Dell IT Academy

Exercício completo

Projeto C# com Entity Framework Core e Consultas LINQ

Neste exercício, será criado um case completo com a aplicação do Entity Frameword com a linguagem LINQ. Desde a criação e um projeto C# com o Entity Framework Core, sua configuração inicial até a realização de consultas usando LINQ.

1. Configurando o Projeto e Instalando Dependências

Abra um terminal e execute o comando para criar um novo projeto .NET Console:

```
dotnet new console -o LojaApp
```

Navegue até o diretório do projeto:

```
cd LojaApp
```

Instalar o Entity Framework Core e o provedor de banco de dados SQL Server executando:

```
dotnet add package Microsoft.EntityFrameworkCore
dotnet add package Microsoft.EntityFrameworkCore.SqlServer
dotnet add package Microsoft.EntityFrameworkCore.Design
```

Instalar a Ferramenta dotnet-ef (Opcional): a ferramenta dotnet-ef facilita a criação e aplicação de migrações:

```
dotnet tool install --global dotnet-ef
```

2. Criando as Classes do Modelo

No VS Code, dentro do diretório do projeto, crie um arquivo c# para cada uma das classes: Categoria, Produto e Venda para representar as tabelas do banco de dados.

Classe Categoria

```
public class Categoria
{
    public int Id { get; set; }
    public string Nome { get; set; }

    public List<Produto> Produtos { get; set; } = new List<Produto>();
}
```

Classe Produto

```
public class Produto
{
    public int Id { get; set; }
    public string Nome { get; set; }
    public decimal Preco { get; set; }
    public DateTime DataAdicionado { get; set; }

    public int CategoriaId { get; set; }
    public Categoria Categoria { get; set; }

    public List<Venda> Vendas { get; set; } = new List<Venda>();
}
```

Classe Venda

```
public class Venda
{
    public int Id { get; set; }
    public int ProdutoId { get; set; }
    public Produto Produto { get; set; }
    public int Quantidade { get; set; }
    public DateTime DataVenda { get; set; }
}
```

3. Configurando o Contexto do Banco de Dados

Crie um Database no SQLServer.

```
CREATE DATABASE MeuProjetoVendas;
```

Crie uma classe de contexto para gerenciar a conexão com o banco de dados e acessar as tabelas.

Classe LojaDbContext

```
using Microsoft.EntityFrameworkCore;

public class LojaDbContext : DbContext
{
    public DbSet<Categoria> Categorias { get; set; }
    public DbSet<Produto> Produtos { get; set; }
    public DbSet<Venda> Vendas { get; set; }

    protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)
    {

optionsBuilder.UseSqlServer("Server=localhost;Database=MeuProjetoVendas;Trusted_Connection=True;"
);
    }
}
```

*Substitua "**Server=localhost;Database=MeuProjetoVendas;Trusted_Connection=True**;" pela string de conexão apropriada ao seu ambiente.

Compile o seu programa!

4. Configurando o Banco de Dados com Migrations

Criar a Primeira Migração:

```
dotnet ef migrations add InitialCreate
```

Aplicar a Migração para Criar o Banco de Dados:

```
dotnet ef database update
```

5. Inserindo Dados nas Tabelas

Na classe Program, crie um método para adicionar dados de exemplo ao banco de dados. Seguem os dados para este exercício.

```
using System;

public class Program
{
    public static void Main()
    {
        using (var context = new LojaDbContext())
        {
            var eletronicos = new Categoria { Nome = "Eletrônicos" };
            var roupas = new Categoria { Nome = "Roupas" };
            var alimentos = new Categoria { Nome = "Alimentos" };

            context.Categorias.AddRange(eletronicos, roupas, alimentos);

            var smartphone = new Produto { Nome = "Smartphone", Preco = 1200.00m, DataAdicionado = DateTime.Now, Categoria = eletronicos };
            var laptop = new Produto { Nome = "Laptop", Preco = 3000.00m, DataAdicionado = DateTime.Now, Categoria = eletronicos };
            var camisa = new Produto { Nome = "Camisa", Preco = 80.00m, DataAdicionado = DateTime.Now, Categoria = roupas };
            var arroz = new Produto { Nome = "Arroz", Preco = 20.00m, DataAdicionado = DateTime.Now, Categoria = alimentos };

            context.Produtos.AddRange(smartphone, laptop, camisa, arroz);

            var venda1 = new Venda { Produto = smartphone, Quantidade = 2, DataVenda = DateTime.Now };
            var venda2 = new Venda { Produto = laptop, Quantidade = 1, DataVenda = DateTime.Now };
            var venda3 = new Venda { Produto = camisa, Quantidade = 3, DataVenda = DateTime.Now };

            context.Vendas.AddRange(venda1, venda2, venda3);

            context.SaveChanges();
        }

        Console.WriteLine("Dados inseridos com sucesso!");
    }
}
```