

Unidad 3

Conectividad de Base de Datos

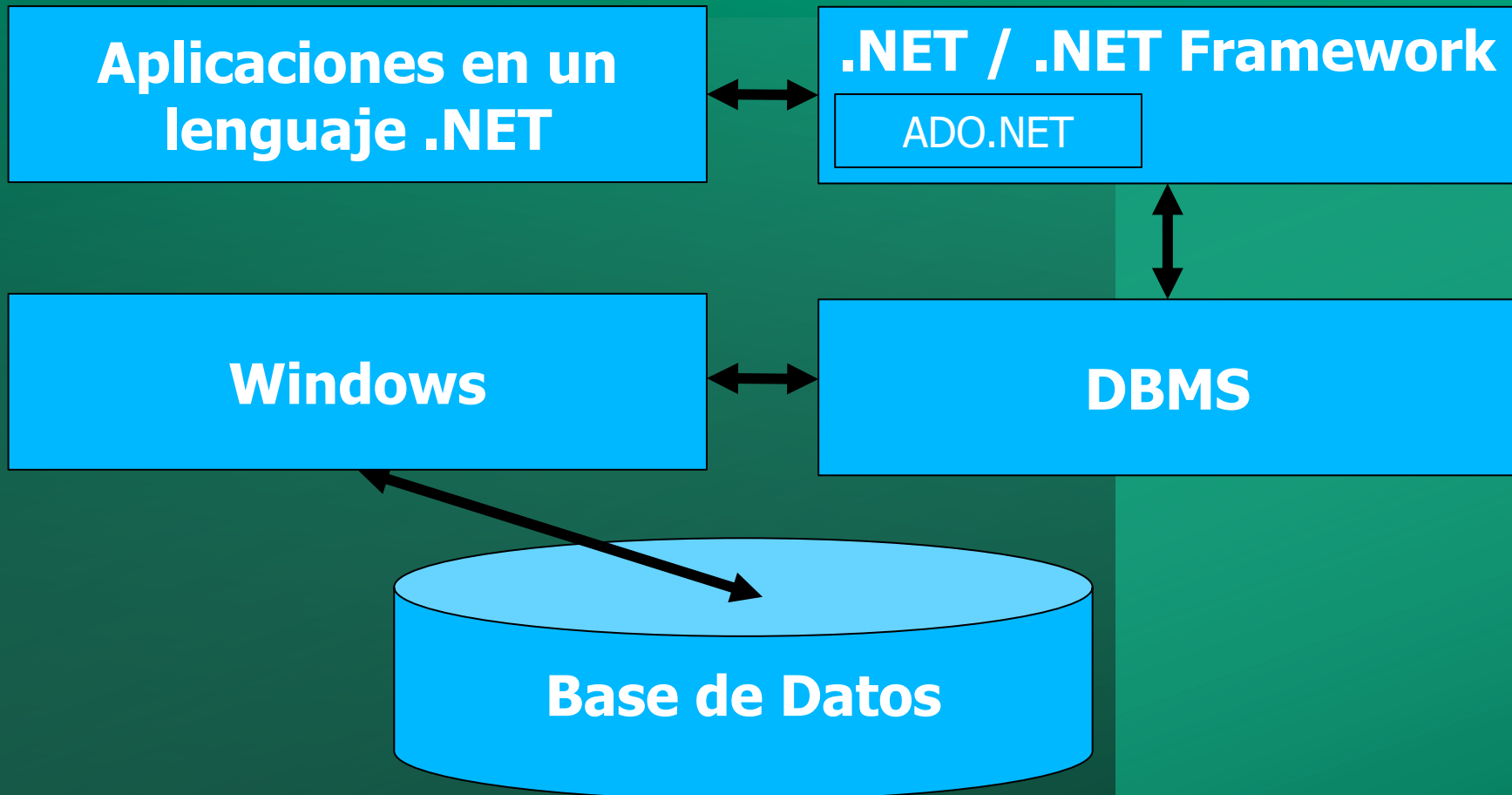
3.1 ADO.NET

3.2 Conectividad entre DML
(huésped) y C# (anfitrión).

3.1 ADO.NET

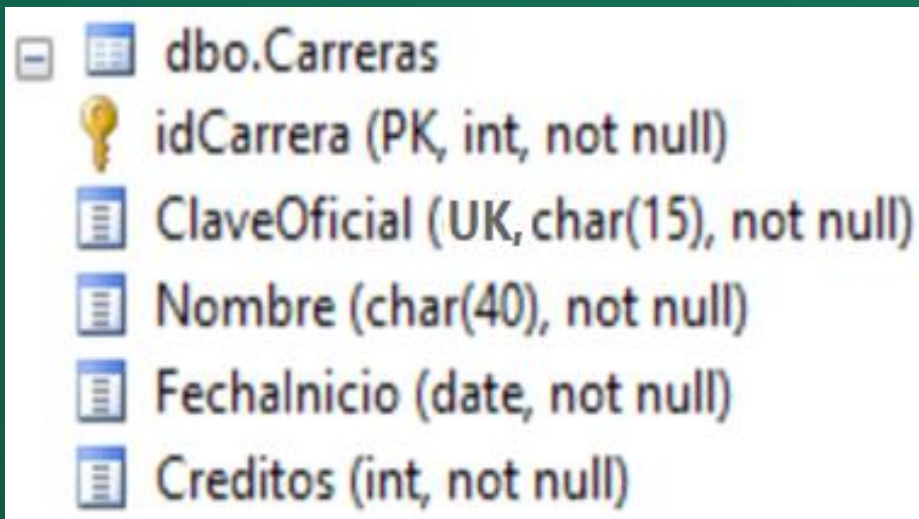
- ADO.NET es un conjunto de clases que provee la funcionalidad para el acceso y manipulación de bases de datos relacionales.
- Forma parte de .NET Framework de Microsoft.
 - .NET Framework es una plataforma de desarrollo de aplicaciones en Windows.
 - Se conforma de una biblioteca de clases para facilitar el desarrollo de aplicaciones de escritorio, web y servicios web.

3.1 ADO.NET



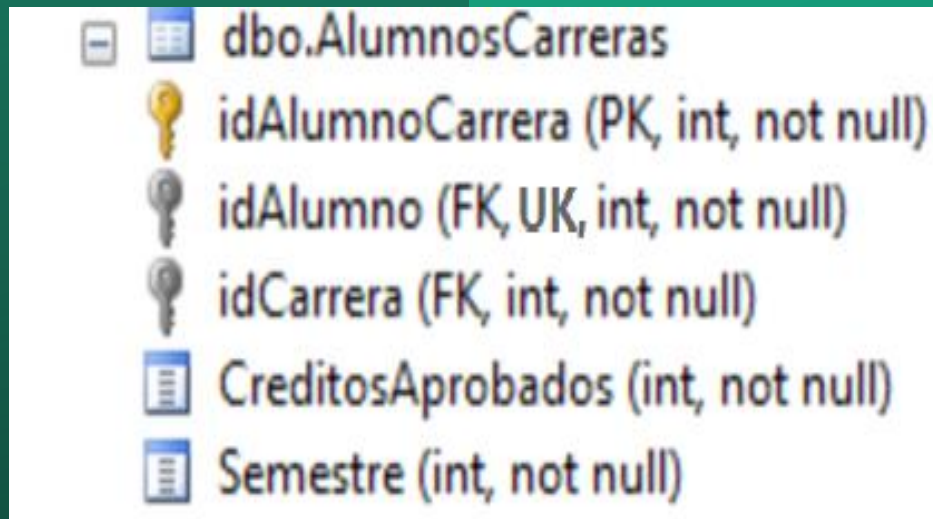
3.2 Conectividad entre DML (huésped) y C# (anfitrión)

- Crear las tablas **Carreras** y **AlumnosCarreras** en la Base de Datos **ITD**, con el esquema que se muestra.
 - No añada tuplas a ninguna de las tablas.



dbo.Carreras

	idCarrera (PK, int, not null)
	ClaveOficial (UK, char(15), not null)
	Nombre (char(40), not null)
	FechaInicio (date, not null)
	Creditos (int, not null)



dbo.AlumnosCarreras

	idAlumnoCarrera (PK, int, not null)
	idAlumno (FK, UK, int, not null)
	idCarrera (FK, int, not null)
	CreditosAprobados (int, not null)
	Semestre (int, not null)

3.2 Conectividad entre DML (huésped) y C# (anfitrión)

Descargar Visual Studio Community 2022 de la URL siguiente:

<https://visualstudio.microsoft.com/es/vs/>

Durante la instalación elija al menos el paquete de desarrollo de aplicaciones **.NET para Escritorio**.

