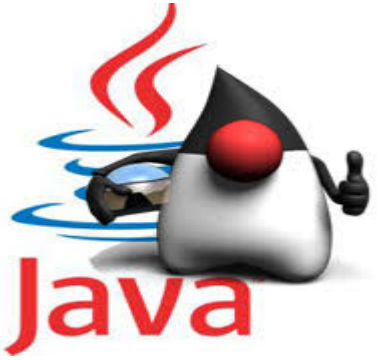
SOU Java
sociedade de usuários java

SOU Java
sociedade de usuários java

Alta Performance com Hibernate

Rhuan Henrique Rocha da Silva
Red Hat Sênior Middleware Consultant / Especialista Java EE

HIBERNATE



Quem Sou Eu?

- **Red Hat Sênior Middleware Consultant na FabricaDS.**
- **Especialista Java EE.**
- **Membro do SouJava {Rio}.**
- **Larga experiência com Projetos Java EE para órgãos governamentais e empresas privadas.**
- **Contribuidor para o Hibernate e JNoSQL.**

Como será a palestra?

- **Breve Introdução ao Hibernate**
- **Demostrando erros comuns que degradam a performance**
- **Utilizando recursos do Hibernate para alcançar a alta performance**

Hibernate

- **Framework para Mapeamento Objeto Relacional (ORM)**
- **Implementação do JPA**
- **Red Hat support.**

Por que utilizar Hibernate?

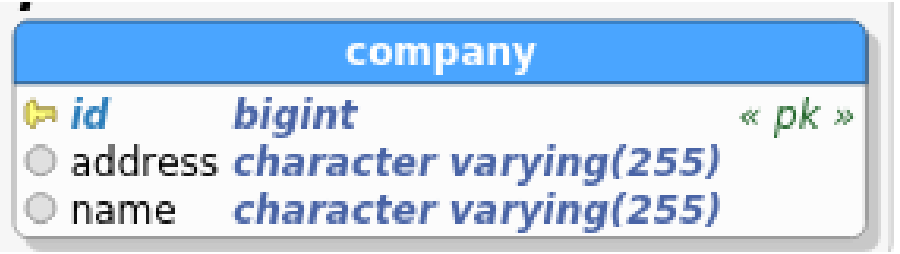
- **Orientação a Objeto X Paradigma Relacional.**
- **Abstração de complexidades na manipulação do JDBC.**
- **Soluções prontas que permitem tunning na recuperação de dados.**

Java™
Enterprise Edition

HIBERNATE

Por que utilizar Hibernate?

```
@Entity
@Table(name="Company")
public class Company {

    @Id
    @GeneratedValue
    private Long id;

    @Column
    private String name;

    @Column
    private String address;

    //Getter and Set
}
```



Java™
Enterprise Edition

HIBERNATE

Por que utilizar Hibernate?

Persistência:

```
Transaction transaction =
session.beginTransaction();

session.save(company);

transaction.commit();
session.close();
```

Buscando dados:

```
Company company =
session.find(Company.class, (Object) id);
```

Java™ Enterprise Edition

HIBERNATE

Relacionamentos

- **@ManyToOne**
- **@OneToOne**
- **@ManyToMany**
- **@OneToMany**

Relacionamentos

```
@ManyToOne
@Fetch(FetchMode.JOIN)
private Profession profession;
```

```
@ManyToMany
private Set<Company> companies;
```

Java™
Enterprise Edition

HIBERNATE

Fetching

- **Direct Fetching**
  - **find(), load(), get(), list(), iterate(), scroll()**
- **Entity Queries**
  - **HQL**

Fetching Strategies

- Join fetching
- Select Fetching
- Subselect Fetching
- Batch Fetching

Fetch Type x Fetch Mode

Fetch Type

- **Momento do Fetching**
- **Altera a quantidade de dados retornado.**

Fetch Mode

- **Seleciona a estratégia de fetching**
- **Altera a quantidade de acessos ao banco.**

N+1 Query

Exemplo de N+1 Query

Código:

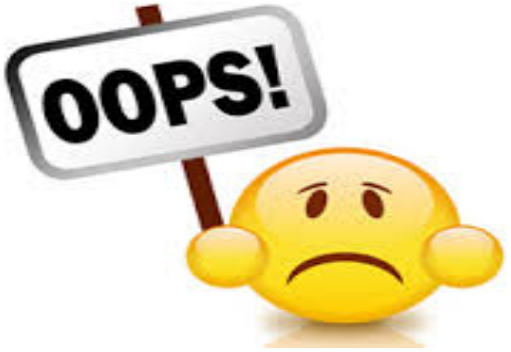
```
Query<Person> query = session.createQuery("select p from Person p where id =:id");
 query.setParameter("id",id);
 Person person = query.getSingleResult();
 System.out.println("Company size:" + person.getCompanies().size());
```

Exemplo de N+1 Query

Query Executada:

```
select person0_.id as id1_1_, person0_.name as name2_1_
    from Person person0_ where person0_.id=?
```

```
select companies0_.Person_id as Person_i1_2_0_,
    companies0_.companies_id as companie2_2_0_,
    company1_.id as id1_0_1_, company1_.address as address2_0_1_,
    company1_.name as name3_0_1_
  from Person_Company companies0_  inner join Company company1_
    on companies0_.companies_id=company1_.id
 where companies0_.Person_id=?
```

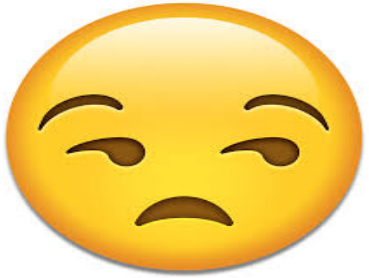


Erros de Mapeamentos das Entidades

Fetch Type Eager

Código:

```
@Entity(name = "Person")
public class Person {

 @Id
 @GeneratedValue
 private Long id;

 @Column
 private String name;

 @ManyToMany(fetch=FetchType.EAGER)
 private Set<Company> companies;

     // Getter e Setter

}
```



Acesso a Atributos com Fetch Type Lazy

Exemplo com Acesso via Get

Entity:

```
@Entity(name = "Person")
public class Person {

 @Id
 @GeneratedValue
 private Long id;

 @Column
 private String name;

 @ManyToMany(fetch=FetchType.LAZY)
 private Set<Company> companies;

     // Getter e Setter
}
```

Exemplo com Acesso via Get

Código 1:

```
System.out.println("Company size:" + person.getCompanies().size());
```

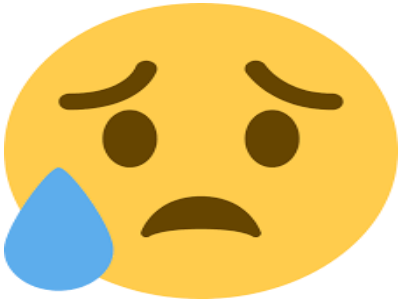
Código 2:

```
person.getCompanies().get(0).getName();
```

Exemplo com Acesso Através de Outros Métodos da Classe

Código:

```
@Override
public boolean equals(Object o) {
    if (this == o) return true;
    if (o == null || getClass() != o.getClass()) return false;
    Person person = (Person) o;
    return Objects.equals(companies, person.companies)&&
            Objects.equals(id, person.id) &&
            Objects.equals(name, person.name);

}
```

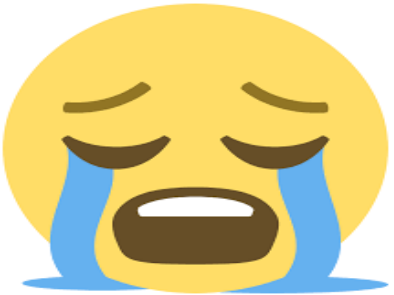


Má utilização do @PostLoad

Exemplo de Má Utilização do @PostLoad

Código:

```
@PostLoad
public void postLoad(){
    this.companySize = this.companies == null ? 0 : this.companies.size();
}
```



Má utilização do NativeQuery

Exemplo de Má Utilização do NativeQuery

Código:

```
Query<Person> query = session.createNativeQuery(
                      "select p1.* from Person p1 left join profession p2 on p1.id = p2.id where p1.id =:id",
                       Person.class);

query.setParameter("id",id);
Person person = query.getSingleResult();
```

