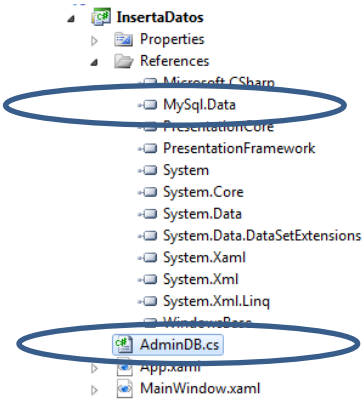
Insertar datos en una tabla MySQL en C# WPF

En el ejemplo anterior la conexión la hacíamos mediante un botón, sin embargo pensando de manera mas realista así no se aplica en la vida real, normalmente debemos dejar que el sistema automáticamente haga la conexión cada que sea necesario y se desconecta de igual manera, vamos a modificar el ejemplo anterior haciendo lo siguiente:

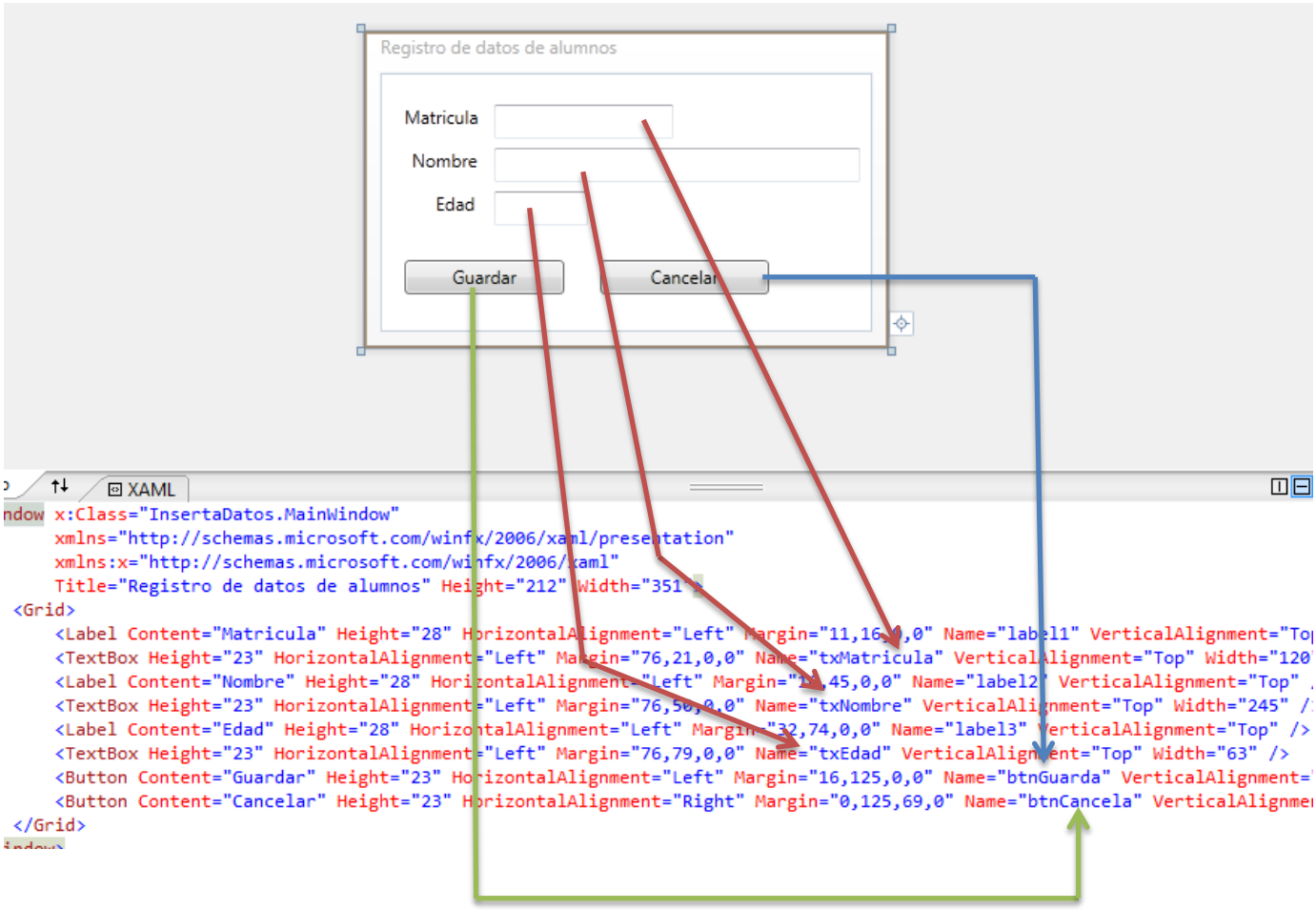
- Creamos un nuevo proyecto WPF y lo llamaremos “InsertaDatos”. Agregamos las referencias a MySQL tal como se hizo en el ejemplo anterios. (*Ventana explorador de soluciones - References*).
- Creamos la misma clase del ejemplo anterior con el mismo nombre y el mismo código.
- Debemos tener algo así en el explorador de soluciones:



