

//Programacion I

//Guia Final

List<Contenido> indice

Paso 1 – Acceso Datos.....	2
Paso 2 – Modelar Entidades.....	3
Paso 3 – Servicio.....	4
Paso 4 – Front.....	7
CargarCombos()	7
CargarLista()	8
Habilitar()	9
Paso 5 – Front Controles.....	10
Botón Grabar.....	10
ValidarDatos()	12
Existe()	14
LimpiarCampos().....	14
Botón Nuevo.....	15
Botón Editar.....	15
Botón Borrar.....	16
Botón Salir.....	16
Botón Cancelar.....	16
List Box Selected Index Changed.....	16

Paso 1 – Acceso Datos

```
using System.Data;
using System.Data.SqlClient;
using System.Linq;
```

Tener un **string de conexion**, la **conexion**, y el **comando**

```
private readonly string stringConexion = @"Data Source=NEHO-PC\SQLEXPRESS;Initial Catalog=Albumes;Integrated Security=True";
private readonly SqlConnection conexion;
private SqlCommand comando;
```

Hacer el constructor inicializando el string de conexión

```
0 referencias
public AccesosDatos()
{
    conexion = new SqlConnection(stringConexion);
}
```

Los primeros métodos son **Desconectar** y **Conectar** para usarlos dentro de los otros métodos que vamos a tener

```
public void Desconectar()
{
    conexion.Close();
}
2 referencias
private void Conectar()
{
    conexion.Open();
    comando = new SqlCommand();
    comando.Connection = conexion;
    comando.CommandType = CommandType.Text;
}
```

Método para hacer un **SELECT** y obtener una **TABLA**, lo usamos para llenar el *combo box* y la *list box*, usa como parámetro un nombre para hacerle select a la tabla que vos quieras

```
0 referencias
public DataTable SelectTabla(string nombre)
{
    DataTable tabla = new DataTable();
    Conectar();
    comando.CommandText = "SELECT * FROM " + nombre; // + " ORDER BY 2";
    tabla.Load(comando.ExecuteReader());
    Desconectar();
    return tabla;
}
```

Método para **Ejecutar**, lo vamos a usar cuando hagamos click en *Grabar* o *Borrar*, se encarga del **insert, update, delete**

```
0 referencias
public bool Ejecutar(string sql, List<Parametro> parametros)
{
    int filasAfectadas = 0;
    Conectar();
    comando.CommandText = sql;
    foreach (Parametro param in parametros)
    {
        comando.Parameters.AddWithValue(param.Clave, param.Valor);
    }
    filasAfectadas = comando.ExecuteNonQuery();
    Desconectar();
    return filasAfectadas == 1;
}
```

Paso 2 – Modelar Entidades

Modelar la entidad **Album**

ALBUMES			
	Column Name	Condensed Type	Nullable
🔑	ID_ALBUM	int	No
	NOMBRE	varchar(100)	Yes
	AUTOR	varchar(100)	Yes
	PUBLICACION	date	Yes
	ID_GENERO	int	Yes
	ID_DISCOGR...	int	Yes
	PRODUCTOR...	varchar(100)	Yes
	DURACION	varchar(8)	Yes
	VINILO	bit	Yes
	CD	bit	Yes
	CASETE	bit	Yes
	DIGITAL	bit	Yes
	STREAMING	bit	Yes

```
0 referencias
public int Id { get; set; }
1 referencia
public string Nombre { get; set; }
0 referencias
public string Autor { get; set; }
0 referencias
public DateTime Publicacion { get; set; }
0 referencias
public int Genero { get; set; }
0 referencias
public int Discografica { get; set; }
0 referencias
public string Productor { get; set; }
0 referencias
public string Duracion { get; set; }
0 referencias
public bool Vinilo { get; set; }
0 referencias
public bool Cd { get; set; }
0 referencias
public bool Casete { get; set; }
0 referencias
public bool Digital { get; set; }
0 referencias
public bool Streaming { get; set; }
```

Por cada *registro* en la tabla álbumes agregamos una *propiedad* a la clase **Album** con el tipo de variable que corresponda.

Hacemos un **override** en el **método ToString()** para que al llenar la *list box* se llene con el *Nombre* (Property) de los álbumes y no con la ubicación del archivo

```
0 referencias
public override string ToString()
{
    ...
    return Nombre;
}
```

Paso 3 – Servicio

La clase que va a ser utilizada por el *front* para ejecutar cosas mediante *acceso datos* en la BD

```
public class Servicio
{
    private AccesoDatos oAccesoDatos;

    0 referencias
    public Servicio()
    {
        oAccesoDatos = new AccesoDatos();
    }
}
```

Hacemos un objeto AccesoDatos **oAccesoDatos** para acceder a los **métodos** de acceso datos **SelectTabla** y **Ejecutar**.

