que va a rellenar de datos el DataSet. Del objeto DataAdapter utilizaremos el método Fill(), que posee dos parámetros; el primero es el DataSet a rellenar de información; y el segundo, una cadena con el nombre que tendrá la tabla creada dentro del DataSet, producto de la ejecución de la consulta.

En el siguiente apartado veremos los objetos DataAdapter, que van a funcionar como intermediarios entre el almacén de datos, y el objeto DataSet, que contiene la versión desconectada de los datos.

Entre los métodos más destacables de la clase DataSet podemos mencionar los siguientes.

- **Clear()**. Elimina todos los datos almacenados en el objeto DataSet, vaciando todas las tablas contenidas en el mismo.

- **AcceptChanges()**. Confirma todos los cambios realizados en las tablas y relaciones contenidas en el objeto DataSet, o bien los últimos cambios que se han producido desde la última llamada al método AcceptChanges.

. • **GetChanges()**. Devuelve un objeto DataSet que contiene todos los cambios realizados desde que se cargó con datos, o bien desde que se realizó la última llamada al método AcceptChanges.

. • **HasChanges()**. Devuelve true o false para indicar si se han realizado cambios al contenido del DataSet desde que fue cargado o bien desde que se realizó la última llamada al método AcceptChanges.

. • **RejectChanges()**. Abandona todos los cambios realizados en las tablas contenidas en el objeto DataSet desde que fue cargado el objeto o bien desde la última vez que se lanzó el método AcceptChanges.

. • **Merge()**. Toma los contenidos de un DataSet y los mezcla con los de otro DataSet, de forma que contendrá los datos de ambos objetos DataSet.

En lo que respecta a las propiedades de la clase DataSet, podemos remarcar las siguientes.

. • **CaseSensitive**. Propiedad que indica si las comparaciones de texto dentro de las tablas distinguen entre mayúsculas y minúsculas. Por defecto tiene el valor False.

. • **DataSetName**. Establece o devuelve mediante una cadena de texto el nombre del objeto DataSet.

. • **HasErrors**. Devuelve un valor lógico para indicar si existen errores dentro de las tablas del DataSet.

. • **Relations**. Esta propiedad devuelve una colección de objetos DataRelation, que representan todas las relaciones existentes entre las tablas del objeto DataSet.

. • **Tables**. Devuelve una colección de objetos DataTable, que representan a cada una de las tablas existentes dentro del objeto DataSet.

Las clases DataAdapter

Como hemos comentado en anteriores apartados, los objetos DataAdapter (SqlDataAdapter y OleDbDataAdapter) van a desempeñar el papel de puente entre el origen de datos y el DataSet, permitiéndonos cargar el DataSet con la información de la fuente de datos, y posteriormente, actualizar el origen de datos con la información del DataSet.

Un objeto DataAdapter puede contener desde una sencilla sentencia SQL, como hemos visto en el apartado anterior, hasta varios objetos Command.

La clase DataAdapter dispone de cuatro propiedades, que nos van a permitir asignar a cada una, un objeto Command (SqlCommand u OleDbCommand) con las operaciones estándar de manipulación de datos. Estas propiedades son las siguientes.

- . • **InsertCommand**. Objeto de la clase Command, que se va a utilizar para realizar una inserción de datos.
- . • **SelectCommand**. Objeto de la clase Command que se va a utilizar para ejecutar una sentencia Select de SQL.
- . • **UpdateCommand**. Objeto de la clase Command que se va a utilizar para realizar una modificación de los datos.
- . • **DeleteCommand**. Objeto de la clase Command que se va a utilizar para realizar una eliminación de datos.

Un método destacable de las clases SqlDataAdapter/OleDbDataAdapter es el método Fill(), que ejecuta el comando de selección que se encuentra asociado a la propiedad SelectCommand, los datos obtenidos del origen de datos se cargarán en el objeto DataSet que pasamos por parámetro.

Para demostrar el uso de los objetos DataAdapter vamos a desarrollar un proyecto con el nombre PruDataAdapter (hacer clic aquí para acceder a este ejemplo). En esta aplicación vamos a utilizar el mismo objeto DataAdapter para realizar una consulta contra una tabla e insertar nuevas filas en esa misma tabla.

En primer lugar diseñaremos el formulario del programa. Como novedad, introduciremos el control DataGridView, que trataremos con más profundidad en un próximo apartado. Baste decir por el momento, que a través del DataGridView visualizaremos una o varias tablas contenidas en un DataSet

Navegación y edición de registros en modo desconectado

Anteriormente vimos la forma de realizar operaciones de edición, en modo conectado, sobre las tablas de una base de datos, empleando los objetos Command. Pero como también ya sabemos, la arquitectura de ADO .NET está orientada a un modelo de trabajo desconectado del almacén de datos, al que recurriremos sólo cuando necesitemos obtener los datos para su consulta y manipulación, o bien, cuando esos mismos datos desconectados, los hayamos modificado y tengamos que actualizarlos en la fuente de datos.

El objeto DataSet, combinado con un grupo de objetos enfocados al mantenimiento de datos desconectados, como son DataAdapter, DataTable, DataRow, etc., nos van a permitir realizar tareas como la navegación entre los registros de una tabla del DataSet, además de la modificación de sus datos en las operaciones habituales de inserción, modificación y borrado de filas.

Data Binding. Enlace de datos a controles

Data Binding es el mecanismo proporcionado por la plataforma .NET, que en aplicaciones con interfaz Windows Forms, enlaza objetos contenedores de datos con los controles del formulario, para poder realizar operaciones automáticas de navegación y edición.

Tipos de Data Binding

Existen dos tipos de enlace de datos: simple y complejo.

- **Enlace simple (Simple Data Binding).** Este tipo de enlace consiste en una asociación entre un control que puede mostrar un único dato y el objeto que actúa como contenedor de datos. El ejemplo más ilustrativo es el control TextBox.

- **Enlace complejo (Complex Data Binding).** En este enlace, el control que actúa como interfaz o visualizador de datos, dispone de la capacidad de mostrar varios o todos los datos del objeto que contiene la información. El control más común es el control DataGridView, que ya hemos visto inicialmente en un apartado anterior, y que trataremos con más detenimiento próximamente.

Elementos integrantes en un proceso de Data Binding

El mecanismo de enlace automático de datos a controles está compuesto por un elevado conjunto de elementos del conjunto de tipos de .NET Framework, entre clases, colecciones, enumeraciones, etc. A continuación vamos a mencionar los más importantes, que emplearemos en el ejemplo desarrollado seguidamente.

- **Binding.** Clase que permite crear un enlace (binding) para un control, indicando la propiedad del control que mostrará los datos, el DataSet del que se extraerá la información, y el nombre de la tabla-columna, cuyos datos pasarán a la propiedad del control.

- **DataBindings.** Colección de que disponen los controles, con la información de enlaces a datos. Gracias a su método Add(), podemos añadir un objeto Binding, para que el control muestre los datos que indica el enlace.

- **BindingContext.** Propiedad de la clase Form, que representa el contexto de enlace a datos establecido en los controles del formulario, es decir, toda la información de enlaces establecida entre los controles y objetos proveedores de datos. Devuelve un objeto de tipo BindingManagerBase.

- **BindingManagerBase.** Objeto que se encarga de administrar un conjunto de objetos de enlace, por ejemplo, los de un formulario, obtenidos a través del BindingContext del formulario.

Ejercicios Prácticos: Enunciados.

UNIDAD I

1. Desarrollar un programa que permita acceder a una tabla de una base de datos y mostrarlos en una grilla.
2. Desarrollar un programa que permita acceder a una tabla de una base de datos y que en la misma se puedan agregar, modificar y borrar registros.
3. Desarrollar un programa que permita acceder a una tabla de una base de datos y que en la misma se puedan agregar, modificar, borrar y consultar registros utilizando procedimientos almacenados.
4. Desarrollar un programa que permita acceder a dos tablas relacionadas de una base de datos y que en la misma se puedan agregar, modificar, borrar y consultar registros utilizando procedimientos almacenados. Para cada operación realizada abrir una transacción y controlar la misma.

UNIDAD II

5. Desarrollar un programa que permita acceder a dos tablas relacionadas de una base de datos aplicando el modelo desconectado. Levantar los datos a un DataSet. Agregar, modificar, borrar y consultar registros. Luego actualizar la base de datos por medio del adaptador.
6. Sobre el ejercicio anterior generar los SQL del adaptador con el objeto CommandBuilder.
7. Desarrollar un programa aplicando todo lo aprendido sobre ADO para administrar un video club.

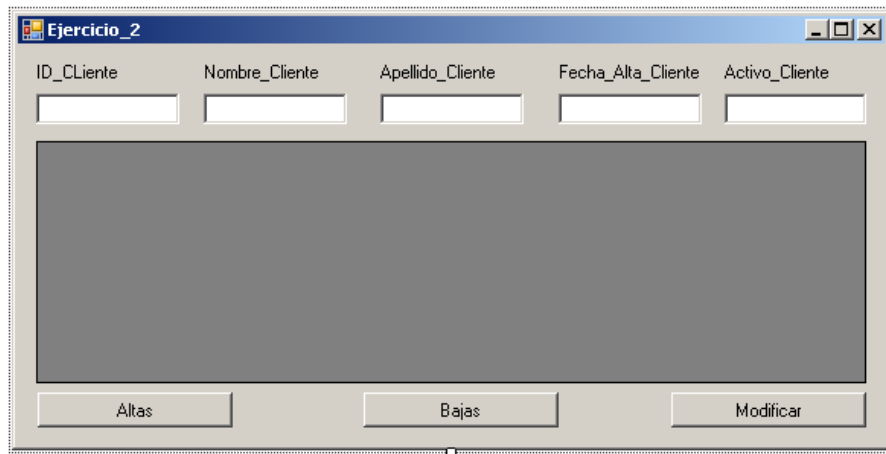
UNIDAD III

8. Desarrollar un programa que permita acceder a los datos de una base de datos aplicando el modelo desconectado. Levantar los datos a un DataSet. Agregar, modificar, borrar y consultar registros. Agregar la posibilidad que los datos del DataSet se guarden en un documento XML y que luego se puedan leer desde allí.

