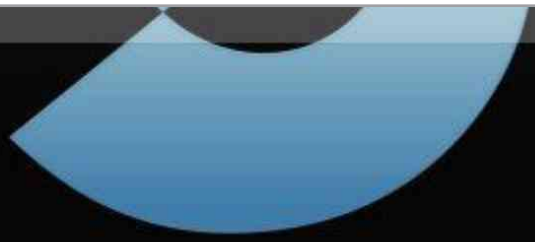
CONEXIÓN SQL SERVER & C#

MANUAL PARA PRINCIPIANTES.



LLAMAS & ZAVALA

Manual para principiantes.

Raúl Antonio Zavala López.

Roberto Llamas Avalos.

Smashwords Edition, Published by Raúl Antonio Zavala López at Smashwords.

Tabla de contenidos.

Agradecimientos.

Nos gustaría agradecer a todas aquellas personas que compraron esta publicación, con el fin de conocer un poco más en que consiste el entorno de establecer una conexión entre SQL SERVER y C#.

Regalo gratis.

El ejemplo desarrollado a lo largo del manual tanto el (Proyecto como el archivo SQL) es posible descargarlos gratuitamente a través del vínculo: http://www.mediafire.com/?ttnuii569fl78a9

Introducción.

El siguiente texto tiene por objetivo principal informar al lector, sea este un conocedor a fondo o no de los sistemas computacionales.

De cómo establecer una comunicación básica entre un sistema gestor de base de datos como lo es (SQL SERVER) en general y un programa desarrollado en lenguaje de programación C#.

A fin de tener la capacidad básica de crear una base de datos principalmente.

Seguido de desarrollar una interface de entrada, es decir un (Programa) por medio del cual se puedan introducir los datos hacia la base.

Para posteriormente realizar manipulaciones de los datos introducidos, es decir poder realizar: ALTAS, BAJAS, MODIFICACIONES Y CONSULTAS.

Sin morir en el intento o simplemente recaer en la frustración de no lograr hacerlo.

De esto es de lo que se trata prácticamente, por ello el lector debe tener al menos una idea de lo que es SQL, SQL SERVER, y C#. Es decir, cómo funcionan estos. Así como de programación en general.

Pues el objetivo de la publicación no es de ser un tratado formal de SQL SERVER, SQL y C# en general, sino de ser una retroalimentación para una necesidad inmediata.

Cabe destacar que la manera en la cual se establece la conexión a la base de datos en el ejemplo del manual puede o no parecer la más apta para algunos lectores, reiterando que solo es un ejemplo... Y la noción central puede moldearse y adaptarse a las necesidades de cada quien.

Por ello, tal manual puede leerse tan solo en una hora... Y tener la capacidad de realizar lo comentado.

Volver al inicio.

~~~

Información adicional.

Entre las herramientas utilizadas a lo largo del curso se encuentran paquetes como:

1.- Microsoft SQL Server 2008 R2 Express

2.- Visual C# Express o más reciente Visual Studio Express para Desktop

Ambos disponibles para ser descargados completamente gratuitamente a través de:

http://www.microsoft.com/es-es/download/details.aspx?id=23650

En el caso del **(Microsoft SQL Server 2008 R2 Express)** o:

http://www.microsoft.com/visualstudio/eng/

pro-ducts/sual-stu-dio-press-pro-ducts

En el caso del **(Visual C# Express o más reciente Visual Studio Express para Desktop)**. Dichos sitios nos proporcionan la modalidad de descargar las herramientas tanto en español como en cualesquier otro idioma.

Estas son las herramientas con las cuales se estará trabajando a lo largo del manual, es decir, es indispensable que se tengan instaladas correctamente.

Para cualquier problema referente a la instalación o en sí mismo como instalarlas es posible encontrar mucha información de por medio en la Web, he incluso el instalador de cada una de las herramientas es muy intuitivo.

Por último cabe anexar que de los dos productos mencionados en el punto 2, utilizaremos **Visual Studio Express para Desktop** por ser la versión más reciente.

Nota: Enlaces Web, verificados el día (12/03/13) posibles problemas de acceso son completamente ajenos a esta publicación.

Fuente de las herramientas en general: http://www.microsoft.com

Si usted encuentra algún error en el manual tal como (Faltas de ortografía) no dude en comunicárnoslo al siguiente correo: soportemanualparaprincipiantes@gmail.com a fin de mejorar la experiencia del usuario.

Volver al inicio.

~~~

Identificación.

Una vez instaladas las herramientas comentadas, necesitamos estar consciente de lo que se va a realizar.

Es decir, como crear la base de datos a utilizar para posteriormente continuar con el proceso de establecer conexión entre C# y SQL SERVER.

Etapas:

1.- Creamos la base de datos a partir de **Microsoft SQL Server 2008 R2 Express**.

2.- Creamos la interfaz de entrada en **Visual Studio Express para Desktop** usando el lenguaje de programación C#.

Realizamos el código correspondiente tanto de programación básica como de las instrucciones de manipulación en SQL.

3.- Procedemos a ejecutar la interfaz (programa) e ingresamos datos a la base.

Todo esto resumido en 3 etapas, cabe destacar que para aprovechar la eficiencia y reutilización de código es posible que se utilicen algunas nociones de P.O.O *(Programación orientada a objetos)*.

Aun así, esto no sugiere que el lector deba conocer a fondo la P.O.O pues se indicara que se está realizado a medida que se utilice.

Nota: En la ejecución del (Proyecto) es necesario que se cree en primer instancia la base de datos por medio del archivo SQL, pues de lo contrario no funcionará el ejemplo.

Volver al inicio.

~~~

Etapa 1.

Para crear una base de datos por medio de SQL, basta con escribir un archivo SQL... Mediante un editor de texto como (Notepad, MacVim, etc.) guardarlo en extensión *.sql* y ejecutarlo en algún gestor de base de datos.

Nosotros utilizaremos *SQL SERVER Management Studio* incluido al momento de instalar Microsoft SQL SERVER R2 Express.

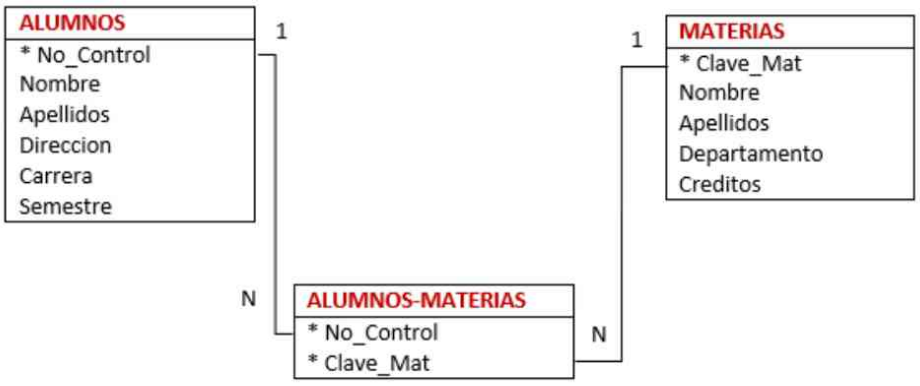
Dicho programa nos sirve como IDE en lo que a una gestión de base de datos se refiere, pues nos permite ejecutar instrucción por instrucción al igual que otros programas como Visual Studio, etc.

Es precisamente este medio en el cual nos dispondremos a crear la base de datos (B.D).

La base de datos para el ejemplo de este manual consistirá en algo muy sencillo que únicamente ilustre el objetivo que se desea.

Pues no se busca implementar una base de datos muy compleja que podría confundir al lector, si este no posee un nivel medio en cuestión de creación de base de datos.

Por tanto manejamos el siguiente modelo:



Nota: Cabe destacar que el modelo relacional utilizado a lo largo del *(Ejemplo)* en lo que se refiere a la base de datos a crear utiliza 3 tablas pues una se encuentra compuesta de las relaciones entre las tablas (ALUMNOS, MATERIAS).. Más no utilizaremos la que corresponde a estas relaciones,

únicamente utilizaremos las principales.

Se creo esta última a fin de tener el modelo relacional completo, si el (Lector) quiere trabajar sobre ella puede hacerlo con total libertad. *Pero en el ejemplo no se utilizará.*

Tal modelo ejemplifica ya las relaciones entre dos tablas, en este caso la relación (ALUMNO-MATERIAS).

Bueno para iniciar a crear la base por medio de SQL y SQL SERVER.

>>Ejecutamos *SQL SERVER Management Studio* e iniciamos sesión de acuerdo a la configuración que colocamos al momento de instalar **Microsoft SQL Server 2008 R2 Express** ya sea por autentificación de Windows o autentificación de SQL SERVER.

Una vez dentro, del *Management Studio* damos clic al botón (Nueva consulta), tal como se muestra en la imagen siguiente:

He iniciamos ya ahora si la programación para crear la base de datos:

Comenzamos indicando la creación de la base, así mismo que utilizaremos esta base de datos en el gestor y no la base de datos (master) por default.

Código en SQL:

