

Curso de C#

Aula 7

Professores

Célio Alencar de Assis
Fábio Duarte Machado



Agenda

- O que é o Entity Framework
- Banco de dados e ORM
- Ler dados do banco de dados
- CRUD no EF

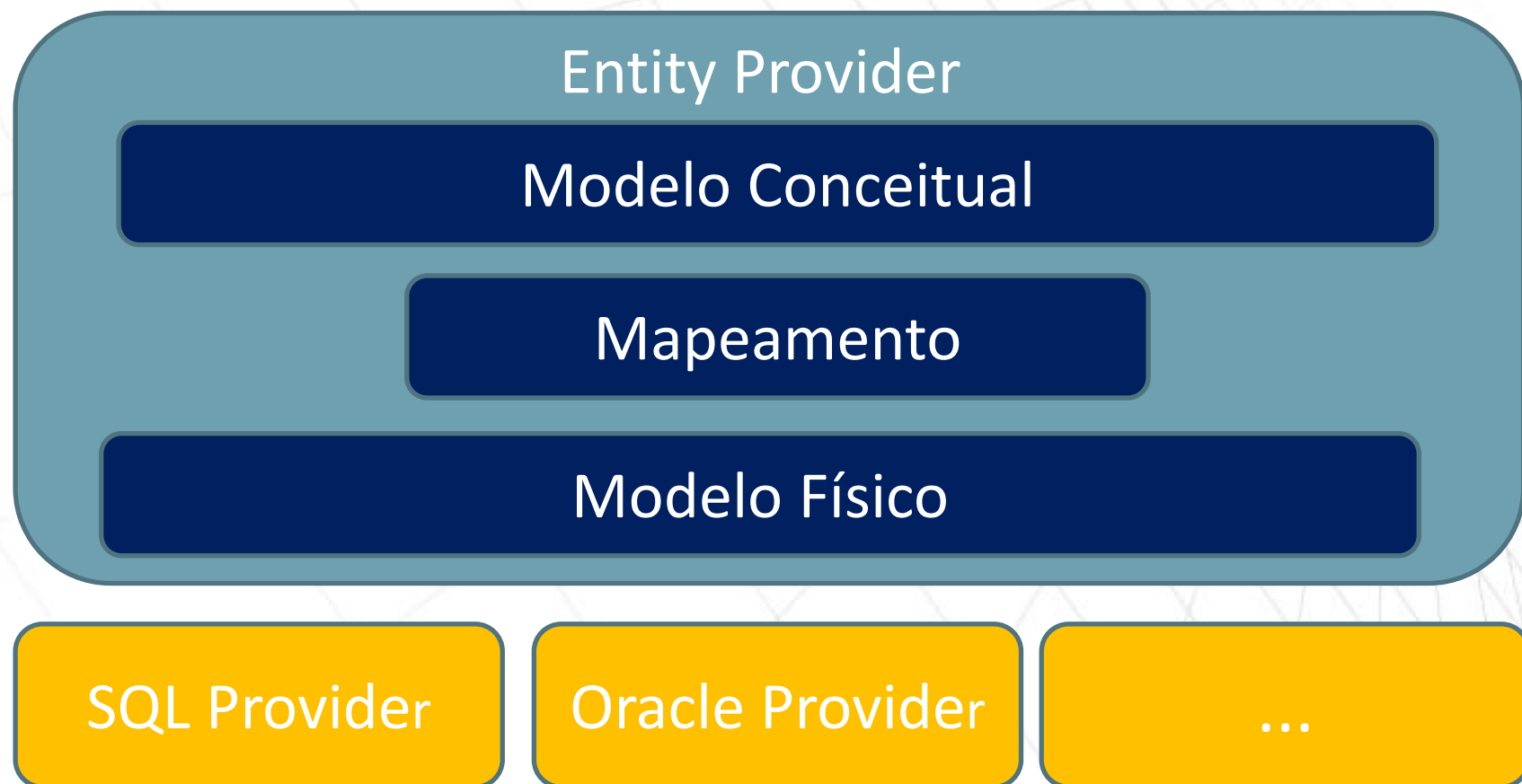
O que é o Entity Framework

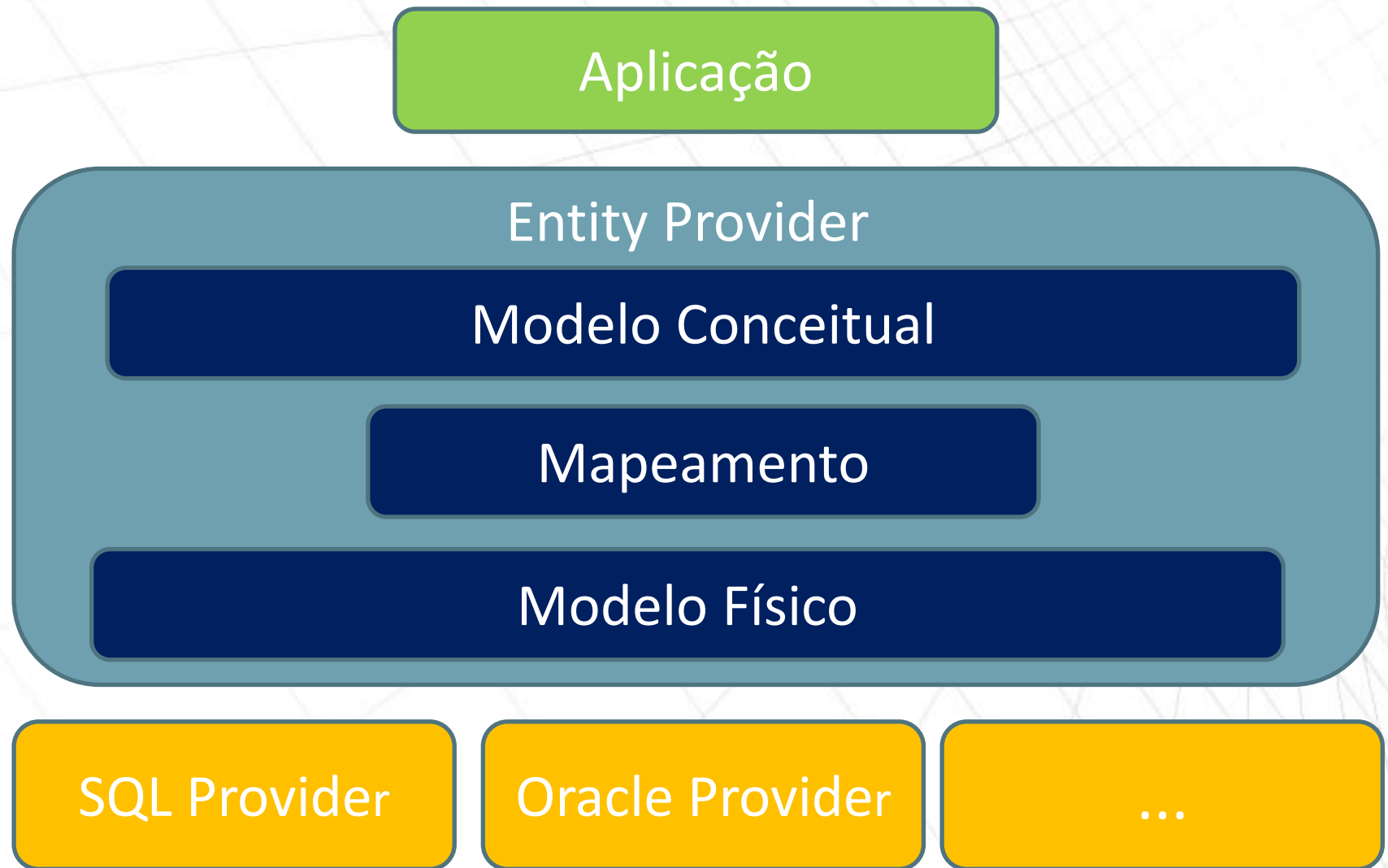
- Extensão do modelo ADO.NET
 - Criado para suportar qualquer banco de dados;
- Como funciona ?
 - Abstrai o modelo do banco de dados:
 - Modelo Conceitual (designer) e Modelo Físico (bd)
 - Faz o mapeamento Objeto Relacional
- Instalação
 - Já faz parte do Visual Studio
 - Instalar o Provider de acordo com o banco

SQL Provider

Oracle Provider

...





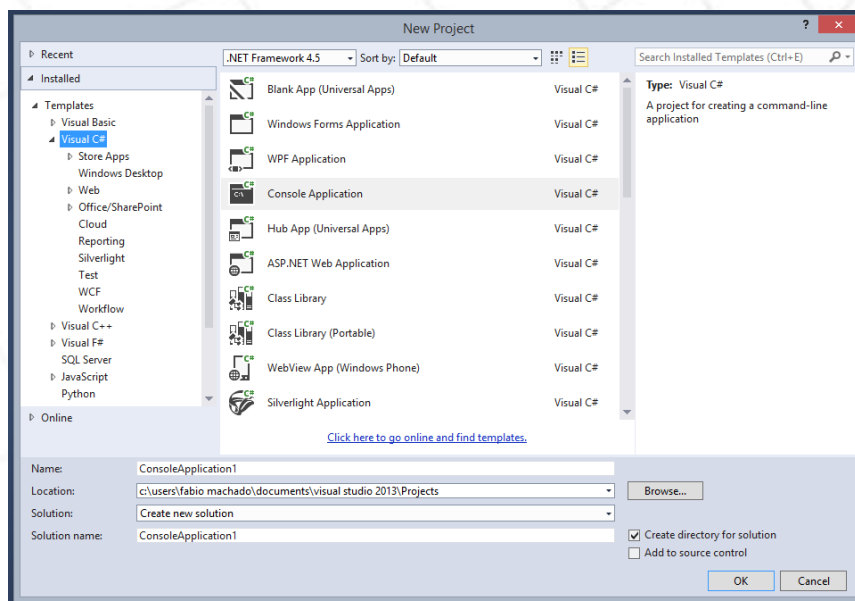
- EF usa o ADO.NET Data Provider
 - Suportado pela maioria dos banco de dados do mercado.
 - <http://msdn.microsoft.com/pt-br/data/dd363565.aspx>

