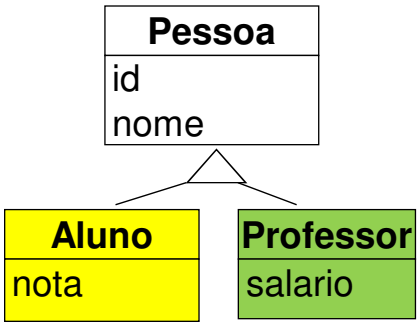
INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
PARAÍBA

PERSISTÊNCIA DE OBJETOS

Fausto Maranhão Ayres

9 JPA: Herança

Exemplo



Mapeamento Default

```
@Entity
public class Pessoa {
    @Id private int id;
    private String nome; ...
}
```

```
@Entity
public class Aluno extends Pessoa {
    private double nota; ...
}

@Entity
public class Professor extends Pessoa {
    private double salario; ...
}
```


Mapeamento Default

```
Pessoa p1 = new Pessoa("joao");
Aluno a1 = new Aluno("maria", 8.0);
Professor pf1 = new Professor("paulo", 1000.0);
Professor pf2 = new Professor("jose", 1500.0);

...
manager.persist(p1);
manager.persist(a1);
manager.persist(pf1);
manager.persist(pf2);
...
```

Coluna Discriminadora:
coluna de controle que identifica os objetos na leitura/gravação.

Tabela PESSOA

id	nome	nota	salario	DTYPE
1	joao			Pessoa
2	maria	8.0		Aluno
3	paulo		1000.0	Professor
4	jose		1500.0	Professor

Desvantagem:
Muitos vazios

Mapeamento Default

A coluna discriminadora é criada, automaticamente, e fica inacessível

```
CREATE TABLE PESSOA (
        ID       INTEGER NOT NULL,
        NOME     VARCHAR(255),
        NOTA     FLOAT,
        SALARIO FLOAT,
        DTYPE VARCHAR(31),
        PRIMARY KEY (ID)
)
```


*Mapeamento 1: **Single_Table***

é definida no topo da hierarquia

```
@Entity
@Inheritance(strategy = InheritanceType.SINGLE_TABLE)
@DiscriminatorColumn(name="Tipo",
                     discriminatorType= DiscriminatorType.INTEGER)
@DiscriminatorValue(1)
public class Pessoa{ ...}
```

valor da coluna discriminadora em cada classe

```
@Entity
@DiscriminatorValue(2)
public class Aluno  extends Pessoa {...}
```

```
@Entity
@DiscriminatorValue(3)
public class Professor  extends Pessoa {...}
```

Resultado

Coluna discriminadora

Tabela PESSOA

id	nome	nota	salario	tipo
1	joao			1
2	maria	8.0		2
3	paulo		1000.0	3
4	jose		1500.0	3

Consultas Polimórficas (**JPQL**)

```
Query q;

q = manager.createQuery("select a from Aluno a");
List<Aluno> lista1 = q.getResultList();

q = manager.createQuery("select p from Professor p");
List<Professor> lista2 = q.getResultList();

q = manager.createQuery("select p from Pessoa p");
List<Pessoa> lista3 = q.getResultList();
```

só alunos

só professores

todos

*Consultas Restritas (**JPQL**)*

```
Query q;

q = manager.createQuery(
"select p from Pessoa p where type(p) in {Professor, Aluno}");

List<Aluno> lista1 = q.getResultList();
```


*Mapeamento 2: **Joined***

```
@Entity
@Inheritance(strategy = InheritanceType.JOINED)
public class Pessoa {
...
```

Tabela PESSOA

id	nome	DTYPE
1	joao	Pessoa
2	maria	Aluno
3	paulo	Professor
4	jose	Professor

Tabela ALUNO

id	nota
2	8.0

id	salario
3	1000.0
4	1500.0

Tabela PROFESSOR

Desvantagem: Necessita de junção para ler aluno e professor

Mapeamento 3: ***Table_per_class***

```
@Entity
@Inheritance(strategy = InheritanceType.TABLE_PER_CLASS)
public class Pessoa{
        @Id private int id;
        private String nome;
```

Não necessita de coluna discriminadora

id	nome
1	joao

PESSOA

id	nome	nota
1	maria	8.0

ALUNO

id	nome	salario
1	paulo	1000.0
2	jose	1500.0

PROFESSOR

Desvantagem:
1-Necessita de vários Selects para ler todos os nomes.
2-Consultas Polimórficas ineficientes

Classe Abstrata Persistente

```
@Entity
public abstract class Pessoa{
        @Id private int id;
        private String nome;  ...
```

Não cria a tabela **Pessoa,** mas o nome da classe pode ser usada nas consultas polimórficas

```
@Entity
public class Aluno  extends Pessoa {...}
```

```
@Entity
public class Professor extends Pessoa {...}
```

Classe Abstrata não Persistente

```
@Entity
@MappedSuperclass
public abstract class Pessoa{
        @Id private int id;
        private String nome;  ...
```

Não cria a tabela **Pessoa** e o nome da classe NÃO pode ser usada nas consultas polimórficas

```
@Entity
public class Aluno extends Pessoa{...}
```

```
@Entity
public class Professor extends Pessoa{...}
```


Renomeação de atributos herdados

```
@Entity
@Inheritance(strategy = InheritanceType.TABLE_PER_CLASS)
public class Pessoa{
        @Id private int id;
        private String nome;
```

id	nome
1	joao

```
@Entity
@AttributeOverride(name="nome",
      column=@Column(name="nomealu"))
public class Aluno  extends Pessoa {...}
```

id	nomealu	nota
1	maria	8.0

```
@Entity
@AttributeOverride(name="nome",
      column=@Column(name="nomeprof"))
public class Professor extends Pessoa {...}
```

id	nomeprof	salario
1	paulo	1000.0
2	jose	1500.0