```
--Ejercicio (Desarrollado) para (CONEXION SQL SERVER Y C#)
--CREACION DE B.D
Create database BD_1
Use BD_1
Go
```

Una vez realizado lo comentado, ejecutamos instrucción por instrucción el código para crear la base de datos.

Mediante el botón (Ejecutar) con un signo de exclamación, ubicado en la barra de herramientas al mismo nivel que el botón (Nueva consulta).

Código en SQL:

--CREACION DE TABLAS.

--ALUMNOS

```
Create table ALUMNOS(
PKNo_Control int primary key,
Nombre char(25),
Apellidos char(25),
Direccion char(40),
Carrera char(30),
Semestre int)
```

--MATERIAS

```
Create table MATERIAS(
PKClave_Mat int primary key,
Nombre char(25),
Creditos int )
```

--ALUMNOS-MATERIAS

```
Create table ALUMNOS_MATERIAS(
FKNo_Control int foreign key(FKNo_Control) references ALUMNOS(PK-
No_Control),
FKClave_Mat int foreign key(FKClave_Mat) references MATERIAS(-
PKClave_Mat),
Primary key (FKNo_Control,FKClave_Mat))
```

Nota: Recomendación iniciar la creación de las tablas a partir de las tablas base como por ejemplo (ALUMNOS, MATERIAS) y no por las tablas derivadas de las relaciones como (ALUMNOS-MATERIAS) pues *SQL SERVER Management Studio* puede indicarnos una alerta de que no se han creado las tablas con las llaves primarias. Esto más que nada porque empleamos relaciones en la tabla (ALUMNOS-MATERIAS).

Una vez programada la base de datos utilizando SQL ejecutamos instrucción

por instrucción el código a excepción del principio que habla sobre la creación de la B.D debido a que ya lo hicimos previamente, a través del mismo botón para verificar que todo el código se encuentre correcto.

De encontrarse todo correcto observaremos el siguiente mensaje:

Comandos completados correctamente.

Si el lector desea visualizar las tablas puede hacer uso de las instrucciones de selección correspondientes a cada tabla, como se muestra:

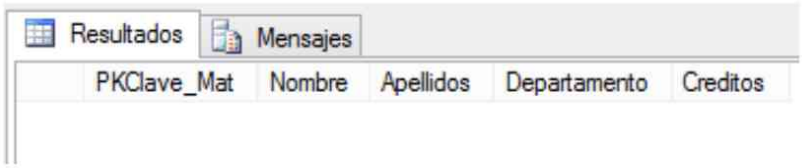
```
--ALUMNOS
Select * from ALUMNOS;
--MATERIAS
Select * from MATERIAS;
--ALUMNOS-MATERIAS
Select * from ALUMNOS_MATERIAS;
```

Dichas instrucciones se ejecutan de igual forma que el código anterior y como resultado *SQL SERVER Management Studio* nos arroga una tabla por (Selección) a la vez.

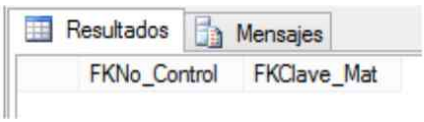
Como se muestra en las siguientes imágenes:

ALUMNOS

MATERIAS



ALUMNOS-MATERIAS



Ahora bien como puede observarse dichas tablas no poseen información alguna, se podría insertar datos mediante la instrucción INSERT de SQL pero

no fuera práctico para un ambiente en el cual se debe estar introduciendo constantemente. Por cuestiones de tiempo y costos.

Es en estos momentos, donde la utilidad de una interfase de entrada (Programa) se ve reflejada pues esta nos permite con mayor rapidez introducir los datos. Como más adelante observaremos.

Volver al inicio.

~~~

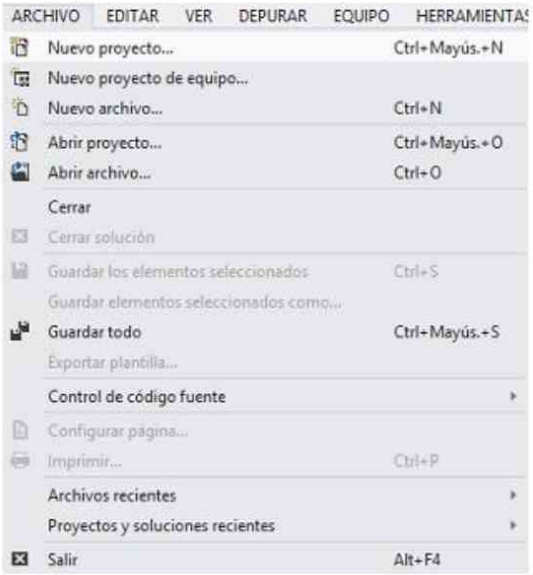
Etapa 2.

Para crear una interfaz de entrada hacia la base de datos, es decir un programa que canalice aquellos datos introducidos por un usuario hacia esta.

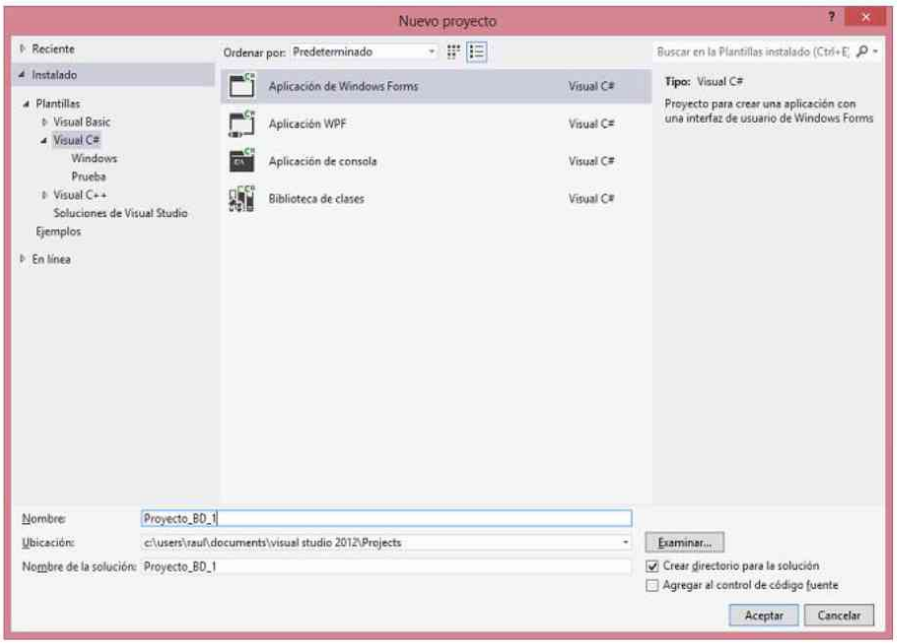
Lo que tenemos que hacer en primera instancia:

1.- Crear un nuevo proyecto en **(Visual Studio Express para Desktop)**.