El código de la clase es casi el mismo (hasta ahora) del ejemplo anterior, solo cambia el espacio de nombres y esta vez también utilizaremos el método “Desconectar” asi que le quietamos la palabra “static” que quede asi como se muestra en este ejemplo.

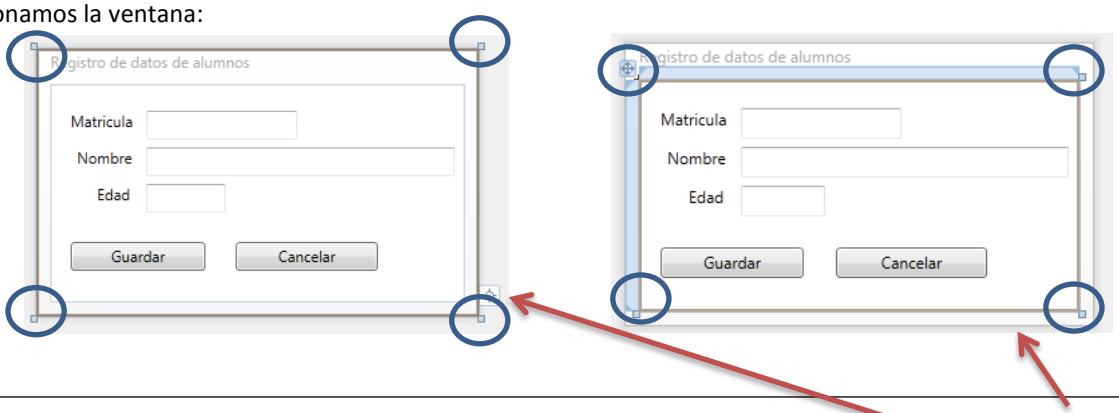
```
using System;
using MySql.Data.MySqlClient;
using System.Windows;

namespace InsertaDatos
{
    class AdminDB
    {
        static MySqlConnection Conex = new MySqlConnection();
        static string serv = "Server=localhost;";
        static string db = "Database=prueba;";
        static string usuario = "UID=root;";
        static string pwd = "Password = masterkey;";

        string CadenaDeConexion = serv + db + usuario + pwd;

        static MySqlCommand Comando = new MySqlCommand();
        static MySqlDataAdapter Adaptador = new MySqlDataAdapter();

        public void Conectar()
        {
            try
              {
                Conex.ConnectionString = CadenaDeConexion;
                Conex.Open();
                   MessageBox.Show("La BD esta ahora coenctada");
              }
            catch (Exception)
            {
                MessageBox.Show("Ocurrio un error al conectar a la BD");
                throw;
            }
        }

        public void Desconectar()
        {
            Conex.Close();
        }
    }
```

La idea es contar con un formulario de captura donde se teclearán los datos de la tabla alumnos mostrada en el ejemplo anterior, un botón que permita guardar esos datos y otro que permita cancelar la operación, para ello damos formato a nuestra ventana WPF como se muestra en la figura, tenemos (*txMatricula, txNombre, txEdad, btnGuarda y btnCancela*).



