06

Utilizando Named Queries

Transcrição

[00:00] Olá, pessoal. Nesse vídeo vamos aprender outra maneira de fazer consultas com a JPQL. Estou na classe "ProdutoDao". Lembra aqueles nossos métodos de consulta? Tínhamos alguns aqui, por exemplo, esse, de buscar produtos pelo nome da categoria. Nós acabamos criando essa *query*: `"SELECT p FROM Produto p WHERE p.categoria.nome = :nome;"` , e o parâmetro `:nome` .

[00:25] Está tudo ok, porém, tem uma outra maneira de fazer consulta, que talvez você encontre em alguns projetos, utilizando um recurso da JPA chamado *named query*, *query* nomeada. A ideia nada mais é do que você extrair essas consultas da classe Dao e deixá-las na própria entidade. Então as consultas ficam juntas com a entidade.

[00:48] Tem gente que prefere fazer dessa maneira, ao invés de deixar a consulta aqui, todas as consultas na classe Dao, deixar as consultas na entidade. Vamos ver como vai funcionar. A ideia é, já que eu estou aqui na classe "ProdutoDao", a entidade é a entidade "Produto". Vou abrir a minha entidade "Produto".

[01:06] Em cima da entidade, tem lá o `@Entity` , o `@Table` , você pode adicionar mais uma anotação, que é o `@NamedQuery` , que vem do pacote "javax.persistence". Como funciona esse *named query*? Você dá um nome para a sua *query*, é tipo um apelido para a sua *query*. Deixa eu voltar para o "ProdutoDao", aqui era `buscarPorNomeDaCategoria".

[01:31] Sei lá, vamos chamar aqui de `@NamedQuery(name = "produtosPorCategoria", )`, ou algo do gênero. E tem outro parâmetro, que é a *query* em si, qual é a *query*? JPQL. Então eu vou extrair da minha classe "ProdutoDao" esse `SELECT` da `buscaPorNomeDaCategoria`. Eu vou extrair ele daqui, "Ctrl + X".

[01:52] E jogar para cá, no `@NamedQuery(name = "produtosPorCategoria", query = "SELECT p FROM PRoduto p WHERE p.categoria.nome = :nome)`. Olha que interessante, deixa eu só dar um "Enter" para quebrar a linha.

[02:00] Existe esse recurso, `@NamedQuery` na própria entidade. Em cima da entidade, em cima da classe, você pode ter essas anotações `@NamedQuery`, então a consulta, as *queries*, elas ficam junto com a entidade, fica mais simples de encontrar as consultas de determinada entidade. Embora, uma desvantagem, é que fica meio estranho.

[02:21] Eu, particularmente, não gosto desta abordagem, porque na entidade, você começa a ter *selects*, você começa a ter JPQL, e para mim, isso, eu particularmente não gosto muito. Prefiro isolar essas consultas e deixar dentro da classe Dao. Mas tem gente que prefere agrupar a consulta na própria entidade, então é capaz de você esbarrar em um projeto um dia e encontrar essas anotações *named query*.

[02:45] E agora, como eu faço a consulta? A consulta, ela está aqui, mas, na verdade, o que está aqui na entidade é só a definição da consulta. Se você quiser rodar essa consulta, como ficaria isso na nossa classe Dao? No "ProdutoDao" você não tem mais essa *string* JPQL.

[03:02] Para você rodar essa consulta, você vai chamar o *entity manager*, só que ao invés de você usar o `createQuery`, você vai chamar o método `createNamedQuery`. O `createNamedQuery` recebe dois parâmetros. O primeiro é o nome da *query*, que é aquele *name* que nós colocamos, `produtosPorCategoria`.

[03:21] Você coloca `em.createNameQuery("produtosPorCategoria", Produto.class`. Só que assim não fica muito claro de onde está vindo essa *query*, de qual entidade. Então, no geral, o pessoal costuma fazer o nome da entidade, ponto e um apelido para a *query*, então a *named query*, eles costumam nomear assim: `("Produto.produtoPorCategoria",`.

[03:42] Vamos à entidade "Produto" e vamos renomear esse *name*. `@NamedQuery(name = "Produto.produtosPorCategoria")`. Só para deixar explícito que essa *named query* vem da entidade "Produto". Vamos voltar para o "ProdutoDao". O segundo parâmetro que o método `createNamedQuery` recebe é qual é a entidade resultante desta consulta, da mesma maneira que o `createQuery`.

[04:07] Então o resto é igual. Perceba que essa nossa consulta, ela tem um parâmetro, `:nome`. Ela foi preenchida na classe Dao, na hora de criar a *named query*, você *seta* o parâmetro da mesma maneira. Essa consulta, nós estávamos chamando ela na nossa classe "CadastroDeProduto". Vamos rodar e ver se tudo continua funcionando. Vamos ver o console.

[00:04:32] Fez o "select", normal, encontrou, tudo certo, não deu erro. Não muda nada, tudo continuou funcionando conforme o esperado. Esse era mais um recurso da JPA que eu queria mostrar para vocês, a *named query*. Você pode ter quantas *named queries* você quiser, pode colocar várias anotações `@NamedQuery` em cima da classe, que funciona normalmente, cada uma tem que ter um nome, que deve ser único.

[04:56] Não pode repetir o nome da *named query* na aplicação, em nenhuma outra entidade. Você chama essa consulta via *entity manager*, `em.createNamedQuery` e passa o *name* da *query*. Então aqui, na classe Dao, por exemplo, eu não teria a JPQL, você só passa a referência, o nome dessa *named query*.

[05:14] Essa é outra abordagem. Eu, particularmente, não gosto muito dessa abordagem, mas tem gente que gosta de fazer dessa maneira e talvez você

encontre, em entidade de projeto que você vai trabalhar essas anotações *named query*. É dessa maneira que se trabalha bem simples, não tem mistério nenhum aqui, funciona perfeitamente. Espero que vocês tenham gostado e eu vejo vocês na próxima aula. Um abraço.