Para ello tan solo debemos de abrir dicho programa y ubicar en la barra de menús superior (ARCHIVO), y seleccionar (Nuevo proyecto) tal como se muestra en la imagen siguiente:



Una vez elegida dicha opción nos aparecerá una ventana en la cual seleccionaremos el tipo de proyecto y en que lenguaje de programación se realizará, en este caso seleccionaremos (Visual C#) seguido de (Aplicación de Windows Forms), como se muestra en la imagen siguiente:



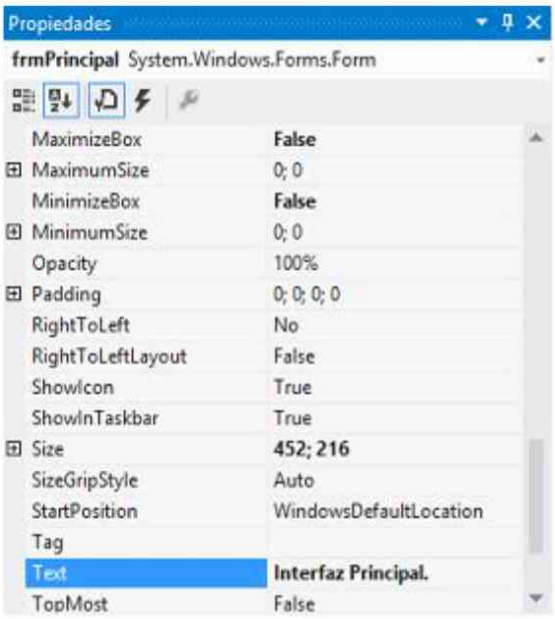
Titularemos para ejemplo de este manual el proyecto como: **Proyecto_BD_1**

Seguido de ello, comenzamos con el proceso de programación básico que ya conocemos.

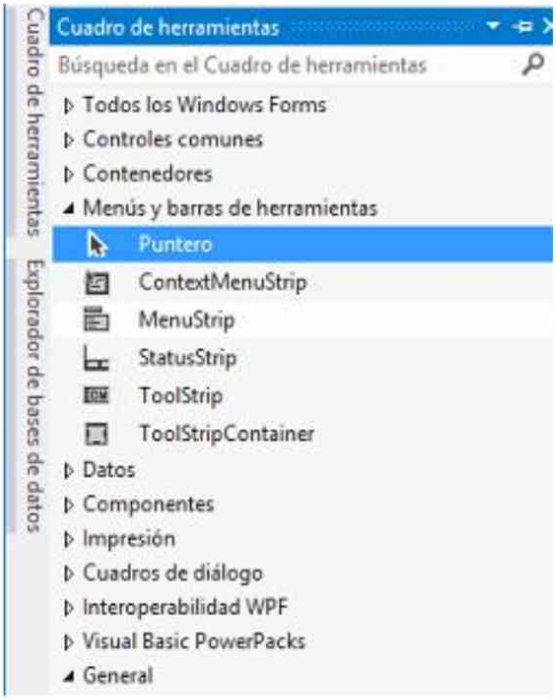
Es decir, titulamos la forma (Principal) como: frmPrincipal.

Así como también establecemos el aspecto *(Color de fondo, Tipo de letra, etc)* todo este tipo de detalles por medio de la barra de propiedades ubicada en la parte derecha de nuestro IDE *(Entorno de desarrollo integrado)* o sea el (Visual Studio).

Como la que se muestra a continuación:

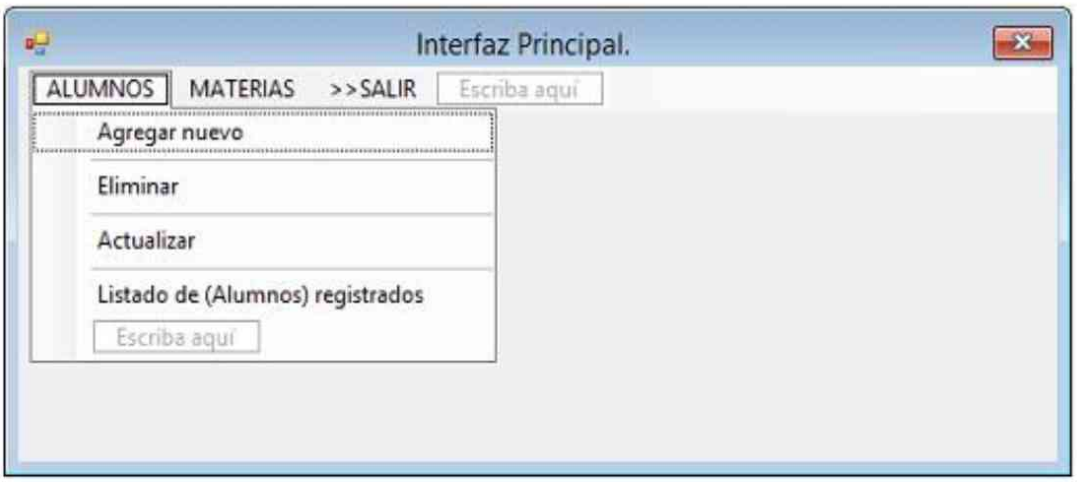


2.- Establecemos un menú a fin de que el usuario que utilice el programa ubique con facilidad las (ALTAS, BAJAS, MODIFICACIONES Y CONSULTAS), esto por medio de la utilización de la herramienta *MenuStrip* ubicada en la parte izquierda del IDE en el *cuadro de herramientas*, como se muestra:



Simplemente seleccionamos la herramienta y la arrastramos a nuestra forma, he inmediatamente iniciamos a construir nuestro (Menú) pues nos creará

una componente que nos permitirá hacerlo, como se muestra:



Indicamos de esta manera los diferentes módulos (ALUMNOS,MATERIAS,SALIR) de nuestra (interface) ejemplo, así como las operaciones a realizar en este caso *(Agregar, Eliminar, Actualizar, Consultar-Listado)*. Lo mismo tanto en el módulo de ALUMNOS como en el de MATERIAS pero con el enfoque de (Materias) por supuesto.

Nota: En lo que se refiere al nombre del componente del (Menú) que por default es comúnmente (MenuStrip1) puede cambiarse a cualesquier nombre que queramos con el fin de referenciarlo más fácil cuando lo ocupemos por medio de la *barra de propiedades* de igual forma.

3.- Vinculamos una forma para cada (Operación), es decir agregamos nuevas formas por medio de la barra de menús por medio de la sección (ARCHIVO) >> Agregar >> Windows forms o por medio del **menú desplegable** al momento de dar clic derecho sobre el proyecto en Agregar >> Windows forms.