Una vez instalado VS2022, ejecútelo y cree un proyecto con las indicaciones que se muestran a partir de la diapositiva siguiente

3.2 Conectividad entre DML (huésped) y C# (anfitrión)

Visual Studio 2022

Abrir recientes

Tareas iniciales



Clonar un repositorio

Obtiene código desde un repositorio en línea, como GitHub o Azure DevOps.



Abrir un proyecto o una solución

Abre un archivo .sln o proyecto de Visual Studio local.



Abrir una carpeta local

Navegar y editar el código en cualquier carpeta



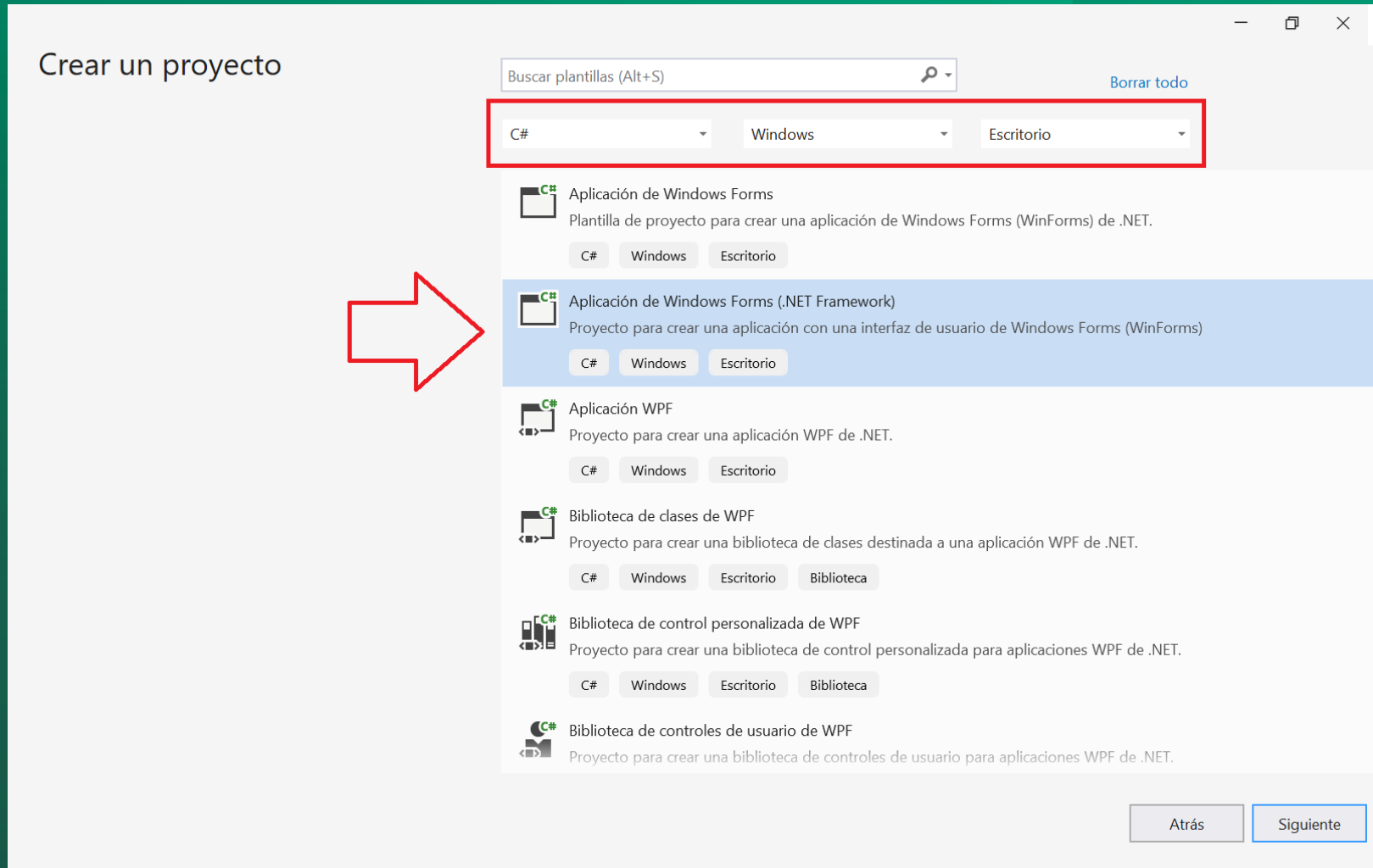
Crear un proyecto

Elija una plantilla de proyecto con la técnica scaffolding de código para comenzar.