Exemplo de Má Utilização do NativeQuery

Query Executada:

```
select person0_.id as id1_1_, person0_.name as name2_1_
      from Person person0_
      where person0_.id=?
```

```
      select profession0_.id as id1_3_0_,
            profession0_.name as name2_3_0_
      from
            Profession profession0_
      where profession0_.id=?
```

Utilizando Fetch Mode

Exemplo de Utilização do Fetch Mode

Codigo Entidade:

```
@Entity(name = "Person")
public class Person {

    @Id
    @GeneratedValue
    private Long id;

    @Column
    private String name;

    @ManyToOne
    @Fetch(FetchMode.JOIN)
    private Profession profession;

     //Getter e Setter
}
```

Exemplo de Utilização do Fetch Mode

Codigo Consulta:

```
Person person = session.find(Person.class, (Object) id);
```

Exemplo de Utilização do Fetch Mode

Query Executada:

```
select
    person0_.id as id1_1_0_, person0_.name as name2_1_0_,
    person0_.profession_id as professi3_1_0_,
  profession1_.id as id1_3_1_,
    profession1_.name as name2_3_1_
from Person person0_
left outer join Profession profession1_
       on person0_.profession_id=profession1_.id
where person0_.id=?
```

Utilizando Join Fetch

Exemplo de Utilização do Join Fetch

Codigo:

```
Query<Person> query = session.createQuery(
            "select p from Person p left join fetch p.profession where p.id =:id");


query.setParameter("id",id);
Person person = query.getSingleResult();
```

Exemplo de Utilização do Join Fetch

Query Executada:

```
select
      person0_.id as id1_1_0_,
      person0_.name as name2_1_0_,
      person0_.profession_id as professi3_1_0_,
      profession1_.id as id1_3_1_,
      profession1_.name as name2_3_1_
from Person person0_
left outer join Profession profession1_
     on person0_.profession_id=profession1_.id
where  person0_.id=?
```

Eliminando N+1 Query em NativeQuery com Entity Query

Exemplo de utilização do Entity Query

Codigo:

```
NativeQuery query = session.createNativeQuery("select {per.*}, {prof.*} from Person per left join profession prof on per.profession_id = prof.id where per.id =:id").setParameter("id",id);

query.addEntity("per", Person.class)
query.addJoin("prof", "per.profession");

List<Object[]> list = query.list();
Person person = null;
for(Object[] tuple : list){
    person = (Person)tuple[0];
}
```

Exemplo de utilização do Entity Query

Query executada:

```
select per.id as id1_1_0_,
    per.name as name2_1_0_,
    per.profession_id as professi3_1_0_,
    prof.id as id1_3_1_,
    prof.name as name2_3_1_
from Person per
left join profession prof
        on per.profession_id = prof.id
where per.id =?
```

Aumentando a Performance com Cache L2

Exemplo de utilização do Cache L2

Propriedades adicionadas ao hibernate.properties:

```
hibernate.cache.use_second_level_cache=true
hibernate.cache.region.factory_class=org.hibernate.cache.ehcache.EhCacheRegionFactory
```

Exemplo de utilização do Cache L2

Codigo da entidade configurada para cache:

```
@Entity(name = "Person")
@Cacheable
@Cache(usage = CacheConcurrencyStrategy.READ_ONLY,region = "teste")
public class Person {

    @Id
    @GeneratedValue
    private Long id;

    @Column
    private String name;

    @ManyToOne
    @Fetch(FetchMode.JOIN)
    private Profession profession;

    //Getter e Setter
}
```

Operações que ativam Cache L2

- **Persistência de dados**
  - **save(), update(), saveOrUpdate()**
- **Recuperação de dados**
  - **Find(), load(), get(), list(), iterate(), scroll()**

Aumentando a Performance com Query Cache

Exemplo de utilização do Query Cache

Propriedades adicionadas ao hibernate.properties:

```
hibernate.cache.use_query_cache=true
```

Codigo da configurada da consulta:

```
Query<Person> query = session.createQuery(
                                    "select p from Person p left join fetch p.profession where p.id =:id");

 query.setParameter("id",id);

query.setCacheable(true);
query.setCacheRegion("teste");

Person person = query.getSingleResult();
```

Obrigado!

Contatos

Twitter: @rhuan080

Email: rhuan080@gmail.com