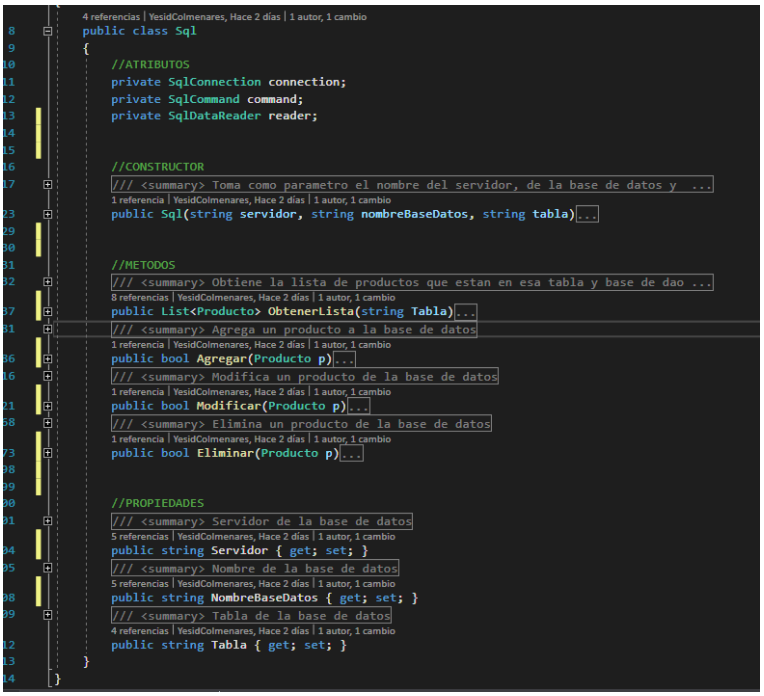
6. Conexión a bases de datos.

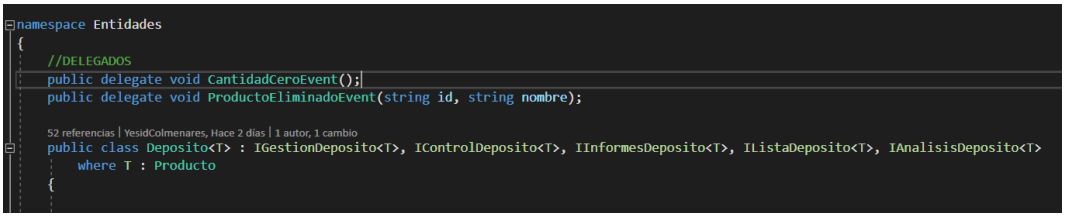
La Clase Sql es la encargada de gestionar la base de datos.
Se encuentran los métodos del CRUD



```
4 referencias | YesidColmenares, Hace 2 días | 1 autor, 1 cambio
public class Sql
{
    //ATRIBUTOS
    private SqlConnection connection;
    private SqlCommand command;
    private SqlDataReader reader;


    //CONSTRUCTOR
    /// <summary> Toma como parametro el nombre del servidor, de la base de datos y  ...
    1 referencia | YesidColmenares, Hace 2 días | 1 autor, 1 cambio
    public Sql(string servidor, string nombreBaseDatos, string tabla)...


    //METODOS
    /// <summary> Obtiene la lista de productos que estan en esa tabla y base de dao ...
    8 referencias | YesidColmenares, Hace 2 días | 1 autor, 1 cambio
    public List<Producto> ObtenerLista(string Tabla)...
    /// <summary> Agrega un producto a la base de datos
    1 referencia | YesidColmenares, Hace 2 días | 1 autor, 1 cambio
    public bool Agregar(Producto p)...
    /// <summary> Modifica un producto de la base de datos
    1 referencia | YesidColmenares, Hace 2 días | 1 autor, 1 cambio
    public bool Modificar(Producto p)...
    /// <summary> Elimina un producto de la base de datos
    1 referencia | YesidColmenares, Hace 2 días | 1 autor, 1 cambio
    public bool Eliminar(Producto p)...


    //PROPIEDADES
    /// <summary> Servidor de la base de datos
    5 referencias | YesidColmenares, Hace 2 días | 1 autor, 1 cambio
    public string Servidor { get; set; }
    /// <summary> Nombre de la base de datos
    5 referencias | YesidColmenares, Hace 2 días | 1 autor, 1 cambio
    public string NombreBaseDatos { get; set; }
    /// <summary> Tabla de la base de datos
    4 referencias | YesidColmenares, Hace 2 días | 1 autor, 1 cambio
    public string Tabla { get; set; }
}
}
```

