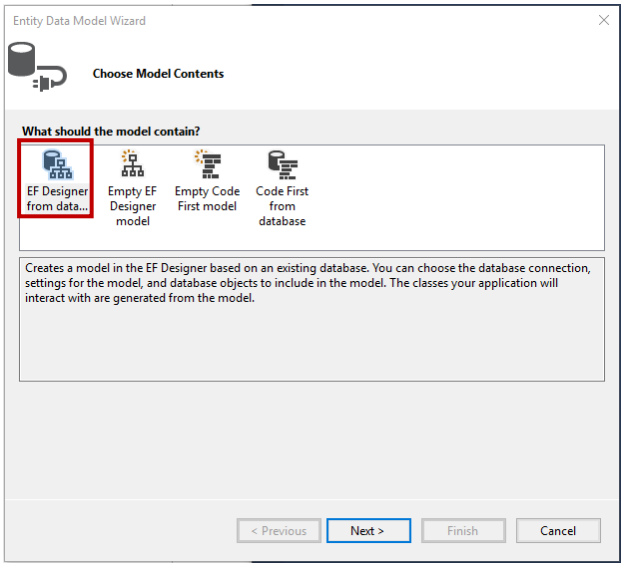
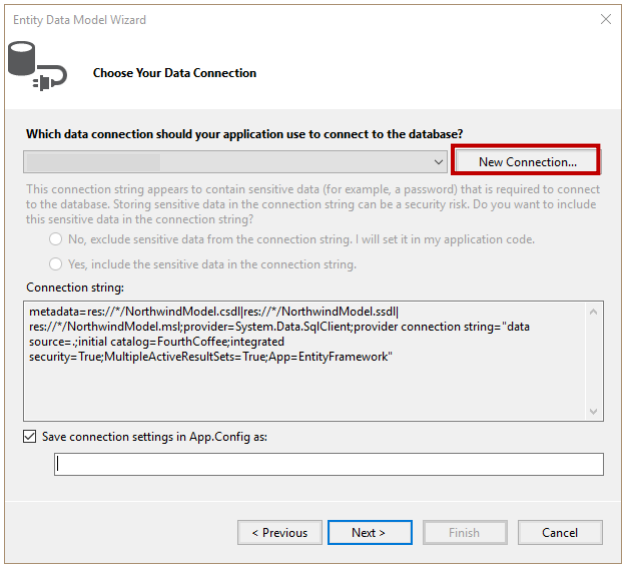
EF 6 → Database First

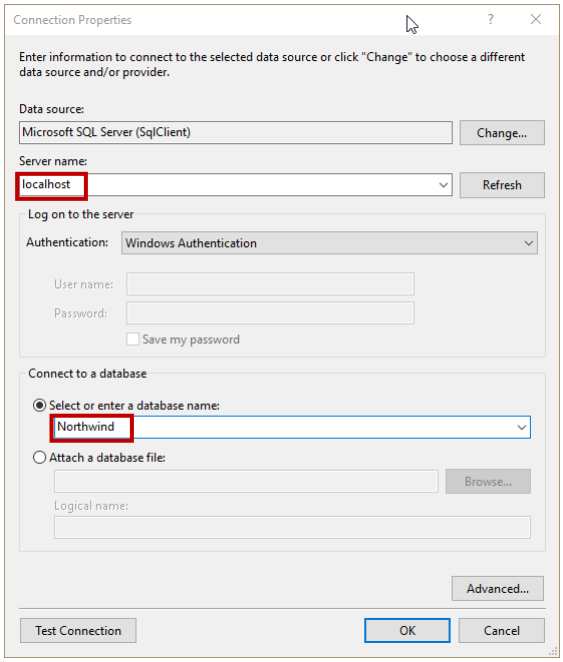
1. Criar um projeto de consola e adicionar ADO.NET Entity Data Model:

1.1. Adicionar um novo item: ADO.NET Entity Data Model
1.2. Mudar o nome do model, por exemplo, NorthwindModel.
1.3. Seguir o wizard Entity Data Model Wizard.
1.4. Analisar o que foi gerado.
1.5. Inserir e listar registos nas tabelas.