Criar um banco de dados no SQL

- Grupo
 - Id, Nome
- Produto
 - Id, Nome, IdGrupo, Custo, Venda, Saldo

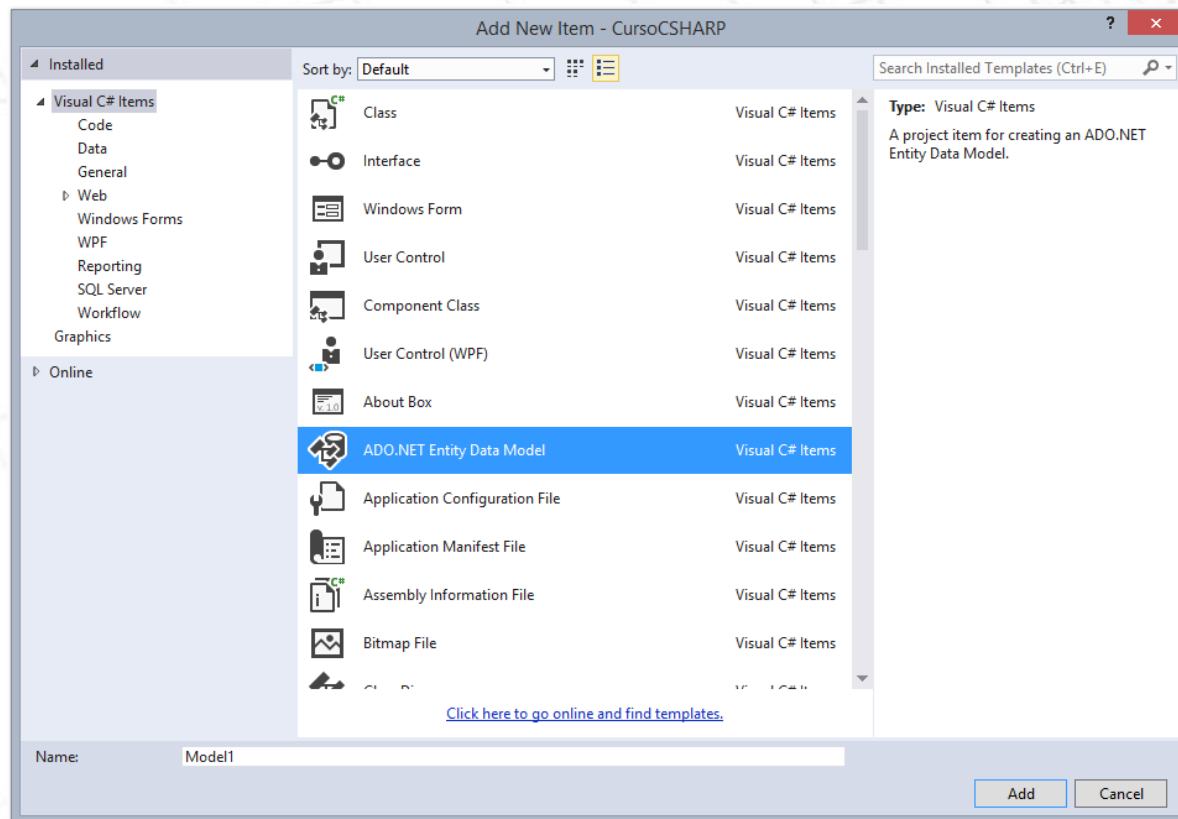
Banco de Dados e ORM

- Criando o mapeamento do Banco com o EF.
 - Criar um projeto Console Application



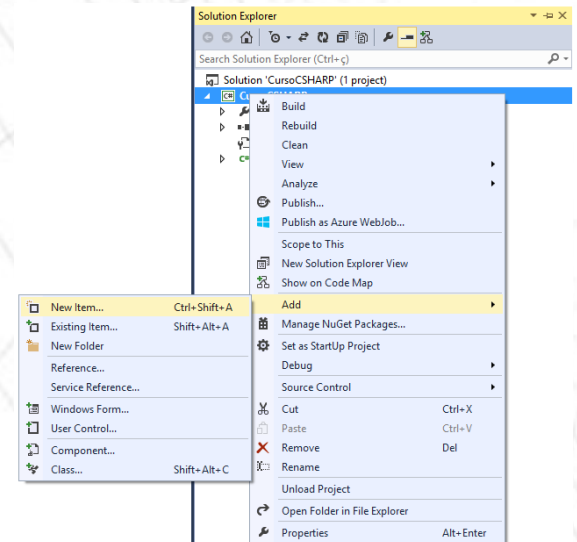
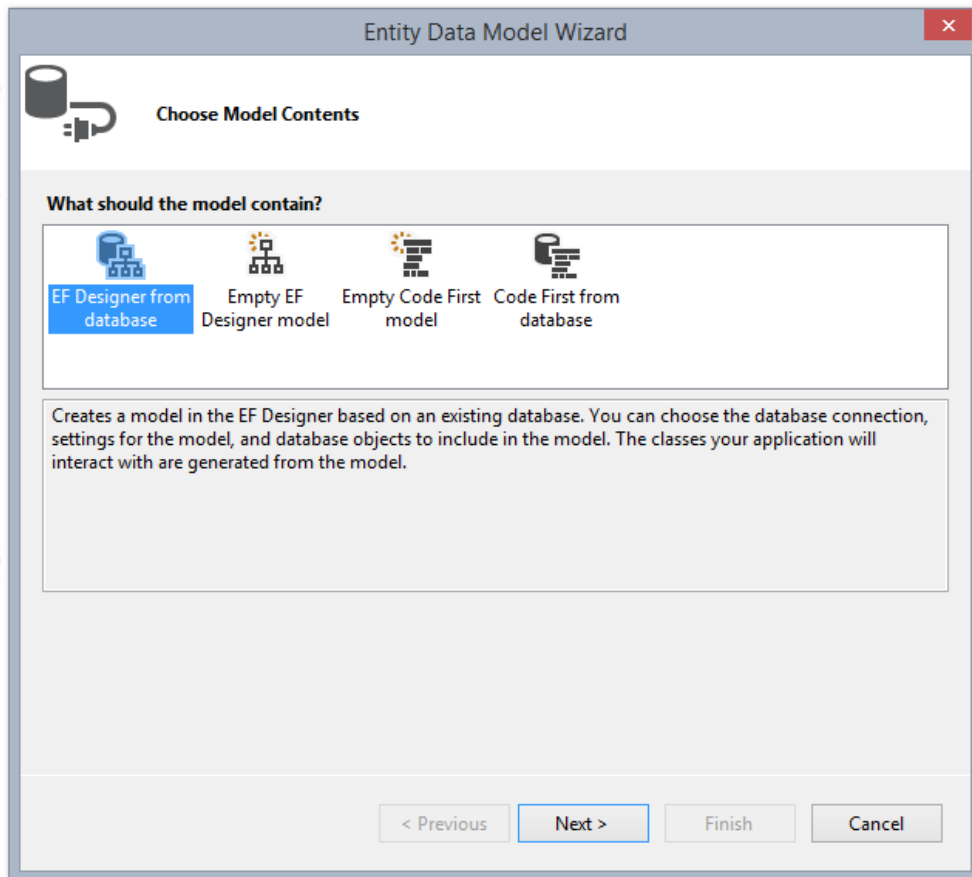
Banco de Dados e ORM

