07

Para saber mais: sincronizando o banco com sua própria aplicação

Depois que sua aplicação estiver madura o bastante para ser promovida, surge a questão: como atualizar o banco de dados daquele ambiente específico? Vimos que em organizações com políticas de acesso mais restritas a ambientes críticos, a solução é gerar um arquivo com o script das migrações e entregar esse arquivo à equipe responsável. Essa tarefa é realizada com o comando `Script-Migration`.

Além disso, também é possível fazer que sua própria aplicação cuide da migração das versões. Ou seja, podemos escrever código em nossa aplicação para que o banco de dados seja sincronizado. Isso é feito através do método de extensão `Migrate`, que está acessível na propriedade `Database` da classe `DbContext`. Essa propriedade representa a instância do banco de dados apontado pelo contexto Entity específico de sua aplicação (no nosso exemplo, `LojaContext`), e expõe métodos que permitem gerenciar o banco apontado pelo contexto, como por exemplo sua criação, exclusão e validação de existência.

O método `Migrate` só pode ser usado em bancos de dados relacionais e fica disponível no pacote `Microsoft.EntityFrameworkCore.Relational`.

Assim, para garantir que todas as migrações estarão aplicadas no banco de dados, podemos escrever:

```
using(var contexto = new LojaContext())
{
  contexto.Database.Migrate();
}
```

Você precisa garantir que esse código seja executado antes de qualquer acesso aos objetos gerenciados pelo contexto. Isso vai depender do tipo de aplicação que será implementada.