Java Persistence API (JPA)
Consultas

Fernando dos Santos
fernando.santos@udesc.br

UDESC
Ibirama

Consultas

- JPA oferece uma linguagem própria para consulta de entidades.
  - Portabilidade entre bancos.

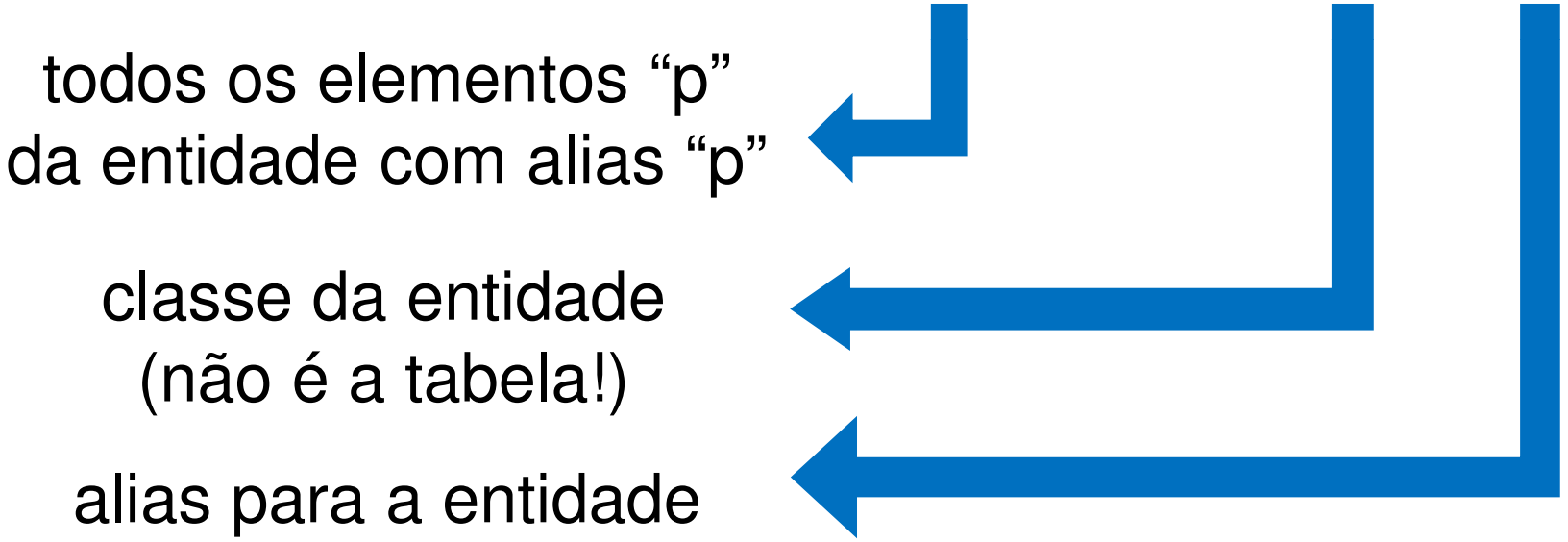
- Consultas são representadas e executadas por um objeto Query.

- Exemplo de consulta para buscar todas as entidades no banco:

```
Query consulta1 = em.createQuery("select p from Produto p");
List<Produto> produtos = consulta1.getResultList();
for(Produto prod : produtos){
    System.out.println("Nome: "+prod.getNome());
}
```

Como escrever consultas

select p from Produto p



Consultas com parâmetros

- Buscar uma única entidade com determinado código:

```
Query consulta2 = em.createQuery("select p from Produto p where p.id = 3");
Produto  prod = (Produto)consulta2.getSingleResult();
System.out.println("Nome: " + prod.getNome());
```

- Busca uma única entidade passando dados via parâmetros:

```
Query consulta3 = em.createQuery("select p from Produto p where p.id = :id");
consulta3.setParameter("id", 1);
Produto prod = (Produto)consulta3.getSingleResult();
System.out.println("Nome: " + prod.getNome());
```

- Operador like

```
Query consulta4 =
            em.createQuery("select p from Produto p where p.nome like :nome");
consulta4.setParameter("nome", “Arroz%");
List<Produto> produtos = consulta4.getResultList();
```

Consultas com funções de agrupamento

- count() e retorno de valor simples

```
Query consulta5 = em.createQuery("select count(p) from Produto p");
Long quantidade = (Long)consulta5.getSingleResult();
System.out.println("Quantidade: " + quantidade);
```

```
Query consulta6 =
        em.createQuery("select count(distinct p.nome) from Produto p");
Long quantidade = (Long)consulta6.getSingleResult();
System.out.println("Quantidade de Nomes Distintos: " + quantidade);
```