– Adicionar o Data Model



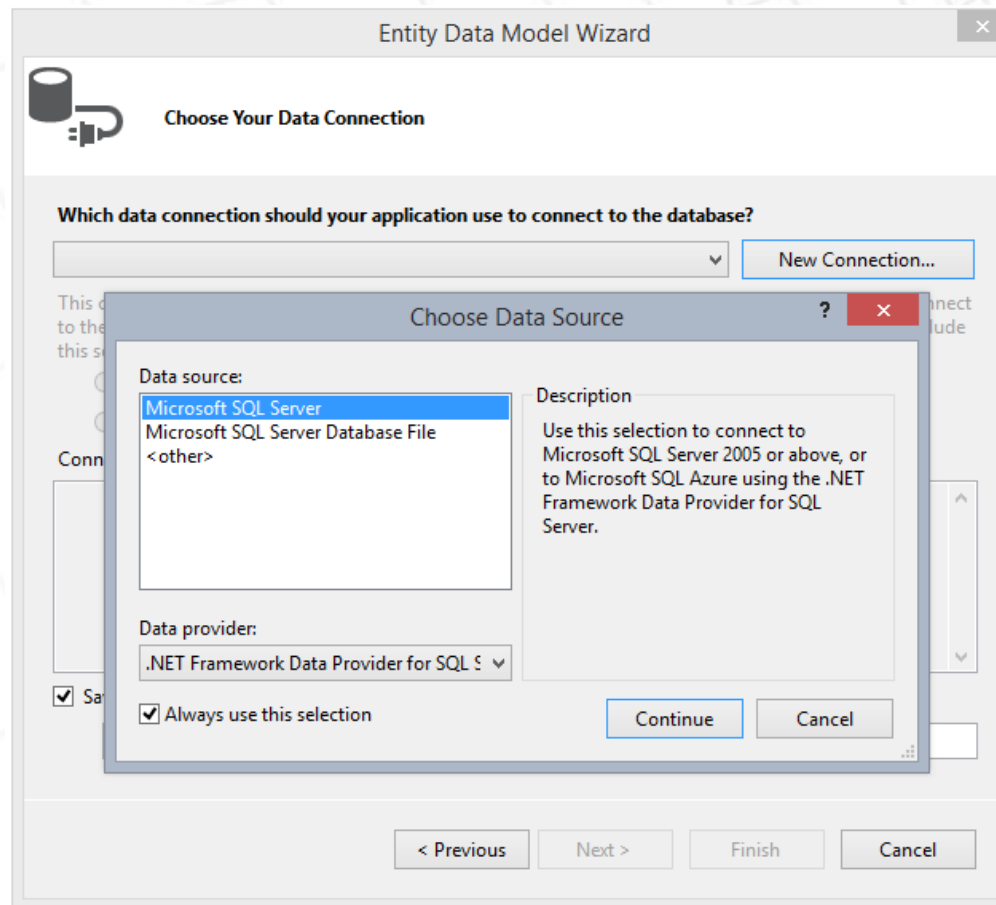
Banco de Dados e ORM

— Adicionar o Data Model



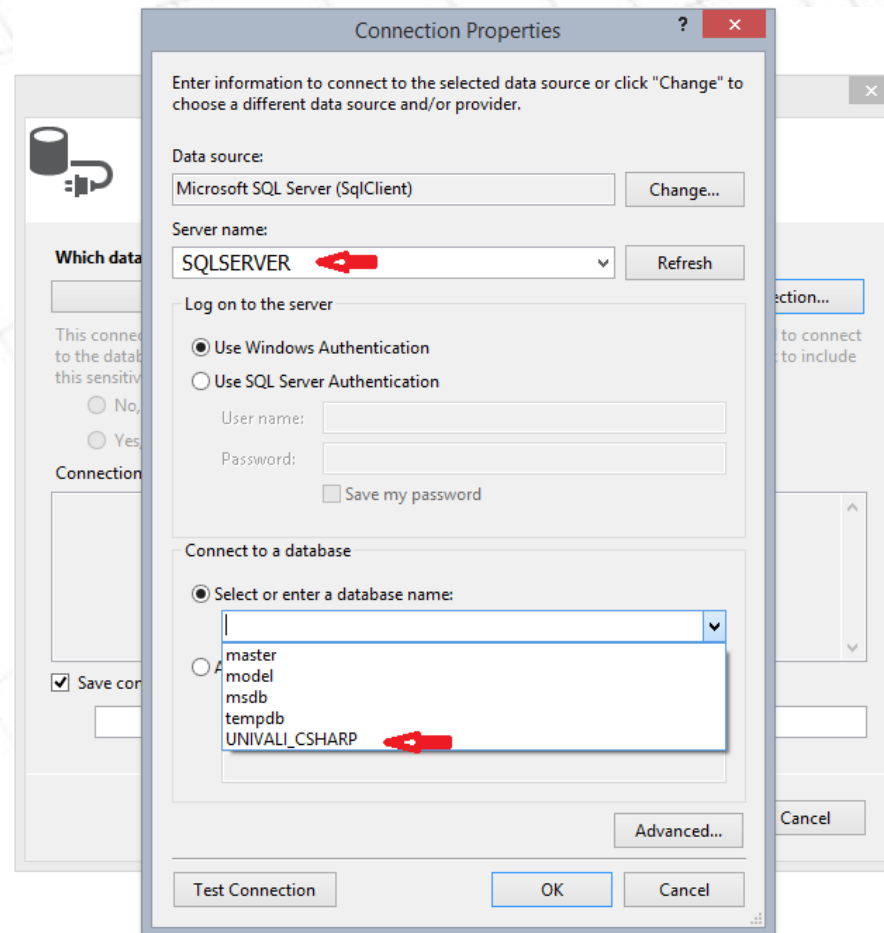
Banco de Dados e ORM

- Criar a conexão



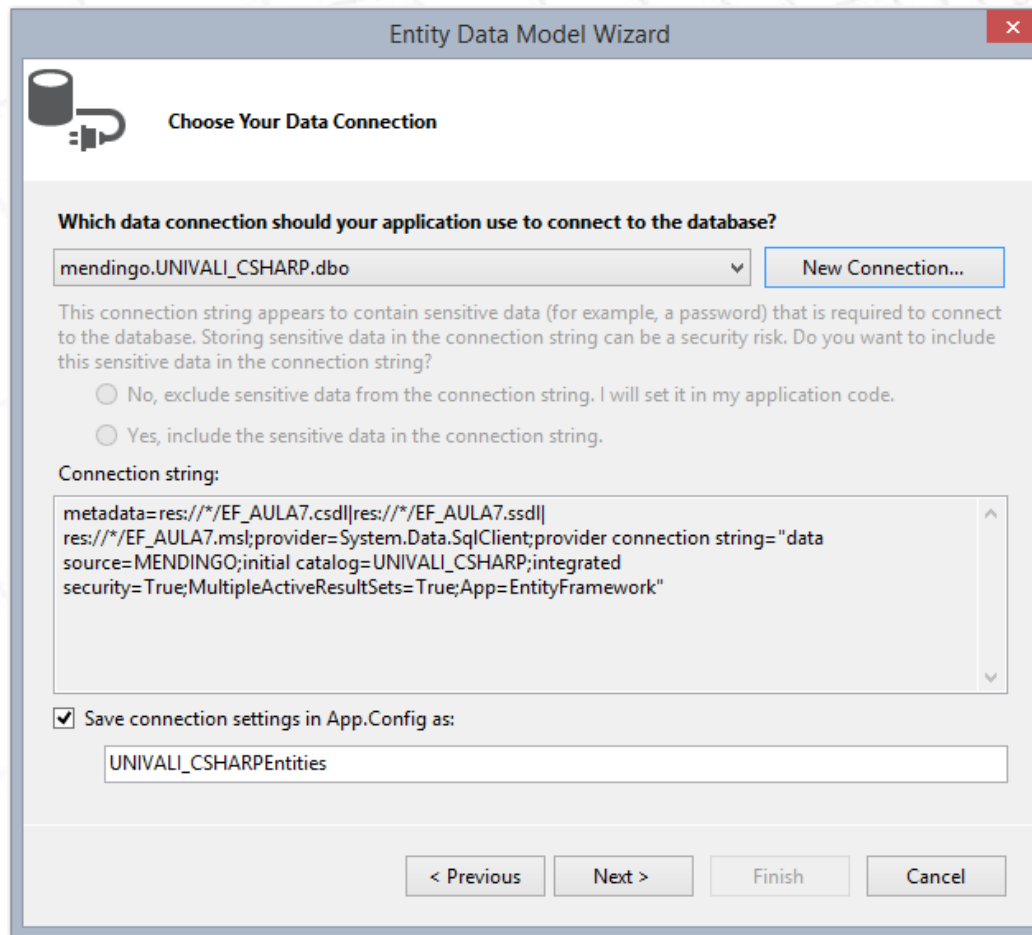
Banco de Dados e ORM

- Selecionar o banco



Banco de Dados e ORM

– Selecionar o banco



The image shows a screenshot of the 'Entity Data Model Wizard' dialog box. The title bar reads 'Entity Data Model Wizard'. The main heading is 'Choose Your Data Connection'. Below this, a question asks: 'Which data connection should your application use to connect to the database?'. A dropdown menu shows 'mendingo.UNIVALI_CSHARP.dbo' and a 'New Connection...' button is to its right. A warning message states: 'This connection string appears to contain sensitive data (for example, a password) that is required to connect to the database. Storing sensitive data in the connection string can be a security risk. Do you want to include this sensitive data in the connection string?'. Two radio buttons are provided: 'No, exclude sensitive data from the connection string. I will set it in my application code.' and 'Yes, include the sensitive data in the connection string.'. Below this, the 'Connection string:' is displayed in a text box with the following content: 'metadata=res://*/EF_AULA7.csdl|res://*/EF_AULA7.ssdl|res://*/EF_AULA7.msl;provider=System.Data.SqlClient;provider connection string="data source=MENDINGO;initial catalog=UNIVALI_CSHARP;integrated security=True;MultipleActiveResultSets=True;App=EntityFramework"'. At the bottom, there is a checkbox 'Save connection settings in App.Config as:' which is checked, followed by a text box containing 'UNIVALI_CSHARPEntities'. Navigation buttons at the bottom include '< Previous', 'Next >', 'Finish', and 'Cancel'.

Entity Data Model Wizard

Choose Your Data Connection

Which data connection should your application use to connect to the database?

mendingo.UNIVALI_CSHARP.dbo New Connection...

