

O que aprendemos?

Transcrição

Recapitularemos o conteúdo apresentado até então. Descobrimos algumas convenções para a definição do tipo e tamanho dos dados para `SqlServer`. Descobrimos essas informações com base na classe `SqlServerTypeMapper`.

CLR	SQL Server
int	int
long	bigint
DateTime	datetime2
Guid	uniqueidentifier
bool	bit
byte	tinyint

CLR	SQL Server
double	float
DateTimeOffset	datetimeoffset
short	smallint
float	real
decimal	decimal(18, 2)
TimeSpan	time

* informações obtidas da classe `SqlServerTypeMapper`

Se não tivéssemos configurado as propriedades das classes, regras implícitas teriam sido utilizadas e o banco de dados não funcionaria da forma desejada. Para realizar as configurações, utilizamos o argumento `TypeName` na anotação `Column`. Aprendemos, também, a configuração da nulidade das colunas. A convenção para a nulidade das colunas se dá pelo tipo de SLR. Se o SLR aceitar `null`, o Entity irá atribuir esse valor à coluna. Os tipos primitivos da SLR são sempre colocados como `not null`. Já tipos complexos - como a `string` - serão considerados nulos. Para quebrarmos essa convenção, utilizamos a anotação `Required`.

Percebemos que o código da classe `Ator` estava misturando informações do mundo relacional com o mundo orientado pelo objeto. Para resolver esse problema, movemos todas as informações de configuração para a classe contexto, dentro do método `OnModelCreating`.

Com isso, iremos analisar o fluxo de mapeamento que o Entity utiliza. Primeiramente, o Entity vai na classe de contexto e verifica a classe que precisa ser mapeada, essa informação está determinada na propriedade `DbSet`. A partir disso, o *software* irá escanear a classe a ser mapeada para identificar se existe alguma anotação. Nas anotações é que as convenções automáticas do Entity serão quebradas. Se não houver nenhuma anotação, o Entity irá para o método `OnModelCreating` e verificará se há alguma regra explícita para depois fechar seu modelo de mapeamento.