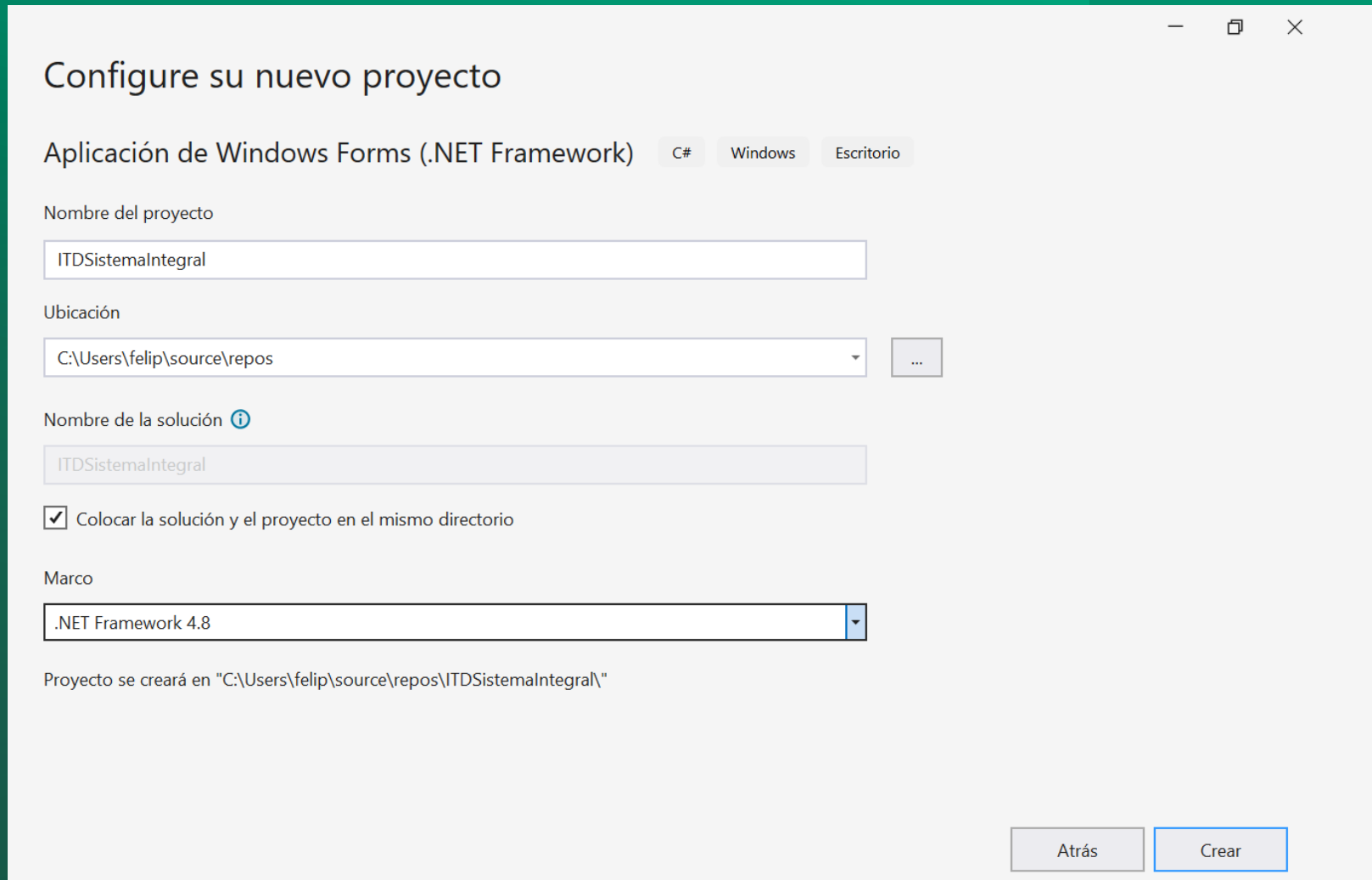


[Continuar sin código →](#)