This connection string appears to contain sensitive data (for example, a password) that is required to connect to the database. Storing sensitive data in the connection string can be a security risk. Do you want to include this sensitive data in the connection string?

☐ No, exclude sensitive data from the connection string. I will set it in my application code.

☐ Yes, include the sensitive data in the connection string.

Connection string:

metadata=res://*/EF_AULA7.csdl|res://*/EF_AULA7.ssdl|
res://*/EF_AULA7.msl;provider=System.Data.SqlClient;provider connection string="data
source=MENDINGO;initial catalog=UNIVALI_CSHARP;integrated
security=True;MultipleActiveResultSets=True;App=EntityFramework"

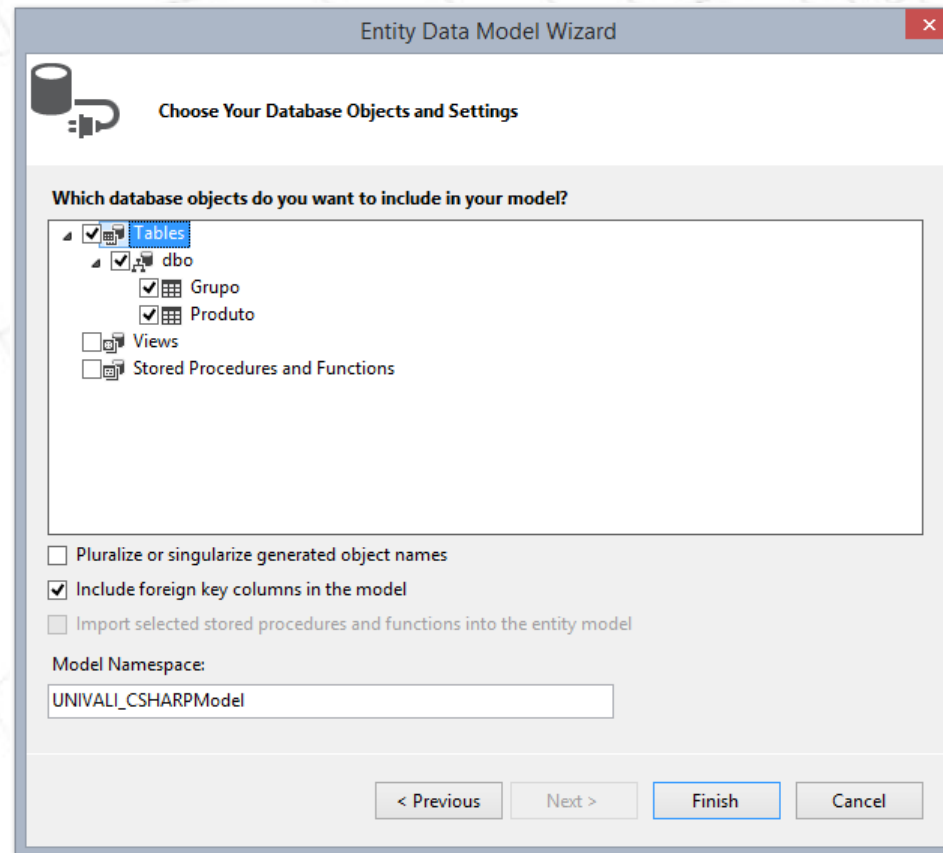
☒ Save connection settings in App.Config as:

UNIVALI_CSHARPEntities

< Previous Next > Finish Cancel

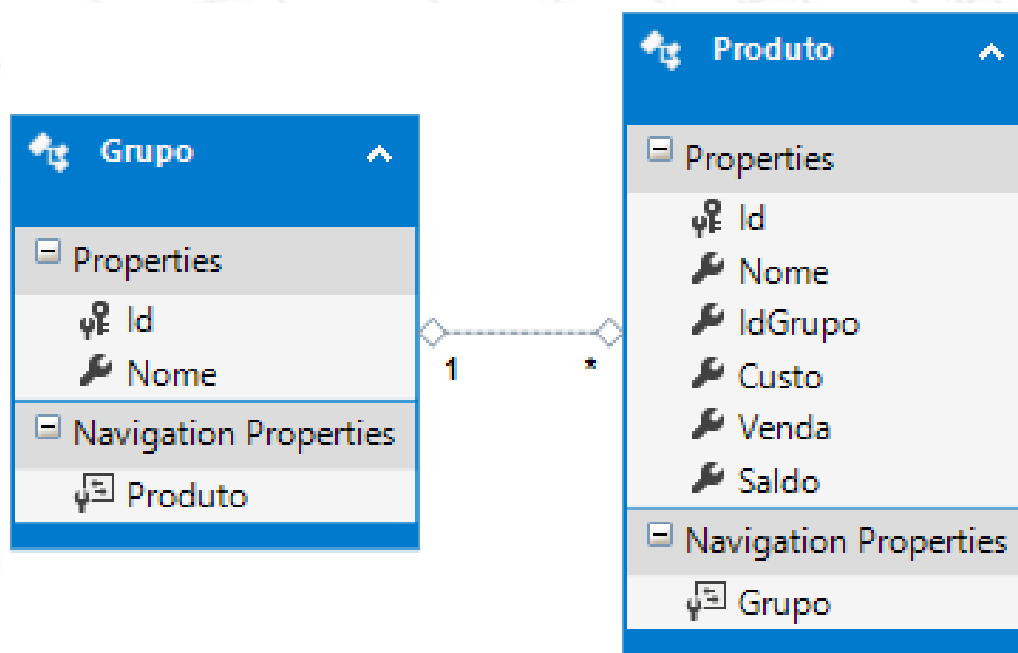
Banco de Dados e ORM

- Selecionar o banco



Banco de Dados e ORM

— Entidades e Associações

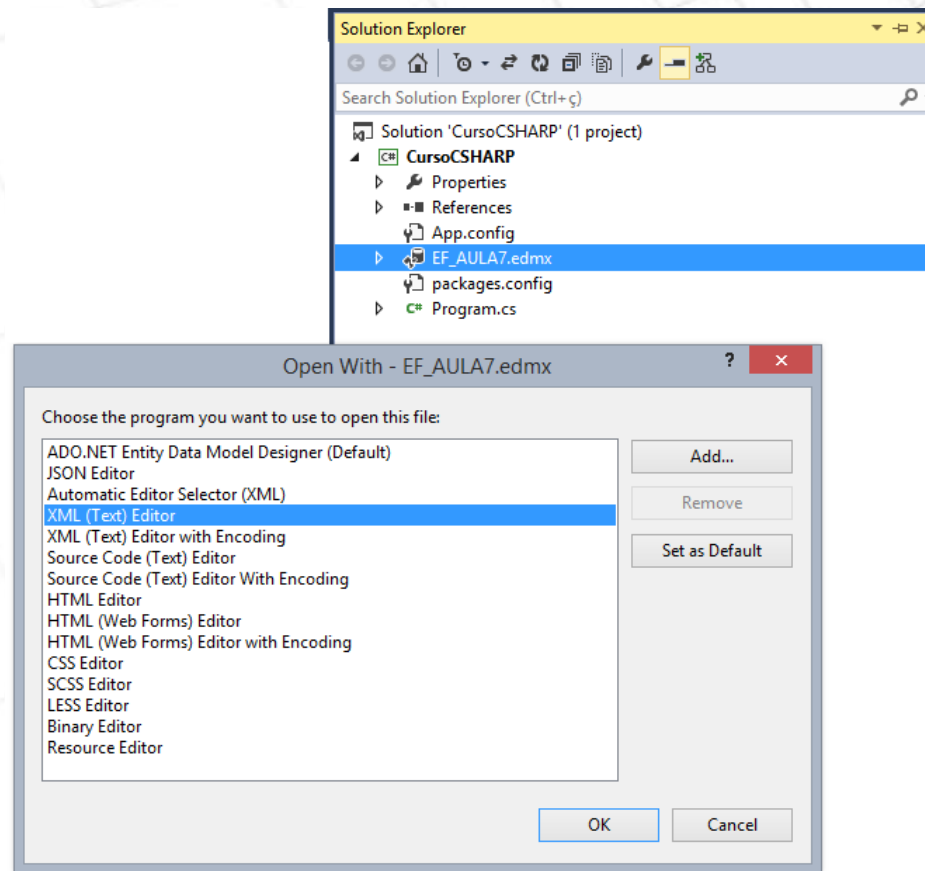


Analizando o EDMx

- O que é o EDMx ?
- Como o banco de dados foi mapeado
 - Tabelas = Entidades
 - Relacionamentos = Associações

Banco de Dados e ORM

- Entendo o arquivo EDMx



Banco de Dados e ORM

- O arquivo EDMx contém o mapeamento entre as entidades (classes) do nosso modelo visual e as tabelas do banco de dados, bem como o tipo do banco de dados. Este arquivo está dividido em 3 partes:
- **SSDL** – contém as informações do banco de dados, como tipo do banco, nome das tabelas, nomes e tipos dos campos;
- **CSDL** – contém as informações do modelo, ou seja, o que fizemos no designer do Entity Framework, como entidades, propriedades e associações.
- **C-S Mapping** – finalmente esta parte do arquivo faz a relação entre o nosso modelo e o banco de dados.

LINQ

- *Language Integrated Query*
 - Linguagem de consulta integrada ao .NET
- Execução das consultas utilizando expressões Lambda
 - Linguagens VB e C#
 - Versão 3.0 do .NET
- Conversão da query LINQ em SQL

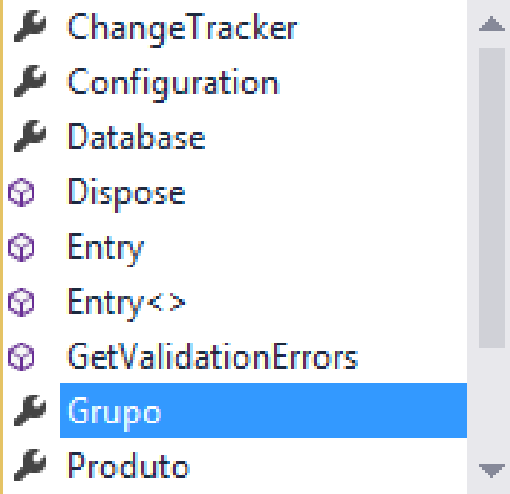
LINQ

- Processo de consulta mais produtivo
 - Validação em tempo de compilação da estrutura das consultas;
 - Temos a vantagem de o IntelliSense nos auxiliar na codificação de nossas consultas;
 - Estamos utilizando uma linguagem *tipada* para execução de nossas consultas;

Executando consultas com LINQ

- Para que as consultas sejam executadas é preciso que seja criada uma instância de um *ObjectContext*.

```
using (UNIVALI_CSHARPEntities context = new UNIVALI_CSHARPEntities()){  
    context.  
}
```



LINQ

- Uma query sempre inicia com o operador **from**, e geralmente termina com o operador **select**
- O operador **from** declara uma variável de interação - (c como por exemplo)

```
using (UNIVALI_CSHARPEntities context = new UNIVALI_CSHARPEntities())
{
    var query = from c in context.Produto select c;
    foreach (var item in query)
    {
        Console.WriteLine("{0} - {1}", item.Id, item.Nome);
    }
    Console.Read();
}
```