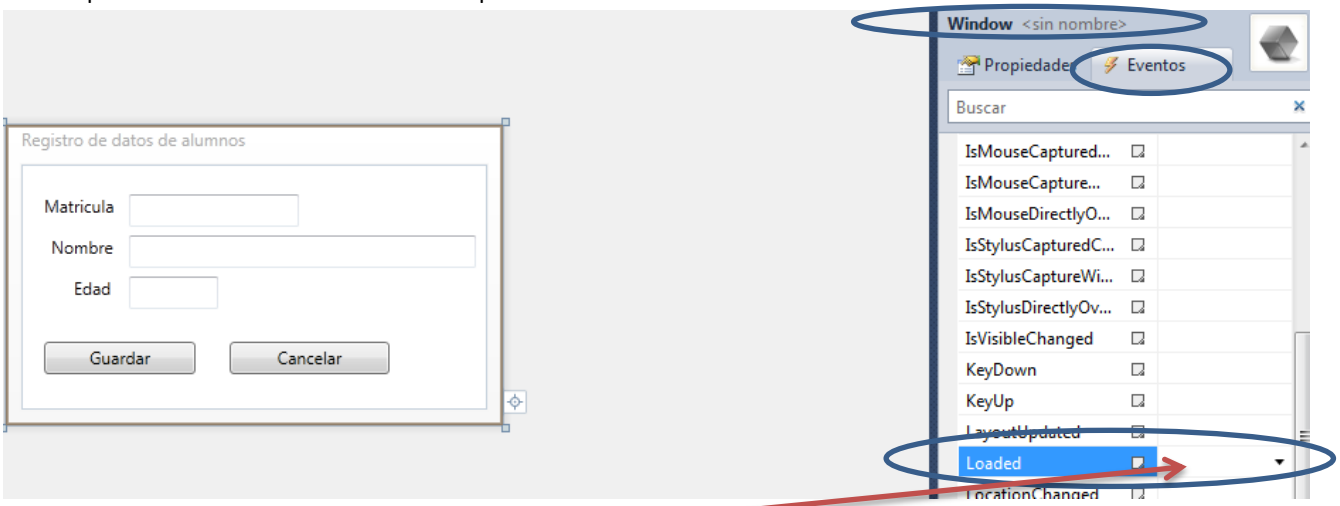
Ya tenemos el diseño, ahora debemos hacerlo funcionar de tal manera que lo que se teclee se envíe a la base de datos en el momento de dar clic en el botón “Guardar”, para ello lo primero es establecer la conexión, en el ejemplo anterior lo hacíamos en un botón, eso ya no nos sirve, lo ideal es que cuando el formulario se abra automáticamente se establezca la conexión, entonces hagamos lo siguiente:

- Seleccionamos la ventana:



Es importante seleccionar de manera adecuada la ventana, en las figuras se muestra la manera correcta y la incorrecta de hacer esta selección, ya que si no lo hacemos de la manera correcta es probable que la conexión no se haga de manera adecuada.

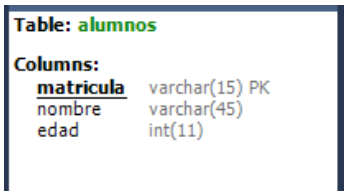
- En el explorador de soluciones buscamos la pestaña eventos en ella buscamos el evento “Loaded”



- Damos doble clic ahí para programar el evento.
- Ahí tecleamos el siguiente código, que si recordamos bien es el mismo código que se ponía en el botón del ejemplo anterior, con esto lo gramos que al cargar el formulario automáticamente se haga la conexión a la base de datos y quede lista para ser administrada.

```
public partial class MainWindow : Window
{
    AdminDB db = new AdminDB();

    public MainWindow()
    {
        InitializeComponent();
    }

    private void Window_Loaded(object sender, RoutedEventArgs e)
    {
        db.Conectar();
    }
```

Algo importante es que el objeto **db** ahora lo ponemos como atributo y no como variable local *(así se usó en el ejemplo anterior)*.