3.2 Conectividad entre DML (huésped) y C# (anfitrión)



3.2 Conectividad entre DML (huésped) y C# (anfitrión)



Configure su nuevo proyecto

Aplicación de Windows Forms (.NET Framework) C# Windows Escritorio

Nombre del proyecto

ITDSistemaIntegral

Ubicación

C:\Users\felip\source\repos

Nombre de la solución ⓘ

ITDSistemaIntegral

☒ Colocar la solución y el proyecto en el mismo directorio

Marco

.NET Framework 4.8

Proyecto se creará en "C:\Users\felip\source\repos\ITDSistemaIntegral\"

Atrás Crear

3.2 Conectividad entre DML (huésped) y C# (anfitrión)

The screenshot displays the Visual Studio IDE with a Windows Form titled "Form1" in Design view. The form contains a title "Registro de Nueva Carrera" and several input fields: "Clave Oficial" (highlighted with a red rectangle), "Nombre", "Fecha de Inicio" (set to 01/10/2024), and "Total de Créditos". At the bottom are "Guardar" and "Cancelar" buttons.

The "Cuadro de herramientas" (Toolbox) on the left shows the "TextBox" control selected. The "Propiedades" (Properties) window on the right shows the properties for the selected control, which is identified as "tbClaveOficial" of type "System.Windows.Forms.TextBox".

The "Lista de errores" (Error List) at the bottom shows 0 Errors, 0 Advertencias, and 0 de 2 Mensajes. The "Lista de errores de búsqueda" (Search Error List) is also visible.

3.2 Conectividad entre DML (huésped) y C# (anfitrión)

Añadir este código al Botón ***Guardar*** (haciendo doble click en el botón)

```
string conexionString = "Data Source=ING-ALANIS-LAP\\SQLEXPRESS;" +  
    "Initial Catalog=ITD;Integrated Security=True;" +  
    "TrustServerCertificate=True";  
try  
{  
    string sFechaInicio = dtpFechaInicio.Value.ToString("yyyy-MM-dd");  
    String comandoString = "insert into Carreras "+  
        "(ClaveOficial, Nombre, FechaInicio, Creditos) values ('" +  
        tbClaveOficial.Text + "',''" + tbNombre.Text + "',''" +  
        sFechaInicio + "',''" + tbCreditos.Text + "')";  
    using (SqlConnection conexion = new SqlConnection(conexionString))  
    {  
        SqlCommand comando = new SqlCommand(comandoString, conexion);  
        comando.Connection.Open();  
        int numTuplas = comando.ExecuteNonQuery();  
        MessageBox.Show("¡" + numTuplas + " Alta(s) Efectuada(s)!");  
        comando.Connection.Close();  
        this.Close();  
    }  
}  
catch (Exception ex)  
{  
    if (ex.Message.ToUpper().Contains("UNIQUE"))  
        MessageBox.Show("¡CLAVE DE CARRERA YA EXISTE! ");  
    else  
        MessageBox.Show("¡Error DESCONOCIDO al dar el alta! " +  
            ex.Message);  
}
```

Para poder usar estas clases de ADO.NET hay que colocar las directivas:
using Microsoft.Data.SqlClient; // NET 8.0
using System.Data.SqlClient; // NET Framework 4.8 o anteriores

3.2 Conectividad entre DML (huésped) y C# (anfitrión)

Form1

Alta de Nueva Carrera

Clave Oficial

Nombre

Fecha de Incorporación ▼

Total de Créditos

3.2 Conectividad entre DML (huésped) y C# (anfitrión)

- La intención de los ejercicios de esta unidad no es aprender el lenguaje C# ni la plataforma de desarrollo .NET.
- El objetivo es enfatizar que las aplicaciones no deben contener restricciones:
 - Los programadores deben limitarse a hacer formularios y atrapar los mensajes de error producidos por el DBMS de manera que todas las aplicaciones, sin importar la plataforma (móvil, web o escritorio) se comporten igual.
 - En mucho software, las aplicaciones de diversas plataformas se comportan de manera distinta porque contienen restricciones que debieran estar establecidas en la base de datos.

