Java Persistence Querying Language (JPQL)

Querying - JPQL

- **Java Persistence Querying Language (JPQL)**
- **Comportamento análogo ao SQL mas funcionando sobre a estrutura de objetos e não sobre a estrutura relacional**

```
select ga from GrupoAutomovel ga where ga.name like :pat

select a from Automovel a where a.numPortas > :n
```

Querying - JPQL

- **O método createQuery do objecto EntityManager cria um objecto Query**

```
(…)
Query query=
        manager.createQuery("select ga from GrupoAutomovel ga");
List<GrupoAutomovel> results=query.getResultList();
(…)
for (GrupoAutomovel g:results)
{
     System.out.println(g);
}
```

Querying - JPQL

- ***Selection – From* clause**
  - *SELECT* queries são usados para ler objetos da base de dados
  - *FROM* define quais as entidades a ler da base de dados.
- **A sintaxe do JPQL FROM é similar ao SQL mas usa o modelo de entidades em vez de tabelas.**
- **No JPQL seguinte é referenciada a entidade `Automovel` e atribuída à variável de identificação `a`**

```
SELECT a FROM Automovel a
```

a partir da qual se pode aceder aos seus atributos e métodos

```
a.grupo   a.grupo.numPortas()   a.matricula()
```

- **A cláusula SELECT pode conter funções agregadas, sub-select, etc**

```
SELECT COUNT(a) FROM Automovel a
SELECT MAX(e.salary) FROM Employee e
```

Querying - JPQL

- **WHERE – filtros**

```
select a from Automovel a where a.marca= :m
```

parâmetro

- **Deve-se usar parâmetros na construção do query. O parâmetro indica-se com um nome precedido de : (:m)**
- **Os parâmetros são preenchidos com o método setParameter do objeto query**

```
Query query=manager.createQuery(
 "select a from Automovel a where a.marca= :m");
query.setParameter("m", "Volkswagen");
```

- **Operadores de WHERE são semelhantes aos do SQL**
  - Expressões matemáticas
  - Expressões de comparação
  - Operadores lógicos
  - Operador LIKE

Querying - JPQL

- **Named queries annotations**

```
@Entity
@NamedQueries({
  @NamedQuery(name="magsOverPrice",
              query="SELECT x FROM Magazine x WHERE x.price > ?1")
  @NamedQuery(name="magsByTitle",
  query="SELECT x FROM Magazine x WHERE x.title = :titleParam")
})
public class Magazine { ... }
```

- **Exemplo de código**

```
(…)
Query query = em.createNamedQuery ("magsOverPrice");
query.setParameter (1, 5.0f);
List<Magazine> results = q.getResultList ();
(…)
```

Querying - JPQL

- **Query parameters**

  ?ordinalParameter

  :namedParameter

  What is the difference? Which one do you prefer?

```
@Entity
@NamedQueries({
  @NamedQuery(name="magsOverPrice",
              query="SELECT x FROM Magazine x WHERE x.price > ?1")
  @NamedQuery(name="magsByTitle",
  query="SELECT x FROM Magazine x WHERE x.title = :titleParam")
})
public class Magazine { ... }
```

Querying - JPQL

- **Alguns métodos do objecto Query**
  - `getResultList()`
    - Execute a SELECT query and return the query results as an untyped List.
  - `getSingleResult()`
    - Execute a SELECT query that returns a single untyped result.
  - `setFirstResult(int startPosition)`
    - Set the position of the first result to retrieve.
  - `executeUpdate()`
    - Execute an update or delete statement.