Como se muestra:

Para el ejemplo agregamos 8 formas (AlumAgregar, AlumElim, AlumActua, ListAlumno, MatAgregar, MatElim, MatActua, ListMaterias), es decir una por cada operación correspondiente a un módulo lo cual nos da 8 operaciones. De igual manera las personalizamos (Damos nombre, color, fuente, etc) por medio de la *barra de propiedades.*

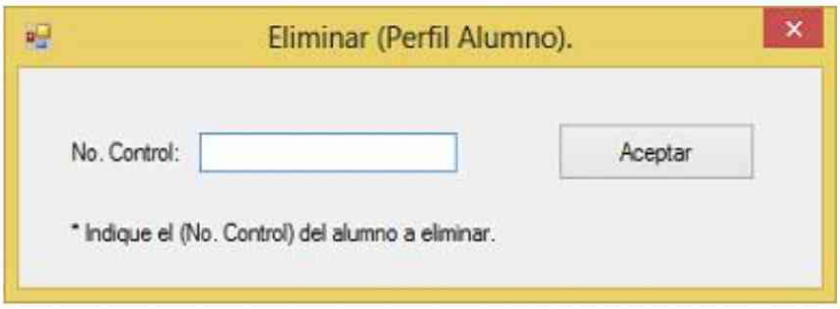
4.- Agregamos nuevos componentes finales a la formas, antes de iniciar la programación como tal añadiendo (Etiquetas y Cajas de texto) que ilustran al usuario los elementos a agregar a la base de datos.

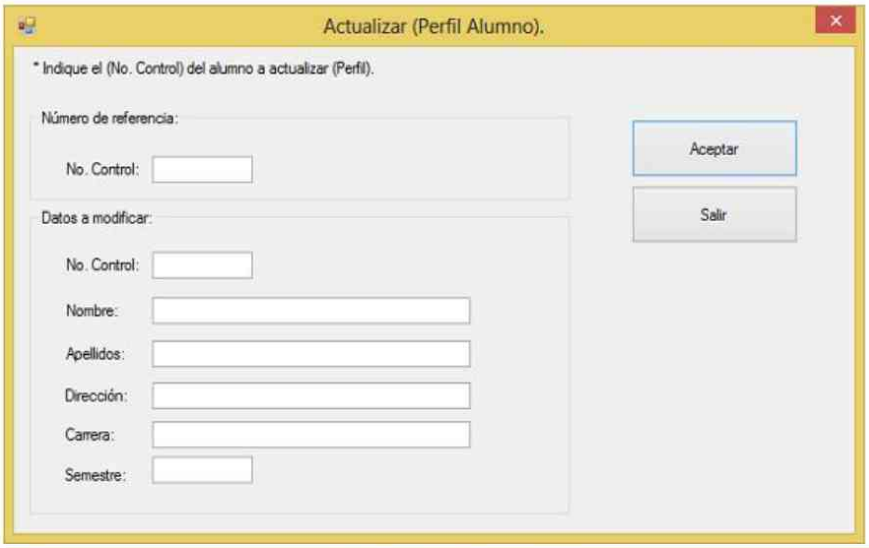
Es decir cada *(Caja de texto)* representaría un valor entrante para cada columna de la base de datos.

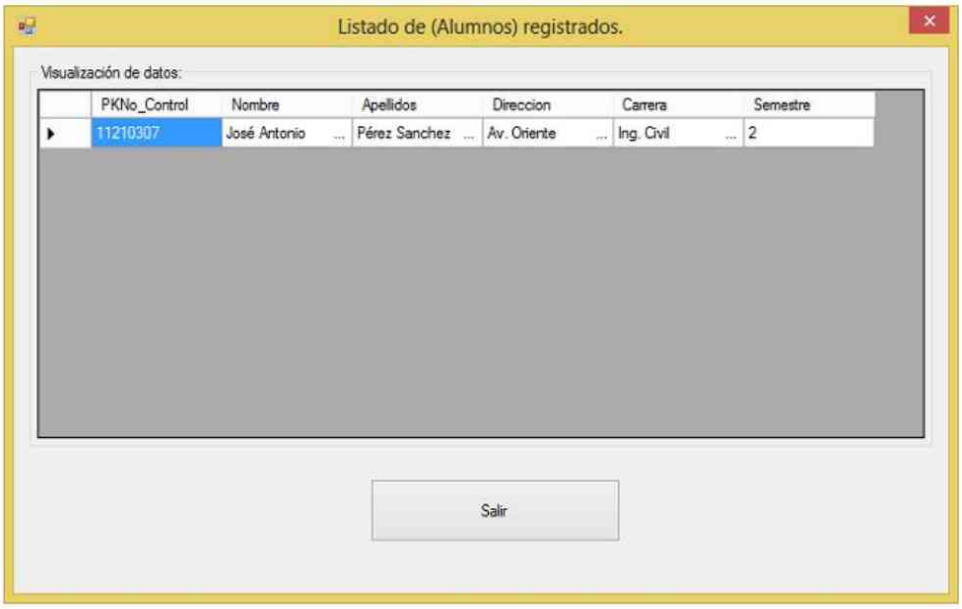
Por ello ponemos una cantidad igual de cajas de texto que de columnas en la base de datos.

Para agregar una caja de texto o etiqueta se sigue la misma lógica que con el (Menú) nos vamos al cuadro de herramientas >> Se busca la herramienta >> Y se arrastra a la forma.

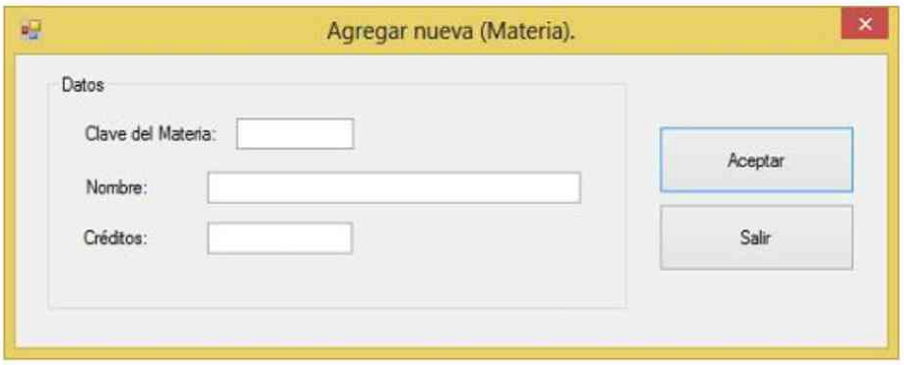
Quedándonos algo como lo siguiente: *En lo que respecta a las formas del módulo (ALUMNOS).*

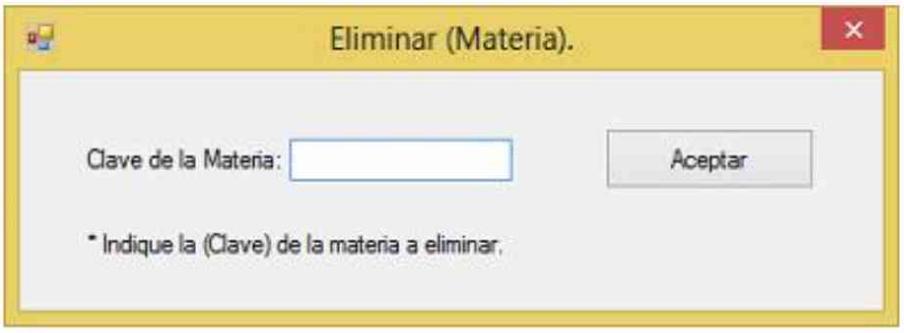


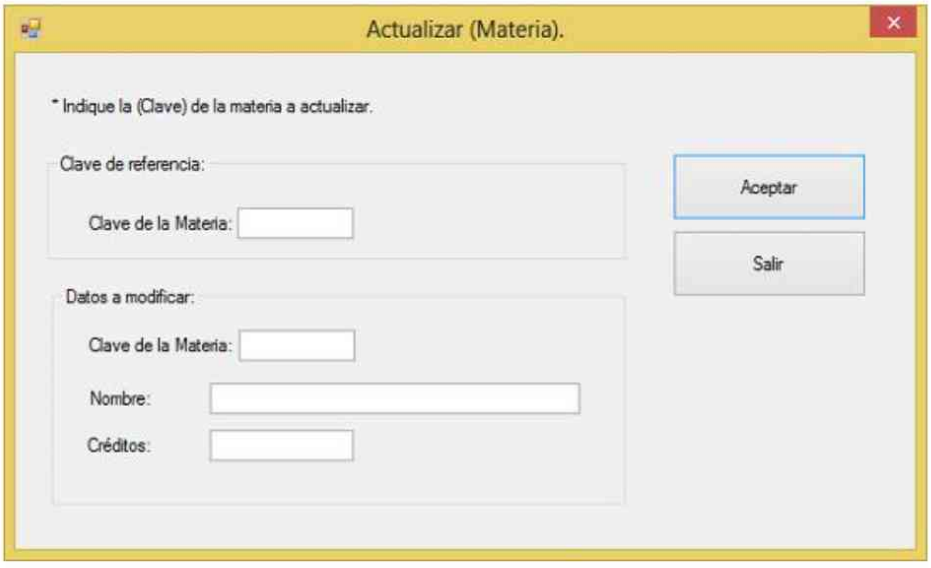




Y en lo que respecta a las formas del módulo (MATERIAS) tenemos:


