INSTITUTO FEDERAL
Rio Grande do Sul

Desenvolvimento de Sistemas I

Pesquisa REST/Spring-Data

Professor: *Jezer Machado de Oliveira*

Spring-data

O Spring-data possui uma poderosa API para geração de pesquisas em repositórios, baseada somente no nome do método

Spring Query Methods

```
@Repository
public interface ProdutoDAO extends CrudRepository<Produto, Integer>{

   List<Produto> findByNome(String nome);
   // Lista todos os produtos com o nome definido na variável nome
}
```

Spring Query Methods

```
@Repository
public interface ProdutoDAO extends CrudRepository<Produto, Integer>{

   List<Produto> findByNomeOrderByValor(String nome);
    // Lista todos os produtos com o nome definido na variável nome, ordenado
   // por valor
}
```

Spring Query Methods

```
@Repository
public interface ProdutoDAO extends CrudRepository<Produto, Integer>{

  List<Produto> findByValorBetween(double menos, double maior);
   // Lista todos os produtos com valor entre menor e maior
}
```

Spring Query Methods

Chave	Exemplo	SQL
And	findBySobrenomeAndNome(String sobrenome, String nome)	… where x.sobrenome = ?1 and x.nome = ?2
Or	findBySobrenomeOrNome(String sobrenome, String nome)	… where x.sobrenome = ?1 or x.nome = ?2
Between	findByAdmissaoBetween(Date inicio, Date termino)	… where x.admissao between 1? and ?2
LessThan	findByIdadeLessThan(Integer idade)	… where x.idade < ?1
GreaterThan	findByIdadeGreaterThan(Integer idade)	… where x.idade > ?1
After	findByAdmissaoAfter(Date data)	… where x.admissao > ?1
Before	findByAdmissaoBefore(Date data)	… where x.admissao < ?1
IsNull	findByIdadeIsNull	… where x.idade is null
IsNotNull,Not Null	findByIdade(Is)NotNull	… where x.idade not null

Spring Query Methods

Chave	Exemplo	SQL
Like	findByNomeLike(String nome)	… where x.nome like ?1
NotLike	findByNomeNotLike(String nome)	… where x.nome not like ?1
StartingWith	findByNomeStartingWith(String nome)	… where x.nome like “?1%“
EndingWith	findByNomeEndingWith(String nome)	… where x.nome like “%?1”
Containing	findByNomeContaining(String nome)	… where x.nome like “%?1%”
OrderBy	findByIdadeOrderBySobrenomeDesc(integer idade)	… where x.idade = ?1 order by x.sobrenome desc
Not	findBySobrenomeNot(String sobrenome)	… where x.sobrenome <> ?1
In	findByIdadeIn(Collection<Idade> idades)	… where x.idade in ?1
NotIn	findByIdadeNotIn(Collection<Idade> idade)	… where x.idade not in ?1
True	findByAtivoTrue()	… where x.ativo = true
False	findByAtivoFalse()	… where x.ativo = false

Spring JPQL Query

```
@Repository
public interface ProdutoDAO extends CrudRepository<Produto, Integer>{

   @Query(value = "SELECT produto FROM Produto produto WHERE
                   produto.nome=:nome AND produto.validade>NOW()")
   List<Produto> pesquisaCustomizada(@Param("nome") String nome);
   // Lista todos os produtos com o nome “nome” que não estão vencidos
}
```

JPQL

- Java Persistence Query Language
- JPQL é uma linguagem de consulta, assim como a SQL, porém orientada a objetos

Spring JPQL Query

JPQL

```
SELECT produto FROM Produto produto WHERE
                produto.nome=:nome AND produto.validade>NOW()"
```

SQL

```
SELECT * FROM produto WHERE nome=:nome AND validade>NOW()
```

JPQL

- Independente de Banco de Dados
- Suporte a funções de agregação ex. max , min, avg, etc.
- Suporte a Atualizações em Lote update e delete
- Suporte a subquery
- Suporte a inner e outer joins

Spring JPQL Query

select
from
[where]
[group by]
[having]
[order by]

Spring JPQL Coleções

SIZE(colecao)

SELECT produto FROM Produto produto WHERE SIZE(produto.marcas) =0

MEMBER OF

SELECT f FROM Funcionario f WHERE f MEMBER OF f.empresa.diretoria

NOT MEMBER OF

SELECT f FROM Funcionario f WHERE f NOT MEMBER OF f.empresa.diretoria

IS EMPTY

SELECT a FROM Autor a WHERE a.livros IS EMPTY

IS NOT EMPTY

SELECT a FROM Autor a WHERE a.livros IS NOT EMPTY

Spring Query – SQL nativo

```
@Repository
public interface ProdutoDAO extends CrudRepository<Produto, Integer>{

   @Query(nativeQuery = true,
    value = "SELECT * FROM produto WHERE nome=:nome AND validade>NOW()")
   List<Produto> pesquisaCustomizada(@Param("nome") String nome);
   // Lista todos os produtos com o nome “nome” que não estão vencidos
}
```

Spring Rest

```
@RestController
public class Produtos {

    @RequestMapping(path = "/produtos/pesquisar/nome",
                    method = RequestMethod.GET)
    public Iterable<Produto> pesquisaPorNome(
            @RequestParam(required = false) String igual,
            @RequestParam(required = false) String contem) {
        if (igual != null) {
            return produtoDAO.findByNome(igual);
        } else {
            return produtoDAO.findByNomeContaining(contem);
        }
    }
```

Spring Rest

```
@RestController
public class Produtos {
.....
   @RequestMapping(path = "/produtos/pesquisar/validade",
                   method = RequestMethod.GET)
   public Iterable<Produto> pesquisaPorValidade(
            @RequestParam
            @DateTimeFormat(pattern = "yyy-MM-dd") Date maior) {
      return produtoDAO.findByValidadeGreaterThan(maior);
   }
```

Spring JPQL Query

JPQL

```
SELECT  a FROM  Automovel a WHERE a.modelo.marca.nome = 'Ferrari’
```

SQL

```
SELECT *
FROM
Automovel auto
LEFT JOIN  Modelo modelo ON(auto.modelo_id = modelo.id)
LEFT JOIN Marca marca ON (modelo.marca_id =marca.id)
WHERE
marca.nome = 'Ferrari'
```