

Programación 3

Back-end en C#

Dr. Andrés Melgar

Arquitectura de SW (estilo arquitectónico monolítico por capas)

Presentación

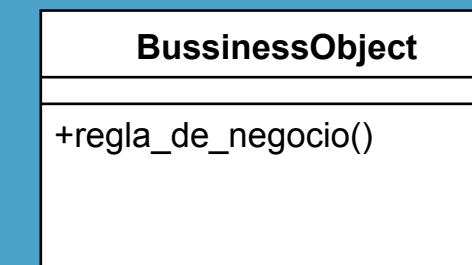
Front-End



Lógica de Negocio

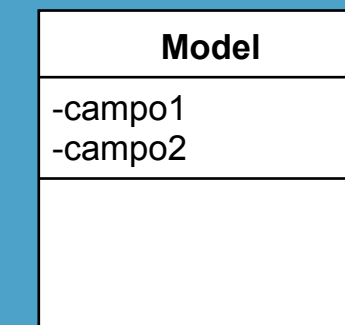


Patrón DAO
(patrón de interacción con BD Data Mapper)



Dominio

Back-End



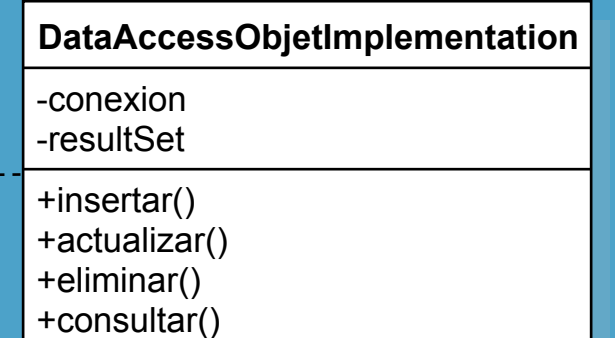
Patrones:

- Data Transfer Object (diseño)
- Identity Field (persistencia)
- Sigle Table Inherence (persistencia)

Persistencia

DataSource

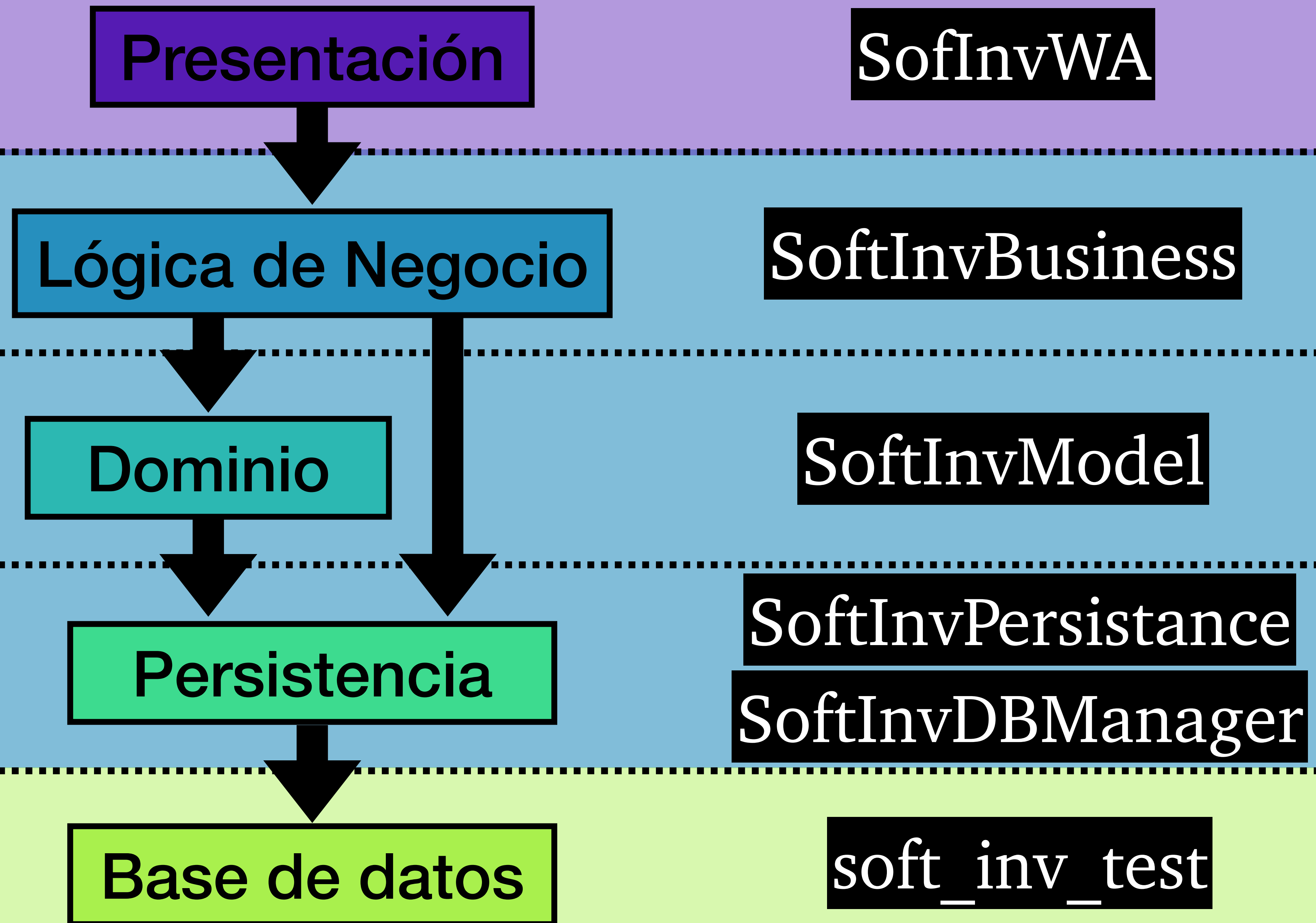
+insertar()
+actualizar()
+eliminar()
+consultar()



Base de datos



Arquitectura de SW (estilo arquitectónico monolítico por capas)



ADO.NET

ActiveX Data Objects for .NET

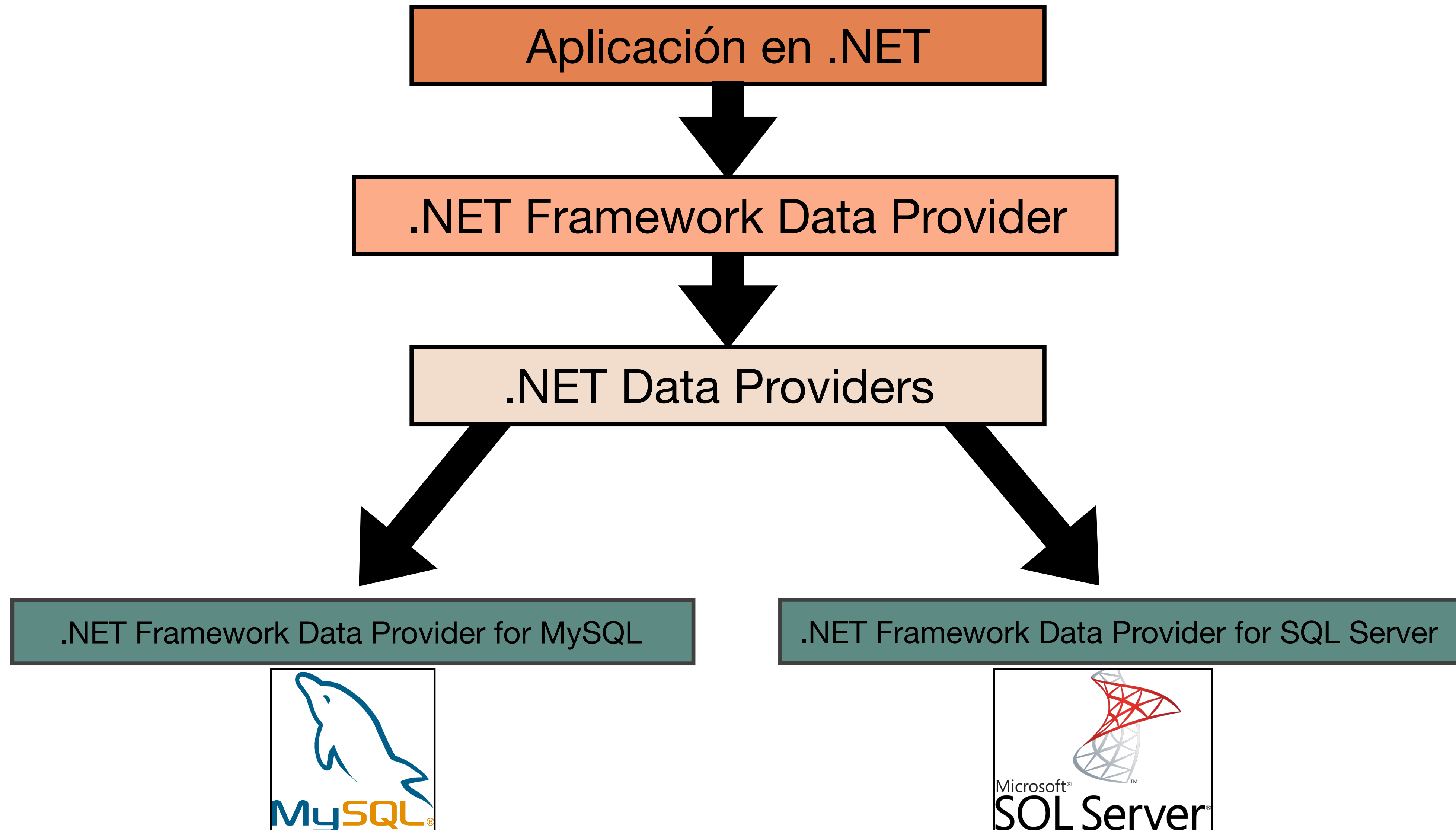
ADO.NET

Definición

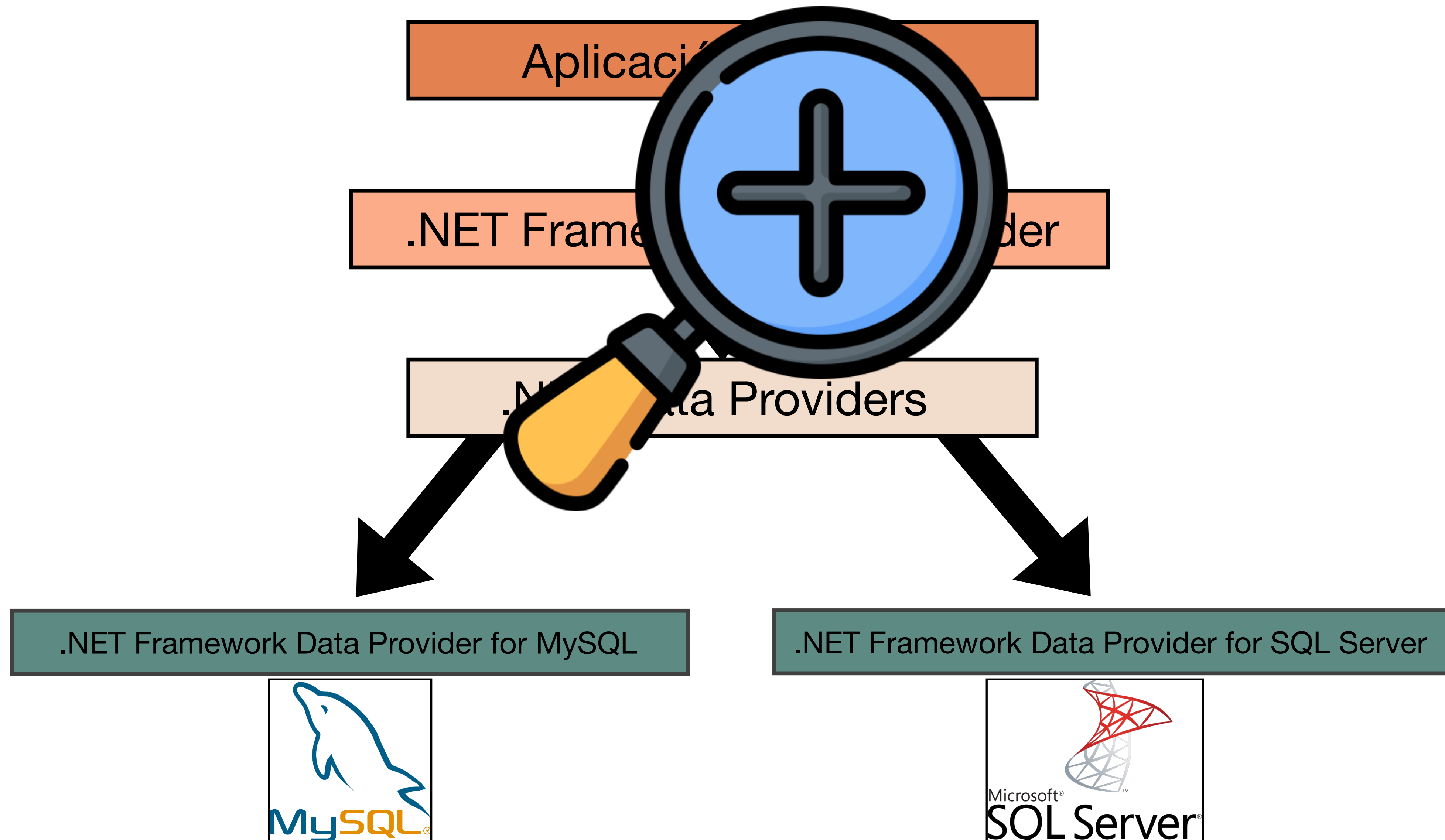
- ADO.NET (*ActiveX Data Objects for .NET*) es un conjunto de clases en .NET que permite a las aplicaciones interactuar con **bases de datos**.
- Modelo basado en proveedores que soporta **múltiples bases de datos** (SQL Server, MySQL, Oracle, entre otros) mediante proveedores específicos (SqlClient, OleDb, Odbc, OracleClient).
- Uso de **comandos parametrizados** que previene SQL Injection al trabajar con SqlCommand y Parameters.