Usamos la carpeta de entidades para tener acceso a la clase **Album**.

```
using AlbumesFinal.Entidades;
```

Hacemos método **GRABAR** para hacer un **Insert** o **Update** dependiendo si el objeto es nuevo o no, eso lo maneja el front, le pasamos entonces el bool de si es nuevo o no, y el objeto en cuestión que vamos a Updatear/Insertar.

```
0 referencias
public bool Grabar(Album oAlbum, bool esNuevo)
{
}
```

Hacemos el **String SQL** para pasarle al **comando** del método **EJECUTAR** del **acceso datos**.

```
bool hecho = false;
string sql;

if (esNuevo)
    sql = "INSERT INTO ALBUMES VALUES(@id, @nombre, @autor, @publicacion, @genero, @discografica, @productor, @duracion, @vinilo, @cd, @casete, @digital, @streaming)";
else
    sql = "UPDATE ALBUMES SET NOMBRE = @nombre, AUTOR = @autor, PUBLICACION = @publicacion, ID_GENERO = @genero, ID_DISCOGRAFICA = @discografica, ID_PRODUCTOR = @productor, DURACION = @duracion, VINILO = @vinilo, CD = @cd, CASETE = @casete, DIGITAL = @digital, STREAMING = @streaming";
```

Dependiendo el caso o se guarda en la variable un Insert o un Update.

Guardamos en una **lista de parámetros** cada propiedad del objeto que vamos a grabar (álbumes en este caso)

```
//Convertimo un objeto Gato en una lista de parámetros
List<Parametro> parametros = new List<Parametro>();
parametros.Add(new Parametro("@id", oAlbum.Id));
parametros.Add(new Parametro("@nombre", oAlbum.Nombre));
parametros.Add(new Parametro("@autor", oAlbum.Autor));
parametros.Add(new Parametro("@publicacion", oAlbum.Publicacion));
parametros.Add(new Parametro("@genero", oAlbum.Genero));
parametros.Add(new Parametro("@discografica", oAlbum.Discografica));
parametros.Add(new Parametro("@productor", oAlbum.Productor));
parametros.Add(new Parametro("@duracion", oAlbum.Duracion));
parametros.Add(new Parametro("@vinilo", oAlbum.Vinilo));
parametros.Add(new Parametro("@cd", oAlbum.Cd));
parametros.Add(new Parametro("@casete", oAlbum.Casete));
parametros.Add(new Parametro("@digital", oAlbum.Digital));
parametros.Add(new Parametro("@streaming", oAlbum.Streaming));
```

Devolvemos como true o false si se pudo grabar para visualizarlo en el front como un *mensaje*.

```
hecho = oAccesoDatos.Ejecutar(sql, parametros);
```

Siguiendo la misma lógica del método Grabar hacemos un método **Delete**

```
0 referencias
public bool Delete(int id)
{
    string sql = "DELETE FROM ALBUMES WHERE ID_ALBUM = @id";
```

El **string SQL** tiene lo que va a usar entonces el **cmd** en **AccesoDatos EJECUTAR**, guardamos en la lista de los parámetros **únicamente el id** para hacer el **delete**.

```
List<Parametro> parametros = new List<Parametro>();
parametros.Add(new Parametro("@id", id));

return oAccesoDatos.Ejecutar(sql, parametros);
```

Y devolvemos como un booleano simplificado en este caso el estado del delete para informar en front.

El último paso del servicio seria llenar el **List Box** con una **lista de objetos para poder clickearlos y llenar los campos**, vamos a hacer un método que utiliza el método **SelectTabla** que hace un **select * from** del nombre de la tabla que quieras y armar un objeto por cada fila que recibamos, luego cada objeto lo agregamos en una lista de objetos álbumes y **devolvemos la lista de objetos en vez de el Data Table** que nos devuelve el SelectTabla del acceso datos.

Tenemos el siguiente Select:

ID_ALBUM	NOMBRE	AUTOR	PUBLICACION	ID_GENERO	ID_DISCOGRAFICA	PRODUCTOR	DURACION	VINILO	CD	CASETE	DIGITAL	STREAMING
1	Ojos Modemos	Charly Garcia	1983-11-05	1	1	Charly Garcia y Joe Blaney	33:04	1	1	1	1	1
2	Piano Bar	Charly Garcia	1984-09-22	2	2	Charly Garcia y Joe Blaney	35:02	1	1	0	1	1
3	Yendo de la cama al living	Charly Garcia	1982-08-28	3	2	Charly Garcia	34:35	1	1	1	1	1
4	Almendra	Almendra	1969-11-29	4	3	Estudios TNT	37:35	1	1	1	1	1
5	Almendra II	Almendra	1970-12-09	4	4	Estudios TNT	79:00	1	1	1	1	1
6	Artaud	Pescado Rabioso	1973-06-26	5	5	Jorge Alvarez y Luis Alberto Spinetta	36:56	1	1	0	1	1
7	Espuma	Duplat	2020-09-04	6	6	Duplat	32:49	0	0	0	1	1
8	In Rainbows	Radiohead	2007-10-10	7	7	Nigel Godrich	42:39	1	1	0	1	1
9	Pablo Honey	Radiohead	1993-02-22	7	8	Sean Slade, Paul Q. Kolderie	42:11	0	1	1	0	1
10	Some Kind of Cadwallader	Algernon Cadwallader	2008-05-24	8	9	Chupacabra	38:40	1	1	0	1	1
11	American Football	American Football	1999-09-14	9	10	Brendan Gamble	40:52	0	1	0	0	1
12	Apricot Princess	Rex Orange County	2017-04-26	10	11	Rex Orange County, Ben Baptie, Jeff Kleinman, T...	39:30	1	1	0	1	1
13	The Getaway	Red Hot Chili Peppers	2016-06-17	7	12	Danger Mouse	53:48	1	1	0	1	1
14	Raro	El Cuarteto de Nos	2006-03-20	7	13	Juan Campodónico	45:19	1	1	0	1	1
15	Portladio	El Cuarteto de Nos	2012-04-25	7	12	Juan Campodónico	49:21	1	1	0	1	1
16	T H E	Tricot	2013-10-02	8	14	Masayuki Nakano	48:18	1	1	0	1	1
17	The Rise and Fall of Ziggy Stardust and the Spid...	David Bowie	1972-06-16	11	3	Ken Scott, David Bowie	38:29	1	1	1	1	1
18	Blackstar	David Bowie	2016-01-08	12	3	David Bowie, Tony Visconti	39:17	1	1	0	1	1
19	este cd no tiene nombre	Superheroes	2008-01-01	7	11	Superheroes	33:10	0	0	0	0	0

