05

Mapeando relacionamentos Muitos-para-Muitos

Transcrição

Conhecendo os relacionamentos

No treinamento usaremos o projeto de um pequeno catálogo de produtos de diferentes lojas, onde os utensílios poderão ser filtrados pelo nome, loja e categoria. Se você já rodou o projeto certamente já observou os três filtros que estão localizados no lado direito da tela. Cada produto é vendido por uma loja e, portanto, possui várias categorias. Por exemplo, conforme podemos observar na classe produto:

```
@Entity
public class Produto {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Integer id;

    private Loja loja;

    private List<Categoria> categorias = new ArrayList<>();

    // código omitido
}
```

Relacionamento muitos-para-muitos

Como vários produtos podem ser vendidos por uma mesma loja, podemos representar esse relacionamento usando `@OneToMany`. Mas, precisamos informar a JPA que um produto contêm diversas categorias relacionadas a si. Podemos pensar nesse relacionamento como do tipo **um para muitos**, o problema dessa abordagem é que cada categoria teria que estar relacionada a **apenas um** produto, entretanto, provavelmente, teremos dois produtos associados a, por exemplo, *tecnologia*. Nesse caso, pode acontecer de termos várias categorias com nomes repetidos cadastrados. Sendo assim, é preciso informar que um produto pode ter várias categorias associadas mas, ao mesmo tempo, essas mesmas categorias podem estar associadas a outros produtos, sem, entretanto, duplicar esses cadastros. Isso caracteriza um relacionamento do tipo **muitos para muitos**. Para marcar a associação como **muitos para muitos** vamos usar a anotação `@ManyToMany`:

```
@Entity
public class Produto {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Integer id;

    private Loja loja;

    @ManyToMany
    private List<Categoria> categorias = new ArrayList<>();
```

```
    // código omitido
}
```

Representando esse relacionamento no banco de dados

Pensando no banco de dados: não é possível representar esse tipo de relacionamento utilizando apenas duas tabelas.

Precisamos dizer que um mesmo produto está relacionado com várias categorias e vice-versa, por hora, temos apenas um cadastro de produto e um de categoria . Para representar isso, normalmente, é utilizada uma tabela que contêm apenas duas colunas:

Coluna 1 - o id de uma das pontas da relação, no nosso caso o produto; Coluna 2- o id para a outra ponta da relação, no nosso caso a categoria;

Essa tabela é chamada de **Tabela de Relacionamento** ou tabela associativa. O único objetivo é relacionar registros de uma tabela com os registros da outra. Essa é a tabela que será gerada pelo Hibernate para representar nosso mapeamento de "muitos para muitos". Pela convenção, a tabela será chamada de **Produto_Categoria**, ou seja, seguindo a fórmula: "nome da classe que contém o relacionamento" + "o nome da classe que foi associada". É interessante notar que, mais uma vez, em nenhum momento vamos precisar fazer referência a essas tabelas que foram criadas. Sempre trabalharemos baseado nos nossos objetos!