Arquitectura ADO.NET



Arquitectura ADO.NET



Arquitectura ADO.NET



Aplicación en .Net

.NET Framework Data Provider

.NET Data Providers

.NET Framework Data Provider for DB

Clase base: DbTransaction

Transaction

Clase base: DbDataReader

Data Reader

Clase base: DbParameter

Parameter

Clase base: DbTransaction

Transaction

Command

Clase base: DbCommand

Clase base: DbConnection

Connection

.NET Data Providers

dll con implementación

Arquitectura ADO.NET

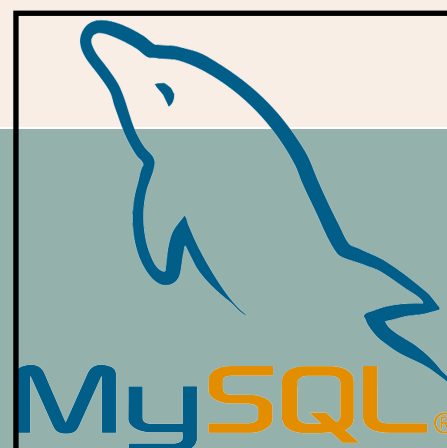


Aplicación en .Net

.NET Framework Data Provider

.NET Data Providers

.NET Framework Data Provider for MySQL



MySqlConnection

Transaction

MySqlCommand

Data Reader

MySqlParameter

Parameter

MySqlCommand

Command

MySqlConnection

Connection

.NET Data Providers

using MySql.Data.MySqlClient;

MySql.Data.dll

Arquitectura ADO.NET



Aplicación en .Net

.NET Framework Data Provider

.NET Data Providers

.NET Framework Data Provider for MSSQL

SqlTransaction

Transaction

SqlDataReader

Data Reader

SqlParameter

Parameter

SqlCommand

Command

SqlConnection

Connection

.NET Data Providers

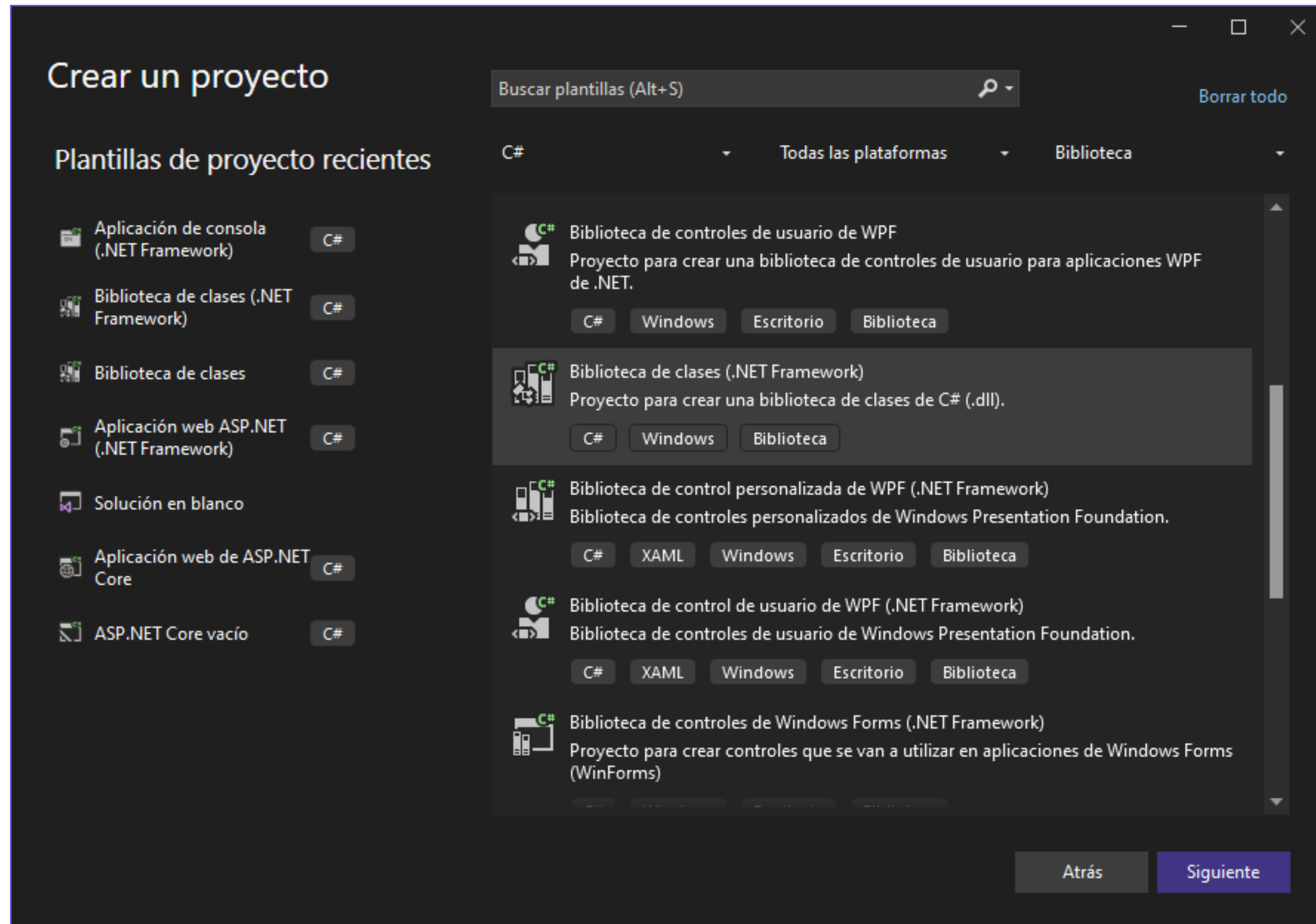
using Microsoft.Data.SqlClient;

Microsoft.Data.SqlClient.dll

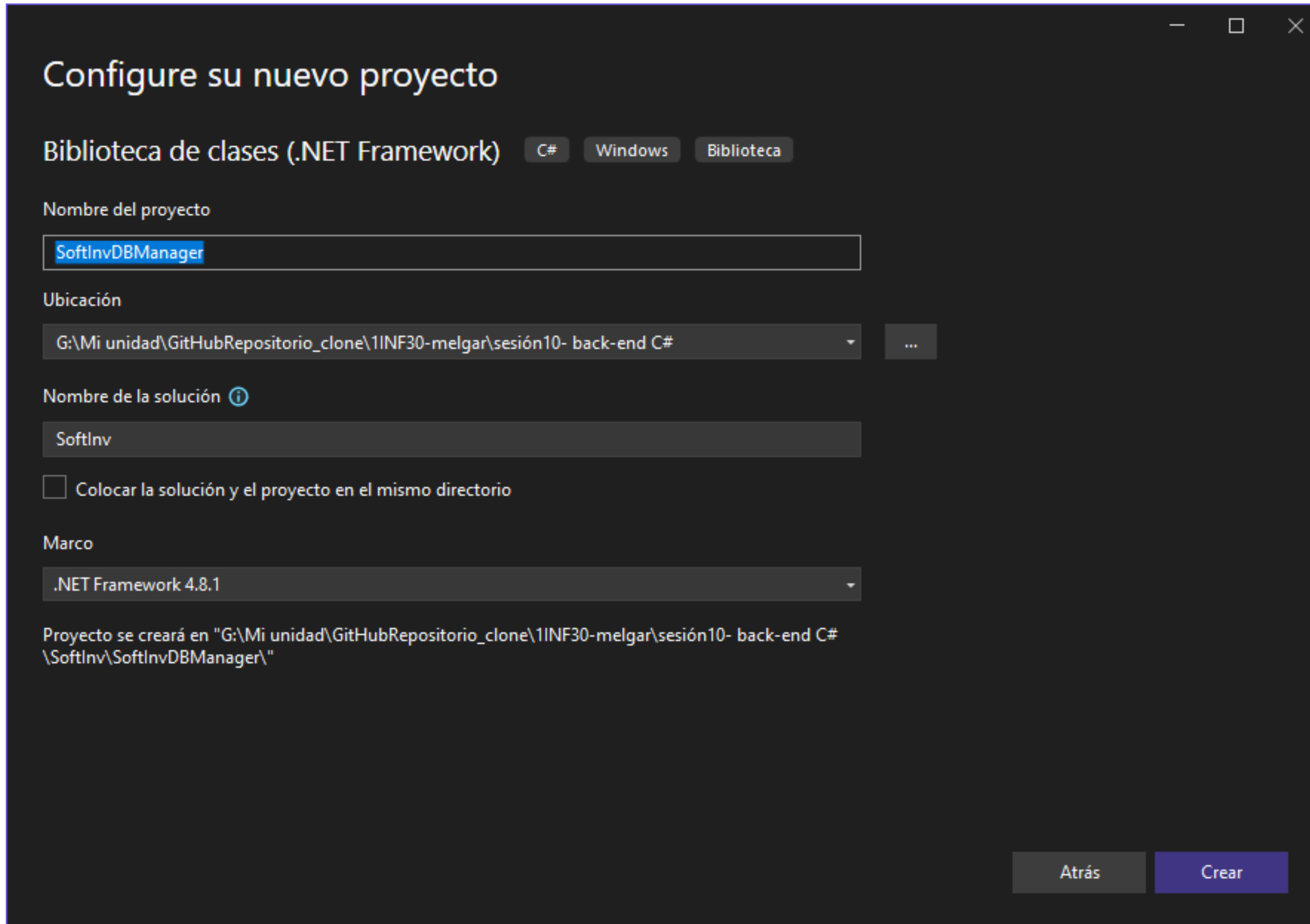


Implementación del DBManager

Creación de Proyecto en VSC (1)



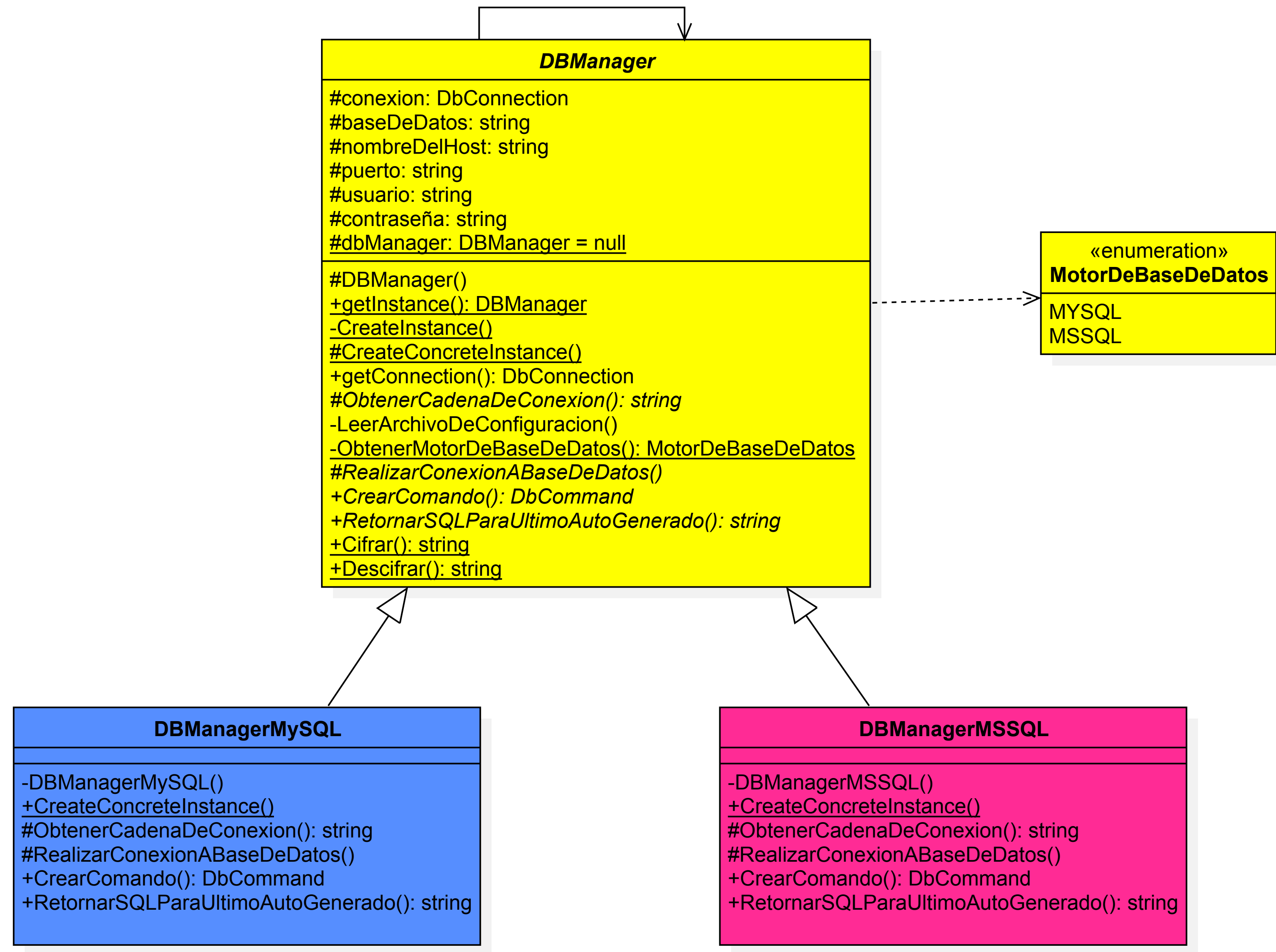
Creación de Proyecto en VSC (2)



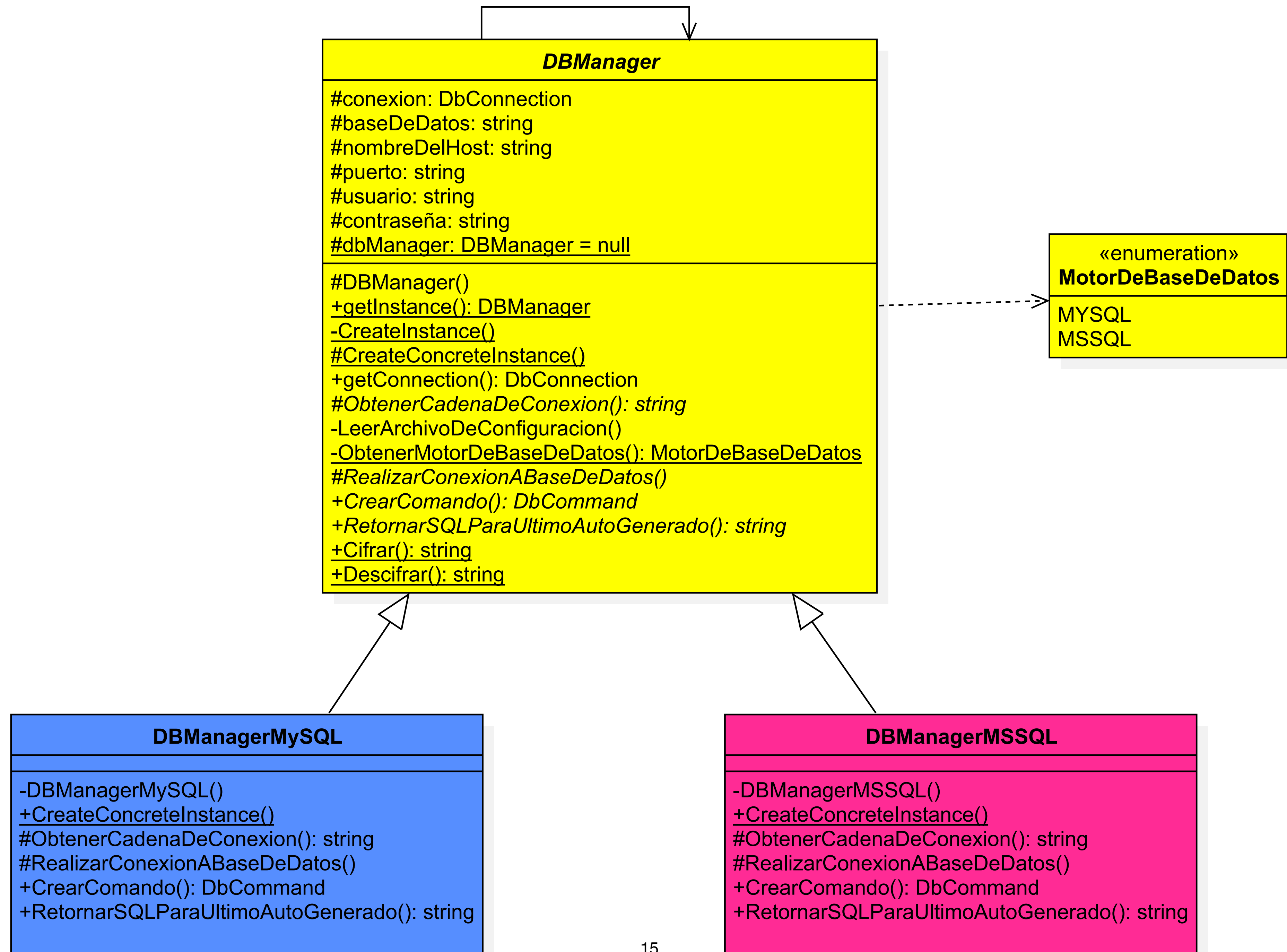
The image shows the 'Configure your new project' dialog in Visual Studio Code. The dialog is dark-themed and contains the following fields and options:

- Configure su nuevo proyecto**: The title of the dialog.
- Biblioteca de clases (.NET Framework)**: A section header with three tabs: **C#**, **Windows**, and **Biblioteca**.
- Nombre del proyecto**: A text input field containing `SoftInvDBManager`.
- Ubicación**: A dropdown menu showing the path `G:\Mi unidad\GitHubRepositorio_clone\1INF30-melgar\sesión10- back-end C#`, followed by a button with three dots (`...`) to browse for a location.
- Nombre de la solución**: A text input field containing `SoftInv`, with an information icon (`i`) to its right.
- ☐ **Colocar la solución y el proyecto en el mismo directorio**: A checkbox that is currently unchecked.
- Marco**: A dropdown menu showing `.NET Framework 4.8.1`.
- Summary text**: A line of text at the bottom stating: "Proyecto se creará en "G:\Mi unidad\GitHubRepositorio_clone\1INF30-melgar\sesión10- back-end C#\SoftInv\SoftInvDBManager\"".
- Buttons**: Two buttons at the bottom right: **Atrás** (Back) and **Crear** (Create).