Primero hacemos una lista de álbumes llamada **"lst"**

```
0 referencias
public List<Album> ListarAlbumes()
{
    List<Album> lst = new List<Album>();

    DataTable t = oAccesoDatos.SelectTabla("Albumes");
```

Y guardamos la respuesta del SelectTabla que es un **Data Table** lo que devuelve en una variable **"t"**.

Creamos un objeto álbum nulo,

```
Album oAlbum = null;
```

Recorremos cada Fila de las columnas de la variable "t" que era el **select * from** álbumes.

```
foreach (DataRow fila in t.Rows)
{
    //Creamos el Objeto
    oAlbum = new Album();

    oAlbum.Id = Convert.ToInt32(fila["ID_ALBUM"]);

    oAlbum.Nombre = fila["NOMBRE"].ToString();
    oAlbum.Autor = fila["AUTOR"].ToString();

    oAlbum.Publicacion = Convert.ToDateTime(fila["PUBLICACION"]);

    oAlbum.Genero = Convert.ToInt32(fila["ID_GENERO"]);
    oAlbum.Discografica = Convert.ToInt32(fila["ID_DISCOGRAFICA"]);

    oAlbum.Productor = fila["PRODUCTOR"].ToString();
    oAlbum.Duracion = fila["DURACION"].ToString();

    oAlbum.Vinilo = Convert.ToBoolean(fila["VINILO"]);
    oAlbum.Cd = Convert.ToBoolean(fila["CD"]);
    oAlbum.Casete = Convert.ToBoolean(fila["CASETE"]);
    oAlbum.Digital = Convert.ToBoolean(fila["DIGITAL"]);
    oAlbum.Streaming = Convert.ToBoolean(fila["STREAMING"]);

    lst.Add(oAlbum);
}
return lst;
```

Y creamos un objeto album nuevo para **guardar cada Columna como una propiedad del objeto**, por cada columna se le guarda dependiendo el nombre de la columna el valor, y se lo convierte a el valor utilizado por el objeto Album.

	ID_ALBUM	NOMBRE	AUTOR	PUBLICACION	ID_GENERO	ID_DISCOGRAFICA	PRODUCTOR	DURACION	VINILO	CD	CASETE	DIGITAL	STREAMING
1	1	Clica Modemos	Charly Garcia	1983-11-05	1	1	Charly Garcia y Joe Blaney	33.04	1	1	1	1	1
2	2	Piano Bar	Charly Garcia	1984-09-22	2	2	Charly Garcia y Joe Blaney	35.02	1	1	0	1	1

Una vez llega a la última columna que es Streaming, **guarda el objeto** álbum en la **lista** de los objetos álbumes.

Para así volver a empezar en la fila 2 por ejemplo Piano Bar.

Finalizamos devolviendo la lista de objetos álbumes

Paso 4 – Front

Usamos las carpetas Entidades y Servicio

```
using AlbumesFinal.Entidades;
```

```
using AlbumesFinal.Servicio;
```

Creamos una **lista** de álbumes y un **objeto servicio**, y el bool **esNuevo** que es una bandera para marcar si el objeto cuando se grabe va a ser un *update* o un *delete*.

```
List<Album> listaAlbumes;  
Service oServicio;  
private bool esNuevo = false;  
1 referencia  
public frmAlbumesABM()  
{  
    InitializeComponent();  
    listaAlbumes = new List<Album>();  
    oServicio = new Servicio();  
}
```

Creamos los siguientes métodos en el **Load** de la ventana

```
1 referencia  
private void frmAlbumesABM_Load(object sender, EventArgs e)  
{  
    CargarCombos();  
    CargarLista();  
    Habilitar(true);  
}
```

CargarCombos()

CargarCombos usa el método del servicio **Select** para cargar los siguientes combos:

Genero: Discografica:

Empezando por Genero.