9. Desarrollar un programa que permita acceder a los datos de una base de datos, y que las consultas se graben en un documento XML para luego ser visualizados desde un navegador.

Uso de Conecction, Command a modo texto, Datareader y Grilla

El presente ejercicio representa una solución del tipo Monolítica que permite el acceso a datos para agregar, eliminar, actualizar, mostrar los datos en una grilla y actualizar los controles desde el click de la grilla.



```
Imports System.Data
```

```
Imports System.Data.SqlClient
```

```
Public Class Ejercicio_2
```

```
    Dim MyConnection As SqlConnection
```

```
    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As  
System.EventArgs) Handles Button1.Click
```

```
        MyConnection.Open()
```

```
        Dim MyCommand As New SqlCommand
```

```
        MyCommand.CommandText = "insert into Cliente  
(CLIE_ID,CLIE_NOMBRE,CLIE_APELLIDO,CLIE_FECHA_ALTA,CLIE_ACTIVO)  
VALUES ('" & TextBox1.Text & "','" & TextBox2.Text & "','" & TextBox3.Text & "','" &  
TextBox4.Text & "','" & TextBox5.Text & "') ;"
```

```
        MyCommand.CommandType = CommandType.Text
```

```
        MyCommand.Connection = MyConnection
```

```
        Try
```

```
            MyCommand.ExecuteNonQuery()
```

```
        Catch ex As Exception
```

```
        Finally
```

```

        MyConnection.Close()
        ActualizarGrilla()
    End Try
End Sub

```

```

Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button2.Click
    MyConnection.Open()
    Dim MyCommand As New SqlCommand
    MyCommand.CommandText = "delete from cliente where clie_id = " &
TextBox1.Text & ""
    MyCommand.CommandType = CommandType.Text
    MyCommand.Connection = MyConnection
    Try
        MyCommand.ExecuteNonQuery()
    Catch ex As Exception
    Finally
        MyConnection.Close()
        ActualizarGrilla()
    End Try
End Sub

```

```

Private Sub Button3_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button3.Click
    MyConnection.Open()
    Try
        Dim MyCommand As New SqlCommand
        MyCommand.CommandText = "update Cliente set CLIE_NOMBRE = " &
TextBox2.Text & ",CLIE_APELLIDO = " & TextBox3.Text & ",CLIE_FECHA_ALTA =
" & TextBox4.Text & ",CLIE_ACTIVO = " & TextBox5.Text & " where clie_id = " &
TextBox1.Text & ""
        MyCommand.CommandType = CommandType.Text
        MyCommand.Connection = MyConnection
        MyCommand.ExecuteNonQuery()
    Catch ex As Exception

```

Finally

MyConnection.Close()

ActualizarGrilla()

End Try

End Sub

Private Sub Ejercicio_2_Load1(ByVal sender As Object, ByVal e As
System.EventArgs) Handles Me.Load

MyConnection = New SqlConnection("Data Source=ANGUS-PC;Initial
Catalog=GESTION;Integrated Security=True")

ActualizarGrilla()

End Sub

Sub ActualizarGrilla()

MyConnection.Open()

Dim MyCommand As New SqlCommand

MyCommand.CommandText = "Select * from Cliente"

MyCommand.CommandType = CommandType.Text

MyCommand.Connection = MyConnection

Dim MyDataReader As SqlDataReader = MyCommand.ExecuteReader

Dim dt As DataTable = New DataTable

dt.Load(MyDataReader)

DataGridView1.DataSource = dt

MyDataReader.Close()

MyConnection.Close()

End Sub

Private Sub DataGridView1_CellClick(ByVal sender As Object, ByVal e As
System.Windows.Forms.DataGridViewCellEventArgs) Handles

DataGridView1.CellClick

DataGridView1.SelectionMode = DataGridViewSelectionMode.FullRowSelect

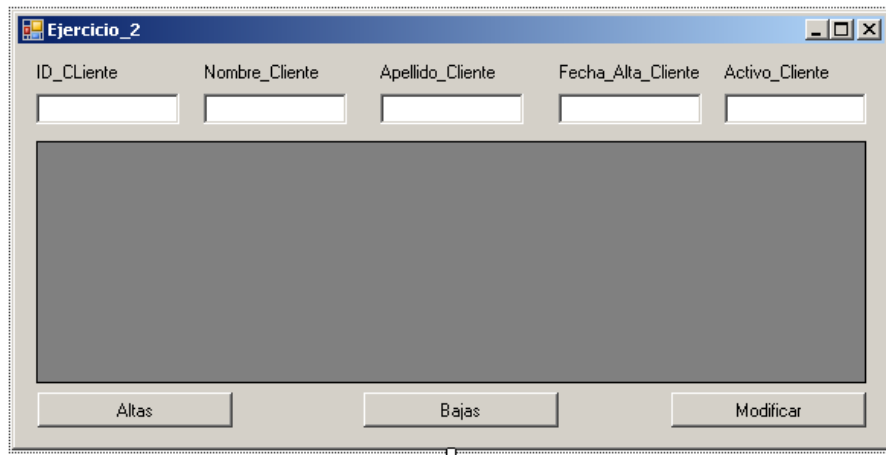
DataGridView1.EditMode = DataGridViewEditMode.EditProgrammatically

Try

TextBox1.Text = Me.DataGridView1.Rows(e.RowIndex).Cells(0).Value

```
TextBox2.Text = Me.DataGridView1.Rows(e.RowIndex).Cells(1).Value
    TextBox3.Text = Me.DataGridView1.Rows(e.RowIndex).Cells(2).Value
    TextBox4.Text = Me.DataGridView1.Rows(e.RowIndex).Cells(3).Value
    TextBox5.Text = Me.DataGridView1.Rows(e.RowIndex).Cells(4).Value
Catch ex As Exception
End Try
End Sub
End Class
```

Use Connection, Command con parámetros, DataReader y Grilla



Imports System.Data

Imports System.Data.SqlClient

Public Class Ejercicio_3

```
Dim Conn As New SqlConnection("Data Source=.;Initial  
Catalog=GESTION;Integrated Security=True")
```

```
Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As  
System.EventArgs) Handles Button1.Click
```

```
Dim Comm As New SqlCommand("spAltaCliente", Conn)  
Comm.CommandType = CommandType.StoredProcedure  
Comm.Parameters.Add("@CLIENTE_ID", SqlDbType.Int).Value =  
Convert.ToInt16(TextBox1.Text)  
Comm.Parameters.Add("@CLIENTE_NOMBRE", SqlDbType.NVarChar).Value =  
Convert.ToString(TextBox2.Text)  
Comm.Parameters.Add("@CLIENTE_APELLIDO", SqlDbType.NVarChar).Value  
= Convert.ToString(TextBox3.Text)  
Comm.Parameters.Add("@CLIENTE_FECHA_ALTA", SqlDbType.Date).Value =  
Convert.ToDateTime(TextBox4.Text)  
Comm.Parameters.Add("@CLIENTE_ACTIVO", SqlDbType.Bit).Value =  
Convert.ToBoolean(TextBox5.Text)  
Try  
Conn.Open()
```

```

        Comm.ExecuteNonQuery()
    Catch ex As Exception
    Finally
        Conn.Close()
        ActualizarGrilla()
    End Try
End Sub

```

Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button2.Click

```

    Dim Comm As New SqlCommand("spBajaCliente", Conn)
    Comm.CommandType = CommandType.StoredProcedure
    Comm.Parameters.Add("@CLIENTE_ID", SqlDbType.Int).Value =
Convert.ToInt16(TextBox1.Text)
    Try
        Conn.Open()
        Comm.ExecuteNonQuery()
    Catch ex As Exception
        MsgBox(ex.Message)
    Finally
        Conn.Close()
        ActualizarGrilla()
    End Try
End Sub

```

Private Sub Button3_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button3.Click

```

    Dim Comm As New SqlCommand("spModificaCliente", Conn)
    Comm.CommandType = CommandType.StoredProcedure
    Comm.Parameters.Add("@CLIENTE_ID", SqlDbType.Int).Value =
Convert.ToInt16(TextBox1.Text)
    Comm.Parameters.Add("@CLIENTE_NOMBRE", SqlDbType.NVarChar).Value =
Convert.ToString(TextBox2.Text)
    Comm.Parameters.Add("@CLIENTE_APELLIDO", SqlDbType.NVarChar).Value
= Convert.ToString(TextBox3.Text)
    Comm.Parameters.Add("@CLIENTE_FECHA_ALTA", SqlDbType.Date).Value =
Convert.ToDateTime(TextBox4.Text)