Diagrama de clases (DBManager)



Mayor detalle sobre diagrama de clases de diseño, notación UML y patrones, los verán en el curso 1INF50 - Diseño de Software



Conector para MySQL (1)

The screenshot shows a web browser window with the address bar displaying `dev.mysql.com/downloads/connector/net/`. The page title is "MySQL Community Downloads" with a breadcrumb "Connector/NET". There are three tabs: "General Availability (GA) Releases" (selected), "Archives", and an information icon. The main heading is "Connector/NET 9.3.0". Below it, a dropdown menu for "Select Operating System:" is set to "Microsoft Windows". A table lists the download options:

Windows (x86, 32-bit), MSI Installer (mysql-connector-net-9.3.0.msi)	9.3.0	1.6M	Download
		MD5: 5fe4764065fece37b08525007dd588c7	Signature

Conector para MySQL (2)

The screenshot shows the Visual Studio IDE with the NuGet package manager open. The 'MySQL.Data' package is selected, and its details are displayed on the right. The left pane shows a list of MySQL-related packages, including 'MySQL.Data', 'MySQLClient', 'Devart.Data.MySql', 'Devart.Data.MySql.EFCore', 'Azure.ResourceManager.MySql', and 'Devart.Data.MySql.EFCore.Design'. The right pane shows the details for 'MySQL.Data', including its version (9.3.0), author (Oracle Corporation), license (GPL-2.0-only WITH Universal-FOSS-exception-1.0), and a list of dependencies.

NuGet: SoftInvDBManager | App.config | DBManager.cs | MotorDeBaseDeDatos.cs

Administrador de paquetes NuGet: SoftInvDBManager

Examinar | Instalado | Actualizaciones

MySQLClient | Incluir versión preliminar | Origen del paquete: nuget.org

MySQL.Data por MySQL, 110M descargas | 9.3.0
MySQL.Data.MySqlClient .Net Core Class Library

MySQLClient por alansav, 82,3K descargas | 5.5.2
A class library which provides a framework for communicating with MySql

Devart.Data.MySql por Devart, 1,7M descargas | 9.4.235
dotConnect for MySQL is a enhanced database connectivity solution built over ADO.NET architecture and a development framework with a number of innov...

Devart.Data.MySql.EFCore por Devart, 786K descargas | 9.4.235.9
dotConnect for MySQL is a high-performance ORM enabled data provider for MySQL 8.0+ including Embedded servers (starting with 4.1), MariaDB, Amaz...

Azure.ResourceManager.MySql por azure-sdk, Microsoft, 351K desc | 1.1.1
Microsoft Azure Resource Manager client SDK for Azure resource provider Microsoft.DBforMySQL.

Devart.Data.MySql.EFCore.Design por Devart, 361K descargas | 9.4.235
dotConnect for MySQL is a enhanced database connectivity solution built over

La licencia de cada paquete la concede su propietario. NuGet no se hace responsable de los paquetes de terceros ni concede ninguna licencia para ellos.

☐ No volver a mostrar

MySQL.Data | nuget.org

Versión: Versión estable más reciente 9 | Instalar

La asignación de origen del paquete está desactivada. Con

Opciones

Descripción
MySQL.Data.MySqlClient .Net Core Class Library

Versión: 9.3.0
Autores: Oracle Corporation
Licencia: GPL-2.0-only WITH Universal-FOSS-exception-1.0
Léame: Ver archivo Léame
Descargas: 110.032.955
Fecha de publicación: martes, 15 de abril de 2025 (15/04/2025)
URL del proyecto: https://dev.mysql.com/downloads/
Notificar abuso: https://www.nuget.org/packages/MySQL.Data/9.3.0/ReportAbuse
Etiquetas: MySql, .NET, Connector, Connector/.NET, netcore, .Net, Core, Conector/Net, coreclr, C/NET, C/Net

Dependencias

Explorador de soluciones

Buscar en Explorador de soluciones (Ctrl+)

Solución "SoftInv" (1 de 1 proyecto)

SoftInvDBManager

Properties

Referencias

- Analizadores
- Microsoft.CSharp
- System
- System.Configuration
- System.Core
- System.Data
- System.Data.DataSetExtensions
- System.Net.Http
- System.Xml
- System.Xml.Linq

Util

- MotorDeBaseDeDatos.cs
- App.config
- DBManager.cs
- DBManagerMSSQL.cs
- DBManagerMySQL.cs

Listo | Agregar al control de código fuente | Seleccionar repositorio

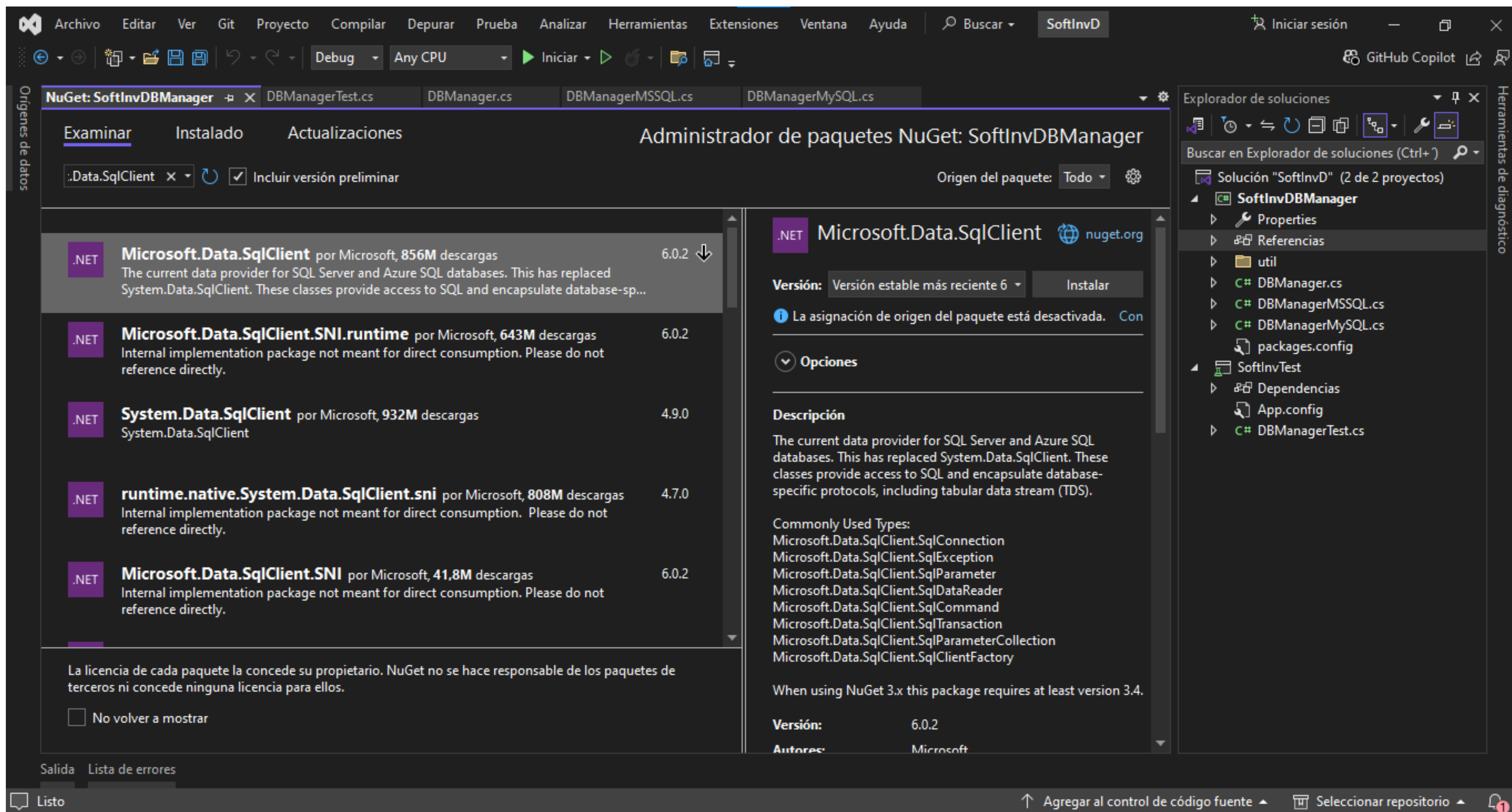
Conector para MSSQL (1)

El soporte extendido de SQL Server 2012 finalizó el 12 de julio de 2022. Entérate de lo que esto significa para ti. [Obtenga más información](#)

Herramientas y conectores de SQL Server

Herramientas	Conectores
Descargar Azure Data Studio	Microsoft ADO.NET for SQL Server
Descargar SQL Server Management Studio (SSMS)	Microsoft JDBC Driver for SQL Server
Descarga SQL Server Data Tools (SSDT).	Microsoft ODBC Driver for SQL Server
Descargar Data Migration Assistant	Native Client Driver for SQL Server

Conector para MSSQL (2)



Archivo Editar Ver Git Proyecto Compilar Depurar Prueba Analizar Herramientas Extensiones Ventana Ayuda Buscar SoftInvD Iniciar sesión GitHub Copilot

Debug Any CPU Iniciar

NuGet: SoftInvDBManager DBManagerTest.cs DBManager.cs DBManagerMSSQL.cs DBManagerMySQL.cs

Examinar Instalado Actualizaciones Administrador de paquetes NuGet: SoftInvDBManager

Origen del paquete: Todo

Microsoft.Data.SqlClient por Microsoft, 856M descargas 6.0.2

Microsoft.Data.SqlClient.SNI.runtime por Microsoft, 643M descargas 6.0.2

System.Data.SqlClient por Microsoft, 932M descargas 4.9.0

runtime.native.System.Data.SqlClient.sni por Microsoft, 808M descargas 4.7.0

Microsoft.Data.SqlClient.SNI por Microsoft, 41,8M descargas 6.0.2

Microsoft.Data.SqlClient

Version: Versión estable más reciente 6 Instalar

La asignación de origen del paquete está desactivada. Con

Opciones

Descripción

The current data provider for SQL Server and Azure SQL databases. This has replaced System.Data.SqlClient. These classes provide access to SQL and encapsulate database-specific protocols, including tabular data stream (TDS).

Commonly Used Types:

- Microsoft.Data.SqlClient.SqlConnection
- Microsoft.Data.SqlClient.SqlException
- Microsoft.Data.SqlClient.SqlParameter
- Microsoft.Data.SqlClient.SqlDataReader
- Microsoft.Data.SqlClient.SqlCommand
- Microsoft.Data.SqlClient.SqlTransaction
- Microsoft.Data.SqlClient.SqlParameterCollection
- Microsoft.Data.SqlClient.SqlClientFactory

When using NuGet 3.x this package requires at least version 3.4.

Version: 6.0.2

Autores: Microsoft

Explorador de soluciones

Buscar en Explorador de soluciones (Ctrl+J)

Solución "SoftInvD" (2 de 2 proyectos)

- SoftInvDBManager
 - Properties
 - Referencias
 - util
 - DBManager.cs
 - DBManagerMSSQL.cs
 - DBManagerMySQL.cs
 - packages.config
- SoftInvTest
 - Dependencias
 - App.config
 - DBManagerTest.cs

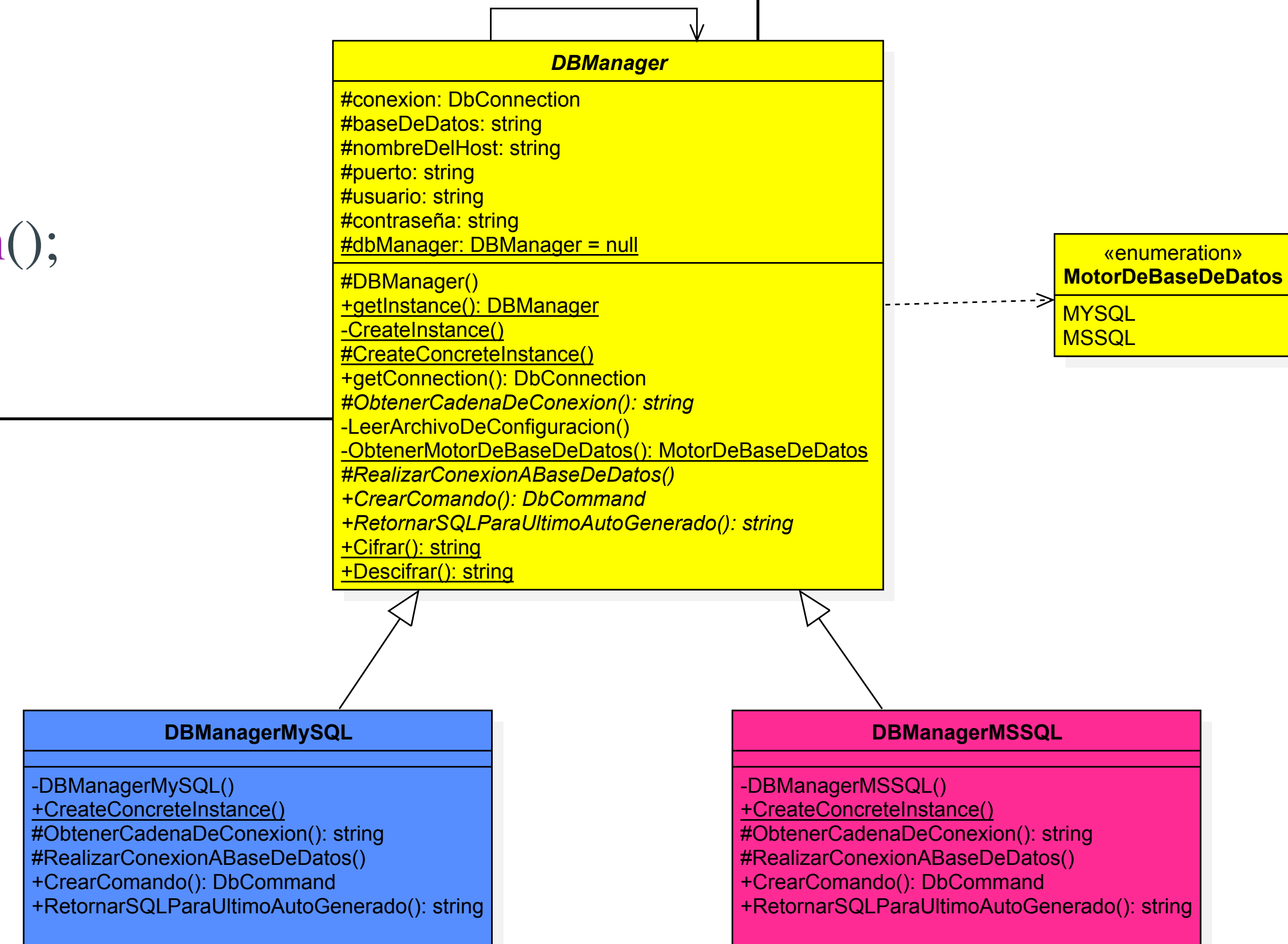
Salida Lista de errores

Listo

Agregar al control de código fuente Seleccionar repositorio

DBManager - creación de instancias (1)