Creamos una tabla para géneros que será la fuente de datos del **cboGeneros**, mandamos como parámetro del objeto servicio, **Select** el **nombre de la tabla** para llenar el combo.

```
1 referencia  
private void CargarCombos()  
{  
    DataTable tablaGeneros = oServicio.Select("Genero");  
    cboGenero.DataSource = tablaGeneros;  
    cboGenero.ValueMember = tablaGeneros.Columns[0].ColumnName; // "ID_GENERO"  
    cboGenero.DisplayMember = tablaGeneros.Columns[1].ColumnName; // "GENERO"  
    cboGenero.DropDownStyle = ComboBoxStyle.DropDownList; // no permite edición, solo selección  
}
```

La tabla tiene como value member la **primera columna** (ID_GENERO)

Y el display la **segunda columna** (GENERO)

Por ultimo el clásico **drop down list**, nunca hay que olvidarse.

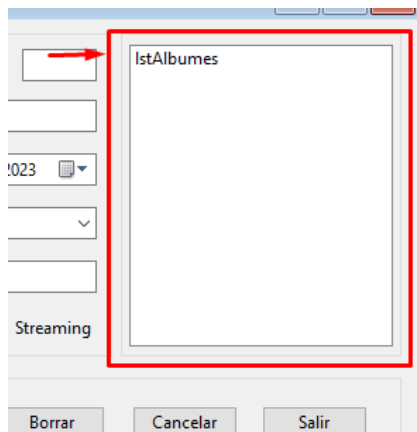
Quedaría así:

```
private void CargarCombos()
{
    DataTable tablaGeneros = oServicio.Select("GENEROS");
    cboGenero.DataSource = tablaGeneros;
    cboGenero.ValueMember = tablaGeneros.Columns[0].ColumnName; // "ID_GENERO"
    cboGenero.DisplayMember = tablaGeneros.Columns[1].ColumnName; // "GENERO"
    cboGenero.DropDownStyle = ComboBoxStyle.DropDownList; // no permite edición, solo selección

    DataTable tablaDisc = oServicio.Select("DISCOGRAFICAS");
    cboDiscografica.DataSource = tablaDisc;
    cboDiscografica.ValueMember = tablaDisc.Columns[0].ColumnName; // "ID_DISCOGRAFICA"
    cboDiscografica.DisplayMember = tablaDisc.Columns[1].ColumnName; // "NOMBRE"
    cboDiscografica.DropDownStyle = ComboBoxStyle.DropDownList; // no permite edición, solo selección
}
```

CargarLista()

CargarLista llena la **ListBox** de los álbumes llamada **lstAlbumes**



Como es fácil confundirse con **lstAlbumes** (ListBox) y **listaAlbumes** (Lista de Objetos) vamos a separarlos por colores

```
1 referencia
private void CargarLista()
{
    listaAlbumes.Clear();
    lstAlbumes.Items.Clear();
    listaAlbumes = oServicio.ListarAlbumes();
    lstAlbumes.Items.AddRange(listaAlbumes.ToArray());
}
```

```
1 referencia
private void CargarLista()
{
    listaAlbumes.Clear();
    lstAlbumes.Items.Clear();
    listaAlbumes = oServicio.ListarAlbumes();
    lstAlbumes.Items.AddRange(listaAlbumes.ToArray());
}
```

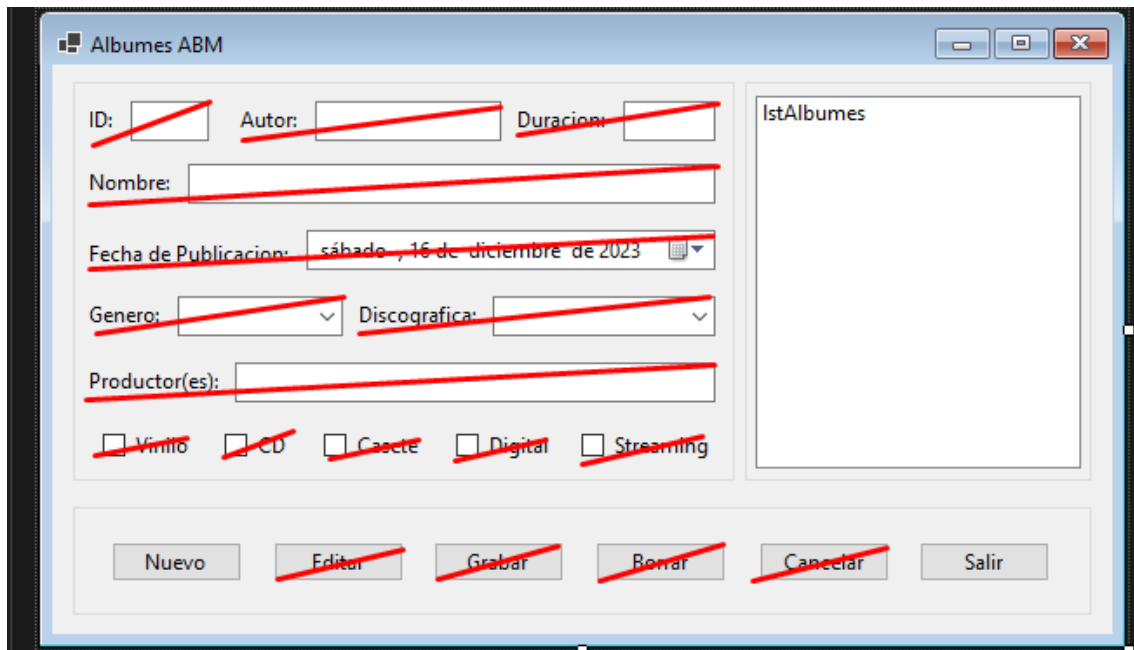

Lo primero que hacemos en el método es **Limpiar** ambas listas, el **ListBox** y la **List<Album>**.