```



```
Comm.Parameters.Add("@CLIENTE_ACTIVO", SqlDbType.Bit).Value =  
Convert.ToBoolean(TextBox5.Text)
```

```
Try  
    Conn.Open()  
    Comm.ExecuteNonQuery()  
Catch ex As Exception  
Finally  
    Conn.Close()  
    ActualizarGrilla()  
End Try  
End Sub
```

```
Private Sub Ejercicio_3_Load(ByVal sender As Object, ByVal e As  
System.EventArgs) Handles Me.Load  
    ActualizarGrilla()  
End Sub
```

```
Sub ActualizarGrilla()  
    Conn.Open()  
    Dim Comm As New SqlCommand("spSelectCliente", Conn)  
    Comm.CommandType = CommandType.StoredProcedure  
    Dim MyDataReader As SqlDataReader = Comm.ExecuteReader  
    Dim dt As DataTable = New DataTable  
    dt.Load(MyDataReader)  
    DataGridView1.DataSource = dt  
    MyDataReader.Close()  
    Conn.Close()  
End Sub
```

```
Private Sub DataGridView1_CellClick1(ByVal sender As Object, ByVal e As  
System.Windows.Forms.DataGridViewCellEventArgs) Handles  
DataGridView1.CellClick  
    DataGridView1.SelectionMode = DataGridViewSelectionMode.FullRowSelect  
    DataGridView1.EditMode = DataGridViewEditMode.EditProgrammatically  
Try  
    TextBox1.Text = Me.DataGridView1.Rows(e.RowIndex).Cells(0).Value
```

```
TextBox2.Text = Me.DataGridView1.Rows(e.RowIndex).Cells(1).Value
    TextBox3.Text = Me.DataGridView1.Rows(e.RowIndex).Cells(2).Value
    TextBox4.Text = Me.DataGridView1.Rows(e.RowIndex).Cells(3).Value
    TextBox5.Text = Me.DataGridView1.Rows(e.RowIndex).Cells(4).Value
Catch ex As Exception
End Try
End Sub

End Class
```

Uso Connection, Command con parámetros, Transacción, DataReader y Grilla



Imports System.Data

Imports System.Data.SqlClient

Public Class Ejercicio_4

Dim Conn As New SqlConnection("Data Source=.;Initial
Catalog=GESTION;Integrated Security=True")

Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button1.Click

Conn.Open()

Dim tr As SqlTransaction

tr = Conn.BeginTransaction

Try

Dim Comm As SqlCommand = New SqlCommand("spAltaCliente")

Comm.CommandType = CommandType.StoredProcedure

Comm.Parameters.Add("@CLIENTE_ID", SqlDbType.Int).Value =
Convert.ToInt16(TextBox1.Text)

Comm.Parameters.Add("@CLIENTE_NOMBRE", SqlDbType.NVarChar).Value
= Convert.ToString(TextBox2.Text)

Comm.Parameters.Add("@CLIENTE_APELLIDO",
SqlDbType.NVarChar).Value = Convert.ToString(TextBox3.Text)

Comm.Parameters.Add("@CLIENTE_FECHA_ALTA", SqlDbType.Date).Value
= Convert.ToDateTime(TextBox4.Text)

```

        Comm.Parameters.Add("@CLIENTE_ACTIVADO", SqlDbType.Bit).Value =
Convert.ToBoolean(TextBox5.Text)
        Comm.Connection = Conn
        Comm.Transaction = tr
        Comm.ExecuteNonQuery()
        tr.Commit()
    Catch ex As Exception
        MsgBox(ex.Message)
        tr.Rollback()
    Finally
        Conn.Close()
        ActualizarGrilla()
    End Try
End Sub

```

```

Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button2.Click
    Dim Comm As SqlCommand = New SqlCommand("spBajaCliente", Conn)
    Comm.CommandType = CommandType.StoredProcedure
    Comm.Parameters.Add("@CLIENTE_ID", SqlDbType.Int).Value =
Convert.ToInt16(TextBox1.Text)
    Conn.Open()
    Dim tr As SqlTransaction
    tr = Conn.BeginTransaction
    Try
        Comm.Transaction = tr
        Comm.ExecuteNonQuery()
        tr.Commit()
    Catch ex As Exception
        MsgBox(ex.Message)
        tr.Rollback()
    Finally
        Conn.Close()
        ActualizarGrilla()
    End Try
End Sub

```

```

Private Sub Button3_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button3.Click
    Dim Comm As SqlCommand = New SqlCommand("spModificaCliente", Conn)
    Comm.CommandType = CommandType.StoredProcedure
    Comm.Parameters.Add("@CLIENTE_ID", SqlDbType.Int).Value =
Convert.ToInt16(TextBox1.Text)
    Comm.Parameters.Add("@CLIENTE_NOMBRE", SqlDbType.NVarChar).Value =
Convert.ToString(TextBox2.Text)
    Comm.Parameters.Add("@CLIENTE_APELLIDO", SqlDbType.NVarChar).Value
= Convert.ToString(TextBox3.Text)
    Comm.Parameters.Add("@CLIENTE_FECHA_ALTA", SqlDbType.Date).Value =
Convert.ToDateTime(TextBox4.Text)
    Comm.Parameters.Add("@CLIENTE_ACTIVO", SqlDbType.Bit).Value =
Convert.ToBoolean(TextBox5.Text)
    Conn.Open()
    Dim tr As SqlTransaction
    tr = Conn.BeginTransaction
    Try
        Comm.Transaction = tr
        Comm.ExecuteNonQuery()
        tr.Commit()
    Catch ex As Exception
        MsgBox(ex.Message)
        tr.Rollback()
    Finally
        Conn.Close()
        ActualizarGrilla()
    End Try
End Sub

```

```

Private Sub Ejercicio_3_Load(ByVal sender As Object, ByVal e As
System.EventArgs) Handles Me.Load
    ActualizarGrilla()
End Sub

```

```

Sub ActualizarGrilla()

```

```

Conn.Open()
    Dim Comm1 As New SqlCommand("spSelectCliente", Conn)
    Comm1.CommandType = CommandType.StoredProcedure
    Dim dr1 As SqlDataReader = Comm1.ExecuteReader
    Dim dt1 As DataTable = New DataTable
    dt1.Load(dr1)
    DataGridView1.DataSource = dt1
    dr1.Close()
    Dim comm2 As New SqlCommand("spSelectTelefono", Conn)
    comm2.CommandType = CommandType.StoredProcedure
    Dim dr2 As SqlDataReader = comm2.ExecuteReader
    Dim dt2 As DataTable = New DataTable
    dt2.Load(dr2)
    DataGridView2.DataSource = dt2
    dr2.Close()
    Conn.Close()
End Sub

```

```

Sub ActualizarGrilla2()
    Conn.Open()
    Dim comm2 As New SqlCommand("spSelectTelefono1", Conn)
    comm2.CommandType = CommandType.StoredProcedure
    comm2.Parameters.Add("@ID_Clie", SqlDbType.Int).Value =
Convert.ToInt16(TextBox1.Text)
    Dim dr2 As SqlDataReader = comm2.ExecuteReader
    Dim dt2 As DataTable = New DataTable
    dt2.Load(dr2)
    DataGridView2.DataSource = dt2
    dr2.Close()
    Conn.Close()
End Sub

```

```

Private Sub DataGridView1_CellClick(ByVal sender As Object, ByVal e As
System.Windows.Forms.DataGridViewCellEventArgs) Handles
DataGridView1.CellClick
    DataGridView1.SelectionMode = DataGridViewSelectionMode.FullRowSelect
    DataGridView1.EditMode = DataGridViewEditMode.EditProgrammatically

```

Try

TextBox1.Text = Me.DataGridView1.Rows(e.RowIndex).Cells(0).Value

TextBox2.Text = Me.DataGridView1.Rows(e.RowIndex).Cells(1).Value

TextBox3.Text = Me.DataGridView1.Rows(e.RowIndex).Cells(2).Value

TextBox4.Text = Me.DataGridView1.Rows(e.RowIndex).Cells(3).Value

TextBox5.Text = Me.DataGridView1.Rows(e.RowIndex).Cells(4).Value

TextBox8.Text = Me.DataGridView1.Rows(e.RowIndex).Cells(0).Value

Catch ex As Exception

End Try

ActualizarGrilla2()

End Sub

Private Sub Button4_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button4.Click

Dim Comm As SqlCommand = New SqlCommand("spAltaTelefono", Conn)

Comm.CommandType = CommandType.StoredProcedure

Comm.Parameters.Add("@Tel_ID", SqlDbType.Int).Value =

Convert.ToInt16(TextBox6.Text)

Comm.Parameters.Add("@Num_Tel", SqlDbType.Int).Value =

Convert.ToInt16(TextBox7.Text)

Comm.Parameters.Add("@CLIENTE_ID", SqlDbType.Int).Value =

Convert.ToInt16(TextBox1.Text)

Conn.Open()

Dim tr As SqlTransaction

tr = Conn.BeginTransaction

Try

Comm.Transaction = tr

Comm.ExecuteNonQuery()

tr.Commit()

Catch ex As Exception

MsgBox(ex.Message)

tr.Rollback()

Finally

Conn.Close()

ActualizarGrilla()

End Try

End Sub

```

Private Sub Button5_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button5.Click
    Dim Comm As SqlCommand = New SqlCommand("spBajaTelefono", Conn)
    Comm.CommandType = CommandType.StoredProcedure
    Comm.Parameters.Add("@Tel_ID", SqlDbType.Int).Value =
Convert.ToInt16(TextBox6.Text)
    Conn.Open()
    Dim tr As SqlTransaction
    tr = Conn.BeginTransaction
    Try
        Comm.Transaction = tr
        Comm.ExecuteNonQuery()
        tr.Commit()
    Catch ex As Exception
        MsgBox(ex.Message)
        tr.Rollback()
    Finally
        Conn.Close()
        ActualizarGrilla()
    End Try
End Sub