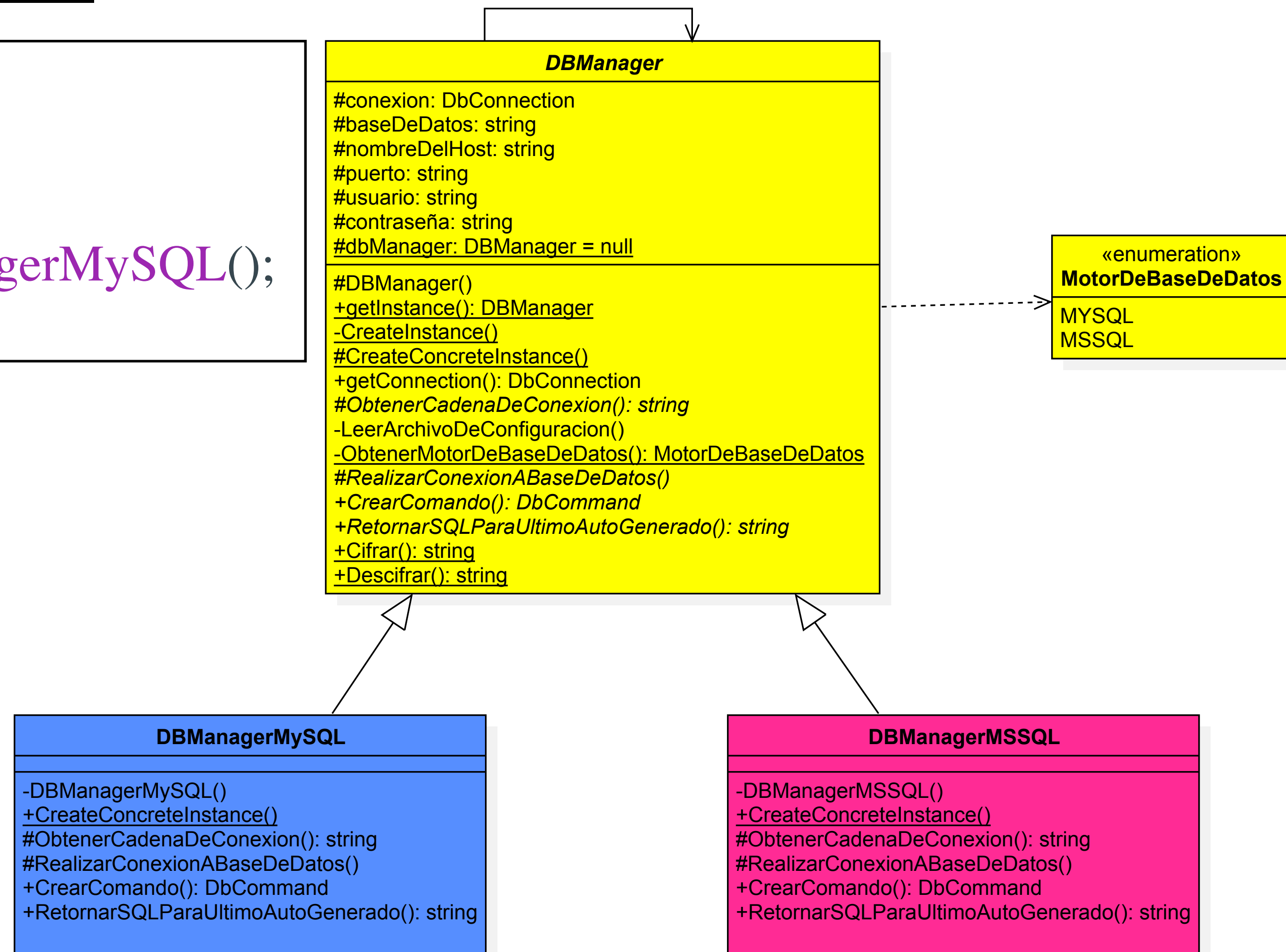
```
private static void CreateInstance()  
{  
    if (DBManager.dbManager == null)  
    {  
        if (DBManager.ObtenerMotorDeBaseDeDatos() == MotorDeBaseDeDatos.MYSQL)  
            DBManagerMySQL.CreateConcreteInstance();  
        else  
            DBManagerMSSQL.CreateConcreteInstance();  
        DBManager.dbManager.LeerArchivoDeConfiguracion();  
    }  
}
```



DBManager - creación de instancias (2)

DBManagerMySQL

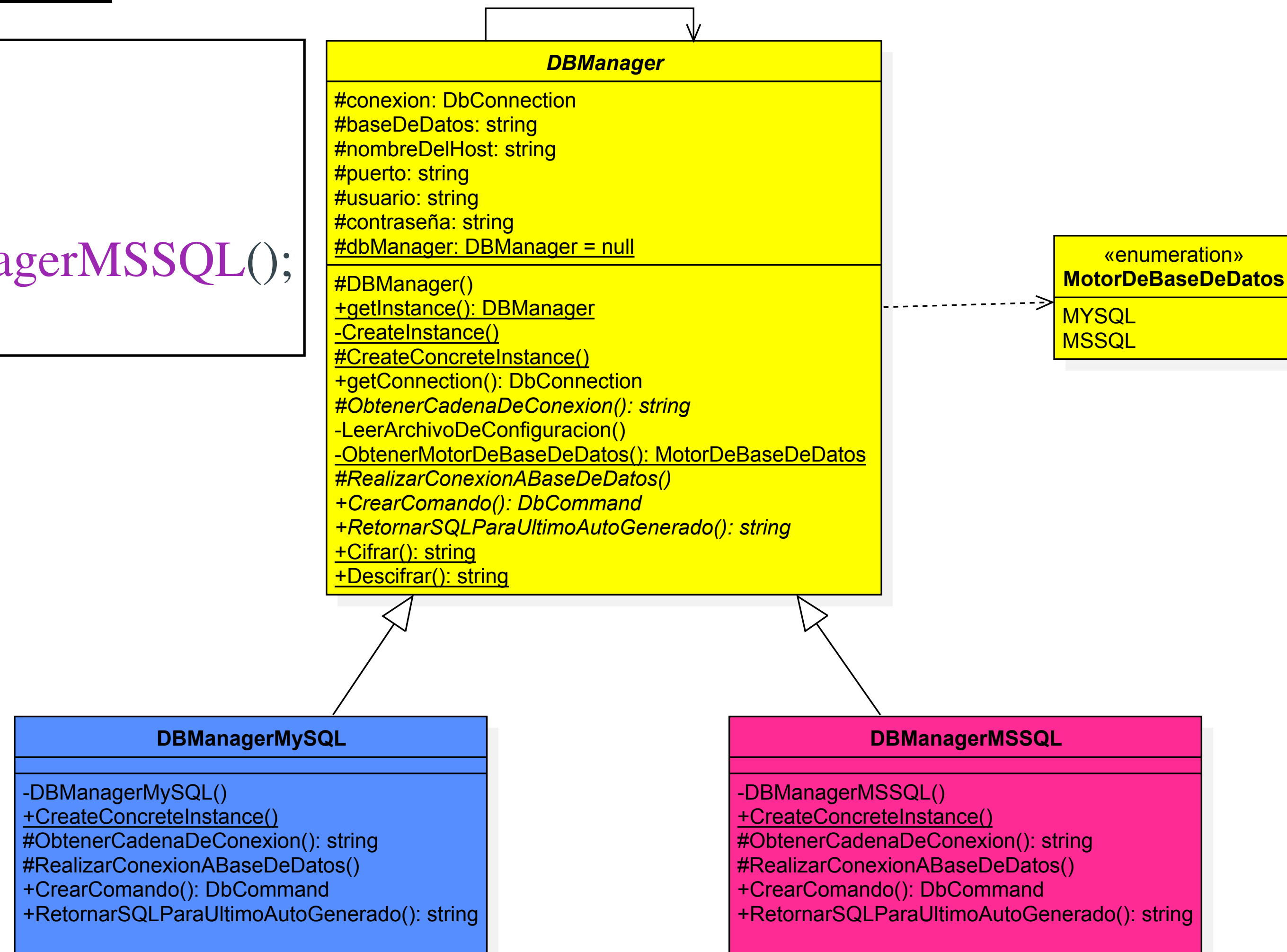
```
public static void CreateConcreteInstance()
{
    if (DBManager.dbManager == null)
        DBManager.dbManager = new DBManagerMySQL();
}
```



DBManager - creación de instancias (3)

DBManagerMSSQL

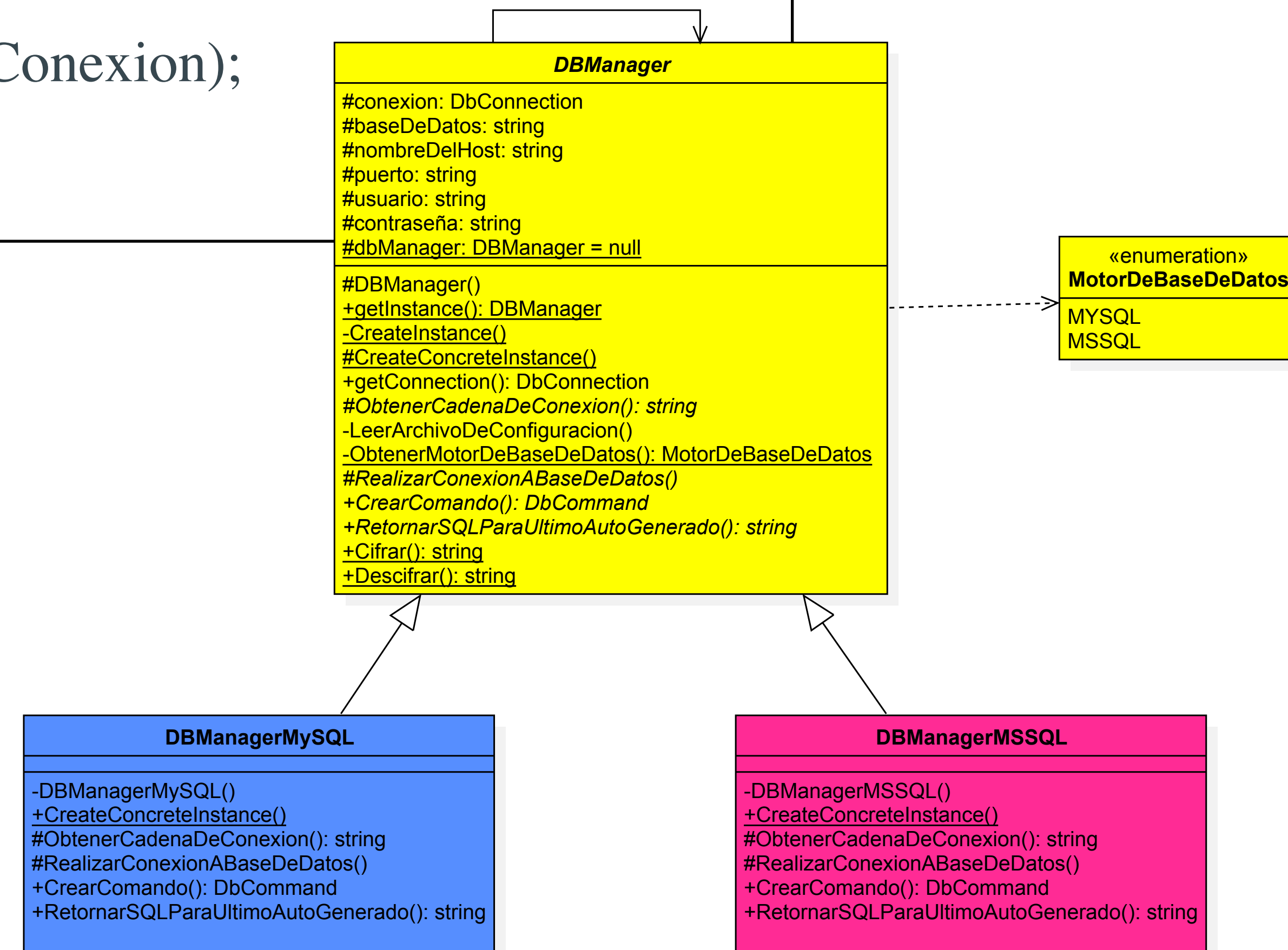
```
public static void CreateConcreteInstance()
{
    if (DBManager.dbManager == null)
        DBManager.dbManager = new DBManagerMSSQL();
}
```



DBManager - conexión a la base de datos (1)