7. Delegados

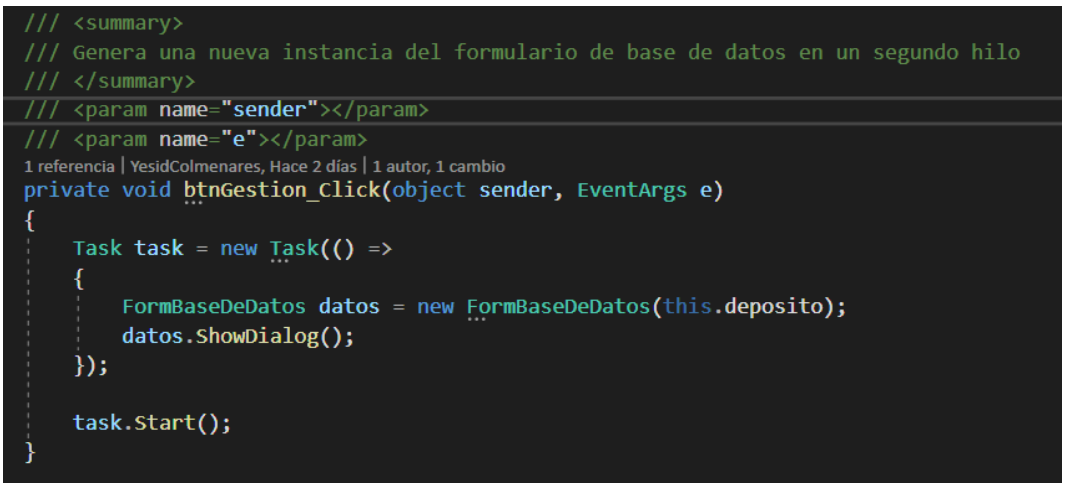
En la clase Depósito declaro dos delegados:
*CantidadCeroEvent
*ProductoEliminadoEvent



```
namespace Entidades
{
    //DELEGADOS
    public delegate void CantidadCeroEvent();
    public delegate void ProductoEliminadoEvent(string id, string nombre);

    52 referencias | YesidColmenares, Hace 2 días | 1 autor, 1 cambio
    public class Deposito<T> : IGestionDeposito<T>, IControlDeposito<T>, IInformesDeposito<T>, IListaDeposito<T>, IAnalisisDeposito<T>
        where T : Producto
    {
```

8. Expresiones lambda e Hilos

En el formulario principal creó un hilo para instanciar el formulario base de datos, con la expresión lambda le paso el delegado de tipo Action al hilo.



```
/// <summary>
/// Genera una nueva instancia del formulario de base de datos en un segundo hilo
/// </summary>
/// <param name="sender"></param>
/// <param name="e"></param>
1 referencia | YesidColmenares, Hace 2 días | 1 autor, 1 cambio
private void btnGestion_Click(object sender, EventArgs e)
{
    Task task = new Task(() =>
    {
        FormBaseDeDatos datos = new FormBaseDeDatos(this.deposito);
        datos.ShowDialog();
    });

    task.Start();
}
```

9. Eventos

En la clase depósito creó dos eventos:

```
52 referencias | YesidColmenares, Hace 2 días | 1 autor, 1 cambio
public class Deposito<T> : IGestionDeposito<T>, IControlDeposito<T>, IInformesDeposito<T>, IListaDeposito<T>, IAnalisisDeposito<T>
    where T : Producto
{

    //EVENTOS
    /// <summary>
    /// Evento que se invoca cuando el producto ingresado al deposito tiene cantidad 0
    /// </summary>
    public event CantidadCeroEvent CantidadCeroEvent;
    /// <summary>
    /// Evento que se invoca cuando el producto es eliminado del deposito
    /// </summary>
    public event ProductoEliminadoEvent ProductoEliminadoEvent;
```