```

```

Private Sub Button6_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button6.Click
    Dim Comm As SqlCommand = New SqlCommand("spModificaTelefono", Conn)
    Comm.CommandType = CommandType.StoredProcedure
    Comm.Parameters.Add("@Tel_ID", SqlDbType.Int).Value =
Convert.ToInt16(TextBox6.Text)
    Comm.Parameters.Add("@Num_Tel", SqlDbType.Int).Value =
Convert.ToInt16(TextBox7.Text)
    Comm.Parameters.Add("@CLIENTE_ID", SqlDbType.Int).Value =
Convert.ToInt16(TextBox1.Text)
    Conn.Open()
    Dim tr As SqlTransaction
    tr = Conn.BeginTransaction
    Try
        Comm.Transaction = tr

```



```

Comm.ExecuteNonQuery()
    tr.Commit()
Catch ex As Exception
    MsgBox(ex.Message)
    tr.Rollback()
Finally
    Conn.Close()
    ActualizarGrilla()
End Try
End Sub

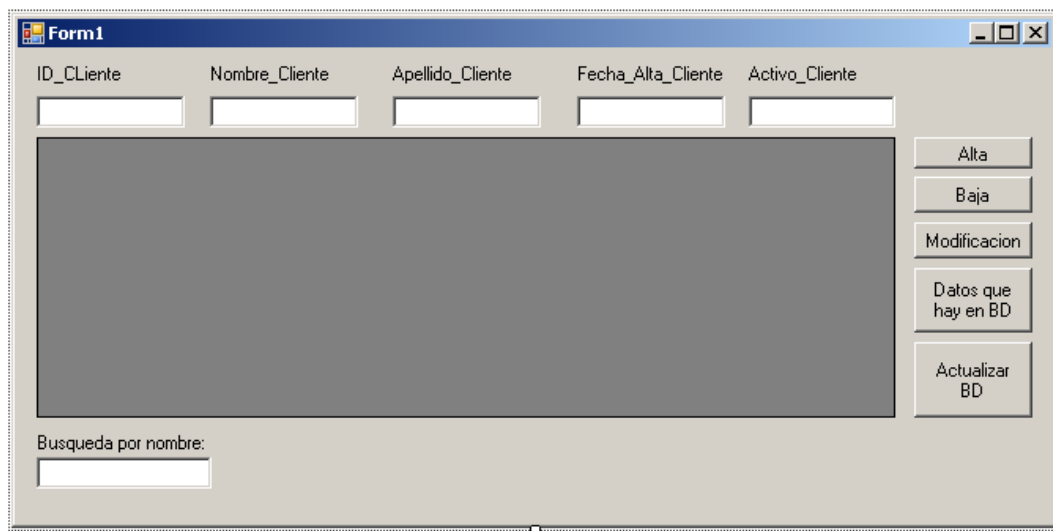
```

```

Private Sub DataGridView2_CellClick(ByVal sender As Object, ByVal e As
System.Windows.Forms.DataGridViewCellEventArgs) Handles
DataGridView2.CellClick
    DataGridView2.SelectionMode = DataGridViewSelectionMode.FullRowSelect
    DataGridView2.EditMode = DataGridViewEditMode.EditProgrammatically
    Try
        TextBox6.Text =
Convert.ToInt16(Me.DataGridView2.Rows(e.RowIndex).Cells(0).Value)
        TextBox7.Text =
Convert.ToInt16(Me.DataGridView2.Rows(e.RowIndex).Cells(1).Value)
        TextBox8.Text =
Convert.ToInt16(Me.DataGridView2.Rows(e.RowIndex).Cells(2).Value)
        TextBox1.Text = Me.DataGridView1.Rows(e.RowIndex).Cells(0).Value
        TextBox2.Text = Me.DataGridView1.Rows(e.RowIndex).Cells(1).Value
        TextBox3.Text = Me.DataGridView1.Rows(e.RowIndex).Cells(2).Value
        TextBox4.Text = Me.DataGridView1.Rows(e.RowIndex).Cells(3).Value
        TextBox5.Text = Me.DataGridView1.Rows(e.RowIndex).Cells(4).Value
    Catch ex As Exception
    End Try
End Sub
End Class

```

Uso de Connection, DataAdapter, CommandBuilder, DataTable con clave primaria, Grilla



Imports System.Data

Imports System.Data.SqlClient

Public Class Ejercicio_5_y_6

Dim Conn As New SqlConnection("Data Source=.;Initial
Catalog=GESTION;Integrated Security=True")

Dim DAdaptador As New SqlDataAdapter("Select * from Cliente", Conn)

Dim DSet As New DataSet

Dim x As Integer

Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button1.Click

DAdaptador.Fill(DSet)

Dim c As Integer = 0

For Each it As DataRow In DSet.Tables(0).Rows

If c < it.Item(0) Then

c = it.Item(0)

End If

Next

Dim DRow As DataRow = DSet.Tables(0).NewRow

DRow.Item(0) = c + 1

DRow.Item(1) = TextBox2.Text

```

DRow.Item(2) = TextBox3.Text
DRow.Item(3) = TextBox4.Text
DRow.Item(4) = TextBox5.Text
DSet.Tables(0).Rows.Add(DRow)
For j As Integer = 0 To DSet.Tables(0).Rows.Count - 1
    If DSet.Tables(0).Rows.Item(j).RowState = DataRowState.Added Then
        DataGridView1.Rows(j).DefaultCellStyle.ForeColor = Color.Green
    End If
Next
End Sub

```

```

Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button2.Click
    x = TextBox1.Text
    Dim z As Integer = MessageBox.Show("¿Realmente desea Borrar los datos del
Cliente " & TextBox1.Text & "?", "Borrar", MessageBoxButtons.YesNo)
    If z = vbYes Then
        For Each i As DataRow In DSet.Tables(0).Rows
            If i.Item(0) = x Then
                i.Delete()
            End If
        Next
    End If
    LimpiarDatos()
    DataGridView1.Refresh()
End Sub

```

```

Private Sub Button3_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button3.Click
    x = TextBox1.Text
    Dim z As Integer = MessageBox.Show("¿Realmente desea modificar los datos
del Cliente " & TextBox1.Text & "?", "Modificar", MessageBoxButtons.YesNo)
    If z = vbYes Then
        For Each i As DataRow In DSet.Tables(0).Rows
            If i.Item(0) = x Then
                i.Item(1) = TextBox2.Text
                i.Item(2) = TextBox3.Text
            End If
        Next
    End If
End Sub

```

```

i.Item(3) = TextBox4.Text
    i.Item(4) = TextBox5.Text
End If
Next
End If
For j As Integer = 0 To DSet.Tables(0).Rows.Count - 1
    If DSet.Tables(0).Rows(j).RowState = DataRowState.Modified Then
        DataGridView1.Rows(j).DefaultCellStyle.ForeColor = Color.Red
    End If
Next
LimpiarDatos()
DataGridView1.Refresh()
End Sub

```

```

Private Sub Button4_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button4.Click
    ActualizarGrilla()
End Sub

```

```

Private Sub Button5_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button5.Click
    Dim CBuilder As New SqlCommandBuilder(DAdaptador)
    DAdaptador.InsertCommand = CBuilder.GetInsertCommand
    DAdaptador.DeleteCommand = CBuilder.GetDeleteCommand
    DAdaptador.UpdateCommand = CBuilder.GetUpdateCommand
    DAdaptador.Update(DSet)
    ActualizarGrilla()
End Sub

```

```

Private Sub TextBox6_TextChanged(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles TextBox6.TextChanged
    Dim DView As New DataView(DSet.Tables(0))
    DView.RowFilter = "CLIE_NOMBRE Like '" & TextBox6.Text & "%'"
    DataGridView1.DataSource = DView
End Sub

```

```

Private Sub Ejercicio_5_Load(ByVal sender As Object, ByVal e As
System.EventArgs) Handles Me.Load
    ActualizarGrilla()
End Sub

```

```

Private Sub DataGridView1_CellClick(ByVal sender As Object, ByVal e As
System.Windows.Forms.DataGridViewCellEventArgs) Handles
DataGridView1.CellClick
    DataGridView1.SelectionMode = DataGridViewSelectionMode.FullRowSelect
    DataGridView1.EditMode = DataGridViewEditMode.EditProgrammatically
    TextBox1.Text = DataGridView1.Rows(e.RowIndex).Cells(0).Value.ToString
    TextBox1.ReadOnly = True
    TextBox2.Text = DataGridView1.Rows(e.RowIndex).Cells(1).Value.ToString
    TextBox3.Text = DataGridView1.Rows(e.RowIndex).Cells(2).Value.ToString
    TextBox4.Text =
DateTime.Parse(DataGridView1.Rows(e.RowIndex).Cells(3).Value)
    TextBox5.Text = DataGridView1.Rows(e.RowIndex).Cells(4).Value.ToString
End Sub

```

```

Sub ActualizarGrilla()
    DAdaptador.Fill(DSet)
    AsignarPK()
    DataGridView1.DataSource = DSet.Tables(0)
    For j As Integer = 0 To DSet.Tables(0).Rows.Count - 1
        DataGridView1.Rows(j).DefaultCellStyle.ForeColor = Color.Black
    Next
End Sub