DBManagerMySQL

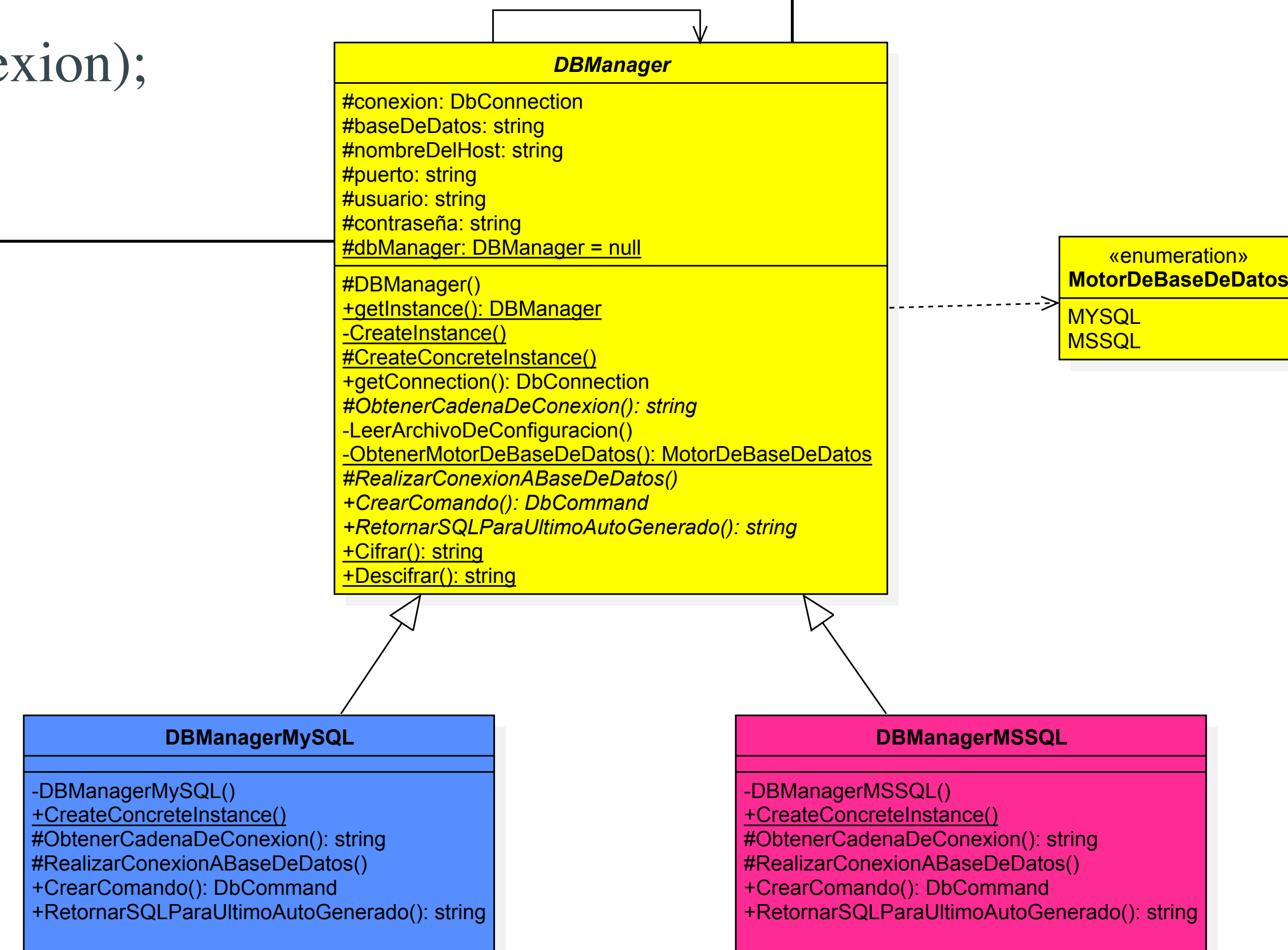
```
protected override void RealizarConexionABaseDeDatos(string cadenaDeConexion)
{
    this.conexion = new MySqlConnection(cadenaDeConexion);
}
```



DBManager - conexión a la base de datos (2)

DBManagerMSSQL

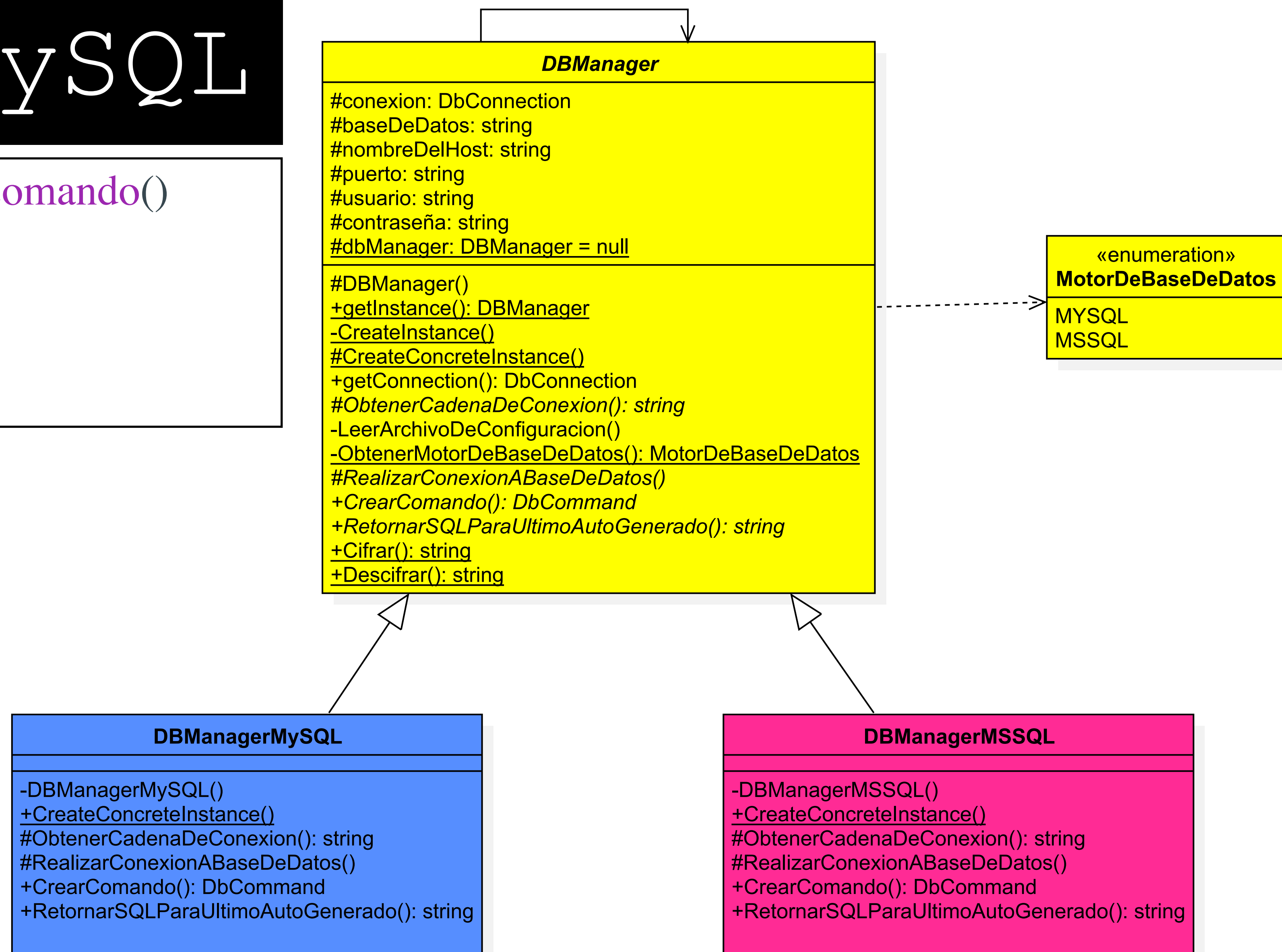
```
protected override void RealizarConexionABaseDeDatos(string cadenaDeConexion)
{
    this.conexion = new SqlConnection(cadenaDeConexion);
}
```



DBManager - crear comando (1)

DBManagerMySQL

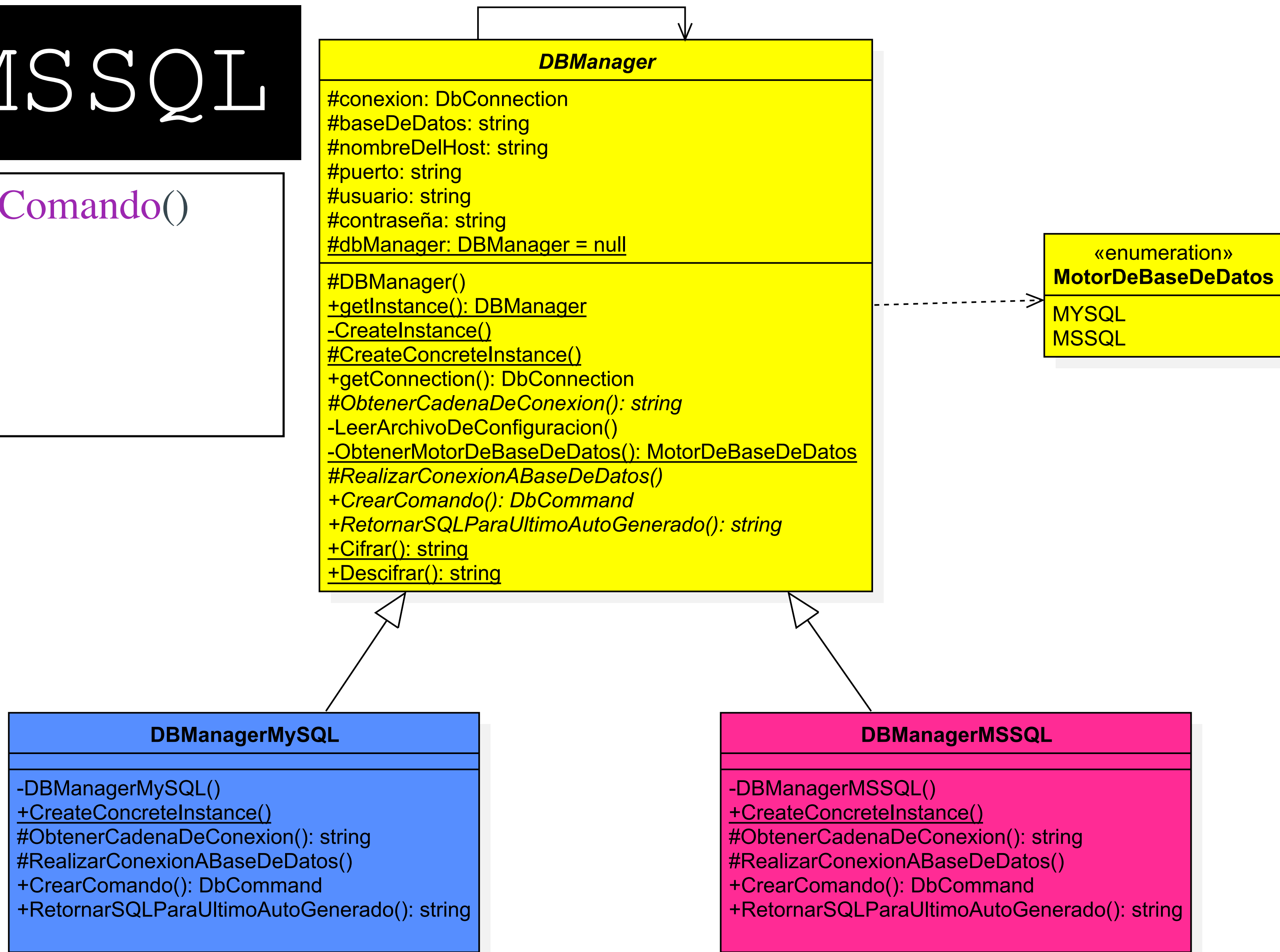
```
public override DbCommand CrearComando()  
{  
    return new MySqlCommand();  
}
```



DBManager - crear comando (2)

DBManagerMSSQL

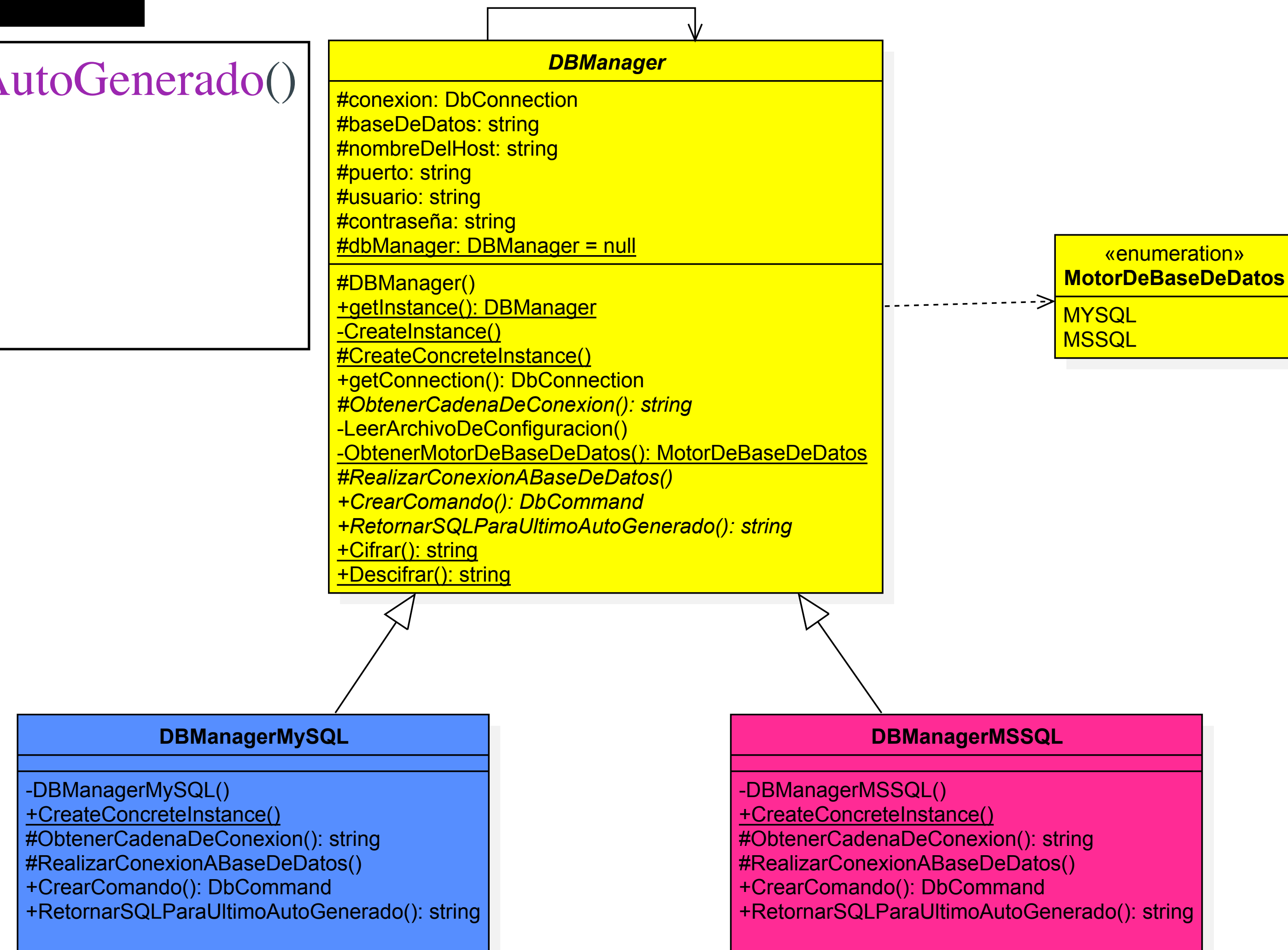
```
public override DbCommand CrearComando()  
{  
    return new SqlCommand();  
}
```



DBManager - SQL para id auto generados (1)