Agregar nueva (Materia).
Datos
Clave del Materia:
Nombre:
Créditos:
Aceptar
Salir


Eliminar (Materia).
Clave de la Materia:
Aceptar
* Indique la (Clave) de la materia a eliminar.


Actualizar (Materia).
* Indique la (Clave) de la materia a actualizar.
Clave de referencia:
Clave de la Materia:
Datos a modificar:
Clave de la Materia:
Nombre:
Créditos:
Aceptar
Salir

5.- Iniciamos con la programación, ubicando los procedimientos (Main) de las respectivas formas y partiendo de la (Interfaz Principal).

En la interfaz principal primeramente programamos los accesos correspondientes a las operaciones de ambos módulos (ALUMNOS, MATERIAS), mediante la utilización de la P.O.O para llamar a las distintas formas.

Código en C#:

```
//ACCESO A (AGREGAR ALUMNO) DESDE MENU
private void altaDeAlumnoToolStripMenuItem_Click(object sender, EventArgs e)
{//Creamos (Vínculo) entre la forma que se encarga de esto y la forma actual.
frmAlumAgregar VinculoAlumAgregar = new frmAlumAgregar(Miconexion);
//Llamamos a la forma encargada a través del (Vínculo).
VinculoAlumAgregar.Show();}
//ACCESO A (ELIMINAR ALUMNO) DESDE MENU
private void bajaDeAlumnoToolStripMenuItem_Click(object sender, EventArgs e)
{//Creamos (Vínculo) entre la forma que se encarga de esto y la forma actual.
```

```
frmAlumElim VinculoAlumBaja = new frmAlumElim(Miconexion);
//Llamamos a la forma encargada a través del (Vínculo).
VinculoAlumBaja.Show();}
//ACCESO A (ACTUALIZAR ALUMNO) DESDE MENU
private void actualizarPerfilAlumnoToolStripMenuItem_Click(object sender,
EventArgs e)
{//Creamos (Vínculo) entre la forma que se encarga de esto y la forma actual.
frmAlumActua VinculoAlumActua = new frmAlumActua(Miconexion);
//Llamamos a la forma encargada a través del (Vínculo).
VinculoAlumActua.Show();}
//ACCESO A (AGREGAR MATERIA) DESDE MENU
private void agregarNuevoPerfilProfesorToolStripMenuItem_Click(object
sender, EventArgs e)
{//Creamos (Vínculo) entre la forma que se encarga de esto y la forma actual.
frmMatAgregar VinculoMatAgregar = new frmMatAgregar(Miconexion);
//Llamamos a la forma encargada a través del (Vínculo).
VinculoMatAgregar.Show();}
//ACCESO A (ELIMINAR MATERIA) DESDE MENU
private void eliminarPerfilProfesorToolStripMenuItem_Click(object sender,
EventArgs e)
{//Creamos (Vínculo) entre la forma que se encarga de esto y la forma actual.
frmMatElim VinculoMatEliminar = new frmMatElim(Miconexion);
//Llamamos a la forma encargada a través del (Vínculo).
VinculoMatEliminar.Show();}
//ACCESO A (ACTUALIZAR MATERIA) DESDE MENU
private void actualizarPerfilProfesorToolStripMenuItem_Click(object sender,
EventArgs e)
{//Creamos (Vínculo) entre la forma que se encarga de esto y la forma actual.
```

```
frmMatActua VinculoMatActualizar = new frmMatActua(Miconexion);
//Llamamos a la forma encargada a través del (Vínculo).
VinculoMatActualizar.Show();}
//Procedimiento de (Salida)
private void sALIRToolStripMenuItem1_Click(object sender, EventArgs e)
{this.Close();}
//ACCESO A (LISTADO DE ALUMNOS) DESDE MENU
private void listadoDeAlumnosRegistradosToolStripMenuItem_Click(object
sender, EventArgs e)
{//Creamos (Vínculo) entre la forma que se encarga de esto y la forma actual.
frmListAlumno VinculoListAlumno = new frmListAlumno(Miconexion);
//Llamamos a la forma encargada a través del (Vínculo).
VinculoListAlumno.Show();}
//ACCESO A (LISTADO DE MATERIAS) DESDE MENU
private void listadoDeMateriasRegistradasToolStripMenuItem_Click(object
sender, EventArgs e)
{//Creamos (Vínculo) entre la forma que se encarga de esto y la forma actual.
frmListMaterias VinculoListMaterias = new frmListMaterias(Miconexion);
//Llamamos a la forma encargada a través del (Vínculo).
VinculoListMaterias.Show();}
```

Ahora procedemos a establecer la **(Conexión a la base de datos)** desde la interfaz principal para redistribuir a las demás formas. *(Siendo esta parte una de las más indispensables en la etapa)*.

Pero antes de realizar tal cosa en necesario extender el **namespace** ubicado en cada forma en el principio del código de:

```
using System.Data A:
using System.Data.SqlClient; //Agregamos el namespace (SqlClient)
```

Para proporcionar la capacidad a las formas de establecer la conexión por

medio de los componentes de (SQL SERVER en Visual Studio, es decir: Librerías).

Aclaración: Cabe destacar que la conexión a la base de datos no se realizará desde el asistente de (Origen de datos) ubicado en HERRAMIENTAS>>Origen de datos, sino se hará por medio de código.

Una vez extendido el **namespace**, procedemos a declarar la conexión en base a la definición de un nuevo (Objeto) o instancia perteneciente a la clase **SqlConnection** y al envió de una cadena que portará toda la información referente a la autentificación, es decir el *id, el password, el nombre del servidor*, tal como se muestra a continuación:

Código en C#:

```
//Establecemos (CONEXION A B.D) desde la (Interfaz principal)
//Obtenemos del (Nombre de la máquina y el nombre del servidor local con SQL SERVER) mediante:
// Nom_Máquina = System.Security.Principal.WindowsIdentity.GetCurrent().Name;
// Nom_SQLSERVER = Environment.MachineName;
//Conexión a B.D (Establecimiento por medio de SQL)
//Id: Nombre del usuario de la maquina.
//Server: Nombre del sql-server.
SqlConnection Miconexion = new SqlConnection("user id=" +System.Security.Principal.WindowsIdentity.GetCurrent().Name + ";"
+ "password=;server=" + Environment.MachineName + "\\" + "SQLEXPRESS;"
+ "Trusted_Connection=yes;"+ "database=BD_1; "+ "connection timeout=30");
```

Se implemento esto:

Nom_Máquina =

System.Security.Principal.WindowsIdentity.GetCurrent().Name;

Nom_SQLSERVER = Environment.MachineName;

Con el fin de que en ciertas computadoras se obtenga automáticamente el (Nombre de la Máquina y el nombre del servidor local) sin necesidad efectuar cambios en el código. De no funcionar tal fragmento de código puede cambiarse a los datos que se tengan como **"id, password, nombre del servidor"** respectivamente.