3.2 Conectividad entre DML (huésped) y C# (anfitrión)

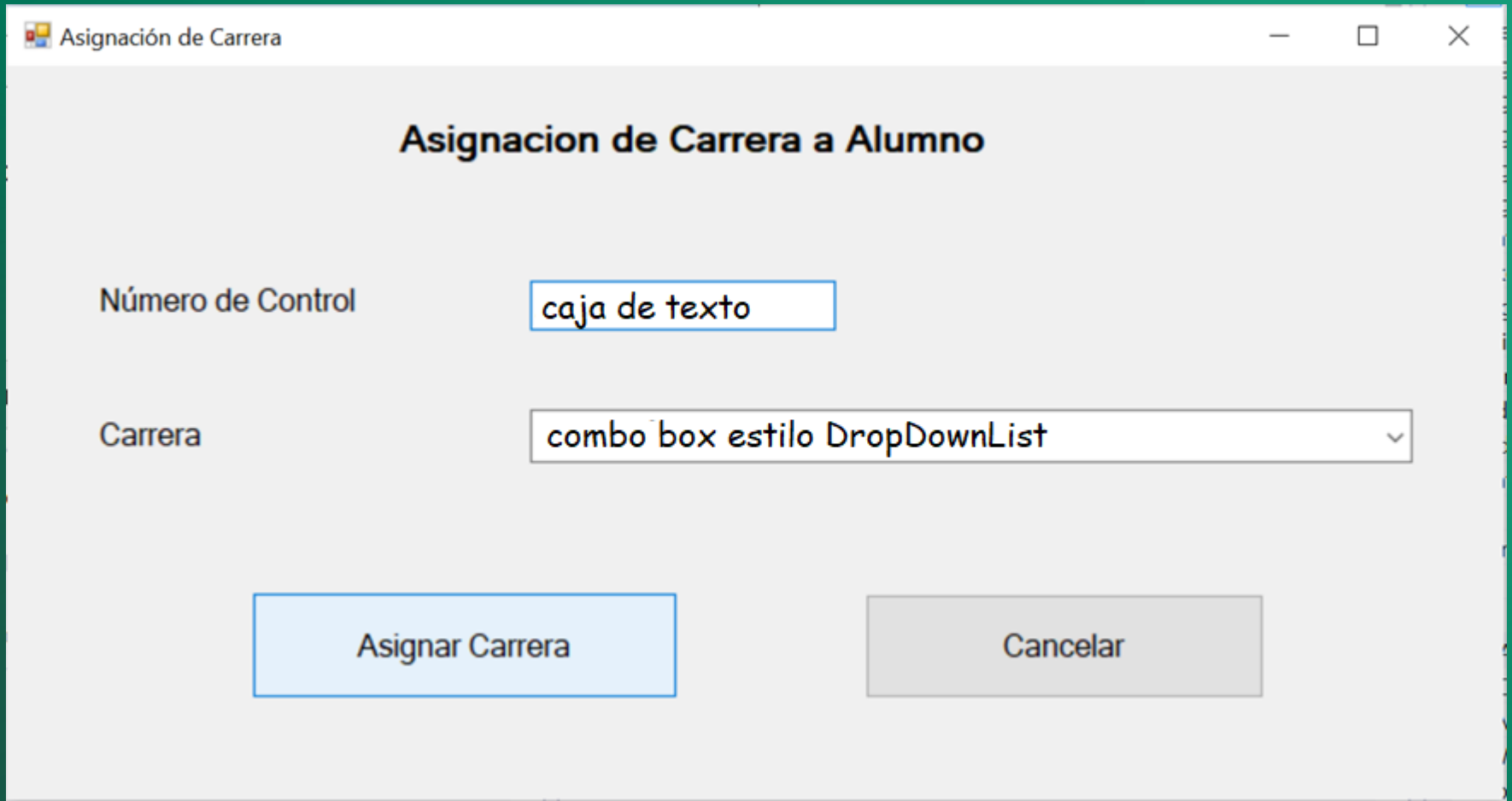
- Añada una restricción check a cada uno de los siguientes atributos y atrape el error para mostrar mensajes adecuados:
 - *ClaveOficial y Nombre.*
 - Un valor ausente en una caja de texto no es un valor NULO, por ello la restricción de NO NULOS no es eficaz.
 - Hay que establecer una restricción check para evitar que se dejen en blanco estos atributos.
 - *Creditos.*
 - Debe encontrarse en el rango de 400 a 440.

3.2 Conectividad entre DML (huésped) y C# (anfitrión)

- En primer lugar establezca en la Base de Datos, las restricciones de la diapositiva anterior y haga diversas pruebas para verificar que se apliquen.
 - Inicialmente no haga ningún cambio a la aplicación.
- Una vez que haya verificado el funcionamiento con las pruebas realizadas, ahora si haga cambios a la aplicación para que los mensajes de error se muestren de la manera más amigable para el usuario.
 - Atrape en la aplicación también el error de truncado si se captura una cadena mayor al tamaño de los atributos en el esquema.

