06

Corrigindo problemas e aplicando as migrações no banco

Transcrição

No vídeo anterior vimos como trabalhar com a ferramenta de migração do Entity. Criamos uma *Migration* representando a evolução da classe `Produto`. Porém quando tentamos executar a migração, recebemos um erro informando que a tabela já existia.

Esse problema foi porque colocamos a criação da tabela e a evolução na mesma execução. Para resolvermos, nós teríamos que separar a criação da evolução da tabela.

Começaremos excluindo a tabela `__EFMigrationsHistory` do banco, e também a pasta `Migrations` do projeto. Além disso, voltaremos a classe produto para a sua forma inicial:

```
namespace Alura.Loja.Testes.ConsoleApp
{
    internal class Produto
    {
        public int Id { get; internal set; }
        public string Nome { get; internal set; }
        public string Categoria { get; internal set; }
        public double Preco { get; internal set; }

        public override string ToString()
        {
            return $"Produto: {this.Id}, {this.Nome}, {this.Categoria}, {this.Preco}";
        }
    }
}
```

Executaremos o comando:

PM> Add-Migration Inicial

Agora que já criamos a *Migration* inicial, voltaremos a colocar a evolução da classe, que ficará da seguinte maneira:

```
namespace Alura.Loja.Testes.ConsoleApp
{
    internal class Produto
    {
        public int Id { get; internal set; }
        public string Nome { get; internal set; }
        public string Categoria { get; internal set; }
        public double PrecoUnitario { get; internal set; }
        public string Unidade { get; set; };

        public override string ToString()
        {
            return $"Produto {this.Id}, {this.Nome}, {this.Categoria}, {this.PrecoUnitario}";
        }
    }
```

```
}
```

Criaremos um nova migração que chamaremos de **Unidade**. O comando será:

```
Add-Migration Unidade
```

Acessando a classe de migração `Unidade`, veremos que ela está bem diferente, refletindo apenas as informações de evolução da tabela. Dentro do método `Up()` temos a instrução de renomear a coluna `Preco` para `PrecoUnitario`, e adicionar uma nova coluna `Unidade`. Já no método `Down`, temos a instrução de deletar a coluna `Unidade` e renomear `PrecoUnitario` para `Preco`.

Tentando executar o comando `Update-Database` receberemos o mesmo erro informando que a tabela já foi criada. A função da tabela `__EFMigrationsHistory` é apenas registrar quais migrações foram executadas no banco de dados. Ao acessá-la, veremos que nenhuma migração foi executada.

Levando isso em consideração, ao tentarmos efetuar a migração, ele executará todas as migrações que estão na pasta `Migrations`, inclusive a migração **Inicial**.

Fingiremos para o banco que a migração **Inicial** já foi executada. Entraremos da classe de migração `Inicial` e comentaremos todo o conteúdo do método `Up()`, em seguida executaremos `Update-Database Inicial`. Isso executará apenas a migração com o nome *Inicial* e irá criar uma linha na tabela de histórico de migração.

Agora podemos executar tirar os comentários da classe `Inicial` e executar o comando `Update-Database`, dessa vez sem passarmos o nome da classe. A sincronização foi efetuada com sucesso.

Acessando o banco de dados, veremos que a coluna `Preco` foi renomeada para `PrecoUnitario` e a coluna `Unidade` foi criada.