Lo segundo es llenar la **List<Album>** con el retorno de **oServicio.ListarAlbumes()** que devolvía un **select * from** del acceso datos de los álbumes, pero estos se traducían a una lista de objetos en vez de una data table.

Luego agregamos al **ListBox** la lista de objetos para poder visualizarla en la ventana.

Habilitar()

Vamos a la ventana y marcamos que necesitamos cuando la ventana se prende y que cosas no se pueden tocar para no romper la lógica



Quedaría así luego de escribir cada control de la ventana y otorgarle un estado luego de lo que marcamos como que no tiene que estar disponible.

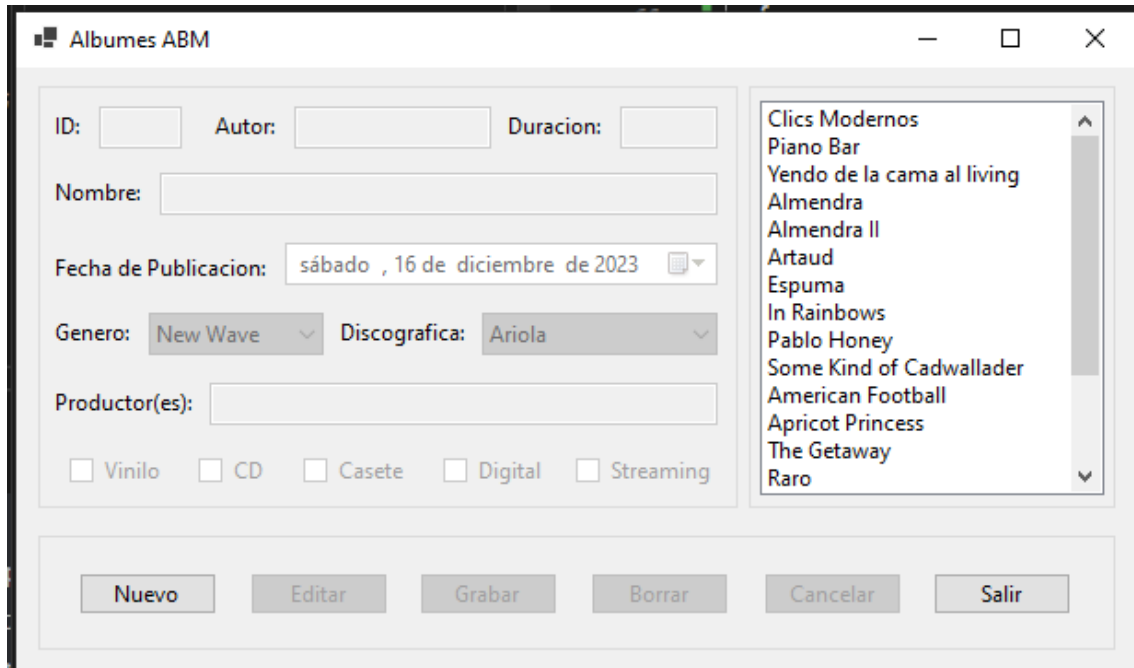
```
private void Habilitar(bool v)
{
    //no
    txtId.Enabled = !v;
    txtAutor.Enabled = !v;
    txtDuracion.Enabled = !v;
    txtNombre.Enabled = !v;
    dtpPublicacion.Enabled = !v;
    cboGenero.Enabled = !v;
    cboDiscografica.Enabled = !v;
    txtProductor.Enabled = !v;
    chkVinilo.Enabled = !v;
    chkCd.Enabled = !v;
    chkCasete.Enabled = !v;
    chkDigital.Enabled = !v;
    chkStreaming.Enabled = !v;
    btnEditar.Enabled = !v;
    btnGrabar.Enabled = !v;
    btnBorrar.Enabled = !v;
    btnCancelar.Enabled = !v;

    //si
    btnNuevo.Enabled = v;
    btnSalir.Enabled = v;
    lstAlbumes.Enabled = v;
}
```

Paso 5 – Front | Controles

Para este punto si ejecutamos el programa tendremos lo siguiente:

- ✓ -La ventana carga con algunos botones bloqueados,
- ✓ -La lista se llena con objetos Album, utilizando el override ToString() para mostrar el Nombre
- ✓ -Los ComboBox están cargados y en la posición [0]



Pero al tocar cualquier botón o al interactuar con la lista nada sucede, de eso nos vamos a encargar en el Paso 5

Primero vamos a hacer funcionar los Botones

Botón Grabar

Empezando por **Grabar**, el más importante, lo primero que hacemos es crear un *método de validación de datos* que analice las ventanas y nos permita saber si es o no valido para proseguir, el método de validación lo Crearemos luego, por el momento lo dejamos vacío.

```
//validacion de los datos
if (ValidarDatos())
{
    // ...
}

private bool ValidarDatos()
{
    throw new NotImplementedException();
}
```