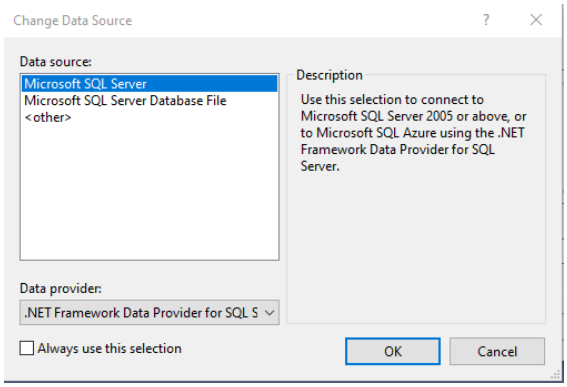
2. Entity Data Model Wizard:

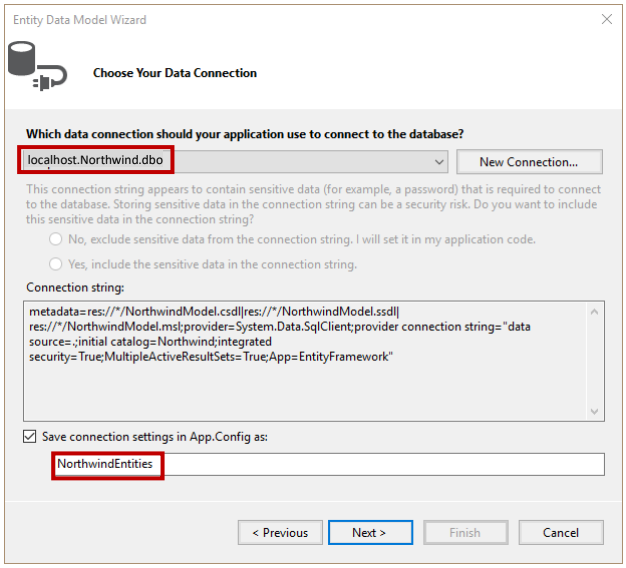


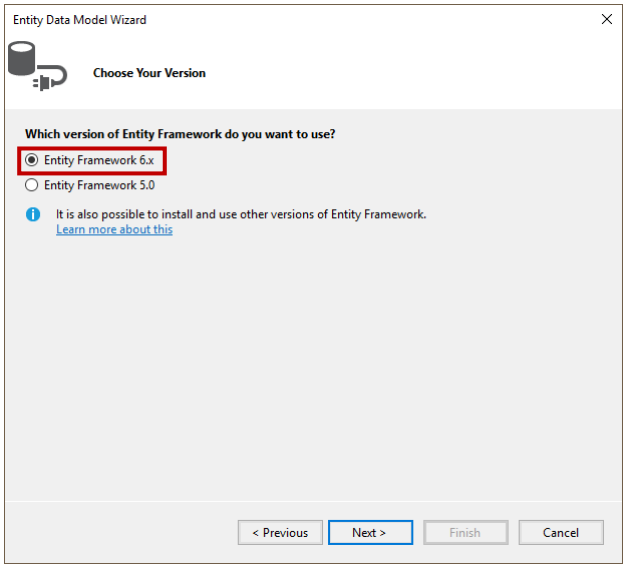


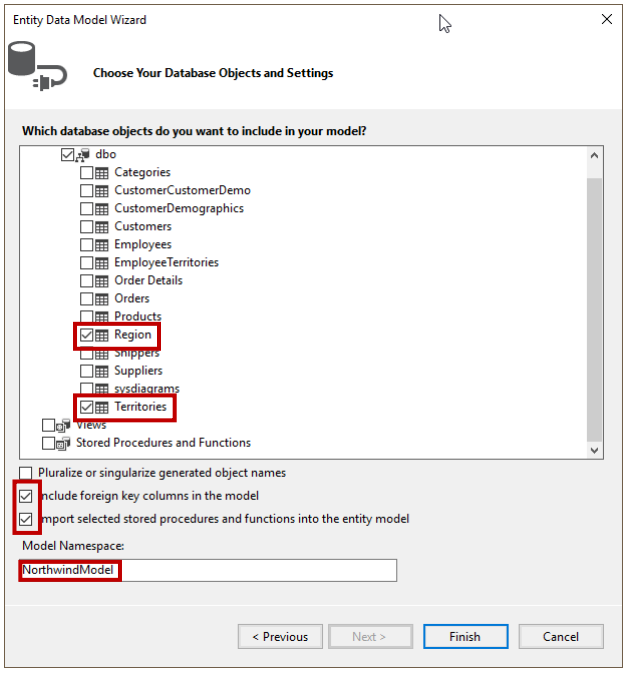


Se necessário, clicar em Change para escolher o Data Source Microsoft SQL Server:

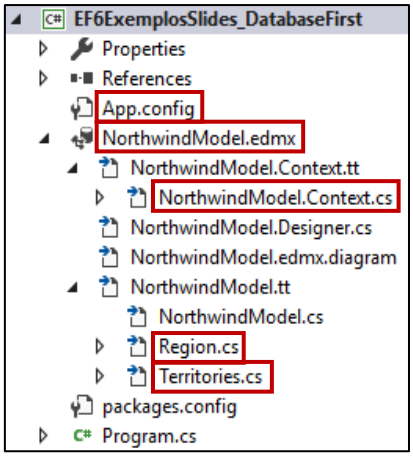








3. Analisar o que foi gerado:



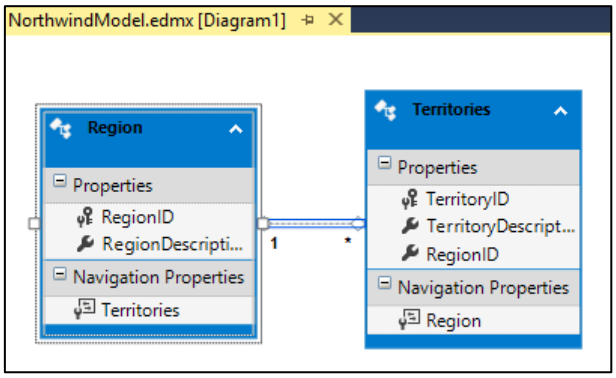
4. App.config:

No ficheiro de configuração da aplicação App.config, foi guardada a connection string criada pelo wizard.

```
<connectionStrings>
  <add
    name="NorthwindEntities"
    connectionString=
        "metadata=res://*/NorthwindModel.csdl|res://*/NorthwindModel.ssdl|res://*/NorthwindModel.msl;
        provider=System.Data.SqlClient;provider connection string=&amp;quot;
        data source=.;
        initial catalog=Northwind;integrated security=True;
        MultipleActiveResultSets=True;
        App=EntityFramework&amp;quot;"
    providerName="System.Data.EntityClient" />
</connectionStrings>
```

5. Modelo de dados - NorthwindModel.edmx:

O modelo de dados é gerado automaticamente a partir das tabelas da base de dados, mostrando as propriedades e relações entre elas.



6. NorthwindModel.Context.cs:

O ficheiro de contexto contém a classe derivada de DbContext e cria uma propriedade para cada classe do modelo, o que corresponde a cada uma das tabelas da base de dados.