- Debemos crear en la clase un método que nos permita hacer el guardado en la tabla, recordemos que tenemos una tabla llama “*alumnos*” con tres campos: matricula, nombre y edad.
- Abrimos la clase “AdminDB” y tecleamos el nuevo método el cual queda de la siguiente manera:

```
public void insertaDatos(string m, string n, int e)
{
    string strSQL = "INSERT INTO alumnos VALUES ('" + m + "','" + n + "'," + e + ");";
    cmd.CommandText = strSQL;
    cmd.Connection = Conex;
    cmd.ExecuteNonQuery();
}
```

Recibe tres parámetros, el primero es el valor de la matricula(m), el segundo es el valor del nombre(n) y el tercero de tipo entero es el valor de la edad(e), el tercero es de tipo entero porque así esta definido en la tabla, es decir, lo que es varchar se traduce en string.

- Ahora solo debemos invocarlo desde el botón “Guardar” asi que lo programamos colocando el siguiente método.

```
private void btnGuarda_Click(object sender, RoutedEventArgs e)
{
    try
    {
       db.insertaDatos(txMatricula.Text, txNombre.Text, Convert.ToInt32(txEdad.Text));
       db.Desconectar();
       this.Close();
    }
    catch
    {
        MessageBox.Show("Error al guardar en la base de datos");
    }
```

- Para terminar programamos el botón “Cancelar”, que lo único que hace es cerrar la conexión y cerrar la ventana.

```
private void btnCancela_Click(object sender, RoutedEventArgs e)
{
   db.Desconectar();
   this.Close();
}
```

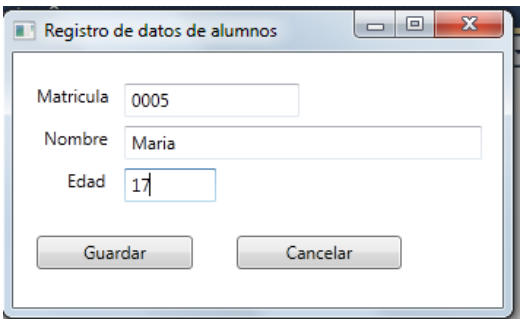
Nota: Es algo molesto que cada que haga la conexión el sistema nos avise que ya se ha hecho, lo correcto es confiar que la conexión se ha hecho, por lo tanto modifiquemos el código de la clase:

```
public void Conectar()
{
    try
    {
        Conex.ConnectionString = CadenaDeConexion;
        Conex.Open();
    }
    catch (Exception)
    {
        MessageBox.Show("Ocurrio un error al conectar a la BD");
        throw;
    }
}
```

Se ha eliminado el mensaje

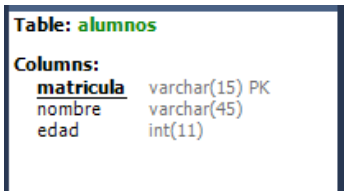
```
public void Conectar()
{
    try
    {
        Conex.ConnectionString = CadenaDeConexion;
        Conex.Open();
        MessageBox.Show("La BD esta ahora conectada");
    }
    catch (Exception)
    {
        MessageBox.Show("Ocurrio un error al conectar a la BD");
        throw;
    }
}
```

Ahora si ya podemos correr el programa y capturar los datos de algún alumno, los datos tecleado serán enviados a la base de datos.



En otro ejemplo veremos como hacer consultas.

Aunque entiende que el aumno debe tener tener conocimientos de SQL hagamos un análisis del código, contamos una tabla de tres campos, dos varchar y un entero, como se muestra en la figura:



En este ejemplo estamos insertando los siguientes datos:

- Matricula = “0005”
- Nombre = “María”
- Edad = 17

El código de SQL para esta inserción sería como se muestra: NOTA Los dos primeros van con comillas simples porque son varchar (*o strings*) y el ultimo parámetro no lleva comillas porque es un valor numérico (*Esto puede variar dependiendo de la versión de SQL que esté utilizando*).

```
INSERT INTO alumnus VALUES(‘0005’,’María’,17);
```

Esta misma cadena se debe general mediante el código, por eso es que no queda de la siguiente manera:

No usa comillas simples

```
public void insertaDatos(string m, string n, int e)
{
    string strSQL = "INSERT INTO alumnos VALUES ('" + m + "','" + n + "'," + e + ");";
    cmd.CommandText = strSQL;
    cmd.Connection = Conex;
    cmd.ExecuteNonQuery();
}
```

Es cuestión de tener cuidado, jugar con los parámetros, sus tipos de datos y saber como formar cadenas con C#.