En base a la respuesta, creamos el objeto utilizando cada control de la ventana para llenar los atributos del objeto Album.

```
//validacion de los datos
if (ValidarDatos())
{
    //crear objeto
    Album oAlbum = new Album();
    oAlbum.Id = Convert.ToInt32(txtId.Text);
    oAlbum.Autor = txtAutor.Text;
    oAlbum.Duracion = txtDuracion.Text;
    oAlbum.Nombre = txtNombre.Text;
    oAlbum.Publicacion = dtpPublicacion.Value;
    oAlbum.Genero = (int)cboGenero.SelectedValue;
    oAlbum.Discografica = (int)cboDiscografica.SelectedValue;
    oAlbum.Productor = txtProductor.Text;
    oAlbum.Vinilo = chkVinilo.Checked;
    oAlbum.Cd = chkCd.Checked;
    oAlbum.Casete = chkCasete.Checked;
    oAlbum.Digital = chkDigital.Checked;
    oAlbum.Streaming = chkStreaming.Checked;
}
```

Si se presionó el botón "Nuevo" antes, lo que vamos a hacer es que maneje la bandera "esNuevo" que **previene que carguemos un álbum nuevo con un id ya cargado**, informándonos, ósea prevenimos un error de insertar algo que *ya tiene un ID*

```
if (esNuevo)
{
    if (Existe(oAlbum.Id))
    {
        MessageBox.Show("Album ya cargado!", "Validación");
        return;
    }
}
```

Para hacer esto necesitamos el método Existe, por el momento lo dejamos vacío.

```
private bool Existe(int id)
{
    throw new NotImplementedException();
}

private bool ValidarDatos()
```

Por ultimo informamos que grabamos bien, Bloqueamos los controles de nuevo, y limpiamos los campos de lo que escribimos o se llenó, esto, es un nuevo método que vamos a crear después, por ultimo cargamos la lista nuevamente con el método **Cargar Lista()** para poder

refrescar lo que cambiamos o agregamos a la base de datos en la **ListBox** y que se vean los cambios

```
MessageBox.Show("Se ha grabado exitosamente!", "Exito", MessageBoxButtons.OK, MessageBoxIcon.Information);  
Habilitar(true);  
LimpiarCampos();  
CargarLista();  
}
```

```
private void LimpiarCampos()  
{  
    throw new NotImplementedException();  
}
```

ValidarDatos()

Por ejemplo nos podemos preguntar si la fecha de publicación es mayor a la del día de hoy, en ese caso estaríamos cargando un álbum que no se publico.

```
private bool ValidarDatos()  
{  
    if (dtpPublicacion.Value >= DateTime.Now)  
    {  
        MessageBox.Show("Ingrese una fecha valida...");  
        dtpPublicacion.Focus();  
        return false;  
    }  
    return true;  
}
```

Podemos agregar también luego de esa pregunta si el TextBox del ID esta vacío.

```
if (dtpPublicacion.Value >= DateTime.Now)  
{  
    MessageBox.Show("Ingrese una fecha val  
    dtpPublicacion.Focus();  
    return false;  
}  
if (txtId.Text == string.Empty)  
{  
    MessageBox.Show("Ingrese un ID...");  
    txtId.Focus();  
    return false;  
}  
return true;
```

Podemos hacer esto para cada TextBox.

Podemos validar si el ID es un **número**, para que **no se ingrese una letra**.

```
try
{
    int codigo = Int32.Parse(txtId.Text);
}
catch (Exception)
{
    MessageBox.Show("El ID solo acepta numeros");
    txtId.Focus();
    return false;
}
```

Por ultimo nos podemos preguntar si no se seleccionó nada en el combo box, ósea si el índice es -1.

```
if (cboDiscografica.SelectedIndex == -1)
{
    MessageBox.Show("No selecciono una discografica");
    cboDiscografica.Focus();
    return false;
}
if (cboGenero.SelectedIndex == -1)
{
    MessageBox.Show("No selecciono un genero");
    cboGenero.Focus();
    return false;
}
```

Me quedo una validación gigantesca, pero es una base, luego podemos poner validaciones varias y personalizadas, como por ejemplo que el productor no sea el mismo que el artista, o que no estén todos los check, o que el año no sea 2020 etc.

```
if (dtpPublicacion.Value == DateTime.Now)
{
    MessageBox.Show("Ingrese una fecha valida...");
    dtpPublicacion.Focus();
    return false;
}
if (txtId.Text == string.Empty)
{
    MessageBox.Show("Ingrese un ID...");
    txtId.Focus();
    return false;
}
if (txtAutor.Text == string.Empty)
{
    MessageBox.Show("El autor esta vacio");
    txtAutor.Focus();
    return false;
}
if (txtDuracion.Text == string.Empty)
{
    MessageBox.Show("La duracion esta vacia");
    txtDuracion.Focus();
    return false;
}
if (txtNombre.Text == string.Empty)
{
    MessageBox.Show("El nombre esta vacio");
    txtNombre.Focus();
    return false;
}
if (txtProductor.Text == string.Empty)
{
    MessageBox.Show("El campo de productor(es) esta vacio");
    txtProductor.Focus();
    return false;
}
if (txtNombre2.Text == string.Empty)
{
    MessageBox.Show("El nombre esta vacio");
    txtNombre2.Focus();
    return false;
}

try
{
    int codigo = Int32.Parse(txtId.Text);
}
catch (Exception)
{
    MessageBox.Show("El ID solo acepta numeros");
    txtId.Focus();
    return false;
}

if (cboDiscografica.SelectedIndex == -1)
{
    MessageBox.Show("No selecciono una discografica");
    cboDiscografica.Focus();
    return false;
}
if (cboGenero.SelectedIndex == -1)
{
    MessageBox.Show("No selecciono un genero");
    cboGenero.Focus();
    return false;
}
```