NorthwindModel.Context.cs

EF6ExemplosSlides_DatabaseFirst

```
1   //------------------------------------------------------------------------------
2   // <auto-generated>
3   //     This code was generated from a template.
4   //
5   //     Manual changes to this file may cause unexpected behavior in your application.
6   //     Manual changes to this file will be overwritten if the code is regenerated.
7   // </auto-generated>
8   //------------------------------------------------------------------------------
9
10  namespace EF6ExemplosSlides_DatabaseFirst
11  {
12      using System;
13      using System.Data.Entity;
14      using System.Data.Entity.Infrastructure;
15
16      public partial class NorthwindEntities : DbContext
17      {
18          public NorthwindEntities()
19              : base("name=NorthwindEntities")
20          {
21          }
22
23          protected override void OnModelCreating(DbModelBuilder modelBuilder)
24          {
25              throw new UnintentionalCodeFirstException();
26          }
27
28          public virtual DbSet<Region> Region { get; set; }
29          public virtual DbSet<Territories> Territories { get; set; }
30      }
31  }
32
```

7. Classes do modelo – Territories.cs e Region.cs:

São geradas as propriedades da classe, o que corresponde aos campos da tabela.

No caso de serem incluídos no modelo tabelas relacionadas na base de dados são geradas dois tipos de propriedades: propriedades escalares, que devolvem um único valor e propriedades de navegação, que representam a relação entre as tabelas.

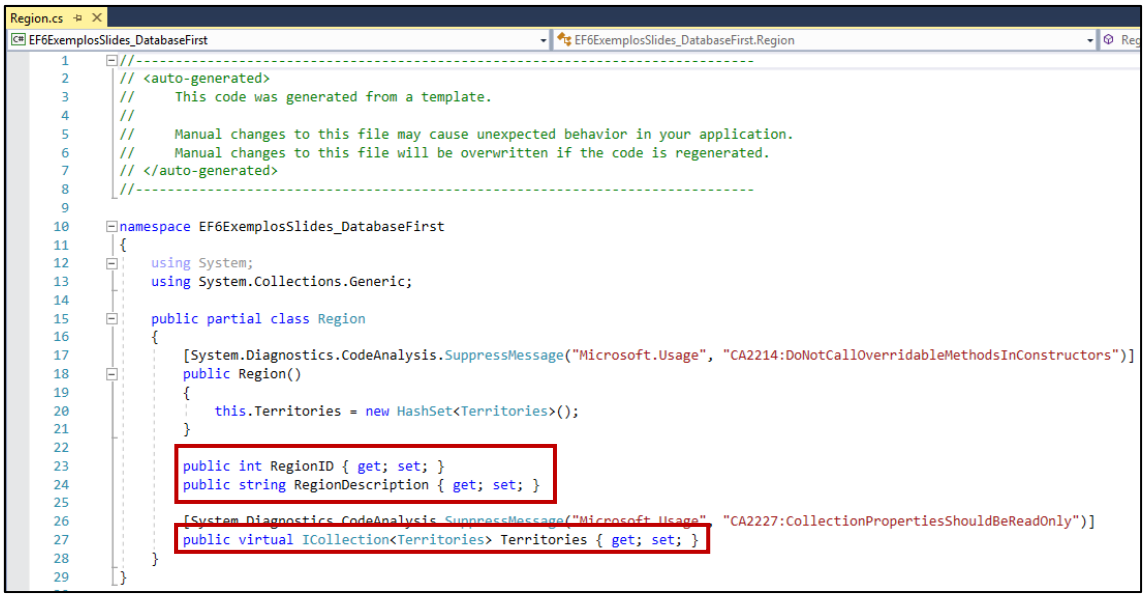
Neste exemplo, as classes Region e Territories têm uma relação de 1-n pela propriedade RegionID, pelo que as propriedades de navegação são:

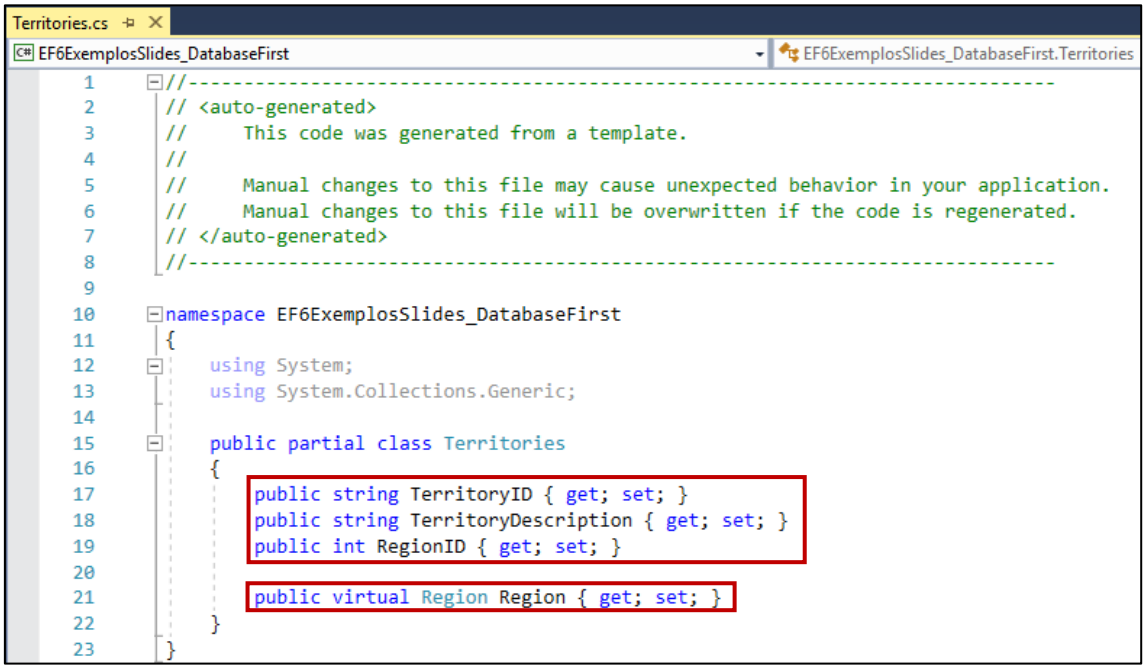
- Na classe Region: 1 região pode ter um ou mais territórios

```
public virtual ICollection<Territories> Territories { get; set; }
```

- Na classe Territories: 1 território só pode ter 1 região

```
public virtual Region Region { get; set; }
```





8. Inserir e listar novos registos:

```
using System;
using System.Linq;

namespace EF6ExemplosSlides_DatabaseFirst
{

    class Program
    {

        static void Main(string[] args)
        {

            using (var db = new NorthwindEntities())  // Classe que herda de DbContext
            {

                #region Nova região
                Region region = new Region();
                region.RegionID = 5;    // Não é identity
                region.RegionDescription = "Norte";
                db.Region.Add(region);

                var count1 = db.SaveChanges();
                Console.WriteLine($"{count1} nova região gravada.\n\n");

                var query1 = db.Region.Select(r => r).OrderBy(r => r.RegionID);
                foreach (var item in query1)
                {
                    Console.WriteLine($"{item.RegionID}: {item.RegionDescription}");
                }
                #endregion

                #region Novo território da nova região
                Territories territories = new Territories();
                territories.TerritoryID = "00001";    // Não é identity
                territories.TerritoryDescription = "Porto";
                territories.RegionID = 5;
                db.Territories.Add(territories);

                var count2 = db.SaveChanges();
                Console.WriteLine($"\n\n\n{count2} novo território gravado.\n\n");

                var query2 = db.Territories.Select(r => r).OrderBy(r => r.TerritoryID);
                foreach (var item in query2)
                {
                    Console.WriteLine($"{item.TerritoryID}: {item.TerritoryDescription}");
                }
                #endregion

                Console.ReadKey();

            }

        }

    }

}
```