```

```

Sub LimpiarDatos()
    TextBox1.Text = ""
    TextBox1.ReadOnly = False
    TextBox2.Text = ""
    TextBox3.Text = ""
    TextBox4.Text = ""
    TextBox5.Text = ""
End Sub

```

```
Sub AsignarPK()  
    DSet.Tables(0).Columns(0).Unique = True  
    DSet.Tables(0).Columns(0).AllowDBNull = False  
    Dim PKcol() As DataColumn = {DSet.Tables(0).Columns(0)}  
    DSet.Tables(0).PrimaryKey = PKcol  
End Sub
```

```
End Class
```

Uso de Connection, Varios DataAdapter relacionados, Procedimientos Almacenados, CommandBuilder, DataTable, Grilla

The screenshot shows a Windows application titled "Video Club" with three main sections:

- Socios:** Includes input fields for ID_Socio, Nombre, Apellido, and Fecha Nacimiento. Below these are buttons for Alta, Baja, and Modifica. A search bar labeled "Busqueda por Nombre" is also present.
- Peliculas:** Includes input fields for ID_Peli, Nombre, Valor, and Stock. Below these are buttons for Alta, Baja, and Modifica. A search bar labeled "Busqueda por Nombre" is also present.
- Pelis Alquiladas:** Includes input fields for ID_Socio, ID_Peli, Fecha de Retiro, Cantidad de Dias, and Devuelta. Below these are buttons for Alquilar Pelicula and Reincorporar Pelicula.

Imports System.Data

Imports System.Data.SqlClient

Public Class Ejercicio_7

#Region "Campos"

Dim Conn As New SqlConnection("Data Source=.;Initial
Catalog=VideoClub;Integrated Security=True")

Dim DAdaptadorSocio As SqlDataAdapter = New SqlDataAdapter("Select * from
Socio", Conn)

Dim DAdaptadorPelicula As SqlDataAdapter = New SqlDataAdapter("select * from
Pelicula", Conn)

Dim DAdaptadorAlquiler As SqlDataAdapter = New SqlDataAdapter("Select * from
Alquiler", Conn)

Dim DSetSocio As New DataSet

Dim DSetPelicula As New DataSet

Dim DSetAlquiler As New DataSet

#End Region

#Region "Eventos del Form"

```
Private Sub Ejercicio_7_Load(ByVal sender As Object, ByVal e As  
System.EventArgs) Handles Me.Load  
    ActualizarGrilla()  
End Sub
```

```
Private Sub DataGridView1_CellClick(ByVal sender As Object, ByVal e As  
System.Windows.Forms.DataGridViewCellEventArgs) Handles  
DataGridView1.CellClick  
    Try  
        Label1.Text = DataGridView1.Rows(e.RowIndex).Cells(0).Value.ToString  
        TextBox1.Text = DataGridView1.Rows(e.RowIndex).Cells(1).Value.ToString  
        TextBox2.Text = DataGridView1.Rows(e.RowIndex).Cells(2).Value.ToString  
        TextBox3.Text =  
DateTime.Parse(DataGridView1.Rows(e.RowIndex).Cells(3).Value.ToString)  
    Catch ex As Exception  
    End Try  
End Sub
```

```
Private Sub DataGridView2_CellClick(ByVal sender As Object, ByVal e As  
System.Windows.Forms.DataGridViewCellEventArgs) Handles  
DataGridView2.CellClick  
    Try  
        Label10.Text = DataGridView2.Rows(e.RowIndex).Cells(0).Value.ToString  
        TextBox4.Text = DataGridView2.Rows(e.RowIndex).Cells(1).Value.ToString  
        TextBox5.Text = DataGridView2.Rows(e.RowIndex).Cells(2).Value.ToString  
        TextBox6.Text = DataGridView2.Rows(e.RowIndex).Cells(3).Value.ToString  
    Catch ex As Exception  
    End Try  
End Sub
```



```

Private Sub DataGridView3_CellClick(ByVal sender As Object, ByVal e As
System.Windows.Forms.DataGridViewCellEventArgs) Handles
DataGridView3.CellClick
    Try
        For Each item As DataGridViewRow In DataGridView1.Rows
            If item.Cells(0).Value =
DataGridView3.Rows.Item(e.RowIndex).Cells(0).Value Then
                item.Selected = True
            End If
        Next
        For Each item As DataGridViewRow In DataGridView2.Rows
            If item.Cells(0).Value =
DataGridView3.Rows.Item(e.RowIndex).Cells(1).Value Then
                item.Selected = True
            End If
        Next
        Label17.Text = DataGridView3.Rows(e.RowIndex).Cells(0).Value.ToString
        Label19.Text = DataGridView3.Rows(e.RowIndex).Cells(1).Value.ToString
        TextBox9.Text =
DateTime.Parse(DataGridView3.Rows(e.RowIndex).Cells(2).Value)
        TextBox10.Text = DataGridView3.Rows(e.RowIndex).Cells(3).Value.ToString
        TextBox11.Text = DataGridView3.Rows(e.RowIndex).Cells(4).Value.ToString
        Label1.Text = Me.DataGridView3.Rows(e.RowIndex).Cells(0).Value
        Label10.Text = Me.DataGridView3.Rows(e.RowIndex).Cells(1).Value
    Catch ex As Exception
    End Try

End Sub

Private Sub TextBox7_TextChanged(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles TextBox7.TextChanged
    Dim DView As New DataView(DSetSocio.Tables(0))
    DView.RowFilter = "Nom_Socio Like '" & TextBox7.Text & "%'"
    DataGridView1.DataSource = DView
End Sub

```

```
Private Sub TextBox8_TextChanged(ByVal sender As System.Object, ByVal e As  
System.EventArgs) Handles TextBox8.TextChanged
```

```
    Dim DView As New DataView(DSetSocio.Tables(1))  
    DView.RowFilter = "Nom_Peli Like '" & TextBox8.Text & "%"  
    DataGridView2.DataSource = DView  
End Sub
```

```
#End Region
```

```
#Region "SUBS"
```

```
Sub Datagrids()  
    DataGridView1.SelectionMode = DataGridViewSelectionMode.FullRowSelect  
    DataGridView1.EditMode = DataGridViewEditMode.EditProgrammatically  
    DataGridView2.SelectionMode = DataGridViewSelectionMode.FullRowSelect  
    DataGridView2.EditMode = DataGridViewEditMode.EditProgrammatically  
    DataGridView3.SelectionMode = DataGridViewSelectionMode.FullRowSelect  
    DataGridView3.EditMode = DataGridViewEditMode.EditProgrammatically  
    DataGridView1.MultiSelect = False  
    DataGridView2.MultiSelect = False  
    DataGridView3.MultiSelect = False  
End Sub
```

```
Sub asdf(ByVal dtsocio As DataTable, ByVal dtPeli As DataTable, ByVal dtAlquiler  
As DataTable)
```

```
    For i = 0 To dtsocio.Columns.Count - 1  
        DataGridView1.AutoSizeColumn(i)  
    Next  
    For j = 0 To dtPeli.Columns.Count - 1  
        DataGridView2.AutoSizeColumn(j)  
    Next  
    For k = 0 To dtAlquiler.Columns.Count - 1  
        DataGridView3.AutoSizeColumn(k)  
    Next  
End Sub
```

```
Sub ActualizarGrilla()
```

Datagrids()

```
If DSetSocio.Tables.Contains(0) Then
    DSetSocio.Tables.Remove(0)
Else
    DSetSocio.Tables.Add(0)
    DAdaptadorSocio.Fill(DSetSocio)
End If
```

```
Dim dtSocio = New DataTable()
DAdaptadorSocio.Fill(dtSocio)
DataGridView1.DataSource = dtSocio
```

```
Dim dtPeli = New DataTable()
DAdaptadorPelicula.Fill(dtPeli)
DSetPelicula.Tables.Add(dtPeli)
DataGridView2.DataSource = DSetPelicula.Tables(0)
```

```
Dim dtAlquiler = New DataTable()
DAdaptadorAlquiler.Fill(dtAlquiler)
If DSetAlquiler.Tables.Contains(0) Then
    DSetAlquiler.Tables.Remove(0)
Else
    DSetAlquiler.Tables.Add(0)
    DAdaptadorAlquiler.Fill(DSetAlquiler)
End If
DataGridView3.DataSource = dtAlquiler
```

```
asdf(dtSocio, dtPeli, dtAlquiler)
ActualizarBD()
End Sub
```

```
Sub ActualizarBD()
    Dim CBPeli As New SqlCommandBuilder(DAdaptadorPelicula)
    DAdaptadorPelicula.InsertCommand = CBPeli.GetInsertCommand
    DAdaptadorPelicula.DeleteCommand = CBPeli.GetDeleteCommand
```

```

DAdaptadorPelicula.UpdateCommand = CBPeli.GetUpdateCommand
DAdaptadorPelicula.Update(DSetPelicula.Tables(0))
Dim CBSocio As New SqlCommandBuilder(DAdaptadorSocio)
DAdaptadorSocio.InsertCommand = CBSocio.GetInsertCommand
DAdaptadorSocio.DeleteCommand = CBSocio.GetDeleteCommand
DAdaptadorSocio.UpdateCommand = CBSocio.GetUpdateCommand
DAdaptadorSocio.Update(DSetSocio.Tables(0))
End Sub
#End Region

```

```

#Region "ABM SOCIO"

