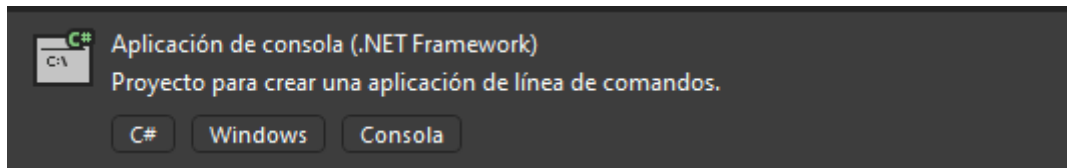


## Práctica 7 Ejercicios de acceso a datos

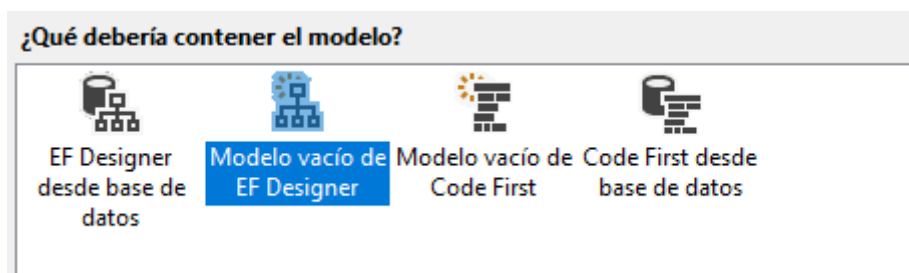
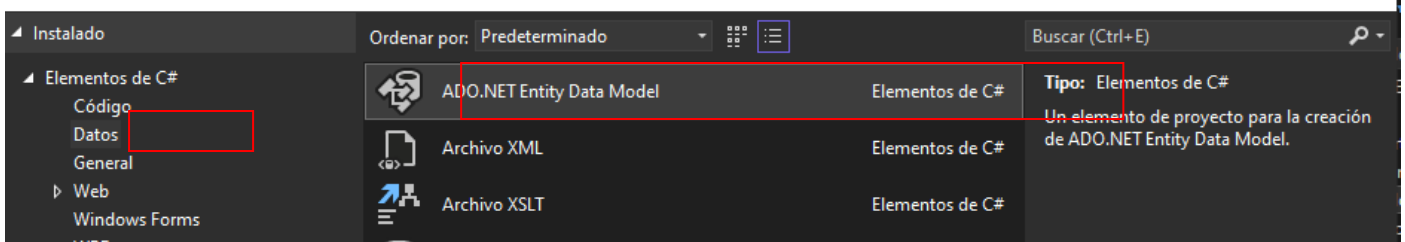
### 1- Crear un proyecto empleando Model First

- Crear un nuevo proyecto

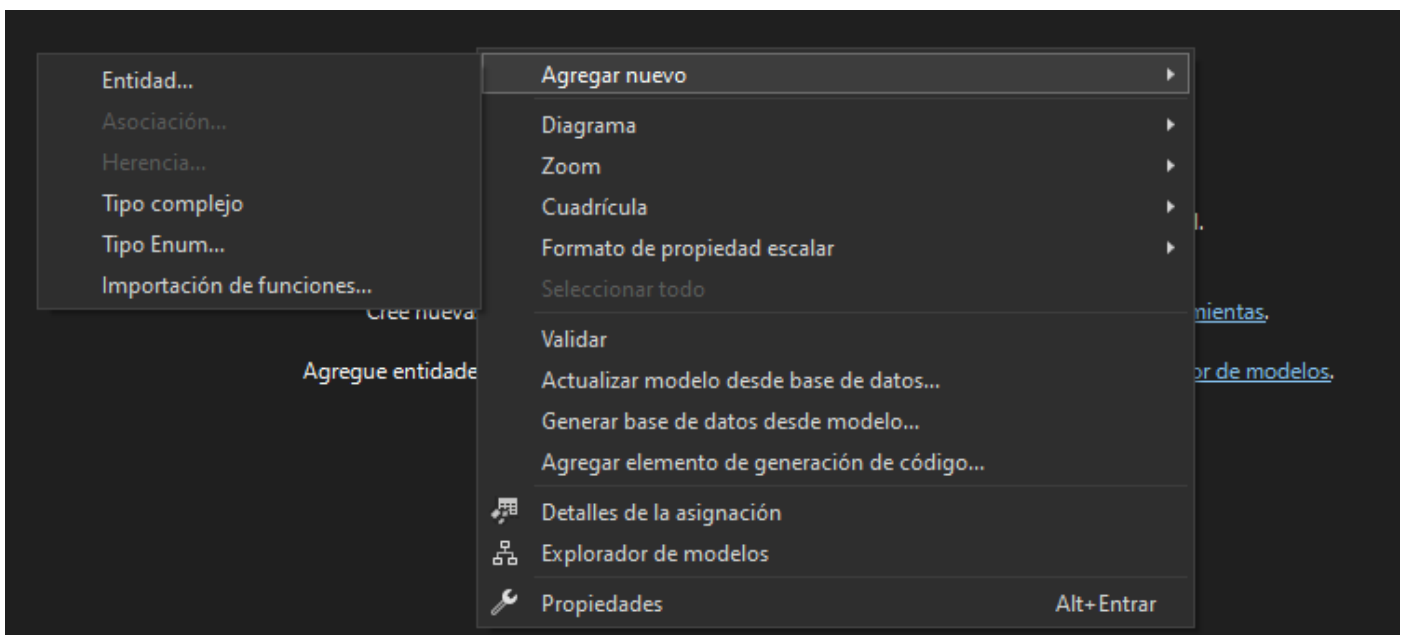


- Crear una capeta Modelo y luego agregar un elemento

Agregar nuevo elemento - P7E1b



- Crear una entidad Person



Agregar entidad



## Propiedades

Nombre de entidad:

Person

Tipo base:

(Ninguno)

Conjunto de entidades:

Persons

## Propiedad de clave

☒ Crear propiedad de clave

Nombre de propiedad:

Id

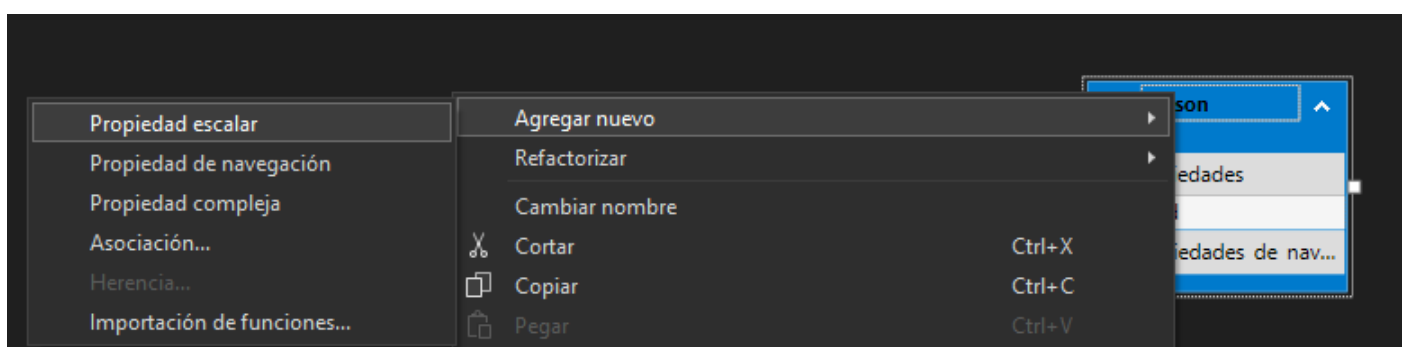
Tipo de propiedad:

Int32

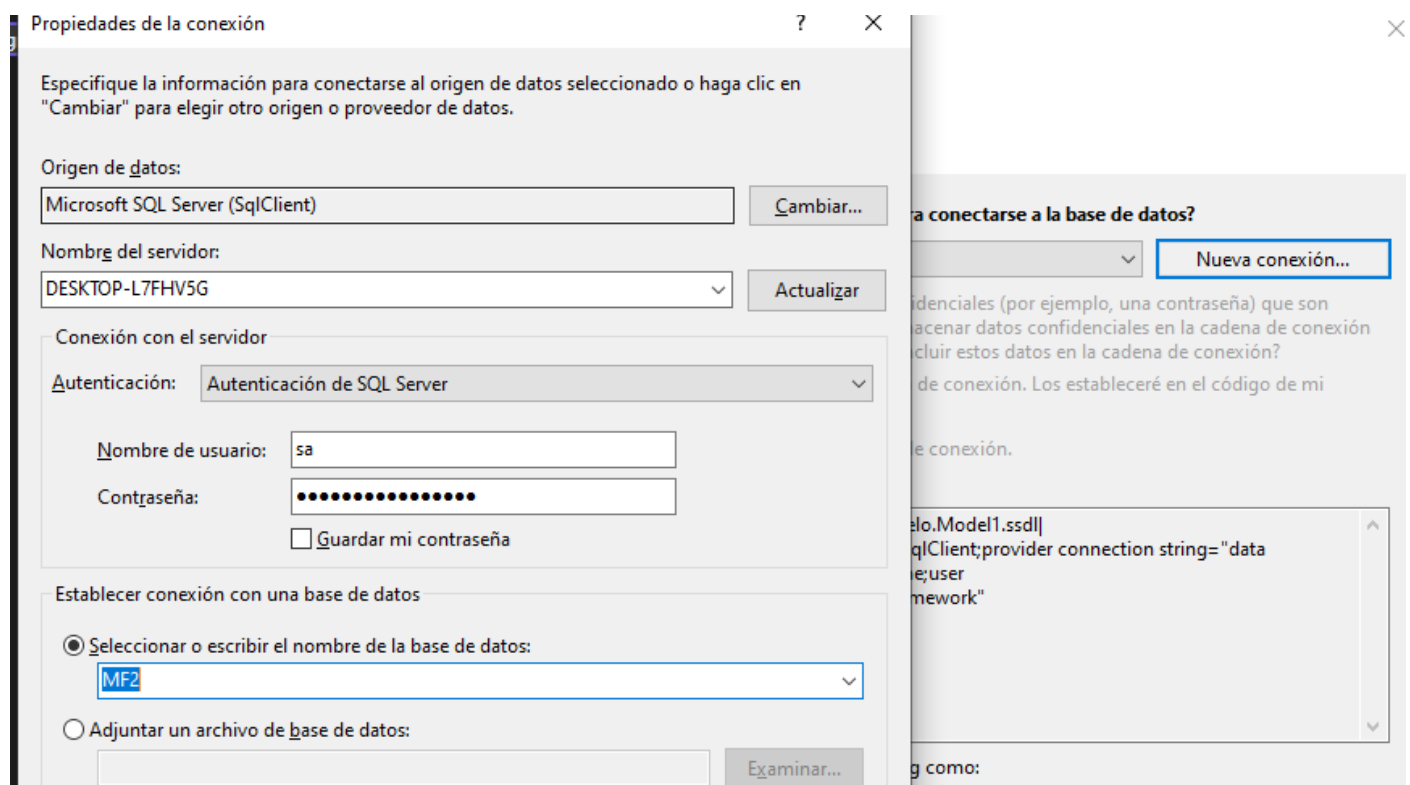
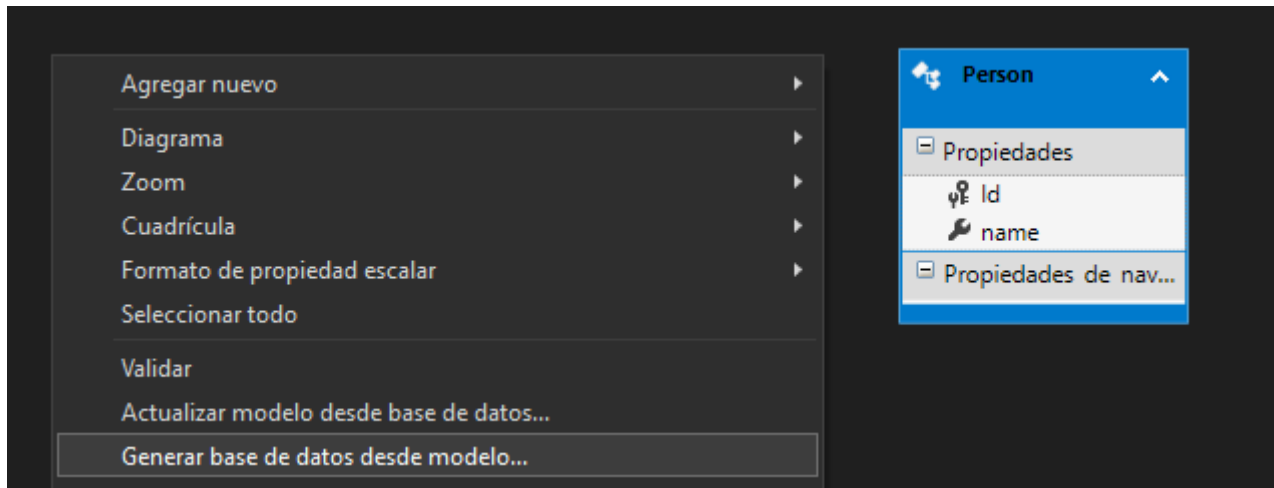
Aceptar