Consultando todos os registros de uma tabela

LINQ

```
using (UNIVALI_CSHARPEntities context = new UNIVALI_CSHARPEntities())
{
    var query = from c in context.Produto select c;
    foreach (var item in query)
    {
        Console.WriteLine("{0} - {1}", item.Id, item.Nome);
    }
    Console.Read();
}
```

Lambda

```
using (UNIVALI_CSHARPEntities context = new UNIVALI_CSHARPEntities())
{
    var query = context.Produto;

    foreach (var item in query)
    {
        Console.WriteLine("{0} - {1}", item.Id, item.Nome);
    }
    Console.Read();
}
```

Consultando os cinco primeiros registros de uma tabela

LINQ

```
using (UNIVALI_CSHPREntities context = new UNIVALI_CSHPREntities())
{
    var query = (from p in context.Produto select p).Take(5);
    foreach (var item in query)
    {
        Console.WriteLine("{0} - {1}", item.Id, item.Nome);
    }
    Console.Read();
}
```

Lambda

```
using (UNIVALI_CSHPREntities context = new UNIVALI_CSHPREntities())
{
    var query = context.Produto.Take(5);
    foreach (var item in query)
    {
        Console.WriteLine("{0} - {1}", item.Id, item.Nome);
    }
    Console.Read();
}
```

Consultando registros através de campos texto

LINQ

```
using (UNIVALI_CSHARPEntities context = new UNIVALI_CSHARPEntities())
{
    var query = from p in context.Produto where p.Nome == "Nokia Lumia 800" select p;
    foreach (var item in query)
    {
        Console.WriteLine("{0} - {1}", item.Id, item.Nome);
    }
    Console.Read();
}
```

Lambda

```
using (UNIVALI_CSHARPEntities context = new UNIVALI_CSHARPEntities())
{
    var query = context.Produto.Where(p => p.Nome == "Nokia Lumia 800");
    foreach (var item in query)
    {
        Console.WriteLine("{0} - {1}", item.Id, item.Nome);
    }
    Console.Read();
}
```

Consultando registros através de campos texto

LINQ

```
using (UNIVALI_CSHARPEntities context = new UNIVALI_CSHARPEntities())
{
    var query = from p in context.Produto where p.Nome.Contains(" NOKIA") select p;
    foreach (var item in query)
    {
        Console.WriteLine("{0} - {1}", item.Id, item.Nome);
    }
    Console.Read();
}
```

Lambda

```
using (UNIVALI_CSHARPEntities context = new UNIVALI_CSHARPEntities())
{
    var query = context.Produto.Where(p => p.Nome.Contains("NOKIA"));
    foreach (var item in query)
    {
        Console.WriteLine("{0} - {1}", item.Id, item.Nome);
    }
    Console.Read();
}
```

Consultando registros através de operadores relacionais

LINQ

```
using (UNIVALI_CSHARPEntities context = new UNIVALI_CSHARPEntities())
{
    var query = from p in context.Produto where p.Custo < 1000 && p.Nome.Contains("Tablet") select p;
    foreach (var item in query)
    {
        Console.WriteLine("{0} - {1}", item.Id, item.Nome);
    }
    Console.Read();
}
```

Lambda

```
using (UNIVALI_CSHARPEntities context = new UNIVALI_CSHARPEntities())
{
    var query = context.Produto.Where(p => p.Custo < 1000 && p.Nome.Contains("TABLET"));
    foreach (var item in query)
    {
        Console.WriteLine("{0} - {1}", item.Id, item.Nome);
    }
    Console.Read();
}
```

Utilizando dados de um relacionamento para filtrar registros

LINQ

```
using (UNIVALI_CSHARPEntities context = new UNIVALI_CSHARPEntities())
{
    var query = from p in context.Produto where p.Grupo.Nome == "Games" select p;
    foreach (var item in query)
    {
        Console.WriteLine("{0} - {1}", item.Id, item.Nome);
    }
    Console.Read();
}
```

Lambda

```
using (UNIVALI_CSHARPEntities context = new UNIVALI_CSHARPEntities())
{
    var query = context.Produto.Where(p => p.Grupo.Nome == "Games");
    foreach (var item in query)
    {
        Console.WriteLine("{0} - {1}", item.Id, item.Nome);
    }
    Console.Read();
}
```


Ordenando registros

LINQ

```
using (UNIVALI_CSHARPEntities context = new UNIVALI_CSHARPEntities())
{
    var query = from p in context.Produto orderby p.Nome select p;
    foreach (var item in query)
    {
        Console.WriteLine("{0} - {1}", item.Id, item.Nome);
    }
    Console.Read();
}
```

Lambda

```
using (UNIVALI_CSHARPEntities context = new UNIVALI_CSHARPEntities())
{
    var query = context.Produto.OrderBy(p => p.Nome);
    foreach (var item in query)
    {
        Console.WriteLine("{0} - {1}", item.Id, item.Nome);
    }
    Console.Read();
}
```

Ordenando registros

LINQ

```
using (UNIVALI_CSHARPEntities context = new UNIVALI_CSHARPEntities())
{
    var query = from p in context.Produto orderby p.Custo descending select p;
    foreach (var item in query)
    {
        Console.WriteLine("{0} - {1} - {2:C}", item.Id, item.Nome, item.Custo);
    }
    Console.Read();
}
```

Lambda

```
using (UNIVALI_CSHARPEntities context = new UNIVALI_CSHARPEntities())
{
    var query = context.Produto.OrderByDescending(p => p.Custo);
    foreach (var item in query)
    {
        Console.WriteLine("{0} - {1} - {2:C}", item.Id, item.Nome, item.Custo);
    }
    Console.Read();
}
```