```

```

Private Sub ALTASOCIO(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button1.Click
    DAdaptadorSocio.Fill(DSetSocio)
    Dim c As Integer = 0
    For Each it As DataRow In DSetSocio.Tables(0).Rows
        If c < it.Item(0) Then
            c = it.Item(0)
        End If
    Next
    If DSetSocio.Tables(0).Rows.Count = 0 Then
        MsgBox(DSetSocio.Tables(0).Rows.Count)
        c = 0
    End If
    Dim dr As DataRow = DSetSocio.Tables(0).NewRow()
    dr(0) = Convert.ToInt16(c + 1)
    dr(1) = TextBox1.Text
    dr(2) = TextBox2.Text
    dr(3) = Convert.ToDateTime(TextBox3.Text)
    DSetSocio.Tables(0).Rows.Add(dr)
    ActualizarGrilla()
    'Try
    ' Dim comm = New SqlCommand("spAltaSocio", Conn)
    ' Dim c As Integer = 0
    ' For Each it As DataRow In DSetSocio.Tables(0).Rows
    '     If c < it.Item(0) Then

```

```

        '      c = it.Item(0)
    '      End If
    '      Next
    '      If DSetSocio.Tables(0).Rows.Count = 0 Then
    '          c = 0
    '      End If
    '      comm.CommandType = CommandType.StoredProcedure
    '      comm.Parameters.Add("@Socio_ID", SqlDbType.Int).Value = c + 1
    '      comm.Parameters.Add("@SOCIO_NOMBRE", SqlDbType.NVarChar).Value =
    TextBox1.Text
    '      comm.Parameters.Add("@SOCIO_APELLIDO", SqlDbType.NVarChar).Value
    = TextBox2.Text
    '      comm.Parameters.Add("@SOCIO_FECHA_NAC", SqlDbType.Date).Value =
    Convert.ToDateTime(TextBox3.Text)
    '      Conn.Open()
    '      comm.ExecuteNonQuery()
    '      Conn.Close()
    'Catch ex As Exception
    '      MsgBox(ex.Message)
    'End Try
End Sub

```

Private Sub BAJASOCIO(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button2.Click

```

    Try
        Dim comm = New SqlCommand("spBajaSocio", Conn)
        comm.CommandType = CommandType.StoredProcedure
        comm.Parameters.Add("@Socio_ID", SqlDbType.Int).Value = Label1.Text
        Conn.Open()
        comm.ExecuteNonQuery()
        Conn.Close()
    Catch ex As Exception
        MsgBox(ex.Message)
    End Try
    ActualizarGrilla()
End Sub

```

Private Sub MODIFICASOCIO(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button3.Click

```
Try
    Dim comm = New SqlCommand("spModificaSocio", Conn)
    comm.CommandType = CommandType.StoredProcedure
    comm.Parameters.Add("@Socio_ID", SqlDbType.Int).Value = Label1.Text
    comm.Parameters.Add("@SOCIO_NOMBRE", SqlDbType.NVarChar).Value =
TextBox1.Text
    comm.Parameters.Add("@SOCIO_APELLIDO", SqlDbType.NVarChar).Value =
TextBox2.Text
    comm.Parameters.Add("@SOCIO_FECHA_NAC", SqlDbType.Date).Value =
Convert.ToDateTime(TextBox3.Text)
    Conn.Open()
    comm.ExecuteNonQuery()
    Conn.Close()
Catch ex As Exception
    MsgBox(ex.Message)
End Try
ActualizarGrilla()
End Sub
#End Region
```

#Region "ABM PELICULA"

```
Private Sub ALTAPELICULA(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button6.Click
    DAdaptadorPelicula.Fill(DSetPelicula)
    Dim c As Integer = 0
    For Each it As DataRow In DSetPelicula.Tables(0).Rows
        If c < it.Item(0) Then
            c = it.Item(0)
        End If
    Next
    If DSetPelicula.Tables(0).Rows.Count = 0 Then
        c = 0
    End If
```

```

Dim dr As DataRow = DSetPelicula.Tables(0).NewRow()
    dr(0) = c + 1
    dr(1) = TextBox4.Text
    dr(2) = TextBox5.Text
    dr(3) = TextBox6.Text
    DSetPelicula.Tables(0).Rows.Add(dr)
    ActualizarGrilla()
End Sub

```

```

Private Sub BAJAPELICULA(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button5.Click
    Dim x As Integer = Label10.Text
    For Each i As DataRow In DSetPelicula.Tables(0).Rows
        If i.Item(0) = x Then
            i.Delete()
        End If
    Next
    ActualizarGrilla()
End Sub

```

```

Private Sub MODIFICAPELICULA(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button4.Click
    Dim x As Integer = Label10.Text
    For Each i As DataRow In DSetPelicula.Tables(0).Rows
        If i.Item(0) = x Then
            i.Item(1) = TextBox4.Text
            i.Item(2) = TextBox5.Text
            i.Item(3) = TextBox6.Text
        End If
    Next
    ActualizarGrilla()
End Sub
#End Region

#Region "Alquileres Peliculas"

```

```

Private Sub ALQUILAR(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button7.Click
    Dim tr As SqlTransaction
    Dim comm = New SqlCommand("spAlquiler", Conn)
    comm.CommandType = CommandType.StoredProcedure
    comm.Parameters.Add("@Socio_ID", SqlDbType.Int).Value =
Convert.ToInt16(TextBox("ingrese el codigo de socio:"))
    comm.Parameters.Add("@PELI_ID", SqlDbType.Int).Value =
Convert.ToInt16(TextBox("ingrese el codigo de pelicula:"))
    comm.Parameters.Add("@Fecha_Ret", SqlDbType.DateTime).Value =
Convert.ToDateTime(TextBox("ingrese fecha de alquiler:"))
    comm.Parameters.Add("@Cantidad_Dias", SqlDbType.Int).Value =
Convert.ToInt16(TextBox("ingrese cantidad de dias de alquiler:"))
    comm.Parameters.Add("@PELI_DEVUELTA", SqlDbType.Bit).Value = False
    Conn.Open()
    tr = Conn.BeginTransaction
    comm.Transaction = tr
    Try
        comm.ExecuteNonQuery()
        tr.Commit()
    Catch ex As Exception
        tr.Rollback()
    End Try
    ActualizarGrilla()
End Sub

```

```

Private Sub DEVOLUCION(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button8.Click
    Try
        If DataGridView3.SelectedRows.Item(0).Cells.Item(4).Value = False Then
            Dim tr As SqlTransaction
            Dim comm As New SqlCommand("spDevolucion", Conn)
            comm.CommandType = CommandType.StoredProcedure
            comm.Parameters.Add("@Socio_ID ", SqlDbType.Int).Value =
Convert.ToInt16(Label17.Text)
            comm.Parameters.Add("@PELI_ID", SqlDbType.Int).Value =
Convert.ToInt16(Label19.Text)

```



```

comm.Parameters.Add("@PELI_DEVUELTA", SqlDbType.Bit).Value = True
    Conn.Open()
    tr = Conn.BeginTransaction
    comm.Transaction = tr
    Try
        comm.ExecuteNonQuery()

        tr.Commit()
    Catch ex As Exception
        tr.Rollback()
    Finally
        Conn.Close()
    End Try
Else
    MsgBox("la pelicula ya fue devuelta")
End If
    ActualizarGrilla()
    Catch ex As Exception
    End Try
End Sub

#End Region

End Class

```

Uso de Connection, Varios DataAdapter relacionados, Procedimientos Almacenados, CommandBuilder, DataTable, Grilla , XML

Imports System.Data

Imports System.Data.SqlClient

Imports System.Xml

Imports System.IO

Public Class Ejercicio_8_y_9

Dim Conn As New SqlConnection("Data Source=.;Initial
Catalog=GESTION;Integrated Security=True")

Dim DAdaptador As New SqlDataAdapter("Select * from Cliente", Conn)

Dim DSet As New DataSet

Dim x As Integer

Dim s As String

Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button1.Click

Try

DAdaptador.Fill(DSet)

Dim c As Integer = 0

For Each it As DataRow In DSet.Tables(0).Rows

```

        If c < it.Item(0) Then
            c = it.Item(0)
        End If
    Next
    Dim DRow As DataRow = DSet.Tables(0).NewRow
    DRow.Item(0) = c + 1
    DRow.Item(1) = TextBox2.Text
    DRow.Item(2) = TextBox3.Text
    DRow.Item(3) = TextBox4.Text
    DRow.Item(4) = TextBox5.Text
    DSet.Tables(0).Rows.Add(DRow)
    For j As Integer = 0 To DSet.Tables(0).Rows.Count - 1
        If DSet.Tables(0).Rows.Item(j).RowState = DataRowState.Added Then
            DataGridView1.Rows(j).DefaultCellStyle.ForeColor = Color.Green
        End If
    Next
    Catch ex As Exception
    End Try

End Sub

```

```

Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button2.Click
    x = TextBox1.Text
    For Each i As DataRow In DSet.Tables(0).Rows
        If i.Item(0) = x Then
            i.Delete()
        End If
    Next
    LimpiarDatos()
    DataGridView1.Refresh()
End Sub

```

```

Private Sub Button3_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button3.Click
    x = TextBox1.Text

