

Persistência de Objetos

Fausto Maranhão Ayres

3 DB4o (linguagem de consulta)

API S.O.D.A.

- **Simple Object Database Access** é uma API de consulta baseada em **nós de grafos** para criação de *consultas dinâmicas*.
- Ela permite **executar consultas no servidor**, reduzindo o tráfego de rede e o tempo de execução da consulta.

Exemplo: Consultar todos objetos da classe Pessoa

```
Query q = manager.query();  
q.constrain(Pessoa.class);  
List<Pessoa> resultados = q.execute();
```

Exemplos - Básicos

Consultar pessoas com
nome igual a "joao"

```
Query q = manager.query();  
q.constrain(Pessoa.class);  
q.descend("nome").constrain("joao");
```

```
List<Pessoa> resultados = q.execute();
```

Outras opções:

- `constrain("joao").contains();` //nome contem joao
- `constrain("joao").like();` //case insensitive
- `constrain(null);` //nome nulo

Exemplos - Básicos

Consultar pessoas com
Idade igual a 10

```
Query q = manager.query();  
q.constrain(Pessoa.class);  
q.descend("idade").constrain(10);
```

```
List<Pessoa> resultados = q.execute();
```

Outras opções:

```
.constrain(10).not(); // idade != 10  
.constrain(10).greater(); // idade > 10  
.constrain(10).greater().equals(). // idade >= 10 && < 50  
and(q.descend("idade").constrain(50).smaller());
```

Exemplos – Navegação entre classes

```
//Classe Telefone -> atributo pessoa -> objeto Pessoa ->
//atributo nome igual a "maria"
Query q = manager.query();
q.constrain(Telefone.class);
q.descend("pessoa").descend("nome").constrain("maria");
q.descend("numero").orderAscending(); //opcional


List<Telefone> resultados = q.execute();
```

Exemplos – Navegação entre classes

```
//Classe Pessoa -> atributo telefones -> objeto Telefone ->
//atributo numero igual a "88000000"
Query q = manager.query();
q.constrain(Pessoa.class);
q.descend("telefones").descend("numero")
                        .constrain("88000000");
List<Pessoa> resultados = q.execute();

//Classe Pessoa -> atributo telefones -> objeto Telefone
//igual ao objeto fornecido
Query q = manager.query();
q.constrain(Pessoa.class);
q.descend("telefones").constrain(new Telefone("88000000"));

List<Pessoa> resultados = q.execute();
```



Exemplos – Navegação entre classes

```
//Classe Livro -> atributo autores -> cada objeto Autor
-> atributo nome igual a "joao"
Query q = manager.query();
q.constrain(Livro.class);
q.descend("autores").descend("nome").constrain("joao");
//q.descend("titulo").orderDescending();
List<Livro> resultados = q.execute();
```

Dica: os campos mais consultados devem ser indexados

Exemplos – Navegação entre classes

```
//Classe Livro -> atributo autores -> cada objeto Autor
-> atributo livros -> cada objeto Livro -> atributo
titulo igual a "java"
Query q = manager.query();
q.constrain(Livro.class);
q.descend("autores").descend("livros").descend("titulo")
    .constrain("java");
//q.descend("titulo").constrain("java").not();
List<Livro> resultados = q.execute();
```

Dica: os campos mais consultados devem ser indexados

USO DE FILTROS CUSTOMIZADOS

fausto.ayres@ifpb.edu.br

9

Filtros customizados

```
...
Query q = manager.query();
q.constrain(Pessoa.class);
q.constrain(new Filtro1() );
List<Pessoa> resultados = q.execute();
}

//classe interna
class Filtro1 implements Evaluation {

    public void evaluate(Candidate candidate) {
        Pessoa p = (Pessoa) candidate.getObject();
        boolean filtro = p.getTelefones().size()==3 ;
        candidate.include(filtro);
    }
}
```

Pessoas que possuem 3 telefones

Esta classe interna pode ser escrita no fim do arquivo

fausto.ayres@ifpb.edu.br

10

Filtros customizados

```
Query q = manager.query();  
q.constrain(Produto.class);  
q.constrain(new Filtro2());  
List<Produto> resultados = q.execute();  
}
```

*Produtos que
possuem algum pedido
acima de 100 unidades*

```
//classe interna  
class Filtro2 implements Evaluation {  
    public void evaluate(Candidate candidate) {  
        Produto p = (Produto) candidate.getObject();  
        boolean filtro = false;  
        for (Pedido ped : p.getPedidos())  
            if (ped.getQuantidade() > 100)  
                filtro = true;  
        candidate.include(filtro);  
    }  
}
```

fausto.ayres@ifpb.edu.br

11

Tamanho do resultado da consulta

- esta operação não gera custo de carga dos objetos

```
Query q = manager.query();  
q.constrain(Pessoa.class);  
  
int i = q.execute().size();
```

fausto.ayres@ifpb.edu.br

12