Por ejemplo:

SqlConnection Miconexion = new SqlConnection("user id=Raul;"+ "password=camila;server=Pruebas"+ "Trusted_Connection=yes;" + "database=BD_1;"+ "connection timeout=30");

Seguido lo que vamos a realizar es enviar la instancia (*Miconexion*) creada como parámetro a través del constructor de cada forma a fin de redistribuir la conexión a la base hacia las otras formas.

Por ejemplo de la siguiente manera:

Código en C#:

//Creamos (Vínculo) entre la forma que se encarga de esto y la forma actual y enviamos a través del constructor la instancia (Miconexion).

frmAlumElim VinculoAlumBaja = new frmAlumElim(Miconexion);

De igual manera en lo que se refiere a las demás formas (frmAlumAgregar, frmAlumActua, etc.).

Nota: Cabe recordar que el constructor es un procedimiento exclusivo que nos permite redistribuir variables (Parámetros) a la clase que pertenece.

Ya en este punto empezamos a trabajar en las demás formas...

Lo primero que vamos a realizar en todas las formas es declarar una *variable/instancia* que se encargará de lo que se envió por medio del constructor desde la *(Interfaz principal)* a la forma en que se encuentre.

Forma: ALUMAGREGAR.CS

De la siguiente manera:

Código en C#:

```
//Definición de la variable (Encargada de recibir parámetro)
//Se declara afuera para que sea global...
SqlConnection MiConexion2;
//CONSTRUCTOR
public frmAlumAgregar(SqlConnection MiConexion)
{InitializeComponent();
//Recibe como parámetro la (Conexión) de la forma principal y la redistribuye...
MiConexion2 = MiConexion; }
```

Después programamos lo restante con respecto a los 2 botones de la forma siguiente:

Nota: Todo código expuesto se encuentra comentado de tal manera que se indica que se está realizando en ese momento, esto con el fin de proporcionar mayor comodidad al lector.

Código en C#:

```
//METODO ENCARGADO DEL BOTON ACEPTAR.
private void cmdAceptar_Click(object sender, EventArgs e)
{//Validacion de campos. (Utilizacion de OR)
if (txtNum.Text == "" || txtNom.Text == "" || txtApellidos.Text == "" || txtDir.Text == "" || txtCarrera.Text == "" || txtSem.Text == "")
{//Mensaje
MessageBox.Show("INSERCION INVALIDA!, FAVOR DE COMPLETAR LOS DATOS EN LOS CAMPOS...", "INFORMACION");
//Posiciona en la primera caja
txtNum.Focus();
}
```

```
else //De lo contrario (Iniciar) - Conexion..
{
//Prueba
try
{
//Abrimos (Conexion).
MiConexion2.Open();
//Generamos (Instruccion de INSERTADO en SQL- Con lo que hay en cajas de texto)
string SQL_INSERTAR = "INSERT INTO ALUMNOS(PKNo_Control,Nombre,Apellidos,Direccion,Carrera,Semestre) values('" + txtNum.Text + "','" + txtNom.Text + "','" + txtApellidos.Text + "','" + txtDir.Text + "','" + txtCarrera.Text + "','" + txtSem.Text + "');";
//Definimos un nuevo elemento (De la clase SqlCommand).
//Esto nos permitira ejecutar la instruccion SQL.
SqlCommand Elem_Comando = new SqlCommand(SQL_INSERTAR, MiConexion2);
//Ejecutamos la instruccion en (Modo Execute).
Elem_Comando.ExecuteNonQuery();
//Cerramos (Conexion local).
MiConexion2.Close();
//Despues...
//Eliminame los objetos (Inutiles ya!) obligando al GC.
Elem_Comando.Dispose();
//Cierra la forma.
this.Close();}
//Obten la excepcion de existir...
catch (SqlException)
```

{//Y despliega mensaje en caso de generarse una (Excepción) por aquello de los distintos tipos de valores insertados o mandados a la B.D.
MessageBox.Show("INSERCION INVALIDA!, COMPLETE DATOS UTILIZANDO EL CORRECTO TIPO DE DATO EN LOS CAMPOS!", "ALERTA");}
//Finalmente haz lo que se deba..
finally
{//Cierra la (Conexion) - Por si acaso no se cerro.
MiConexion2.Close();
}}}

//METODO ENCARGADO DEL BOTON SALIR.

private void cmdSalir_Click(object sender, EventArgs e)
{//Este objeto (Cerrarlo) - Utilizamos el (This).
this.Close();
}

IMPORTANTE: Es probable que de acuerdo a como está programada la base de datos en la etapa 1, es decir el margen de longitud de los campos. Puedan existir ocasiones en el programa (ejemplo) al momento de agregar datos que se genere una excepción de parte del *SqlCommand* y ejecute el mensaje (Ubicado) en el catch. *Más esto no significa que el ejemplo se encuentre incorrecto. Sino que no se crearon los campos en la base de datos con una longitud considerada estándar.*

Forma: ALUMELIM.CS
De la siguiente manera:

Código en C#:

//Definición de la variable (Encargada de recibir parametro)
//Se declara afuera para que sea global...
SqlConnection MiConexion3;
//CONSTRUCTOR

```
public frmAlumElim(SqlConnection MiConexion)
{
InitializeComponent();
//Recibe como parametro la (Conexion) de la forma principal y la redis-
tribuye...
MiConexion3 = MiConexion; }
```

Después programamos lo restante con respecto al único botón de la forma siguiente:

Código en C#:

```
//METODO ENCARGO DEL BOTON ACEPTAR.
private void cmdAceptar_Click(object sender, EventArgs e)
{//Validacion de campos.
if (txtNum.Text == "")
{//Mensaje
MessageBox.Show("ELIMINACION INVALIDA!, FAVOR DE COMPLETAR
LOS DATOS EN LOS CAMPOS...", "INFORMACION");}
else //De lo contrario (Iniciar) - Conexion.
{//Prueba
try
{//Abrimos (Conexion).
MiConexion3.Open();
//Generamos (Instruccion SQL- Con lo que hay en cajas de texto)
string SQL_ELIMINAR = "DELETE FROM ALUMNOS WHERE(PK-
No_Control='" + txtNum.Text + "')";
//Definimos un nuevo elemento (De la clase SqlCommand).
//Esto nos permitira ejecutar la instruccion SQL.
SqlCommand Elem_Comando = new SqlCommand(SQL_ELIMINAR, Mi-
Conexion3);
```

```
//Ejecutamos la instruccion en (Modo Execute sin consulta)
Elem_Comando.ExecuteNonQuery();
//Cerramos (Conexion local).
MiConexion3.Close();
//Cerrar la forma.
this.Close();}
//Obten la excepcion de existir...
catch (SqlException)
{//Y despliega mensaje en caso de generarse una (Excepción) por aquello de los distintos tipos de valores insertados o mandados a la B.D.
MessageBox.Show("ELIMINACION INVALIDA!, COMPLETE DATOS UTILIZANDO EL CORRECTO TIPO DE DATO EN LOS CAMPOS!", "ALERTA");}
//Finalmente haz lo que se deba...
finally
{//Cerrar (Conexion) - Por si acaso
MiConexion3.Close();}
}
```

Aclaración: Cabe destacar que en esta parte del ejemplo el introducir un (*Número de control*) en el caso de los ALUMNOS o (*Clave de materia*) en el caso de las MATERIAS que no se encuentre en la base de datos no genera una excepción de parte de SQL-SERVER de tal manera que se indique como error en C#, pues regresa únicamente que se borraron o filas, en el ejemplo no se implementó el chequeo de si son o filas indicar que no existe registro con tal identificación. Más esto no quiere decir que no se pueda realizarse.