```

```
For Each i As DataRow In DSet.Tables(0).Rows
```

```
    If i.Item(0) = x Then
```

```
        i.Item(1) = TextBox2.Text
```

```
        i.Item(2) = TextBox3.Text
```

```
        i.Item(3) = TextBox4.Text
```

```
        i.Item(4) = TextBox5.Text
```

```
    End If
```

```
Next
```

```
For j As Integer = 0 To DSet.Tables(0).Rows.Count - 1
```

```
    If DSet.Tables(0).Rows(j).RowState = DataRowState.Modified Then
```

```
        DataGridView1.Rows(j).DefaultCellStyle.ForeColor = Color.Red
```

```
    End If
```

```
Next
```

```
LimpiarDatos()
```

```
DataGridView1.Refresh()
```

```
End Sub
```

```
Private Sub Button4_Click(ByVal sender As System.Object, ByVal e As  
System.EventArgs) Handles Button4.Click
```

```
    ActualizarGrilla()
```

```
End Sub
```

```
Private Sub Button5_Click(ByVal sender As System.Object, ByVal e As  
System.EventArgs) Handles Button5.Click
```

```
    Dim CBuilder As New SqlCommandBuilder(DAdaptador)
```

```
    DAdaptador.InsertCommand = CBuilder.GetInsertCommand
```

```
    DAdaptador.DeleteCommand = CBuilder.GetDeleteCommand
```

```
    DAdaptador.UpdateCommand = CBuilder.GetUpdateCommand
```

```
    ActualizarGrilla()
```

```
End Sub
```

```
Private Sub TextBox6_TextChanged(ByVal sender As System.Object, ByVal e As  
System.EventArgs) Handles TextBox6.TextChanged
```

```

Dim DView As New DataView(DSet.Tables(0))
    DView.RowFilter = "CLIE_NOMBRE Like '" & TextBox6.Text & "%'"
    DataGridView1.DataSource = DView
End Sub

Private Sub Form1_Load(ByVal sender As Object, ByVal e As System.EventArgs)
Handles Me.Load
    ActualizarGrilla()

End Sub

Private Sub DataGridView1_CellClick(ByVal sender As Object, ByVal e As
System.Windows.Forms.DataGridViewCellEventArgs) Handles
DataGridView1.CellClick

    TextBox1.Text = DataGridView1.Rows(e.RowIndex).Cells(0).Value.ToString
    TextBox1.ReadOnly = True
    TextBox2.Text = DataGridView1.Rows(e.RowIndex).Cells(1).Value.ToString
    TextBox3.Text = DataGridView1.Rows(e.RowIndex).Cells(2).Value.ToString
    TextBox4.Text =
DateTime.Parse(DataGridView1.Rows(e.RowIndex).Cells(3).Value)
    TextBox5.Text = DataGridView1.Rows(e.RowIndex).Cells(4).Value.ToString
End Sub

Sub ActualizarGrilla()
    DAdaptador.Fill(DSet)
    AsignarPK()
    DataGridView1.DataSource = DSet.Tables(0)
    For j As Integer = 0 To DSet.Tables(0).Rows.Count - 1
        DataGridView1.Rows(j).DefaultCellStyle.ForeColor = Color.Black
    Next
    DataGridView1.SelectionMode = DataGridViewSelectionMode.FullRowSelect
    DataGridView1.EditMode = DataGridViewEditMode.EditProgrammatically
    DataGridView2.SelectionMode = DataGridViewSelectionMode.FullRowSelect

```

```
        DataGridView2.EditMode = DataGridViewEditMode.EditProgrammatically
    End Sub
```

```
Sub LimpiarDatos()
    TextBox1.Text = ""
    TextBox1.ReadOnly = False
    TextBox2.Text = ""
    TextBox3.Text = ""
    TextBox4.Text = ""
    TextBox5.Text = ""
End Sub
```

```
Sub AsignarPK()
    DSet.Tables(0).Columns(0).Unique = True
    DSet.Tables(0).Columns(0).AllowDBNull = False
    Dim PKcol() As DataColumn = {DSet.Tables(0).Columns(0)}
    DSet.Tables(0).PrimaryKey = PKcol
End Sub
```

```
Private Sub Button6_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button6.Click
    Dim DAXML As New SqlDataAdapter("select * from Cliente ", Conn)
    Dim DsXML As New DataSet
    File.Delete("C:\Users\angus\Desktop\LUG\file.xml")
    DAXML.Fill(DsXML, "Cliente")
    DsXML.WriteXml("C:\Users\angus\Desktop\LUG\file.xml")
    DsXML.Clear()
End Sub
```

```
Private Sub Button7_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button7.Click
    Dim DsXML As New DataSet()
    Try
        DsXML.Clear()
        DsXML.ReadXml("C:\Users\angus\Desktop\LUG\file.xml")
        DataGridView2.DataSource = DsXML.Tables(0)
    End Try
End Sub
```

Catch ex As Exception

End Try

End Sub

Private Sub Button8_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button8.Click

Dim ds As New DataSet

Dim dt As New DataTable

Dim dr As DataRow

dt.Columns.Add(New DataColumn("SQL", GetType(String)))

dr = dt.NewRow()

dr.Item(0) = DAdaptador.InsertCommand.CommandText.ToString

dt.Rows.Add(dr)

dr = dt.NewRow()

dr.Item(0) = DAdaptador.DeleteCommand.CommandText.ToString

dt.Rows.Add(dr)

dr = dt.NewRow()

dr.Item(0) = DAdaptador.UpdateCommand.CommandText.ToString

dt.Rows.Add(dr)

ds.Tables.Add(dt)

ds.WriteXml("C:\Users\angus\Desktop\LUG\file_2.xml")

System.Diagnostics.Process.Start("C:\Users\angus\Desktop\LUG\file_2.xml")

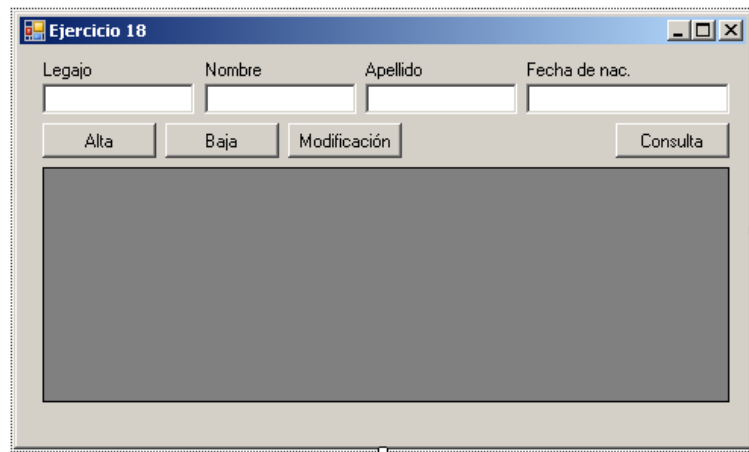
End Sub

End Class

Modelo 3 Capas

Sistema Universidad

Interfaz del Usuario



CAPA UI

Clase ClienteVista

```
Public Class ClienteVista
```

```
    Private _alu As BLL.Alumno
```

```
    Public Property Alumno() As BLL.Alumno
```

```
        Get
```

```
            Return _alu
```

```
        End Get
```

```
        Set(ByVal value As BLL.Alumno)
```

```
            _alu = value
```

```
        End Set
```

```
    End Property
```

```
    Private _aluD As BLL_Dinamica.Alumno_Dinamico
```

```
    Public Property AlumnoD() As BLL_Dinamica.Alumno_Dinamico
```

```
        Get
```

```
            Return _aluD
```

```
        End Get
```

```
        Set(ByVal value As BLL_Dinamica.Alumno_Dinamico)
```



```

    _aluD = value
    End Set
End Property

Sub New()
    Me.Alumno = New BLL.Alumno
    Me.AlumnoD = New BLL_Dinamica.Alumno_Dinamico
End Sub
End Class

```

CAPA BLL

Clase Alumno

```
Public Class Alumno
```

```

    Private _leg As String
    Public Property Legajo() As String
    Get
        Return _leg
    End Get
    Set(ByVal value As String)
        _leg = value
    End Set
End Property

```

```

    Private _nom As String
    Public Property Nombre() As String
    Get
        Return _nom
    End Get
    Set(ByVal value As String)
        _nom = value
    End Set
End Property

```

```

    Private _ape As String
    Public Property Apellido() As String

```

```

    Get
        Return _ape
    End Get
    Set(ByVal value As String)
        _ape = value
    End Set
End Property

Private _fecha As Date
Public Property FechaNac() As Date
    Get
        Return _fecha
    End Get
    Set(ByVal value As Date)
        _fecha = value
    End Set
End Property

Sub New()

End Sub

Sub New(ByVal QueLeg As String, ByVal QueNom As String, ByVal QueApe As
String, ByVal QueFechas As Date)
    Me.Legajo = QueLeg
    Me.Nombre = QueNom
    Me.Apellido = QueApe
    Me.FechaNac = QueFechas
End Sub

End Class

```

CAPA BLL DINAMICO

Clase Alumno_Dinámico

```
Public Class Alumno_Dinamico
```

Implements DAL.IABMC(Of BLL.Alumno)

Private _AluDa As DAL.AlumnoDAL

Public Property AluDat() As DAL.AlumnoDAL

Get

Return _AluDa

End Get

Set(ByVal value As DAL.AlumnoDAL)

_AluDa = value

End Set

End Property

Sub New()

Me.AluDat = New DAL.AlumnoDAL

End Sub

Public Sub Alta(ByVal QueT As BLL.Alumno) Implements DAL.IABMC(Of
BLL.Alumno).Alta

Me.AluDat.Alta(QueT)