DBManagerMySQL

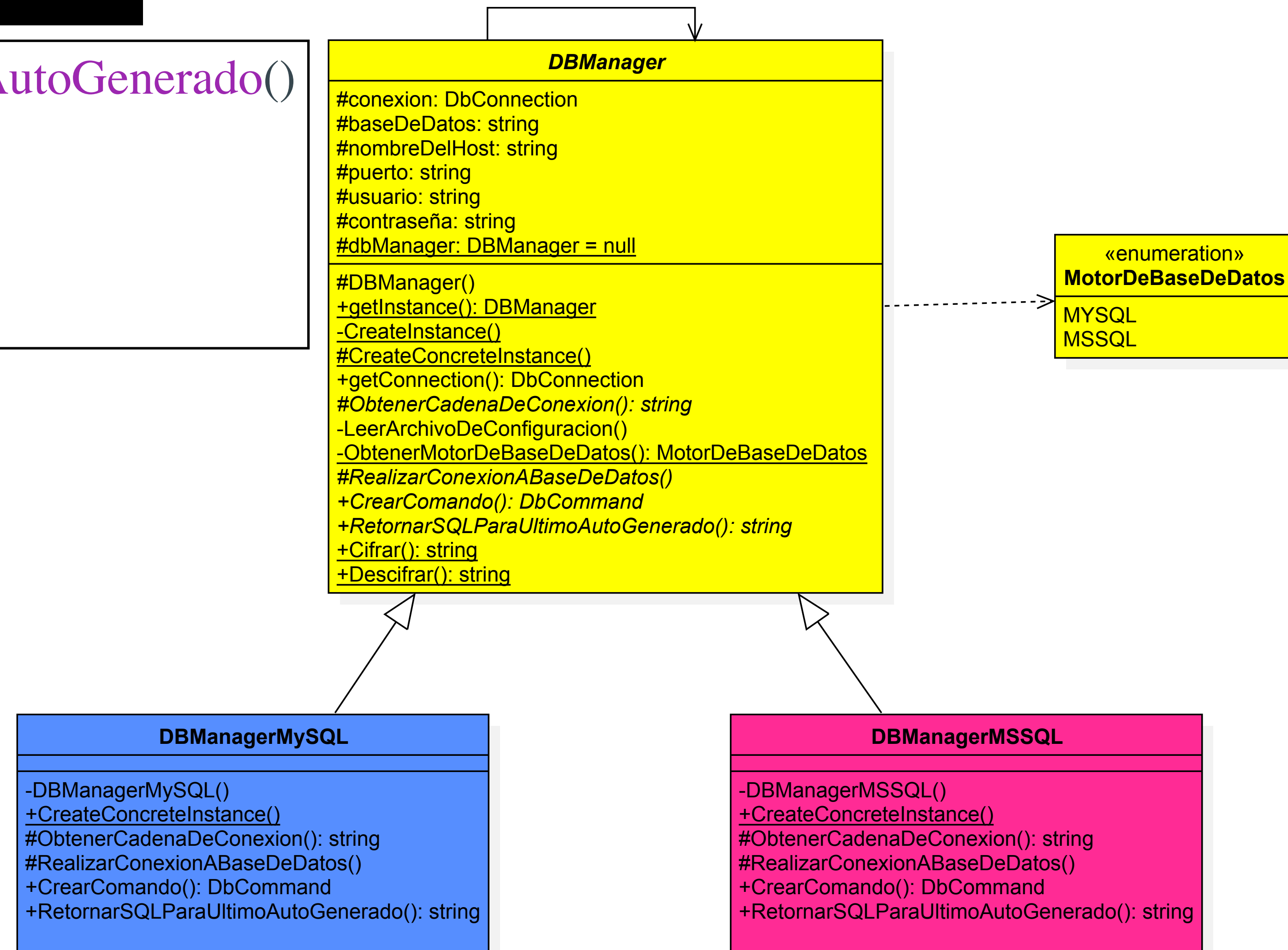
```
public override string RetornarSQLParaUltimoAutoGenerado()
{
    string sql = "select @@last_insert_id as id";
    return sql;
}
```



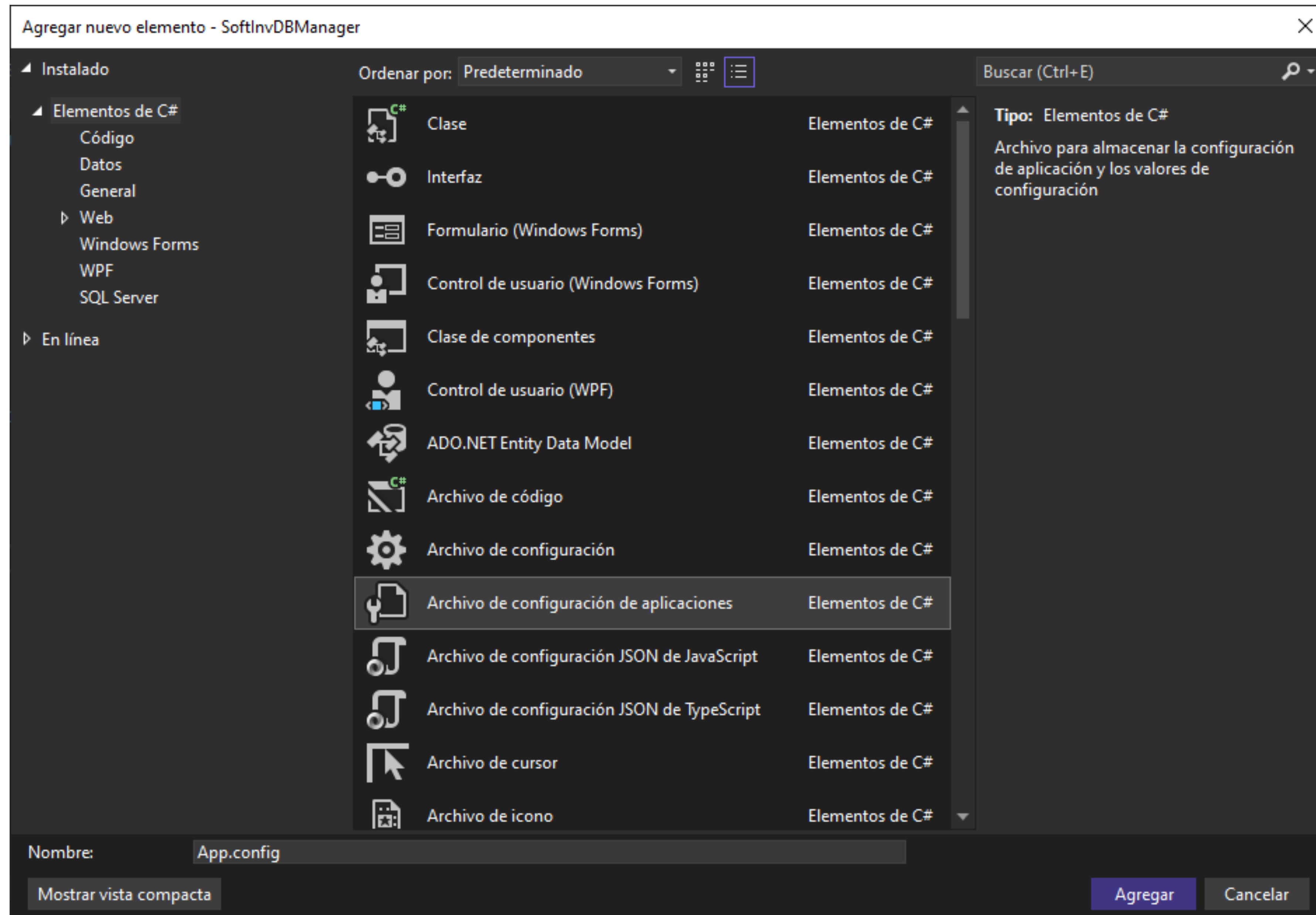
DBManager - SQL para id auto generados (2)

DBManagerMSSQL

```
public override string RetornarSQLParaUltimoAutoGenerado()
{
    string sql = "select @@IDENTITY as id";
    return sql;
}
```

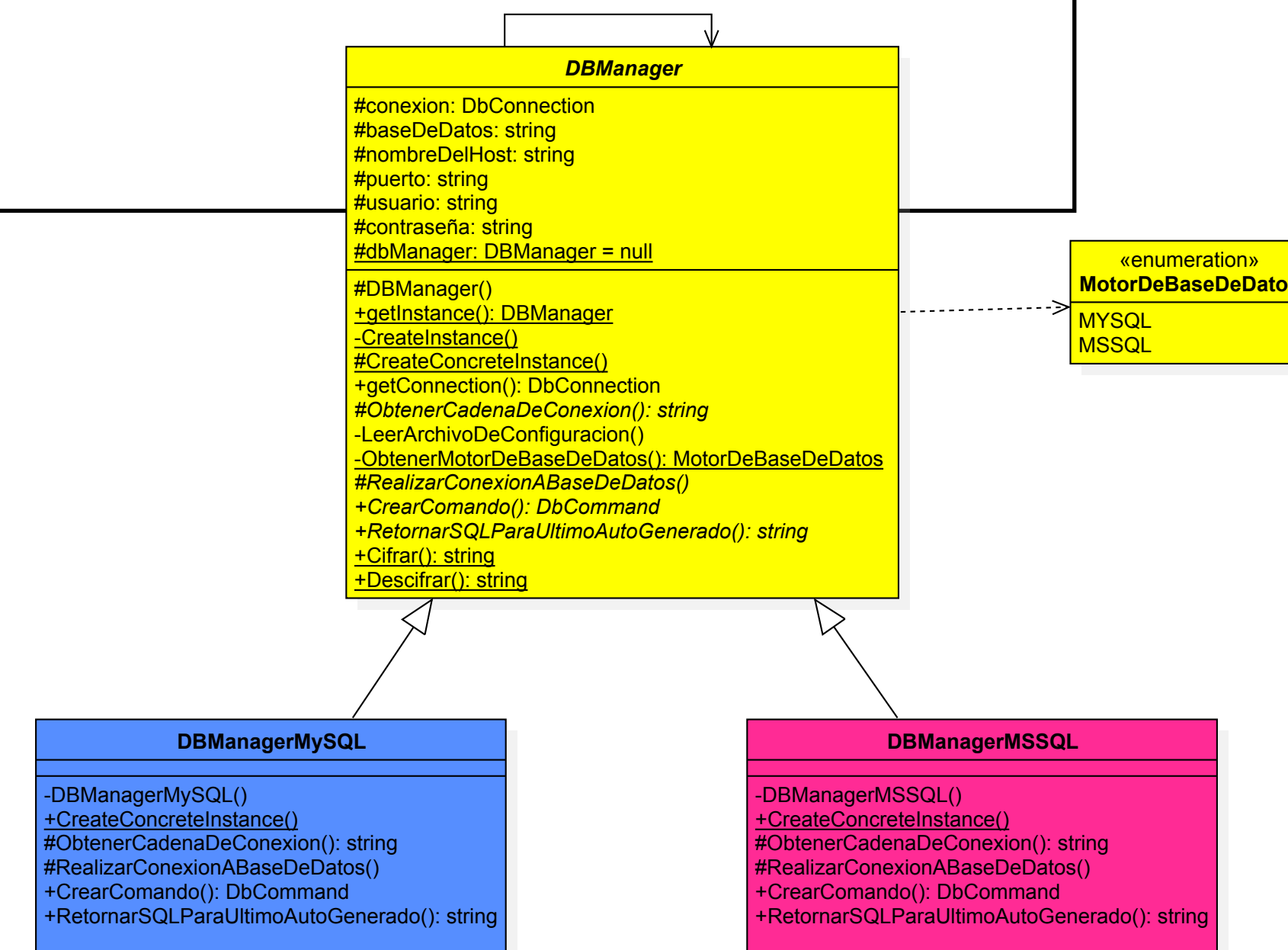


DBManager - leer configuración (1)



DBManager - leer configuración (2)

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
  <appSettings>
    <add key="tipoDeBaseDeDatos" value="MYSQL" />
    <add key="baseDeDatos" value="soft_inv_test" />
    <add key="nombreDeHost" value="localhost" />
    <add key="puerto" value="3306" />
    <add key="usuario" value="admin" />
    <add key="contraseña" value="UABYAG8AZwByAGEAbQBhAGMAaQBvAG4AMwAhAA == " />
  </appSettings>
</configuration>
```



DBManager - leer configuración (3)

```
using System.Configuration;
```

```
private void LeerArchivoDeConfiguracion()
```

```
{
```

```
    this.baseDeDatos = ConfigurationManager.AppSettings["baseDeDatos"];
```

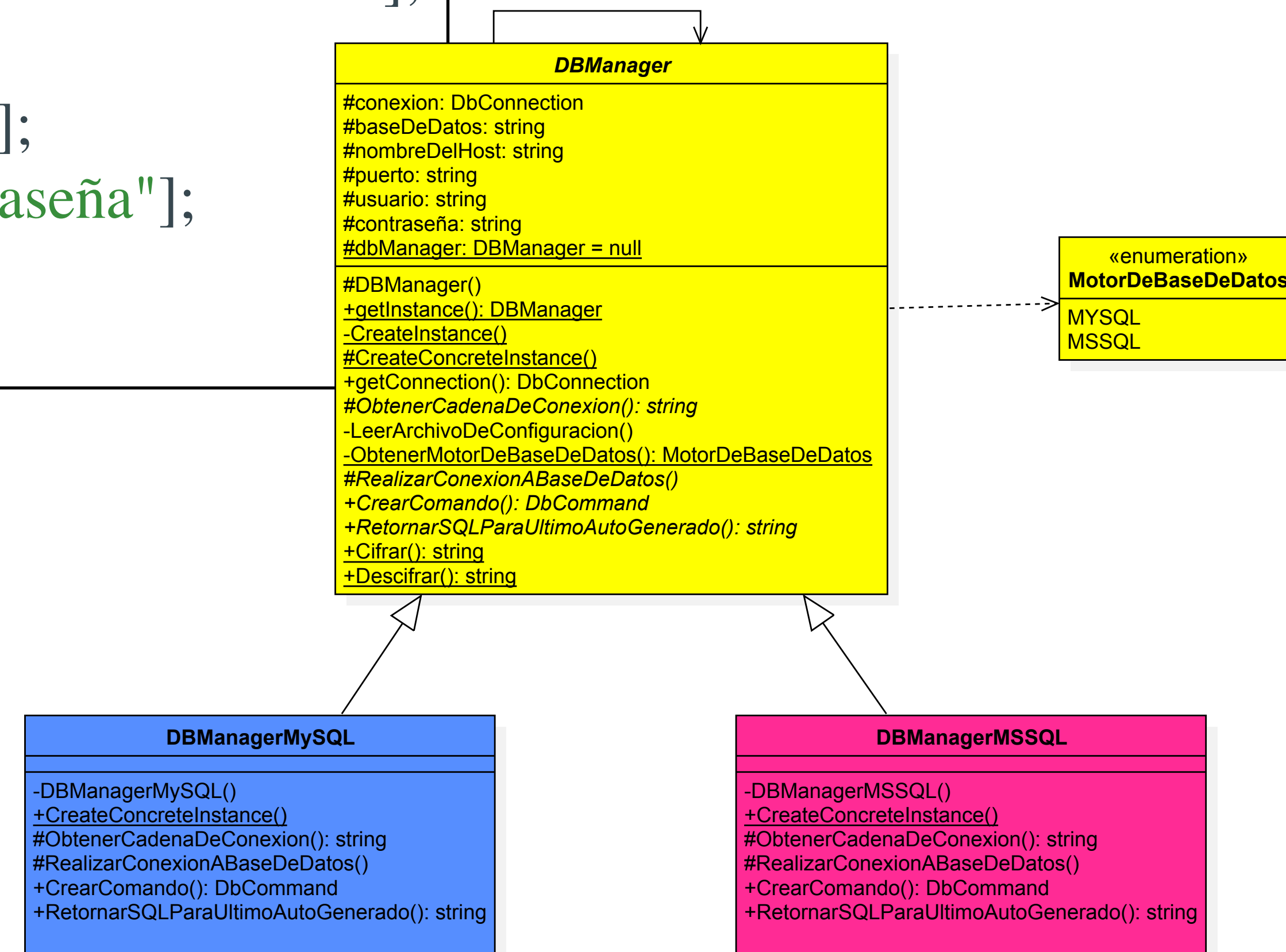
```
    this.nombreDeHost = ConfigurationManager.AppSettings["nombreDeHost"];
```