3.2 Conectividad entre DML (huésped) y C# (anfitrión)

Crear este formulario para la asignación de la carrera a un estudiante



The screenshot shows a Windows application window titled "Asignación de Carrera". The window has a standard Windows title bar with minimize, maximize, and close buttons. The main content area has a light gray background and is titled "Asignacion de Carrera a Alumno" in bold black text. Below the title, there are two input fields. The first is labeled "Número de Control" and contains the text "caja de texto". The second is labeled "Carrera" and contains the text "combo box estilo DropDownList" with a downward arrow on the right. At the bottom of the window, there are two buttons: "Asignar Carrera" (highlighted in light blue) and "Cancelar" (grayed out).

Asignación de Carrera

Asignacion de Carrera a Alumno

Número de Control caja de texto

Carrera combo box estilo DropDownList

Asignar Carrera Cancelar

```

string conexionString = "Data Source=ING-ALANIS-LAP\\SQLEXPRESS;" +
    "Initial Catalog=ITD;Integrated Security=True;TrustServerCertificate=True";
using (SqlConnection conexion = new SqlConnection(conexionString))
{
    //
    // Obtener idAlumno
    //
    string consulta = "SELECT idAlumno FROM Alumnos WHERE NumControl='" + tbNumControl.Text + "'";
    SqlCommand comando = new SqlCommand(consulta, conexion);
    try
    {
        comando.Connection.Open();
        if (comando.ExecuteScalar() == null)
            throw new ArgumentException("Número de Control no Existe");
        iIdAlumno = (Int32)comando.ExecuteScalar();
        //MessageBox.Show("Alumno " + iIdAlumno.ToString());
    }
    catch (Exception ex)
    {
        string mensajeError = ex.Message;
        MessageBox.Show(ex.Message);
        this.Close();
    }
    comando.Connection.Close();
    //
    // Obtener idCarrera
    //
    consulta = "SELECT idCarrera FROM Carreras WHERE Nombre='" + cbCarrera.Text + "'";
    comando = new SqlCommand(consulta, conexion);
    try
    {
        comando.Connection.Open();
        if (comando.ExecuteScalar() == null)
            throw new ArgumentException("Carrera no Existe");
        iIdCarrera = (Int32)comando.ExecuteScalar();
    }
    catch (Exception ex)
    {
        string mensajeError = ex.Message;
        MessageBox.Show(ex.Message);
        this.Close();
    }
    comando.Connection.Close();
}

```

ExecuteScalar() regresa el valor de la primera tupla y la primera columna del resultado de la consulta. El resto de las columnas y tuplas se ignoran.

Añadir este código al Botón
Asignar Carrera

```

//
// Añadir la tupla en AlumnosCarreras
//
try
{
    String comandoString = "INSERT INTO AlumnosCarreras VALUES (" +
        iIdAlumno.ToString() + "," + iIdCarrera.ToString() + ",0,1)";
    using (SqlConnection conexion = new SqlConnection(conexionString))
    {
        SqlCommand comando = new SqlCommand(comandoString, conexion);
        comando.Connection.Open();
        int numTuplas = comando.ExecuteNonQuery();
        MessageBox.Show("¡Asignación Efectuada Correctamente!");
        tbNumControl.Text = "";
    }
}
catch (Exception ex)
{
    if (ex.Message.ToUpper().Contains("FOREIGN") && ex.Message.ToUpper().Contains("idAlumno"))
        MessageBox.Show("¡Número de Control no existe! ");
    else if (ex.Message.ToUpper().Contains("FOREIGN") && ex.Message.ToUpper().Contains("idCarrera"))
        MessageBox.Show("¡Carrera no existe! ");
    else if (ex.Message.ToUpper().Contains("UNIQUE"))
        MessageBox.Show("¡ERROR. Alumno ya tiene carrera asignada! ");
    else
        MessageBox.Show(ex.Message);
}
//
// Final de Añadir Tupla en AlumnosCarreras
//

```

Este código es complemento de la diapositiva anterior

3.2 Conectividad entre DML (huésped) y C# (anfitrión)

Asignación de Carrera

Asignacion de Carrera a Alumno

Número de Control

Carrera

Asignar carrera a un estudiante

Asignación de Carrera

Asignacion de Carrera a Alumno

Número de Control

Carrera

Intentar asignar otra carrera al mismo estudiante

¡ERROR! Alumno ya tiene carrera asignada!