Cancelar

d. Crear una nueva Propiedad escalar en Person



e. Generar la BD y una nueva conexión



Microsoft Visual Studio



La base de datos 'MF2' no existe o no tiene permiso para verla.

¿Desea intentar crearla?

Sí

No

Asistente para generar base de datos

Elegir la conexión de datos

¿Qué conexión de datos debe usar la aplicación para conectarse a la base de datos?

desktop-l7fhv5g.MF2.dbo Nueva conexión...

Esta cadena de conexión parece contener datos confidenciales (por ejemplo, una contraseña) que son necesarios para conectarse con la base de datos. Almacenar datos confidenciales en la cadena de conexión puede suponer un riesgo para la seguridad. ¿Desea incluir estos datos en la cadena de conexión?

☐ No, excluir datos confidenciales de la cadena de conexión. Los estableceré en el código de mi aplicación.

☒ Sí, incluir datos confidenciales en la cadena de conexión.

Cadena de conexión:

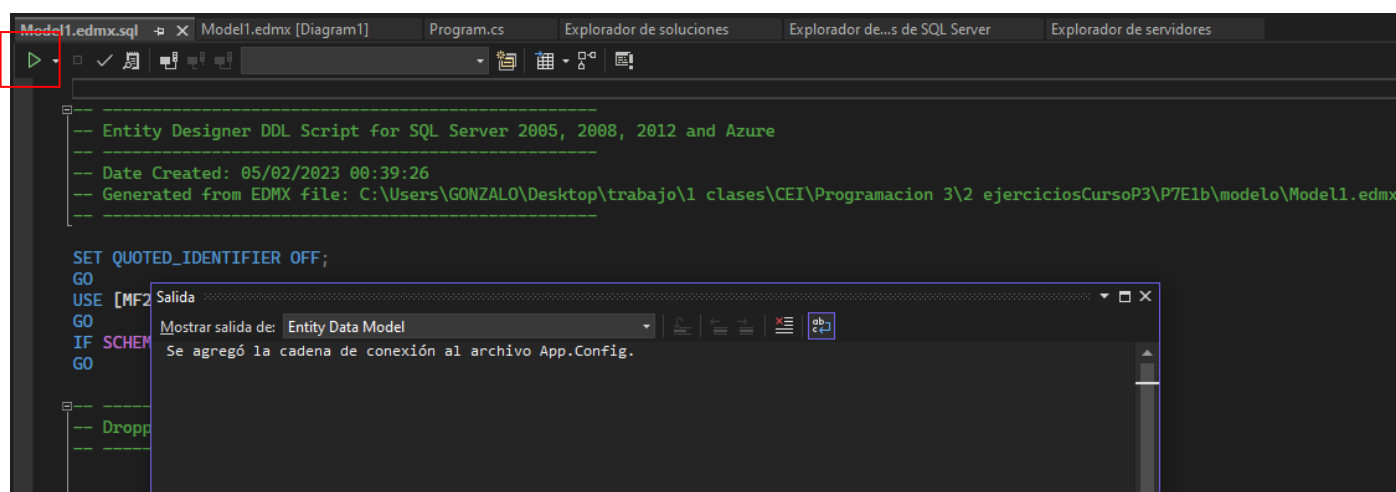
```
metadata=res://*/modelo.Model1.csdl|res://*/modelo.Model1.ssdl|
res://*/modelo.Model1.msl;provider=System.Data.SqlClient;provider connection string="data
source=DESKTOP-L7FHV5G;initial catalog=MF2;user
id=sa;password=*****;MultipleActiveResultSets=True;App=EntityFramework"
```

☒ Guardar configuración de conexión en App.Config como:

Model1Container

< Anterior Siguiente > Finalizar Cancelar

f. Finalmente



```
-- Entity Designer DDL Script for SQL Server 2005, 2008, 2012 and Azure
--
-- Date Created: 05/02/2023 00:39:26
-- Generated from EDMX file: C:\Users\GONZALO\Desktop\trabajo\1 clases\CEI\Programacion 3\2 ejerciciosCursoP3\P7E1b\modelo\Model1.edmx
--
SET QUOTED_IDENTIFIER OFF;
GO
USE [MF2]
GO
IF SCHEMA_ID(N'@@SERVERNAME', N'dbo') IS NULL
GO
-- Dropp
```

Salida

Mostrar salida de: Entity Data Model

Se agregó la cadena de conexión al archivo App.Config.

