08

# Para saber mais: Grafos de relacionamento com EntityGraphs

Uma alternativa às "queries" planejadas é utilizar um recurso que entrou na versão *2.1* da especificação, chamado *EntityGraphs*. Com esse recurso podemos dizer à JPA quais relacionamentos queremos trazer nas "queries". Alguns *providers* já forneciam um recurso parecido como no caso do **Hibernate Fetch Profiles** e o **Eclipse Link Fetch Groups**.

Para montar um `EntityGraphs` basta usarmos a anotação `@NamedEntityGraphs` na declaração da classe:

```
@NamedEntityGraphs()

@Entity
public class Produto {
```

Essa anotação recebe uma lista de `@NamedEntityGraph` com todos os grafos de relacionamento que queremos montar para essa entidade:

```
@NamedEntityGraphs({
    @NamedEntityGraph(name = "produtoComCategoria")
})

@Entity
public class Produto {
```

Ao montar um grafo, precisamos definir um nome e todos os atributos que queremos trazer junto ao grafo:

```
@NamedEntityGraphs({
    @NamedEntityGraph(name = "produtoComCategoria",
                      attributeNodes = {
                            @NamedAttributeNode("categorias")
                      })
})

@Entity
public class Produto {
```

Ao realizar a busca normalmente pela lista de produtos, devemos dizer qual grafo queremos utilizar passando uma `dica` na query:

```
public List<Produto> getProdutos() {
        return em.createQuery("select distinct p from Produto p", Produto.class)
                .setHint("javax.persistence.loadgraph", em.getEntityGraph("produtoComCategoria"))
                .getResultList();
}
```

Utilizando `EntityGraphs` podemos receber no método o nome do grafo e assim alternar entre os grafos em tempo de execução.