Aceptar

Unidad 3

Conectividad de Base de Datos

Ejercicio:

Cree un formulario para registrar los datos de un alumno en forma completa (tanto los que pertenecen a la tabla Alumnos como Personas) y guardarlos en la Base de Datos en las tablas correspondientes.

Unidad 3

Conectividad de Base de Datos

Considere que debe hacer en primer lugar el insert en la tabla ***Personas***.

Tabla **PERSONAS**

<u>IdPersona</u>	Nombre	Apellidos	Calle	<u>NumExt</u>	<u>Poblacion</u>	<u>Pais</u>	<u>FechaNac</u>	Sexo	CURP
1	Parejita	López	Zarco	123	Cd de México	México	07-02-1981	Hombre	L1
2	<u>Johaness</u>	Gutenberg	Carlomagno	1	Mainz	Alemania	12-01-1398	Hombre	G2
3	Benito	Juárez García	Monte Albán	100	Cd de México	México	21-03-1806	Hombre	J4
4	Luis	Pasteur	Campos Elíseos	234	París	Francia	20-03-1850	Hombre	P1
5	Abraham		Calle del	347	Jerusalén	Israel	11-04-1890	Hombre	A0
6	José	Revueltas	Negrete	1002	Durango	México	24-03-1982	Hombre	R7
7	Lorena	Ochoa	Fresno	1410	Guadalajara	México	23-06-1981	Mujer	O1
8	Aristóteles		Templo Atenea	542	Atenas	Grecia	23-07-1905	Hombre	A1
9		Tchaikovski	Plaza Roja	471	Moscú	Rusia	13-08-1920	Hombre	T4
10		Botticelli	Filipo <u>Lippi</u>	2	Floencia	Italia	07-09-1919	Hombre	B9
11	José Luis	López	Francisco Zarco	123	Cd de México	México	09-03-1961	Hombre	JLL01
12	<u>Friele</u>	<u>Gensfleisch</u>	Carlomagno	1	Mainz	Alemania	09-03-1370	Hombre	GF7
13	Antonio	Maza	Guelaguetza	201	Oaxaca	México	15-05-1780	Hombre	AM01
14	Margarita	Maza	Monte Albán	100	Cd de México	México	19-08-1820	Mujer	MM01

Unidad 3

Conectividad de Base de Datos

Una vez ejecutado el *insert* en la tabla ***Personas***, se obtiene el valor de la *idPersona* y se podrá completar el *insert* en la tabla ***Alumnos***.

Tabla **ALUMNOS**

<u>IdAlumno</u>	<u>NumControl</u>	<u>EscuelaProcede</u>	<u>IdPersona</u>
1	98040151	Prepa PUMAS	1
2	97040587	Palacio Nacional	3
3	97040014	Colegio Vizcaya	6
4	96040121	LPGA	7
5	98040150	Colegio Alemán	2

Unidad 3

Conectividad de Base de Datos

Observaciones:

- Como en la tabla ***Alumnos*** hay integridad referencial hacia ***Personas***, primero hay que añadir la tupla en personas para obtener el valor de la ***idPersona*** y poder insertar la tupla en *Alumnos*.
- El control de las excepciones con ***try-catch*** debe considerar las llaves únicas en ambas tablas.
- Considere, que en la tabla *Personas*, los valores válidos para el atributo *Sexo* no deben asignarse como constantes sino tomarse de la Base de Datos:
 - Si se toman de una tabla de dominio, el procedimiento es sencillo como se pudo ver en el ejemplo de asignación de carrera.
 - Si se tiene el dominio en una restricción Check, los valores se pueden tomar del Diccionario de Datos (tabla ***sys.check_constraints***) como lo veremos en la siguiente unidad.

Unidad 3

Conectividad de Base de Datos

Algunas situaciones a considerar para los formularios Alta de Alumnos o Maestros:

- ¿Qué acciones se deben tomar si en la tabla personas ya existe una tupla para la persona que se está registrando?
 - ¿Primero capturar la CURP para verificar si ya existen datos de esa persona en la BD?
 - ¿Hacer un *try-catch* para que en caso de que falle el *insert* por llave única en la tabla Personas se muestre la información para efectos de revisión y solo añadir los datos en la tabla Alumnos o Maestros según corresponda?