```
    this.puerto = ConfigurationManager.AppSettings["puerto"];
```

```
    this.usuario = ConfigurationManager.AppSettings["usuario"];
```

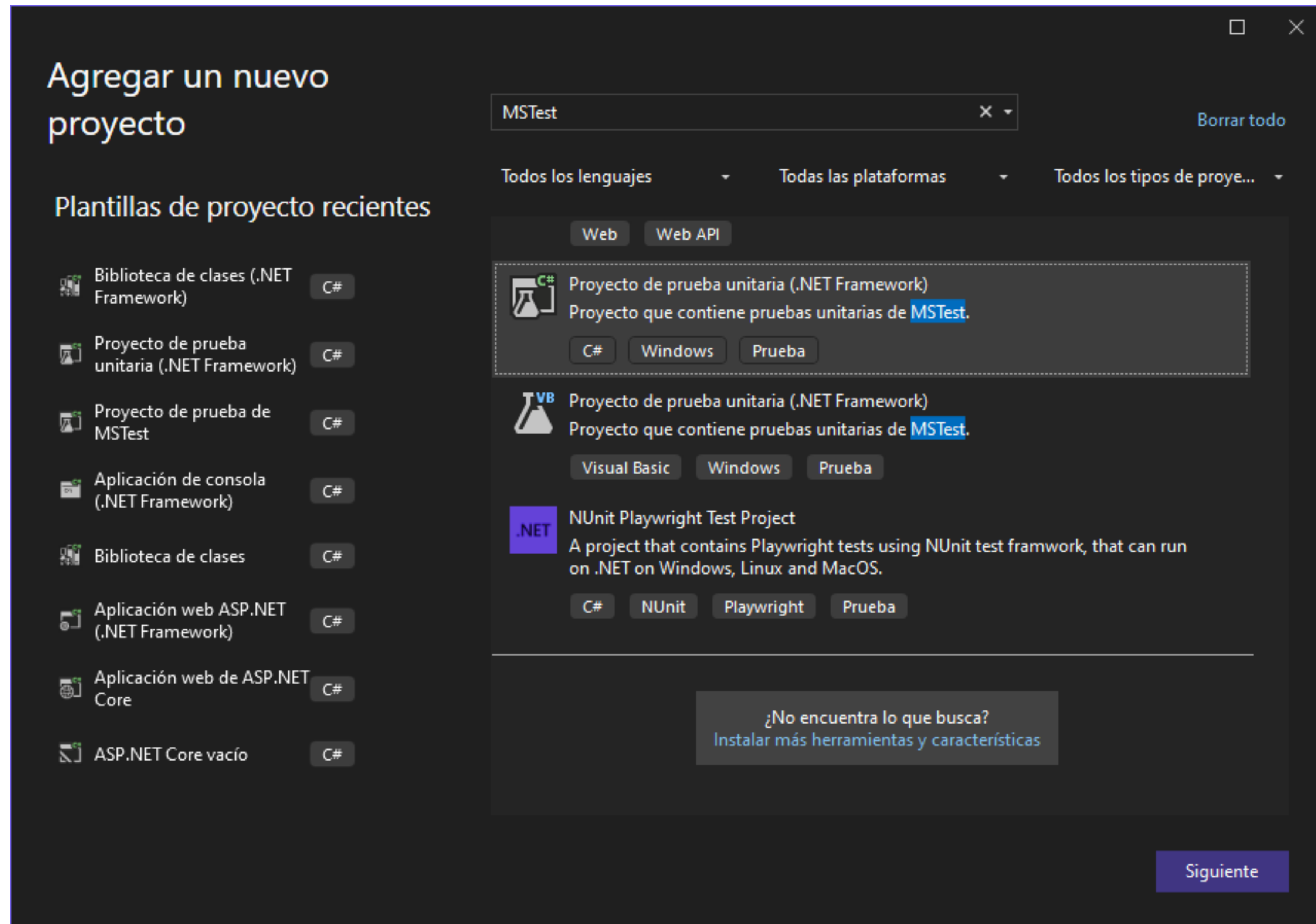
```
    this.contraseña = ConfigurationManager.AppSettings["contraseña"];
```

```
}
```

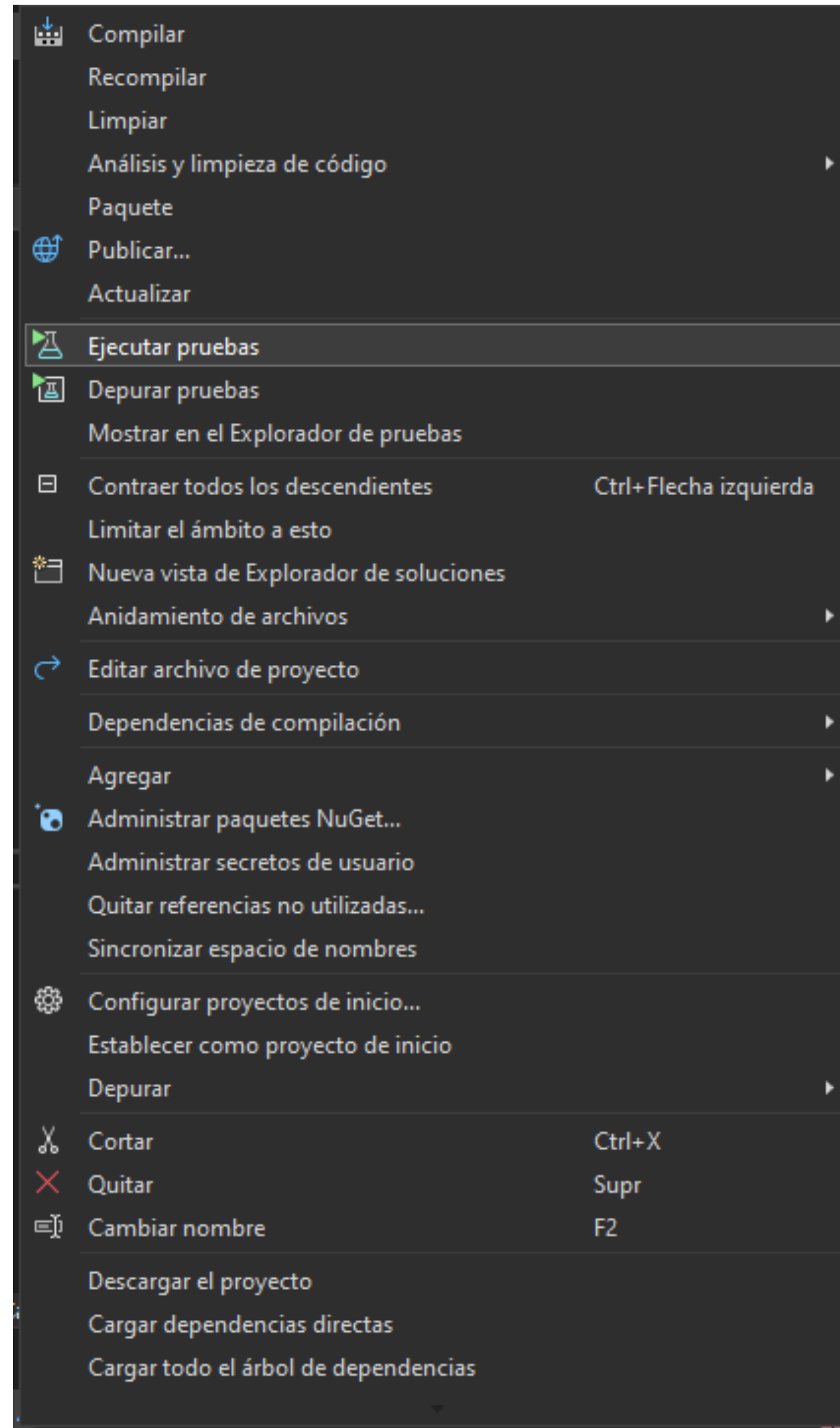


Pruebas

Creación del Proyecto



Ejecución de las pruebas



Clase de Prueba

```
using System.Data.Common;
using SoftInv.Db;

namespace SoftInvTest
{
    [TestClass]
    public class DBManagerTest
    {
        [TestMethod]
        public void TestGetInstance()
        {
            Console.WriteLine("GetInstance");
            DBManager dBManager1 = DBManager.Instance;
            DBManager dBManager2 = DBManager.Instance;
            Assert.IsNotNull(dBManager1);
            Assert.IsNotNull(dBManager2);
            Assert.AreEqual(dBManager1, dBManager2);
        }

        [TestMethod]
        public void TestGetConnection()
        {
            Console.WriteLine("GetConnection");
            DbConnection conexion = DBManager.Instance.Connection;
            Assert.IsNotNull(conexion);
            conexion.Open();
        }
    }
}
```

Capa de Dominio

Modelo Relacional

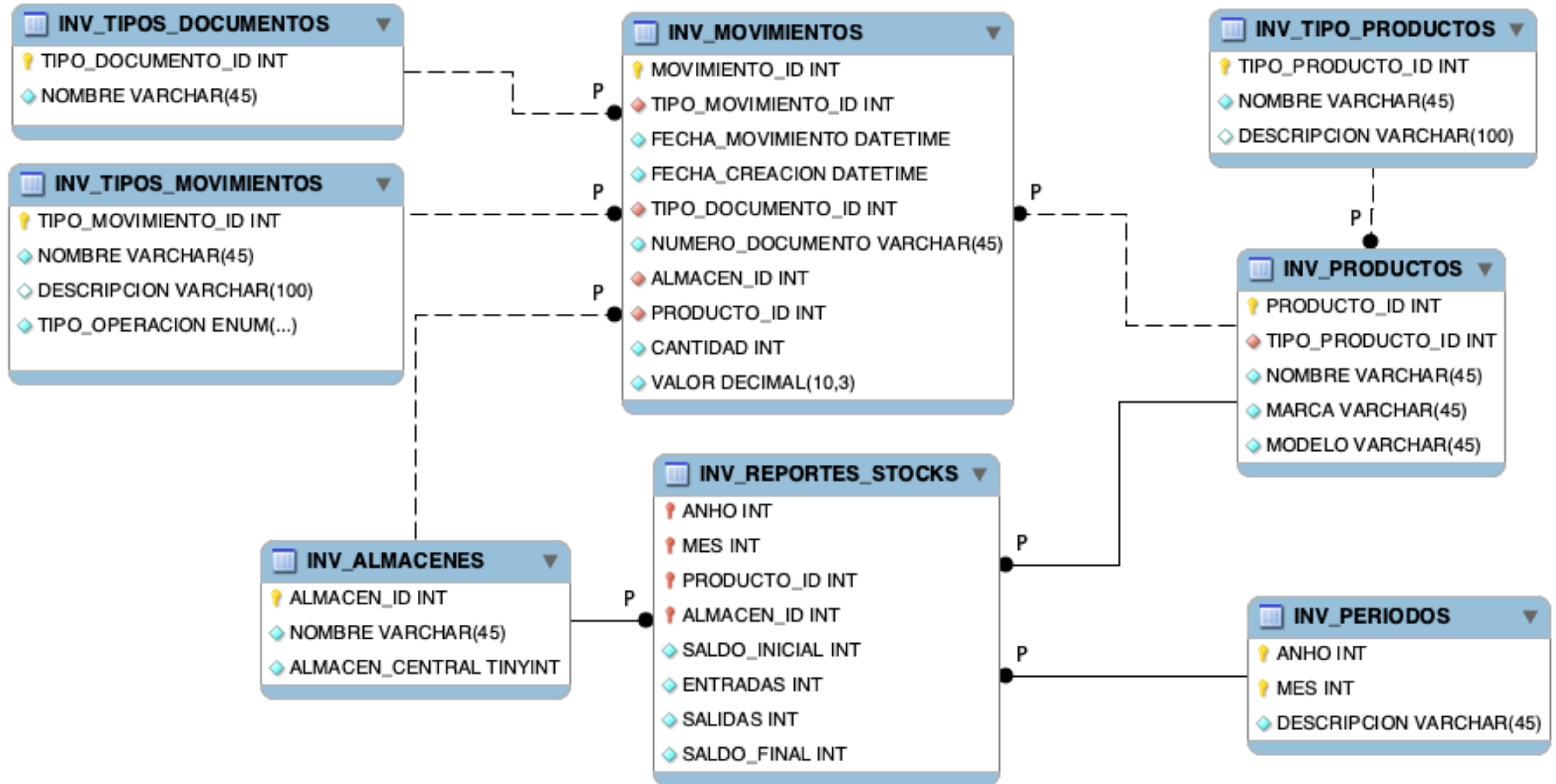
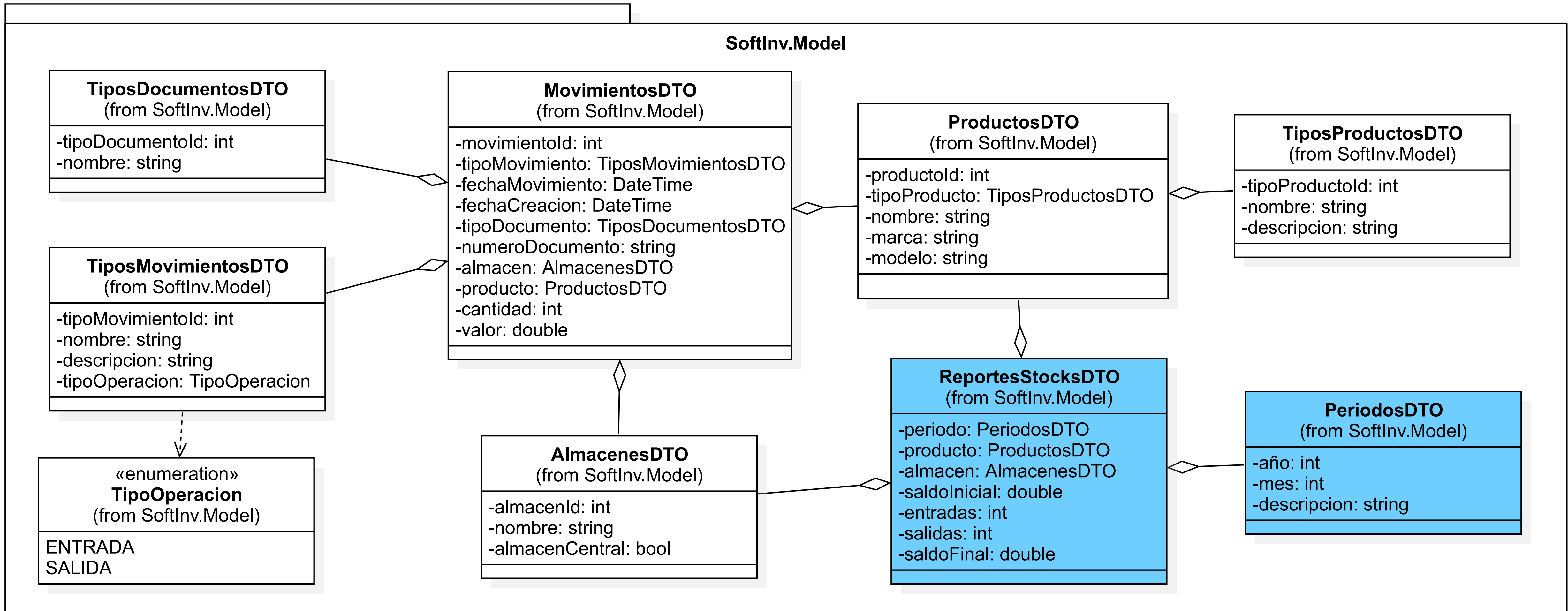


Diagrama de Clases



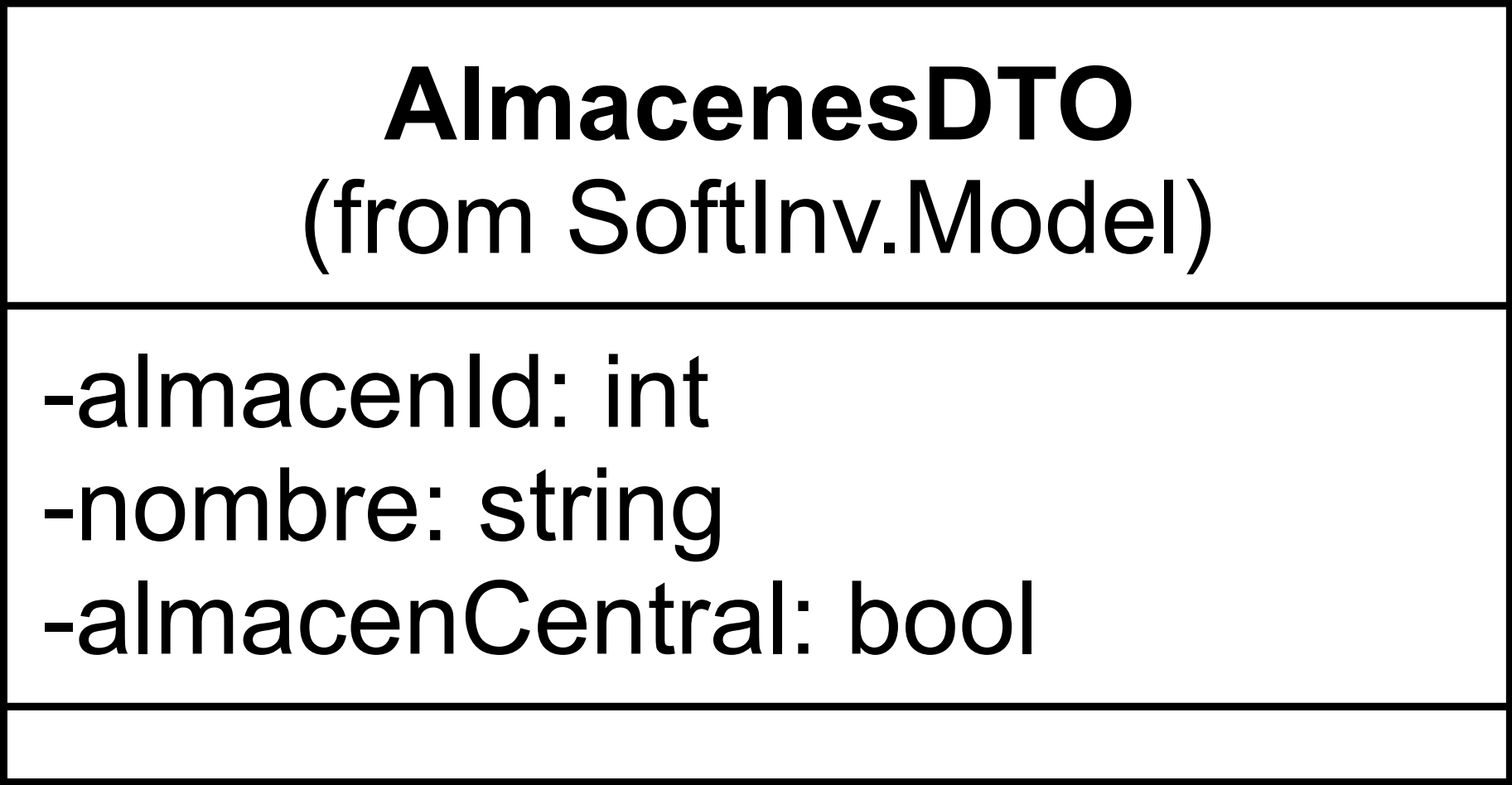
Implementación de los POJO (*Plain Old Java Object*)