Existe()

Existe recorre cada ítem de la lista de álbumes que tenemos, **es la lista que usa el ListBox** como fente de información por ejemplo, y por cada álbum chequea si el ID es el mismo que el ID ingresado, si los dos son iguales, rompe el ciclo y devuelve una bandera, que nos indica que ese álbum ya se cargó y que no se puede hacer un insert. Esto depende de la bandera previa "esNuevo"

```
private bool Existe(int pk)
{
    bool encontro = false;
    for (int i = 0; i < listaAlbumes.Count; i++)
    {
        if (listaAlbumes[i].Id == pk)
        {
            encontro = true;
            break;
        }
    }
    return encontro;
}
```

LimpiarCampos()

Para limpiar toda la pantalla tenemos un **GroupBox** gigantesco encapsulando todos los **controles** donde se ingresan datos, este GroupBox es clave para limpiar rápido los campos, pero si no lo tenemos no pasa nada porque vamos a tener que limpiar cada campo uno por uno.

El nombre del GroupBox es:



Entonces recorreremos cada control del GroupBox con un ciclo Foreach, Por cada control, si el control es *Tipo de Control* lo limpiamos, o lo devolvemos al estado original.

```

1 referencia
private void LimpiarCampos()
{
    foreach (Control control in this.gbDatos.Controls)
    {
        if (control is ComboBox)
        {
            ((ComboBox)control).SelectedIndex = -1;
        }

        if (control is TextBox)
        {
            control.Text = String.Empty;
        }

        if (control is DateTimePicker)
        {
            ((DateTimePicker)control).Value = DateTime.Now;
        }

        if (control is CheckBox)
        {
            ((CheckBox)control).Checked = false;
        }
    }
}

```

Botón Nuevo

Marcamos la bandera **EsNuevo**, Habilitamos los controles, y cancelamos el botón Editar y Borrar, **Limpiamos los campos** para poder crear un objeto nuevo

```

1 referencia
private void btnNuevo_Click(object sender, EventArgs e)
{
    Habilitar(false);
    esNuevo = true;
    btnBorrar.Enabled = false;
    btnEditar.Enabled = false;
    LimpiarCampos();
}

```

Botón Editar

Es prácticamente igual, solo que este se activa si se cliquea algo en el **ListBox** previamente. Como también el botón Borrar.

```

1 referencia
private void btnEditar_Click(object sender, EventArgs e)
{
    Habilitar(false);
    esNuevo = false;
    btnBorrar.Enabled = false;
    btnEditar.Enabled = false;
}

```

Botón Borrar

Utilizamos el **índice** de la **ListBox** ya que es el mismo que el índice de la **List<Album>** para obtener el ID del álbum seleccionado y luego de confirmarlo lo borramos, y actualizamos y limpiamos la pantalla.

```
private void btnBorrar_Click(object sender, EventArgs e)
{
    int index = lstAlbumes.SelectedIndex;
    if (MessageBox.Show("Seguro que desea borrar el Album seleccionado?", "Confirmación", MessageBoxButtons.YesNo, MessageBoxIcon.Question) == DialogResult.Yes)
    {
        if (oServicio.Delete(ListaAlbumes[index].Id))
        {
            MessageBox.Show("El Album fue eliminado.", "Información");
            Habilitar(true);
            LimpiarCampos();
            CargarLista();
        }
        else
        {
            MessageBox.Show("No se pudo borrar el Album seleccionado!", "Error");
        }
    }
}
```

Botón Salir

Salir luego de una confirmación cierra la ventana.

```
1 referencia
private void btnSalir_Click(object sender, EventArgs e)
{
    if (MessageBox.Show("Seguro que desea salir?", "Confirmación", MessageBoxButtons.YesNo, MessageBoxIcon.Question) == DialogResult.Yes)
    {
        this.Close();
    }
}
```

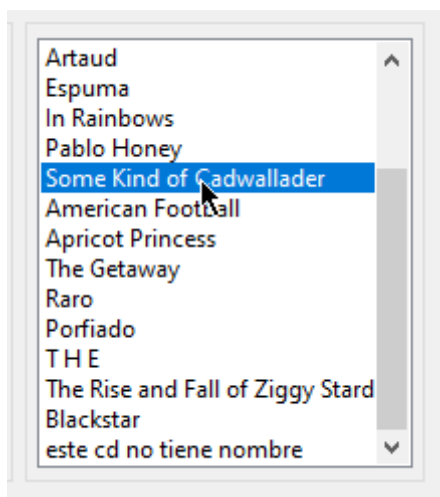
Botón Cancelar

Cancelar devuelve la pantalla a su **estado inicial** luego de una confirmación.

```
private void btnCancelar_Click(object sender, EventArgs e)
{
    if (MessageBox.Show("Seguro que desea cancelar? Los cambios no guardados se perderan", "Confirmación", MessageBoxButtons.YesNo, MessageBoxIcon.Question) == DialogResult.Yes)
    {
        LimpiarCampos();
        CargarLista();
        Habilitar(true);
    }
}
```

List Box Selected Index Changed

El evento que maneja el **click** sobre un ítem de la **ListBox**.