Conectar

History

**Examinar**

Escriba aquí para filtrar la lista

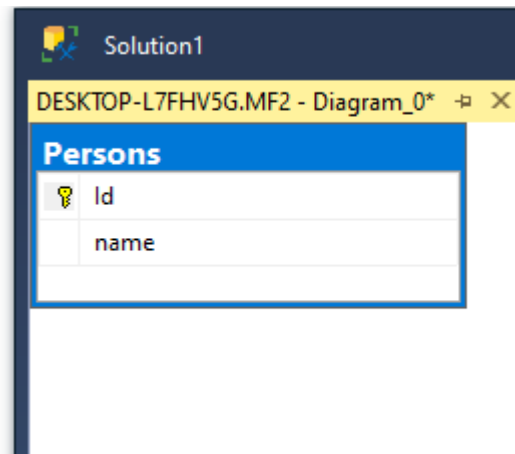
Local

DESKTOP-L7FHV5G  
DESKTOP-L7FHV5G  
MSSQLLocalDB

Red

Azure

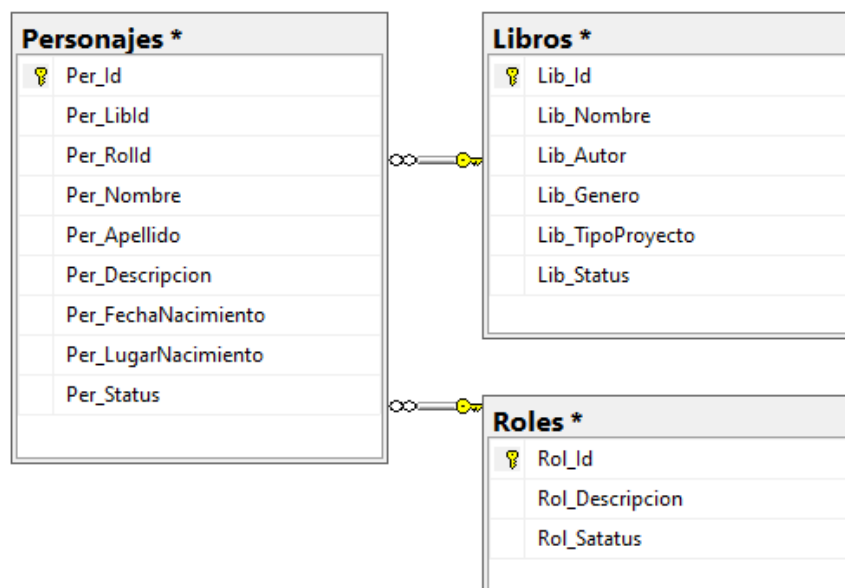
gestionAlumnos  
GestionPedidos  
MF  
MF2  
Diagramas de base de datos  
Tablas  
Vistas  
Recursos externos  
Sinónimos  
Programación  
Service Broker  
Almacenamiento  
Seguridad



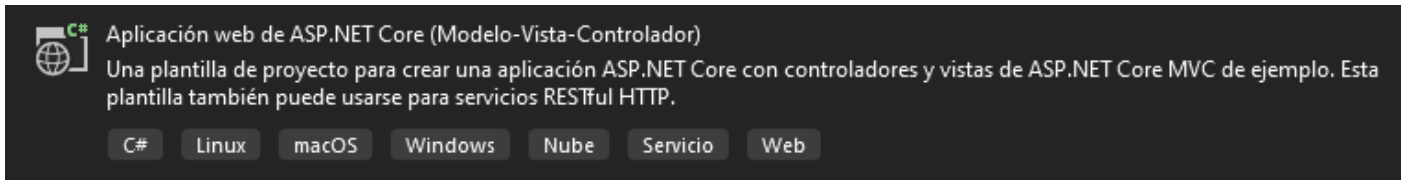
## 2- Crear un proyecto empleando Database First.

Para esto puede seguir los pasos:

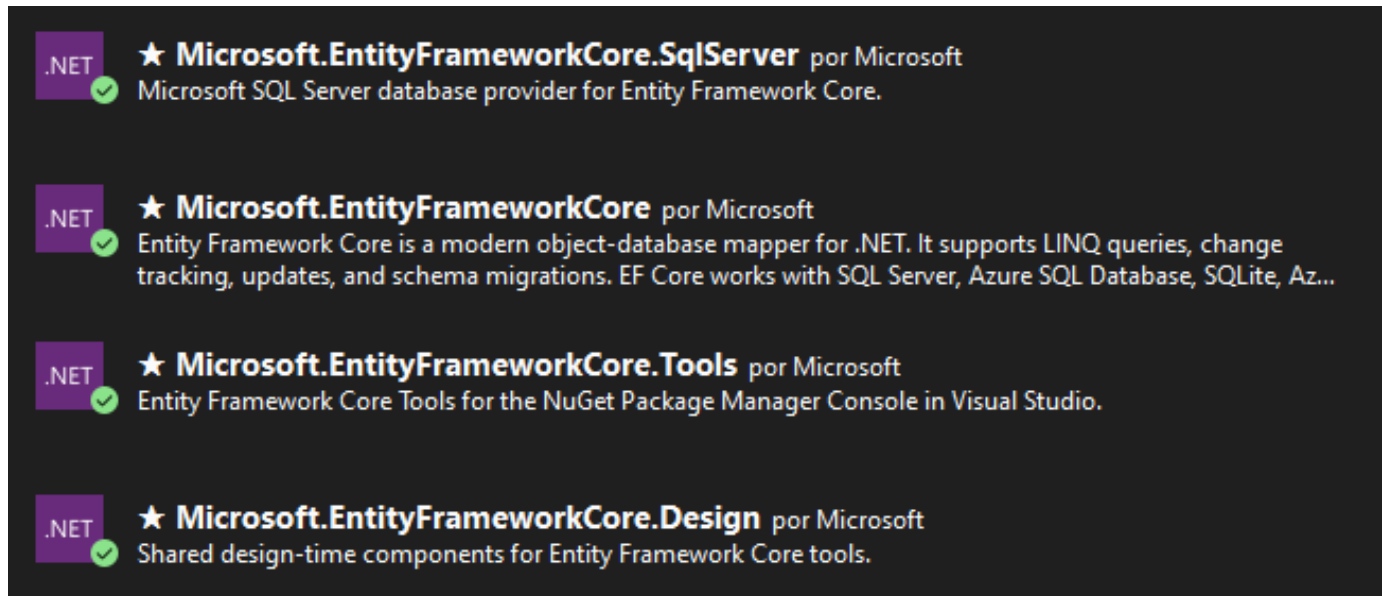
- Crear la base de datos db\_escritorio ejecutando el script de SQL del archivo Practica7-E2.txt



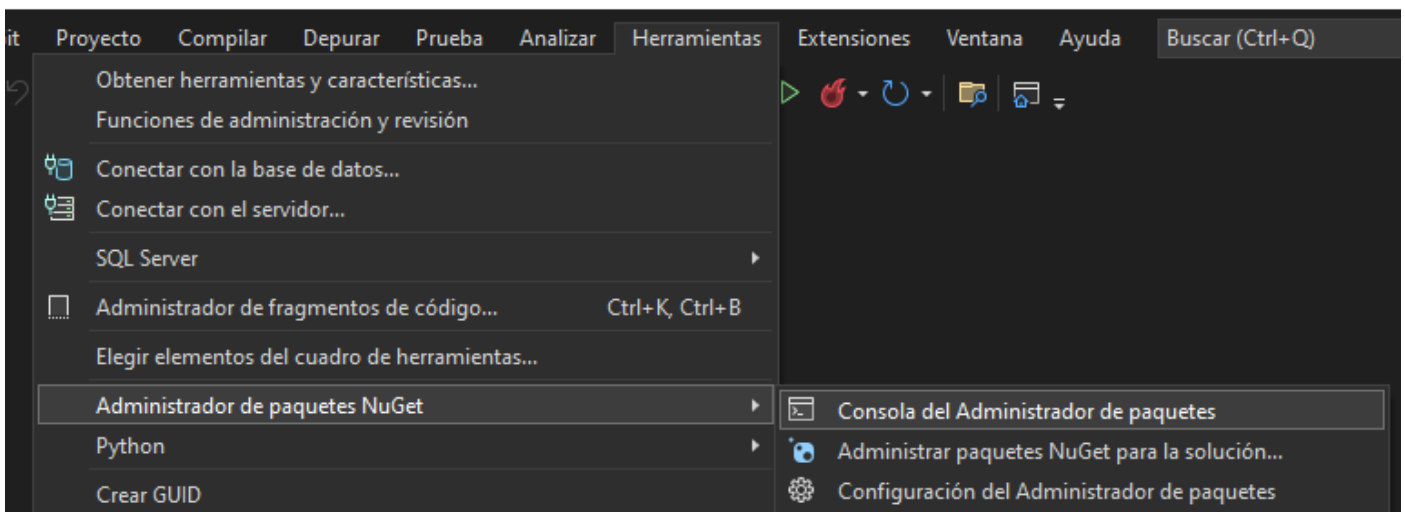
b. Crear un proyecto en VS según:



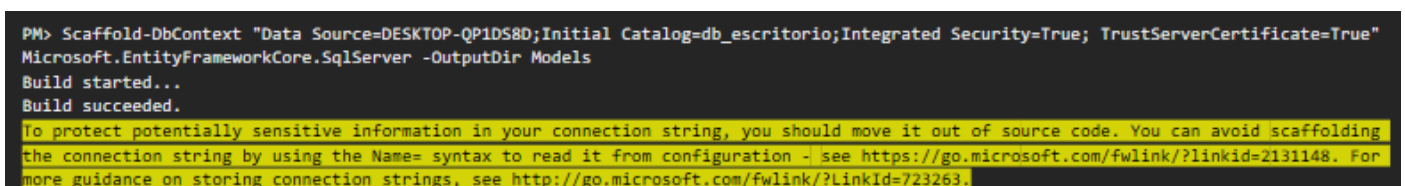
c. En VS vaya a Administrar paquetes de NuGet y descargar los siguientes paquetes:

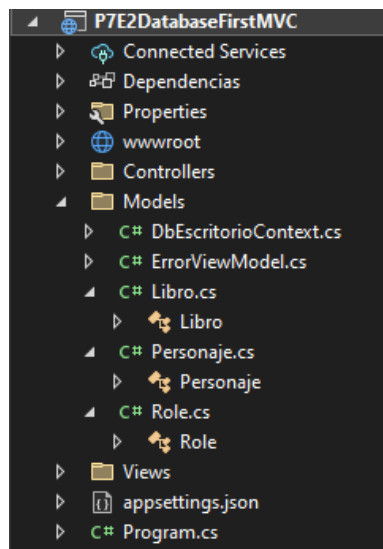


d. Luego ejecute la consola del Administrador de paquetes



e. Finalmente ejecute la línea de conexión del archivo Practica7-E2.txt

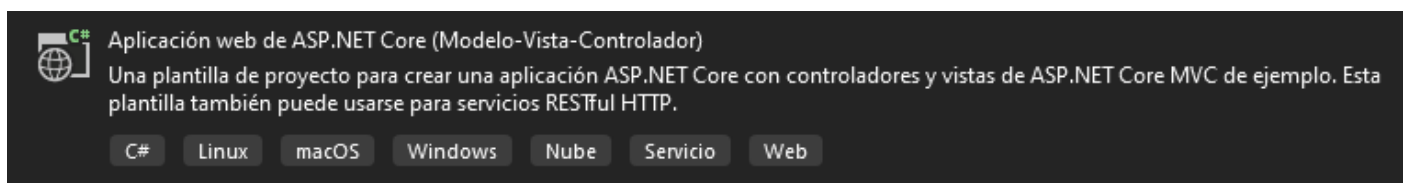




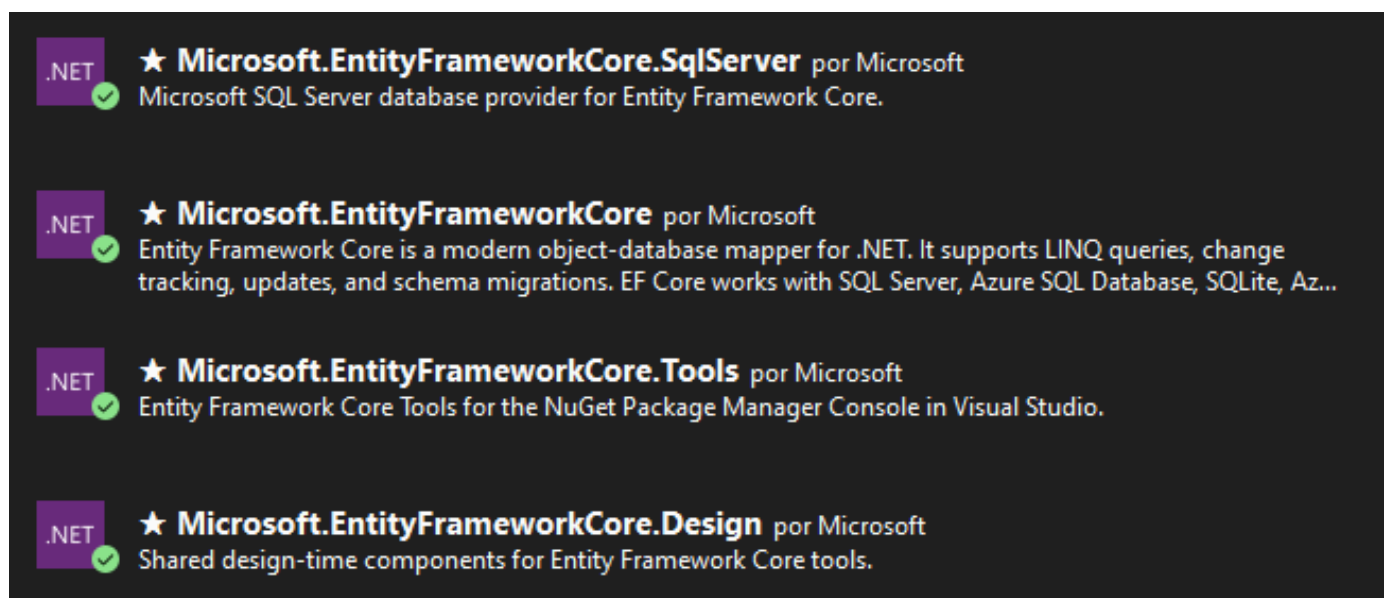
### 3- Crear un proyecto empleando Code First.

Para esto puede seguir los pasos:

- Crear un proyecto en VS según:



- En VS vaya a Administrar paquetes de NuGet y descargar los siguientes paquetes:



- Crear las siguientes clases:

```
public class Personajes
{
    [Key]
    [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
    0 referencias
    public int Per_Id { get; set; }

    0 referencias
    public int Per_LibId { get; set; }

    0 referencias
    public Libros Libro { get; set; }

    0 referencias
    public int Per_RolId { get; set; }

    0 referencias
    public Roles Per_Rol { get; set; }

    [Required]
    [StringLength(255)]
    0 referencias
    public string Per_Nombre { get; set; }

    [Required]
    [StringLength(255)]
    0 referencias
    public string Per_Apellido { get; set; }

    [Required]
    [StringLength(10)]
    0 referencias
    public string Per_Descripcion { get; set; }

    [Required]
    [StringLength(255)]
    0 referencias
    public DateTime Per_FechaNacimiento { get; set; }

    [Required]
    [StringLength(255)]
    0 referencias
    public string Per_LugarNacimiento { get; set; }

    [Required]
    [StringLength(255)]
    0 referencias
    public string Per_Status { get; set; }
}
```



```

public class Roles
{
    [Key]
    [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
    0 referencias
    public int Lib_Id { get; set; }

    [Required]
    [StringLength(255)]
    0 referencias
    public string Rol_Descripcion { get; set; }

    [Required]
    [StringLength(255)]
    0 referencias
    public string Rol_Status { get; set; }

    0 referencias
    public ICollection<Personajes> Personajes { get; set; }
}