```
namespace SoftInv.Model
{
  public class AlmacenesDTO
  {
    private int? almacenId;
    private string nombre;
    private bool? almacenCentral;

    public AlmacenesDTO()
    {
      this.AlmacenId = null;
      this.Nombre = null;
      this.AlmacenCentral = null;
    }

    public AlmacenesDTO(int almacenId, string nombre, bool almacenCentral)
    {
      this.AlmacenId = almacenId;
      this.Nombre = nombre;
      this.AlmacenCentral = almacenCentral;
    }

    public int? AlmacenId { get => almacenId; set => almacenId = value; }
    public string Nombre { get => nombre; set => nombre = value; }
    public bool? AlmacenCentral { get => almacenCentral; set => almacenCentral = value; }
  }
}
```



Capa de Persistencia

Clase Base (1)

DAOImplBase

```
public abstract class DAOImplBase
{
    protected string nombreDeTabla;
    protected BindingList<Columna> listaColumnas;
    protected DbConnection conexion;
    protected DbTransaction transaccion;
    protected DbCommand comando;
    protected DbDataReader lector;
    protected bool mostrarSentenciaSQL;
    protected bool retornarLlavePrimaria;
    protected bool usarTransaccion;
    ...
}
```

Clase Base (2)

DAOImplBase

```
protected void AbrirConexion()
{
    this.conexion = DBManager.Instance.Connection;
    this.conexion.Open();
}
```

```
protected void CerrarConexion()
{
    if (this.conexion != null)
    {
        this.conexion.Close();
    }
}
```

```
protected void IniciarTransaccion()
{
    this.AbrirConexion();
    this.transaccion = this.conexion.BeginTransaction();
}
```

```
protected void ComitarTransaccion()
{
    this.transaccion.Commit();
    this.transaccion = null;
}
```

Clase Base (3)

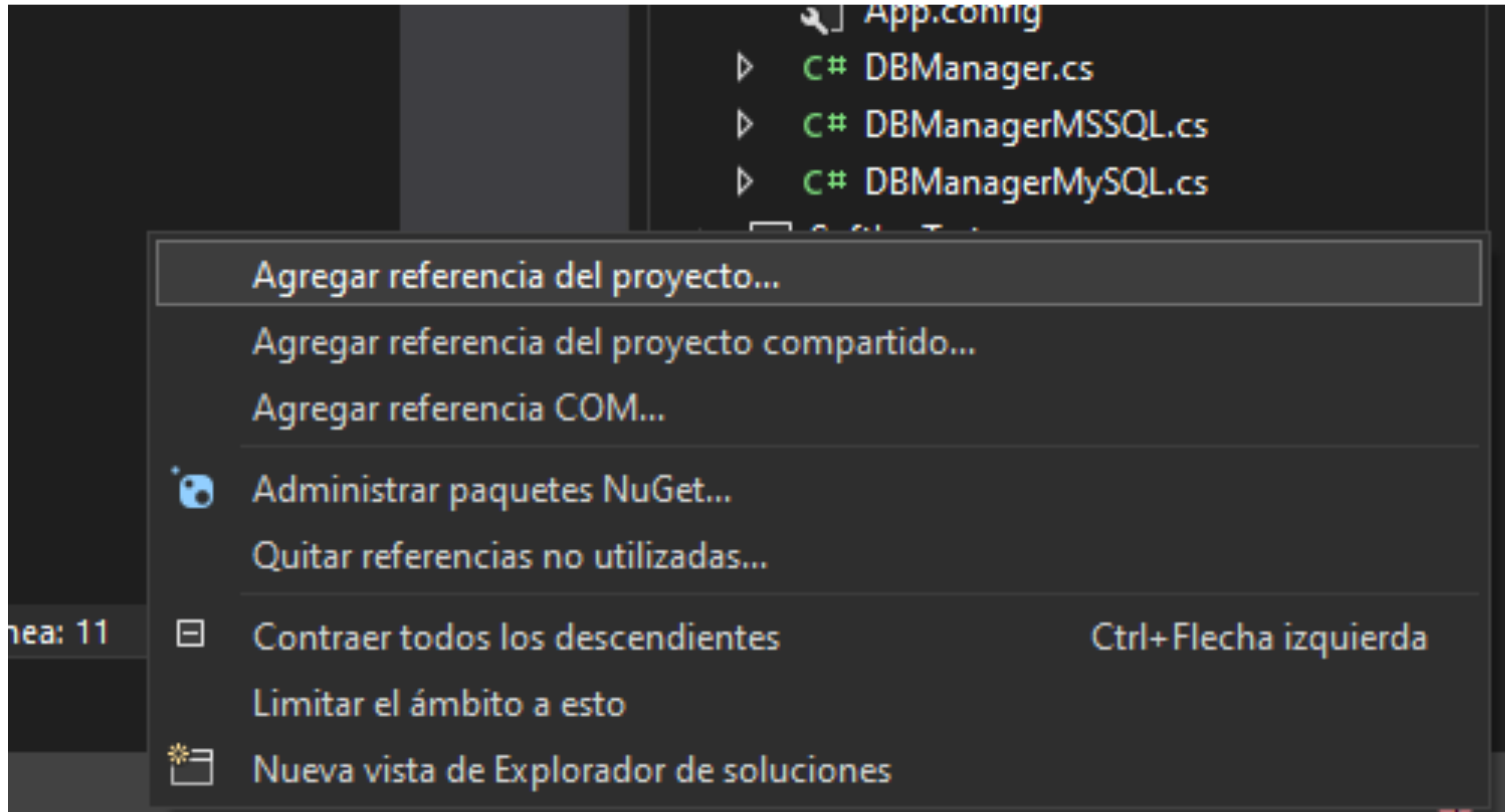
DAOImplBase

```
protected void RollbackTransaccion()
{
    if (this.transaccion != null)
    {
        this.transaccion.Rollback();
    }
    this.transaccion = null;
}
```

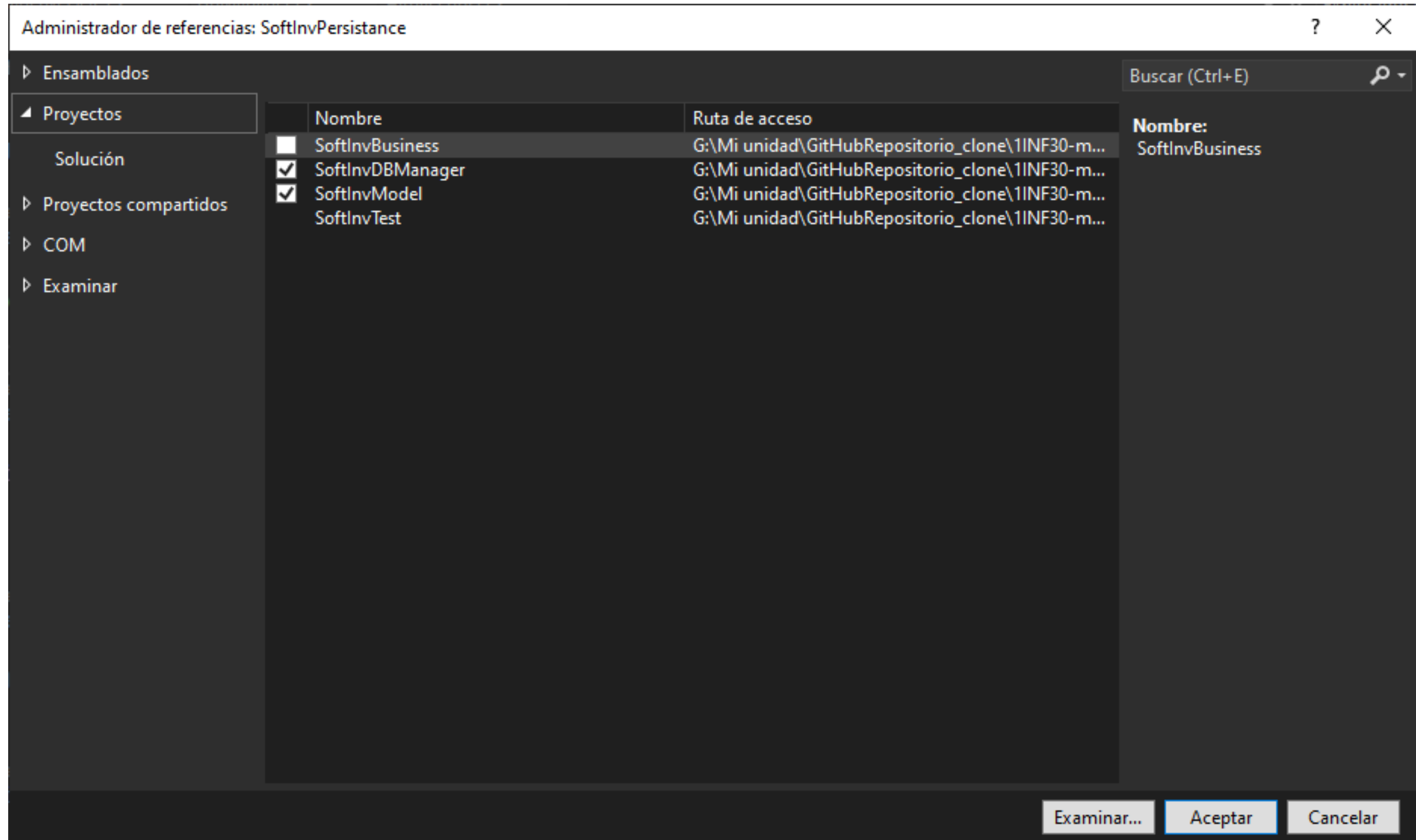
```
protected void ColocarSQLenComando(string sql)
{
    this.comando.Connection = this.conexion;
    this.comando.CommandText = sql;
    this.comando.CommandType = System.Data.CommandType.Text;
}
```

```
protected void EjecutarConsultaEnBD()
{
    this.lector = this.comando.ExecuteReader();
}
```

Referencia a Dependencias (1)



Referencia a Dependencias (2)



Capa de Negocio

Referencia a Dependencias