Invocación de los Eventos

```
24 referencias | YesidColmenares, Hace 2 días | 1 autor, 1 cambio
public void Agregar(T t)
{
    if (this.Contiene(t))
    {
        throw new DepositoExistException();
    }
    else
    {
        if (t.Cantidad == 0)
        {
            if (this.CantidadCeroEvent is not null)
            {
                this.CantidadCeroEvent.Invoke();
            }
        }
        this.productos.Add(t);
    }
}
```

```
8 referencias | YesidColmenares, Hace 2 días | 1 autor, 1 cambio
public bool Eliminar(T t)
{
    bool valor = false;

    if (!this.Contiene(t))
    {
        throw new DepositoNotExistException();
    }
    else
    {
        valor = this.productos.Remove(t);

        if (valor && this.ProductoEliminadoEvent is not null)
        {
            this.ProductoEliminadoEvent.Invoke(t.Id, t.Nombre);
        }
    }

    return valor;
}
```

Manejador de eventos

```
//MANEJADOR DE EVENTOS PROPIOS
/// <summary>
/// Captura el evento que es generado cuando un producto es eliminado del deposito,
/// mostrando en un MessageBox el nombre y el id del producto.
/// </summary>
/// <param name="id">Id del producto eliminado</param>
/// <param name="nombre">Nombre del producto eliminado</param>
1 referencia | YesidColmenares, Hace 2 días | 1 autor, 1 cambio
private void Deposito_ProductoEliminadoEvent(string id, string nombre)
{
    MessageBox.Show($"Eliminaste el producto ID: {id}, Nombre: {nombre}", "Aviso", MessageBoxButtons.OK, MessageBoxIcon.Warning);
}
/// <summary>
/// Captura el evento que es generado cuando se crea un producto sin cantidad de stock,
/// avisa en un MessageBox
/// </summary>
1 referencia | YesidColmenares, Hace 2 días | 1 autor, 1 cambio
private void Deposito_CantidadCeroEvent()
{
    MessageBox.Show("Agregaste un producto con cantidad: 0", "Advertencia", MessageBoxButtons.OK, MessageBoxIcon.Warning);
}
```

10. Metodos de extension

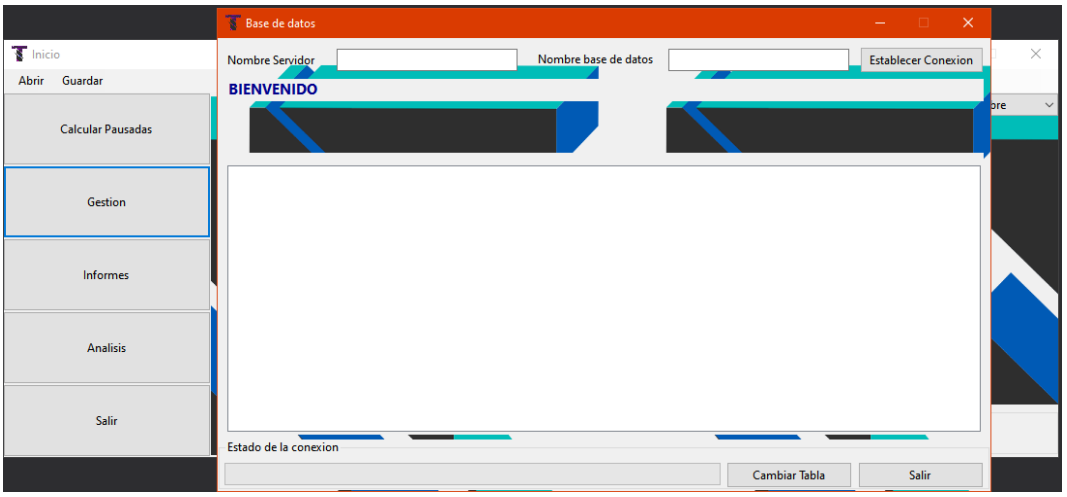
De la clase Sql generó un método de extensión llamado ProbarConexion

```
0 referencias | 0 cambios | 0 autores, 0 cambios
public static class SqlExtension
{
    /// <summary> Realiza una prueba de conexion a la base de datos
    1 referencia | 0 cambios | 0 autores, 0 cambios
    public static bool ProbarConexion(this Sql sql, string servidor, string nombreBaseDatos)
    {
        bool valor = true;
        SqlConnection connection;

        try
        {
            if (servidor != "" && nombreBaseDatos != "")
            {
                using (connection = new SqlConnection($"Server={servidor};Database={nombreBaseDatos};Trusted_Connection=True;"))
                {
                    connection.Open();
                }
            }
            else
            {
                valor = false;
            }
        }
        catch (Exception)
        {
            valor = false;
        }

        return valor;
    }
}
```