```

```

public class Libros
{
    [Key]
    [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
    0 referencias
    public int Lib_Id { get; set; }

    [Required]
    [StringLength(255)]
    0 referencias
    public string Lib_Nombre { get; set; }

    [Required]
    [StringLength(255)]
    0 referencias
    public string Lib_Autor { get; set; }

    [Required]
    [StringLength(255)]
    0 referencias
    public string Lib_Genero { get; set; }

    [Required]
    [StringLength(10)]
    0 referencias
    public string Lib_TipoProyecto { get; set; }

    [Required]
    [StringLength(255)]
    0 referencias
    public string Lib_Status { get; set; }

    0 referencias
    public ICollection<Personajes> Personajes { get; set; }
}

```

```
public class MyDbContext : DbContext
{
    0 referencias
    public DbSet<Libros> Libros { get; set; }
    0 referencias
    public DbSet<Personajes> Personajes { get; set; }
    0 referencias
    public DbSet<Roles> Roles { get; set; }
    0 referencias
    public MyDbContext(DbContextOptions<MyDbContext>options) : base(options)
    {
    }
}
```

d. Modificar las clases:

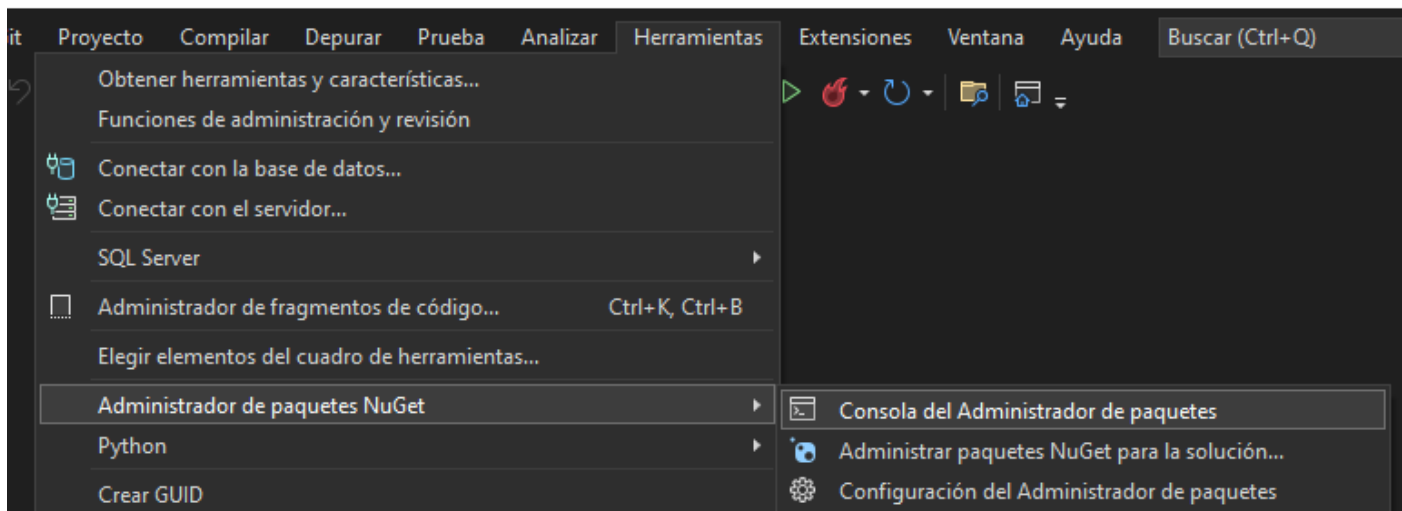
appsettings.json:

```
"AllowedHosts": "*",
"ConnectionStrings": { "Connection": "Data Source=DESKTOP-QP1DS8D;Initial Catalog=db_escritorio2;Integrated Security=True; TrustServerCertificate=True" }
```

Program:

```
// Conexion BD
builder.Services.AddDbContext<MyDbContext>(options =>
options.UseSqlServer("name=ConnectionStrings:Connection"));
```

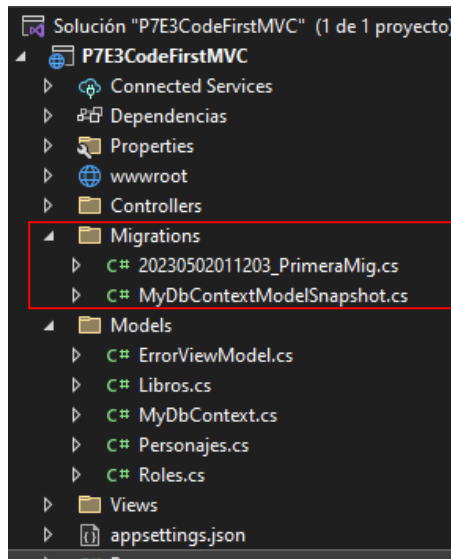
e. Luego ejecute la consola del Administrador de paquetes



f. Finalmente ejecute:

Add-Migration PrimeraMig

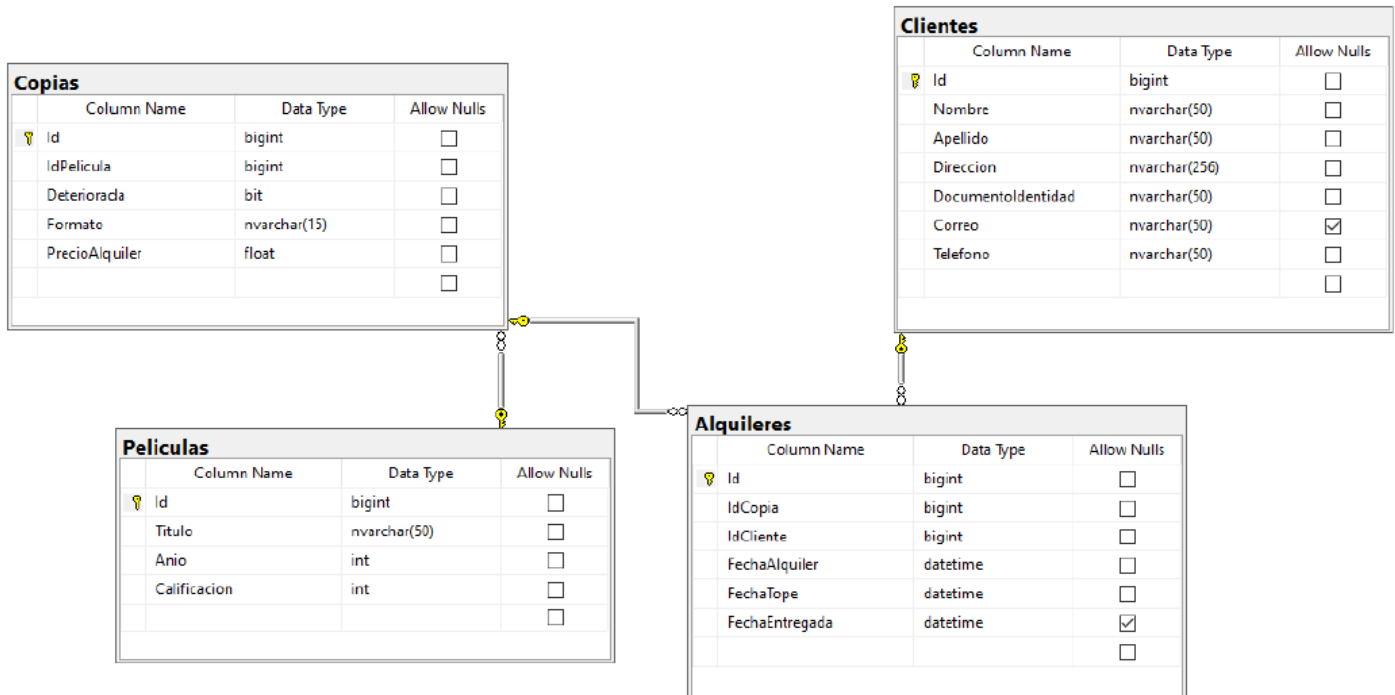
Update-Database



#### 4- Crear una nueva base datos llamada PRG3\_EF\_PRO usando SQL Management Studio.

1. Ejecutar el script “Tablas y datos – Acceso Base Datos 0” para generar las tablas e insertar los datos.
2. Crear un nuevo proyecto en Visual Studio que implemente lo siguiente:
  - a. Una capa de acceso a datos, conteniendo un proyecto que implemente Entity Framework y contiene una modelo de datos creado con la modalidad “Code First desde base de datos”.
  - b. Una aplicación de consola, que cuente con un menú interactivo y permita realizar las siguientes operaciones sobre nuestra base de datos:
    - i. Insertar un nuevo autor
    - ii. Insertar un nuevo Libro (El libro debe pertenecer a un actor existente).
    - iii. Dado el identificador de un libro, modificar la cantidad de ventas de este.
    - iv. Consultar los libros disponibles ordenados por la cantidad de ventas, de mayor a menor, mostrando el nombre del autor.
    - v. Consultar los autores ordenados de manera alfabética
    - vi. Dado el identificador de un libro, borrar el mismo de la base de datos.

## 5- Dado el siguiente diagrama, crear una nueva base de datos en SQL Server, llamada PRG3\_EF\_PR1.



1. En un nuevo proyecto en Visual Studio, utilizando el patrón de arquitectura de software por capas, implementar lo siguiente:

Para gestionar un negocio de alquiler de películas, es necesario contar con un sistema que permita realizar la administración de los clientes, las películas en stock y de los alquileres realizados.

2. Para esto se definen los siguientes requisitos:

- i. Contar con un menú y sub menús interactivos en una aplicación de consola que guíe el usuario en las distintas operaciones de gestión.
- ii. Poder dar de alta nuevos clientes.
- iii. Poder dar de alta nuevas películas (Las calificaciones de una película van del 1 al 10).
- iv. Poder dar de alta nuevas copias de una película (Los formatos de las películas son DVD y BluRay).
- v. Registrar nuevos alquileres (Se asume que la fecha de alquiler es la fecha de creación del registro, el periodo de alquiler siempre es 3 días), tener en cuenta que solo se podrán registrar alquileres de copias que estén disponibles para alquilar.
- vi. Actualización de alquileres activos, poder marcar un alquiler como entregado.
- vii. Actualización de copias, poder marcar una copia como deteriorada.
- viii. Tener la posibilidad de realizar las siguientes consultas:
  - a) Copias en stock, mostrando los datos de la película.
  - b) Clientes registrados.
  - c) Copias disponibles para alquilar (Que no cuentan con un alquiler activo), mostrando los datos de la película.
  - d) Historial de alquileres mostrando, además, el título de la película, el formato, nombre del cliente, datos de contacto e id.
  - e) Historial de alquileres para un cliente, mostrando primero los alquileres activos, si es que cuenta con alguno (Se debe solicitar el id de cliente previo a mostrar sus alquileres).
  - f) Alquileres activos, mostrando además título de la película, formato, nombre del cliente, datos de contacto e id (Se entiende por activo aquellos alquileres aún no entregados).
  - g) Alquileres activos cuya fecha tope fue excedida, mostrando además título de la película, formato, nombre del cliente, datos de contacto e id.