Forma: ALUMACTUA.CS

Omitimos lo de la variable o instancia creada al principio pues se asume que el lector, ya comprendió el punto. Y vamos directo al grano de la

programación de los botones.

Código en C#:

```
//METODO ENCARGADO DEL BOTON ACEPTAR.
private void cmdAceptar_Click(object sender, EventArgs e)
{//Validacion de campos. (Utilizacion de OR)
if (txtNum0.Text == "" || txtNum.Text == "" || txtNom.Text == "" || txtApellidos.Text == "" || txtDir.Text == "" || txtCarrera.Text == "" || txtSem.Text == "")
{//Mensaje
MessageBox.Show("MODIFICACION INVALIDA!, FAVOR DE COMPLETAR LOS DATOS EN LOS CAMPOS...", "INFORMACION");
//Posiciona en la primera caja
txtNum0.Focus();}
else //De lo contrario (Iniciar) - Conexion..
{//Prueba
try
{
//Abrimos (Conexion).
MiConexion4.Open();
//Generamos (Instruccion SQL- Con lo que hay en cajas de texto)
string SQL_MODIFICAR = "UPDATE ALUMNOS " + "SET PKNo_Control = " + txtNum.Text +",Nombre = " + "'" + txtNom.Text + "'" + ",Apellidos =" + "'" + txtApellidos.Text + "'" + ",Direccion =" + "'" + txtDir.Text + "'" + ",Carrera =" + "'" + txtCarrera.Text + "'" + ",Semestre =" + "'" + txtSem.Text + "'" +
"WHERE PKNo_Control = " + txtNum0.Text + ";";
//Definimos un nuevo elemento (De la clase SqlCommand).
//Esto nos permitira ejecutar la instruccion SQL.
SqlCommand Elem_Comando = new SqlCommand(SQL_MODIFICAR, MiConexion4);
```

```
//Ejecutamos la instruccion en (Modo Execute sin consulta)
Elem_Comando.ExecuteNonQuery();
//Cerramos (Conexion local).
MiConexion4.Close();
//Cerrar la forma.
this.Close();}
//Obten la excepcion de existir...
catch (SqlException)
{//Y despliega mensaje en caso de generarse una (Excepción) por aquello de los distintos tipos de valores insertados o mandados a la B.D.
MessageBox.Show("MODIFICACION INVALIDA!, COMPLETE DATOS UTILIZANDO EL CORRECTO TIPO DE DATO EN LOS CAMPOS!", "ALERTA");}
//Finalmente haz lo que se deba..
finally
{//Cerrar (Conexion) - Por si acaso
MiConexion4.Close();
}}}
//METODO ENCARGADO DEL BOTON SALIR.
private void cmdSalir_Click(object sender, EventArgs e)
{//Este objeto (Cerrarlo) - Utilizamos el (This).
this.Close();
}
```

Por último ilustramos como realizar una (Consulta o Listado) simple de aquellos datos introducidos a la base de datos.

Mediante lo que se conoce como ***(Vincular un datagrid o Grilla de datos)*** con una tabla mediante una conexión previamente realizada.

Forma: LISTALUMNO.CS

Omitimos lo de la *variable o instancia* creada al principio pues se asume que el lector, ya comprendió el punto. Y vamos directo al grano de la programación de los botones.

Código en C#:

```
//METODO ENCARGADO DEL BOTON (SALIR)
private void cmdSalir_Click(object sender, EventArgs e)
{//Cierra la forma
this.Close();}
//PROCEDIMIENTO QUE CARGA LOS DATOS EN LA TABLA1 (DATAGRIDVIEW) AL MOMENTO DE ABRIR FORMA
private void frmListAlumno_Load(object sender, EventArgs e)
{//Abrir (Conexión)
MiConexion8.Open();
//Consulta (Seleccion en B.D)
string SQL_SELECT = "SELECT * FROM ALUMNOS";
/*Definición del (Adaptador de datos) este proporciona
* un flujo de datos entre la B.D y la conexion que permite llenar
* el conjunto de datos (Dataset))*/
SqlDataAdapter Adaptador_Datos = new SqlDataAdapter(SQL_SELECT, MiConexion8); //Consulta
//Definicion de conjunto de datos.
DataSet Conjunto = new DataSet();
//Llena el conjunto con lo que hay en el adaptor (Adaptor lleno debido a que realizo consulta previamente)
Adaptador_Datos.Fill(Conjunto,"ALUMNOS");
//Llena la (Tabla1) con los datos del (Conjunto).
Tabla1.DataSource = Conjunto;
Tabla1.DataMember = "ALUMNOS";
```

//Una vez llenado el (Conjunto de datos) - Cierrame la conexion con la B.D
MiConexion8.Close();}

Nota: Aquellos lectores en los cuales el IDE no les reconozca la parte del **DataSet**, favor de verificar si se encuentra referenciada la siguiente línea:
using System.Data; //DATA
Pues en la mayoría de los casos este es el problema.

Con esto daríamos por concluida la etapa 2, pues no es necesario ilustrar como se realizarían las operaciones correspondientes a las otras 4 formas restantes, pues sería en el mismo sentido que se hizo para las formas de (ALUMNOS).

