01

Colunas não necessárias no negócio

Transcrição

Não mapeamos ainda a coluna `last_update`.

```
CREATE TABLE [actor] (
    actor_id int NOT NULL IDENTITY,
    first_name varchar(45) NOT NULL,
    last_name varchar(45) NOT NULL,
    last_update DATETIME NOT NULL,
    CONSTRAINT PK_ actor PRIMARY KEY  NONCLUSTERED (actor_id)
)
```

A coluna `last_update` não contém nenhuma informação para o valor de negócio. A coluna apenas informa qual foi a última vez que um registro foi modificado ou atualizado. Veremos o que fazer quando temos colunas no banco de dados que não serão refletidas na camada de negócios. O Entity denomina esse tipo de coluna de **shadow properties**.

Ao tentarmos configurar uma shadow property, veremos que ela não existe na classe `Ator`.

```
namespace Alura.Filmes.App.Negocio
{

    public class Ator
    {

        public int Id { get; set; }
        public string PrimeiroNome { get; set; }
        public strig UltimoNome { get; set; }

        public override string ToString()
        {
            return $"Ator ({Id}): {PrimeiroNome} {UltimoNome}";
        }
    }
}
```

A única opção é trabalhar no `OnModelCreating` utilizando o `modelBuilder`. Nesse caso, ao escrevermos a propriedade `last_update`, não podemos utilizar a expressão lambda, pois ela não está definida na classe. Teremos de dizer que trata-se de uma `DateTime`.

```
namespace Alura.Filmes.App.Dados
{
    public class AluraFilmesContexto : DbContext
    {
        public DbSet<Ator> Atores { get; set; }

        protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)
        {
```

```
            optionsBuilder.UseSqlServer("Server=(localdb)mssqllocaldb;Database=AluraFilmes;Trusted_(
        }

        protected override void OnModelCreating(ModelBuilder modelBuilder)
        {
            modelBuilder.Entity<Ator>()
                .ToTable("actor");

            modelBuilder.Entity<Ator>()
                .Property(a => a.Id)
                .HasColumnName("actor_id");

            modelBuilder.Entity<Ator>()
                .Property(a => a.PrimieroNome)
                .HasCoumnName("first_name")
                .HasColumnType("varchar(45)")
                .IsRequired();

            modelBuilder.Entity<Ator>()
                .Property(a => a.UltimoNome)
                .HasColumnType("varchar(45)")
                .IsRequired();

            modelBuilder.Entity<Ator>()
            .Property<DateTime>("last_update");
        }
    }
}
```

Temos uma questão: se `DateTime` aceita o valor `null`. Testaremos isso criando uma variável do tipo `DateTime` chamada `aniversario`. Percebemos que `DateTime` não aceita o valor `null` por tratar-se de uma estrutura. Ao olharmos para o nosso script, percebemos que `last_update` não pode ter valor nulo.

```
CREATE TABLE actor (
    actor_id int NOT NULL IDENTITY,
    first_name varchar(45) NOT NULL,
    last_name varchar(45) NOT NULL,
    last_update DATETIME NOT NULL,
    CONSTRAINT PK_ actor PRIMARY KEY  NONCLUSTERED (actor_id)
)
```

Iremos gerar novamente um novo script para averiguar se as informações estão coerentes com o original. Novamente, fazemos o procedimento padrão para geração do novo script (abriremos o console, removeremos a migração anterior, adicionaremos a nova e geramos o script)

```
CREATE TABLE [actor] (
    [actor_id] int NOT NULL IDENTITY,
    [first_name] varchar(45) NOT NULL,
    [last_name varchar(45) NOT NULL,
    [last_update] datetime2 NOT NULL,
    CONSTRAINT PK_ actor PRIMARY KEY  NONCLUSTERED (actor_id)
);
```

Há uma discrepância com relação ao script original devido ao `datetime2`. Para resolvermos essa diferença, iremos novamente à classe contexto e definiremos o tipo da propriedade como `datetime`.

```
namespace Alura.Filmes.App.Dados
{
    public class AluraFilmesContexto : DbContext
    {
        public DbSet<Ator> Atores { get; set; }

        protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)
        {
            optionsBuilder.UseSqlServer("Server=(localdb)mssqllocaldb;Database=AluraFilmes;Trusted_(
        }

        protected override void OnModelCreating(ModelBuilder modelBuilder)
        {
            modelBuilder.Entity<Ator>()
                .ToTable("actor");

            modelBuilder.Entity<Ator>()
                .Property(a => a.Id)
                .HasColumnName("actor_id");

            modelBuilder.Entity<Ator>()
                .Property(a => a.PrimieroNome)
                .HasCoumnName("first_name")
                .HasColumnType("varchar(45)")
                .IsRequired();

            modelBuilder.Entity<Ator>()
                .Property(a => a.UltimoNome)
                .HasColumnType("varchar(45)")
                .IsRequired();

            modelBuilder.Entity<Ator>()
            .Property<DateTime>("last_update");
            .HasColumnType ("datetime")
        }
    }
}
```

Iremos gerar novamente o script e compara-lo ao original.

```
CREATE TABLE [actor] (
    [actor_id] int NOT NULL IDENTITY,
    [first_name] varchar(45) NOT NULL,
    [last_name varchar(45) NOT NULL,
    [last_update] datetime NOT NULL,
    CONSTRAINT PK_ actor PRIMARY KEY  NONCLUSTERED (actor_id)
);
```

Temos um script mais parecido com o original.