End Sub

Public Sub Baja(ByVal QueT As BLL.Alumno) Implements DAL.IABMC(Of
BLL.Alumno).Baja

Me.AluDat.Baja(QueT)

End Sub

Public Sub Consulta(ByVal QueT As BLL.Alumno) Implements DAL.IABMC(Of
BLL.Alumno).Consulta

Me.AluDat.Consulta(QueT)

End Sub

Public Function ConsultaRango(ByVal QueT1 As BLL.Alumno, ByVal QueT2 As
BLL.Alumno) As System.Collections.Generic.List(Of BLL.Alumno) Implements
DAL.IABMC(Of BLL.Alumno).ConsultaRango

Return Me.AluDat.ConsultaRango(QueT1, QueT2)

End Function

```

Public Sub Modificacion(ByVal QueT As BLL.Alumno) Implements DAL.IABMC(Of
BLL.Alumno).Modificacion
    Me.AluDat.Modificacion(QueT)
End Sub
End Class

```

CAPA DAL

Interfaz IABMC

```

Public Interface IABMC(Of T)

    Sub Alta(ByVal QueT As T)
    Sub Baja(ByVal QueT As T)
    Sub Modificacion(ByVal QueT As T)
    Sub Consulta(ByVal QueT As T)
    Function ConsultaRango(ByVal QueT1 As T, ByVal QueT2 As T) As List(Of T)

End Interface

```

Clase AlumnoDal

```

Public Class AlumnoDAL

    Implements IABMC(Of BLL.Alumno)

    Public Sub Alta(ByVal QueT As BLL.Alumno) Implements IABMC(Of
BLL.Alumno).Alta
        Dim dt As DataTable = Servicios_DAL.Comando.ObjStructureTable("Select * from
Alumno")
        Dim dr As DataRow = dt.NewRow
        dr.Item(0) = QueT.Logajo

```

```

dr.Item(1) = QueT.Nombre
dr.Item(2) = QueT.Apellido
dr.Item(3) = QueT.FechaNac
dt.Rows.Add(dr)
Servicios_DAL.Comando.Actualizar("Select * from Alumno", dt)
End Sub

```

```

Public Sub Baja(ByVal QueT As BLL.Alumno) Implements IABMC(Of
BLL.Alumno).Baja
    Dim dt As DataTable = Servicios_DAL.Comando.ObjDataTable("Select * from
Alumno where Leg=" + QueT.Legajo + "")

    If dt.Rows.Count > 0 Then
        dt.Rows(0).Delete()
        Servicios_DAL.Comando.Actualizar("Select * from Alumno", dt)
    End If
End Sub

```

```

Public Sub Consulta(ByVal QueT As BLL.Alumno) Implements IABMC(Of
BLL.Alumno).Consulta
    Dim dt As DataTable = Servicios_DAL.Comando.ObjDataTable("Select * from
Alumno where Leg = " + QueT.Legajo + "")

    If dt.Rows.Count > 0 Then
        QueT.Nombre = dt.Rows(0).Item(1)
        QueT.Apellido = dt.Rows(0).Item(2)
        QueT.FechaNac = dt.Rows(0).Item(3)
    End If
End Sub

```

```

Public Function ConsultaRango(ByVal QueT1 As BLL.Alumno, ByVal QueT2 As
BLL.Alumno) As System.Collections.Generic.List(Of BLL.Alumno) Implements
IABMC(Of BLL.Alumno).ConsultaRango
    Dim dt As DataTable = Servicios_DAL.Comando.ObjDataTable("Select * from
Alumno where Leg>=" + QueT1.Legajo + " and Leg<=" + QueT2.Legajo + "")

    Dim lista As New List(Of BLL.Alumno)

    If dt.Rows.Count > 0 Then
        Dim dr As DataRow = dt.NewRow

```

```

    For Each r As DataRow In dt.Rows
        lista.Add(New BLL.Alumno(dr.Item(0), dr.Item(1), dr.Item(2), dr.Item(3)))
    Next
End If
Return lista
End Function

Public Sub Modificacion(ByVal QueT As BLL.Alumno) Implements IABMC(Of
BLL.Alumno).Modificacion
    Dim dt As DataTable = Servicios_DAL.Comando.ObjDataTable("Select * from
Alumno where Leg=" + QueT.Legajo + "")
    If dt.Rows.Count > 0 Then
        dt.Rows(0).Item(1) = QueT.Nombre
        dt.Rows(0).Item(2) = QueT.Apellido
        dt.Rows(0).Item(3) = QueT.FechaNac
        Servicios_DAL.Comando.Actualizar("Select * from Alumno", dt)
    End If
End Sub
End Class

```

CAPA de SERVICIOS DAL

Clase Comando

Imports System.Data

Imports System.Data.SqlClient

Public Class Comando

Private Shared _com As SqlCommand

Shared Function ObjComando(ByVal SelectCommand As String, ByVal QueCon As SqlConnection) As SqlCommand

_com = New SqlCommand

_com.CommandText = SelectCommand

_com.CommandType = CommandType.Text

_com.Connection = QueCon

Return _com

End Function

Shared Function ObjDataTable(ByVal SelectCommand As String) As DataTable

Dim da As New SqlDataAdapter(ObjComando(SelectCommand, Conexion.ObjConexion))

Dim dt As New DataTable

da.Fill(dt)

Return dt

End Function

Shared Function ObjStructureTable(ByVal SelectCommand As String) As DataTable

Dim da As New SqlDataAdapter(ObjComando(SelectCommand, Conexion.ObjConexion))

Dim dt As New DataTable

da.FillSchema(dt, SchemaType.Mapped)

Return dt

End Function

Shared Sub Actualizar(ByVal SelectCommand As String, ByVal DT As DataTable)

Dim da As New SqlDataAdapter(ObjComando(SelectCommand, Conexion.ObjConexion))

Dim cb As New SqlCommandBuilder(da)

```
da.InsertCommand = cb.GetInsertCommand
da.DeleteCommand = cb.GetDeleteCommand
da.UpdateCommand = cb.GetUpdateCommand
da.Update(DT)
End Sub
End Class
```

Clase Conexion

```
Imports System.Data
```

```
Imports System.Data.SqlClient
```

```
Public Class Conexion
```

```
Private Shared _con As SqlConnection
```

```
Shared Function ObjConexion() As SqlConnection
```

```
    _con = New SqlConnection("Data Source=.;Initial Catalog=Prueba;Integrated  
Security=True")
```

```
    Return _con
```

```
End Function
```

```
End Class
```


Public Class Form1

```
Private Sub btnSal_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
    Me.Close()
End Sub
```

Dim alu As New ClienteVista

```
Private Sub btnAlta_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles btnAlta.Click
    If Me.TxtLegajo1.Valido = True And Me.TxtFechaNac1.Valido = True Then
        alu.Alumno.Legajo = Me.TxtLegajo1.Text
        alu.Alumno.Nombre = Me.txtNom.Text
        alu.Alumno.Apellido = Me.txtApe.Text
        alu.Alumno.FechaNac = Convert.ToDateTime(Me.TxtFechaNac1.Text)
        alu.AlumnoD.Alta(alu.Alumno)
    Else
        MsgBox("Mal ingresado un dato!")
    End If
End Sub
```

```
Private Sub btnBaja_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles btnBaja.Click
```

```
alu.Alumno.Legajo = Me.TxtLegajo1.Text
alu.AlumnoD.Baja(alu.Alumno)
refrescarGrilla()
```

End Sub

```
Private Sub btnMod_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
```

Handles btnMod.Click

```
alu.Alumno.Legajo = Me.TxtLegajo1.Text
alu.Alumno.Nombre = Me.txtNom.Text
alu.Alumno.Apellido = Me.txtApe.Text
alu.Alumno.FechaNac = Convert.ToDateTime(Me.TxtFechaNac1.Text)
alu.AlumnoD.Modificacion(alu.Alumno)
refrescarGrilla()
```

End Sub

```
Private Sub btnCon_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
```

Handles btnCon.Click

```
alu.Alumno.Legajo = Me.TxtLegajo1.Text
alu.AlumnoD.Consulta(alu.Alumno)
```

```
Me.TxtLegajo1.Text = ""
```

```
Me.txtNom.Text = ""
```

```
Me.txtApe.Text = ""
```

```
Me.TxtFechaNac1.Text = Nothing
```

```
Dim ListaUnAlumno As New List(Of Object)
```

```
ListaUnAlumno.Add(alu.Alumno)
```

```
DataGridView1.DataSource = ListaUnAlumno
```

End Sub

```
Private Sub Form1_Load(ByVal sender As Object, ByVal e As System.EventArgs) Handles
```

Me.Load

```
refrescarGrilla()
```

```
DataGridView1.EditMode = DataGridViewEditMode.EditProgrammatically
```

```
DataGridView1.SelectionMode = DataGridViewSelectionMode.FullRowSelect
```

End Sub

Sub refrescarGrilla()

DataGridView1.DataSource = Servicios_DAL.Comando.ObjDataTable("Select * from
Alumno")

End Sub

Private Sub DataGridView1_CellClick(ByVal sender As Object, ByVal e As
System.Windows.Forms.DataGridViewCellEventArgs) Handles DataGridView1.CellClick
TxtLegajo1.Text = DataGridView1.Rows(e.RowIndex).Cells(0).Value

End Sub

End Class

Bibliografía :

Visual basic 2005 Express de Jorge Serrano Pérez

Visual Basic Net – Manual de Programación

Carpeta de Tps del alumno Carlos Mentaberry