Lo primero que hacemos es localizamos en la **List<Album>** el objeto que clickeamos usando el índice seleccionado (*guardado como int*) del **ListBox**.


```
int selected = lstAlbumes.SelectedIndex;

if (selected != -1)
{
    Album oAlbum = listaAlbumes[selected];
}
```

Creamos un álbum a partir del ítem que está en esa posición, Luego hacemos lo opuesto al botón guardar, en vez de llenar los atributos del objeto con los campos, llenamos los campos con los atributos del objeto de la lista

```
txtId.Text = oAlbum.Id.ToString();
txtAutor.Text = oAlbum.Autor;
txtDuracion.Text = oAlbum.Duracion;
txtNombre.Text = oAlbum.Nombre;
dtpPublicacion.Value = oAlbum.Publicacion;
cboGenero.SelectedValue = oAlbum.Genero;
cboDiscografica.SelectedValue = oAlbum.Discografica;
txtProductor.Text = oAlbum.Productor;
chkVinilo.Checked = oAlbum.Vinilo;
chkCd.Checked = oAlbum.Cd;
chkCasete.Checked = oAlbum.Casete;
chkDigital.Checked = oAlbum.Digital;
chkStreaming.Checked = oAlbum.Streaming;
```

Por ultimo agregamos una mínima lógica de botones para encender Editar y Borrar, cancelar Nuevo y Salir, y traer Cancelar.

```
btnEditar.Enabled = true;
btnBorrar.Enabled = true;
btnNuevo.Enabled = false;
btnCancelar.Enabled = true;
btnSalir.Enabled = false;
```

Y ya está, la ventana funciona por completo. Podemos agregar **LimpiarCampos()** como último detalle al evento **Load** de la ventana.

Albumes ABM

ID: 19 Autor: Superheroes Duracion: 33:10

Nombre: este cd no tiene nombre

Fecha de Publicacion: martes , 1 de enero de 2008

Genero: Rock alternati Discografica: Self-released

Productor(es): Superheroes

☐ Vinilo ☐ CD ☐ Casete ☐ Digital ☒ Streaming

Artaud
Espuma
In Rainbows
Pablo Honey
Some Kind of Cadwallader
American Football
Apricot Princess
The Getaway
Raro
Porfiado
T H E
The Rise and Fall of Ziggy Stard
Blackstar
este cd no tiene nombre

Nuevo Editar Grabar Borrar Cancelar Salir

Para probar si graba voy a cargar un álbum como prueba.

Albumes ABM

ID: 20 Autor: La Maquina de Hac Duracion: 38:41

Nombre: Películas

Fecha de Publicacion: viernes , 1 de julio de 1977

Genero: Rock progresi Discografica: Talent/Microfon

Productor(es): La Maquina de Hacer Pajaros

☒ Vinilo ☒ CD ☒ Casete ☒ Digital ☒ Streaming

Clics Modernos
Piano Bar
Yendo de la cama al living
Almendra
Almendra II
Artaud
Espuma
In Rainbows
Pablo Honey
Some Kind of Cadwallader
American Football
Apricot Princess
The Getaway
Raro

Nuevo Editar Grabar Borrar Cancelar Salir

Exito
Se ha grabado exitosamente!
Aceptar

```
JOIN DISCOGRAFICAS D ON D.ID_DISCOGRAFICA = A.ID_DISCOGRAFICA
ORDER BY 1 DESC
```

select * from albumes

ID	NOMBRE	AUTOR	PUBLICACION	GENERO	DISCOGRAFICA	PRODUCTOR/ES	DURACION	VINILO	CD	CASETE	DESCARGA DIGITAL	STREAMING
20	Películas	La Maquina de Hacer Pajaros	1977-07-01	Rock progresivo	Talent/Microfon	La Maquina de Hacer Pajaros	38:41	1	1	1	1	1

El objeto se grabó efectivamente en la BD, y se actualiza la lista cuando se graba.

Albumes ABM

ID: Autor: Duracion:

Nombre:

Fecha de Publicacion: sábado , 16 de diciembre de 2023

Genero: Discografica:

Productor(es):

☐ Vinilo ☐ CD ☐ Casete ☐ Digital ☐ Streaming

Espuma
In Rainbows
Pablo Honey
Some Kind of Cadwallader
American Football
Apricot Princess
The Getaway
Raro
Porfiado
T H E
The Rise and Fall of Ziggy Stard
Blackstar
este cd no tiene nombre
Películas

Nuevo Editar Grabar Borrar Cancelar Salir