- count() e retorno de muitas “colunas” (simples e/ou de objetos)

```
Query consulta7 =
    em.createQuery("select p.nome, count(p) from Produto p group by p.nome");
List<Object[]> linhas = consulta7.getResultList();
for (Object[] linha : linhas) { // cada elemento é um vetor, representa a linha
   String nome = (String)linha[0];
   Long count = (Long)linha[1];
   System.out.println(“nome: "+nome+" Quantidade: “+count);
}
```

Consultas com subconsultas

- Selecionar todas os produtos com valor maior que a média:
- Base:

ID	Nome	Preço
1	Arroz	1.50
2	Açúcar	2.00
3	Feijão	3.75

Resultado:

ID	Nome	Preço
3	Feijão	3.75

```
Query consulta =
        em.createQuery("select p1 from Produto p1
                                where  p1.precoUnitario >=
                              ( select avg(p2.precoUnitario)
                                from Produto p2 )“
                              );
List<Produto> produtos = consulta.getResultList();
for(Produto prod : produtos){
        System.out.println("Nome: "+prod.getNome());
}
```

Consultas com comparação na mesma entidade

- Selecionar todos os produtos cujo valor é maior que o valor de um produto com determinado código:
- Base:

ID	Nome	Preço
1	Arroz	1.50
2	Açúcar	2.00
3	Feijão	3.75

Resultado:

ID	Nome	Preço
2	Açúcar	2.00
3	Feijão	3.75

```
Query consulta = em.createQuery("select p1
                                   from Produto p1, Produto p2
                                  where p1.precoUnitario > p2.precoUnitario
                                    and p2.id = :id");
consulta.setParameter("id", 1);
List<Produto> produtos = consulta.getResultList();
for(Produto prod : produtos){
    System.out.println("Nome: "+prod.getNome());

}
```

Consultas com verificação de intervalo

- Selecionar todos os produtos cujo preço esteja entre um valor **mínimo** e **máximo**, informados como parâmetro:
- Base:

ID	Nome	Preço
1	Arroz	1.50
2	Açúcar	2.00
3	Feijão	3.75

Resultado:

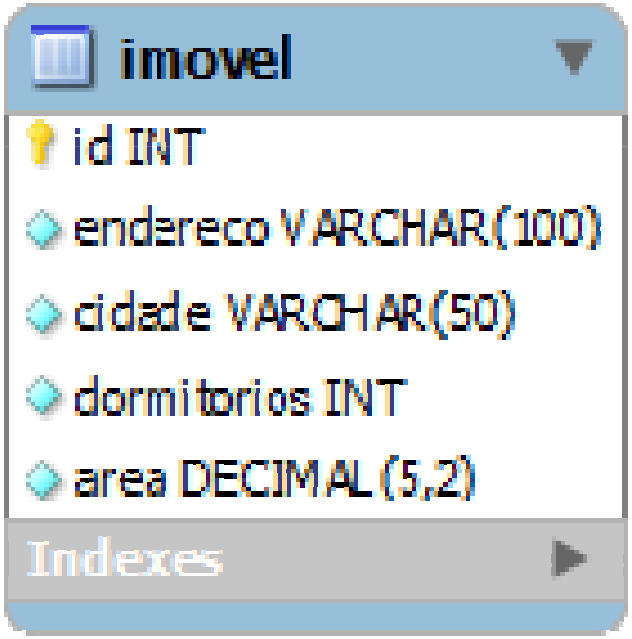
ID	Nome	Preço
1	Arroz	1.50
2	Açúcar	2.00

```
Query consulta = em.createQuery("select p
                                   from Produto p
                                  where p.precoUnitario
                                        between :valorMin and :valorMax");
consulta.setParameter("valorMin", 1.50f);    // " f " força conversão para float
consulta.setParameter("valorMax", 2.50f);
List<Produto> produtos = consulta.getResultList();
for (Produto prod : produtos) {
    System.out.println("Nome: " + prod.getNome());

}
```

Exercício

- Criar opções de consulta no sistema de imóveis:



  - imóveis de uma determinada cidade (informada pelo usuário)
  - imóveis com área superior a média de áreas
  - imóveis com a quantidade de quartos dentro de uma faixa informada pelo usuário (minimo e máximo)
  - quantidade de imóveis cadastrados
  - imóveis cadastrados por cidade

Bibliografia

- BAUER, Christian; KING, Gavin. **Java Persistence com Hibernate.** Rio de Janeiro: Ciência Moderna, 2007. 844 p.

- BURKE, Bill; MONSON-HAEFEL, Richard. **Enterprise JavaBeans 3.0**. 5.ed. São Paulo: Prentice Hall, 2007. 538 p.

- **The Java EE 6 Tutorial**, parte VI (Persistence)
  - http://download.oracle.com/javaee/6/tutorial/doc/