Volver al inicio.

~~~

Etapa 3.

Hemos llegado a la parte final de este manual, realizaremos un registro completo en lo que se refiere a los (ALUMNOS) en la base de datos creada por medio de la interfaz creada de igual manera.

A fin de dejar, otros ejemplos para las futuras pruebas de los lectores...

1.- Ejecutamos nuestra interfaz.

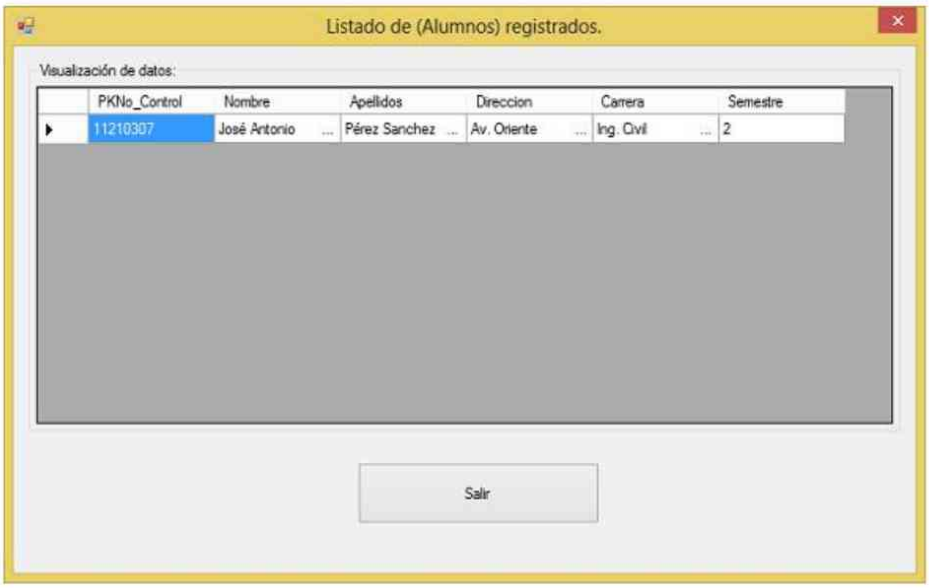
2.- Accedemos a ALUMNOS >> Agregar, e insertamos el siguiente registro.

Tal como se muestra en la imagen siguiente:

3.- Damos (Aceptar) para guardar el registro, seguido salimos de la forma.

4.- Ahora accesamos al (Listado de Alumnos registrados) a fin de observar que el registro se inserto con éxito.

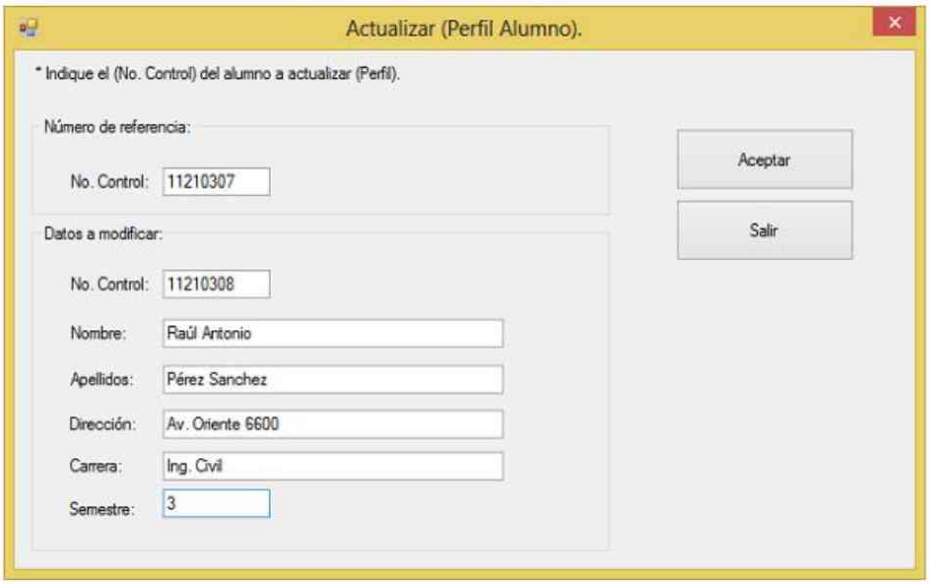
Tal como se muestra en la imagen siguiente:



Como puede observarse si se logro con éxito.

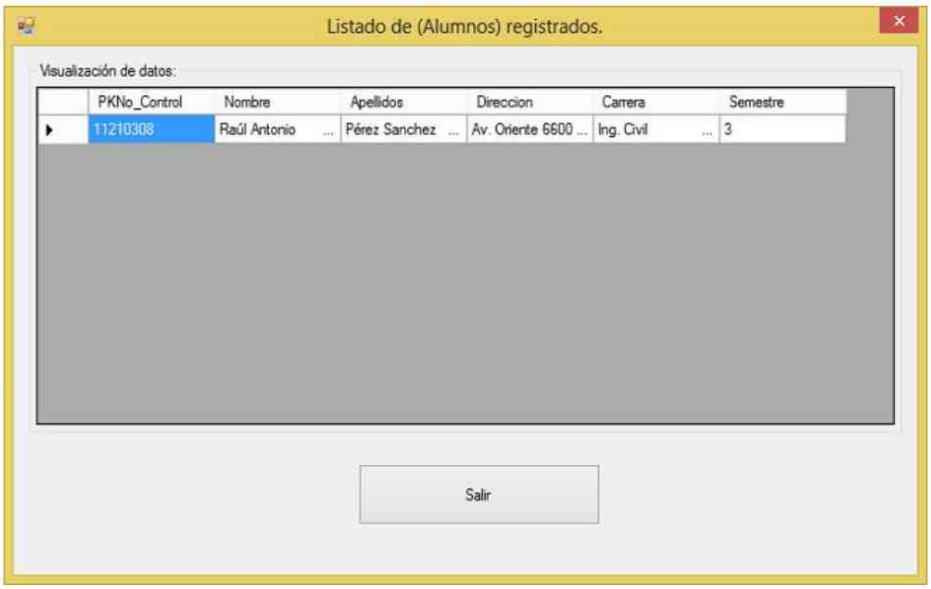
5.- Ahora procedemos a modificar los parámetros del registro, es decir algún campo del registro (Por ejemplo: el nombre) de **José Antonio** a **Raúl Antonio** junto con el semestre de **2** a **3** por medio del número de control para ello.

Tal como se muestra en la imagen siguiente:



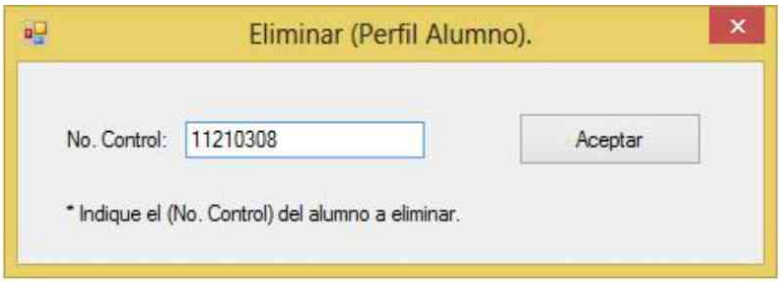
6.- De nuevo accesamos al (Listado de Alumnos registrados) a fin de observar que el registro se modificó con éxito.

Tal como se muestra en la imagen siguiente:



7.- Por último eliminamos dicho registro insertado a base del número de control y accesamos al (Listado de Alumnos registrados) a fin de observar que el registro se elimino con éxito.

Tal como se muestra en las imágenes siguientes:



Volver al inicio.

###

Acerca de los autores.

Raúl Antonio Zavala López.

Actualmente soy estudiante del I.T.T (Instituto Tecnológico de Tijuana) es decir: futuro aspirante a (Ing. en Sistemas Computacionales).

Me considero una persona apasionada de las ciencias exactas, más específicamente de la (Matemáticas), pues considero que tales nos permiten expresar fenómenos tan complejos con una simplicidad considerable... Tomando como criterio aquel mundo tan caótico donde vivimos actualmente. La experiencia como estudiante así mismo como instructor de asesorías impartidas a alumnos del I.T.T, han logrado consolidarme a tal grado de poder contextualizar todos aquellos problemas generalmente presentados como estudiante al momento de cursar un ambiente media-superior. Teniendo como producto algunas publicaciones, principalmente destinadas a compartir un poco de mis conocimientos a todos aquellos estudiantes que lo requieran. Esto y muchas otras cosas más es lo que soy...

Roberto Llamas Avalos.

Nací el 8 de junio en Tijuana Baja California México, hijo de dos excepcionales personas, actualmente estudio en el Instituto Tecnológico de Tijuana (I.T.T) la carrera de Ingeniería en sistemas computacionales, aunque previamente obtuve el título de técnico en electrónica.

Me interesan las matemáticas (se podría decir las ciencias exactas), al igual que su relación con la programación, debido a que esto es base para la creación de videojuegos. De igual manera me apasiona la física más precisamente la mecánica cuántica (aunque se fundamenta en la probabilidad), la existencia de fenómenos, eventos que ocurren en la naturaleza los cuales tratan de ser descritos matemáticamente por esta rama me intriga, otro aspecto que me cautiva de la mecánica cuántica es la computación cuántica y como revolucionará la computación y la programación, dando un cambio radical a lo que se conoce actualmente.

Tome la decisión de convertirme en autor de libros matemáticos y tratar de que estos sean lo más simples y sencillos posibles, debido a que pienso que las matemáticas deben ser simples y elegantes y no temidas por el estudiante promedio (sin generalizar), al ser estudiante e instructor de asesorías a colegas (alumnos), he podido comprender como y de qué forma se pueden abordar temas tan básicos que pueden tornarse complicados si estos no son comprendidos correctamente o temas tan complicados que pueden llegar a aterrar a los alumnos.

Conéctate con nosotros en línea:

Raúl Antonio Zavala López.

Twitter: @RauZaLo

Facebook: https://www.facebook.com/raulantonio.zavalalopez

Smashwords: http://www.smashwords.com/profile/view/raulzavala

Roberto Llamas Avalos.

Twitter: @RobertoLlamasAv

Facebook: https://www.facebook.com/roberto.llamas.RLLA11

Volver al inicio.