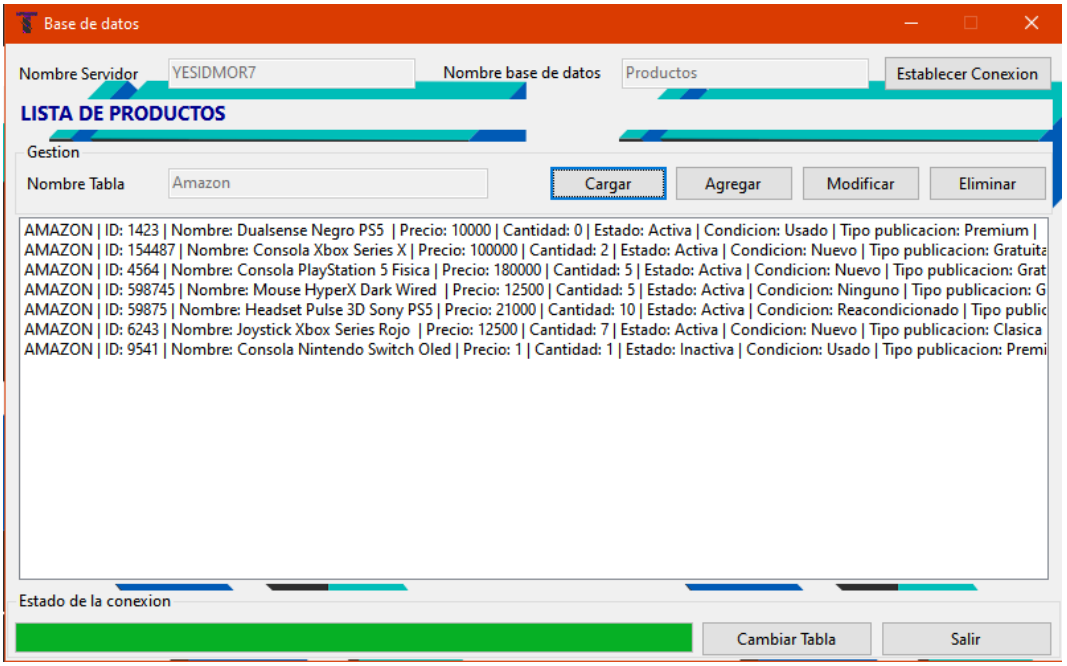
Interfaz gráfica - Windows Form

1. **Botón Gestion**

   Genera una nueva instancia del formulario base de datos.



   Ingreso el nombre del servidor, de la base de datos y el de la tabla para acceder a la lista de los productos.
   Cargar, Agregar, Modificar y Eliminar son el CRUD de la base de datos, utilizo la clase Sql.
   Con el botón cambiar Tabla puedo seleccionar otra Tabla que esté en la misma base de datos y servidor.



2. **Botón Análisis**

   Genera un análisis de todos los productos que hay en el depósito.