CURSO DE HIBERNATE 5

OpenWebinars

HIBERNATE

(1) INTRODUCCION

Persistencia, desfase objecto-relacional, ORM. Productos y estándares

(2) HIBERNATE

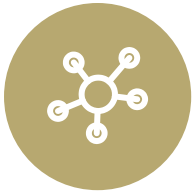
Más que un ORM. Comparativa con otros productos. JPA. Maven. Módulos

(3) PRIMER PROYECTO

Hibernate.cfg.xml, EntityManager y persistence.xml

(4) ENTIDADES

Definición del modelo del dominio. Entidades y ciclo de vida. XML y anotaciones. Tipos de datos.



(5) ASOCIACIONES

ManyToOne, OneToMany, OneToOne, ManyToMany

HIBERNATE

(6) ELEMENTOS AVANZADOS

Campos calculados, herencia.

(7) COLECCIONES

Mapeo de colecciones. Tipos (list, set, map). Colecciones ordenadas (sorted vs. ordered).

(8) GENERACION DEL ESQUEMA

Customización del proceso de generación del esquema.

(9) CONTEXTO DE PERSISTENCIA

Almacenamiento, recuperación y borrado de entidades.

(10) TRANSACCIONES

Control de concurrencia. Patrones y antipatrones.

HIBERNATE

(12)
ENVERS

Introducción a la
auditoria de entidades.

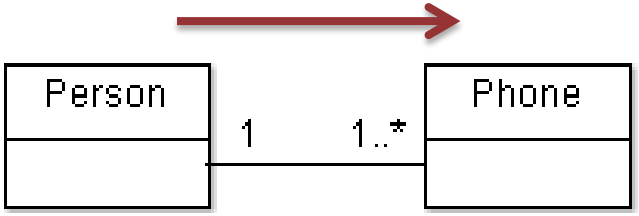
(11)
CONSULTAS
HPQL VS JPQL

Consultas con
parámetros,
Anotaciones. SQL nativo

1.

ASOCIACIONES UNO A MUCHOS UNIDIRECCIONALES

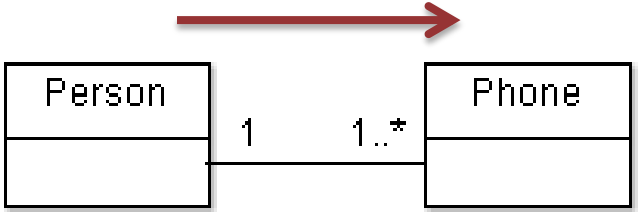
ASOCIACIONES UNO A MUCHOS UNIDIRECCIONALES




```
@OneToMany(cascade = CascadeType.ALL, orphanRemoval = true)
private List<Phone> phones = new ArrayList<>();
```

Estas propiedades indican que el ciclo de vida de un Phone se circunscribe al ciclo de vida del Person con el que están asociados. Si se desasignan de la lista, Hibernate se encarga de eliminarlos.

ASOCIACIONES UNO A MUCHOS UNIDIRECCIONALES



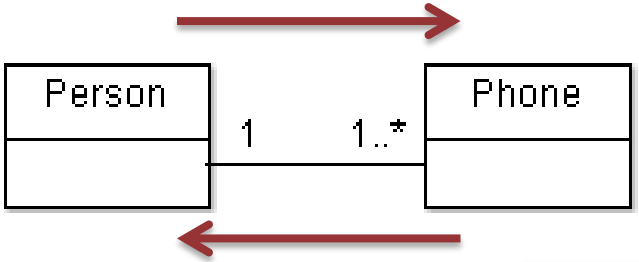

```
Person person = new Person("Pepe");
Phone phone1 = new Phone("954000000");
Phone phone2 = new Phone("600000000");

person.getPhones().add(phone1);
person.getPhones().add(phone2);
em.persist(person);
em.flush();

person.getPhones().remove(phone1);
```

Gestionamos el ciclo de vida de un Phone se a través de la lista que tiene un `Person`.

2.

ASOCIACIONES UNO A MUCHOS BIDIRECCIONALES

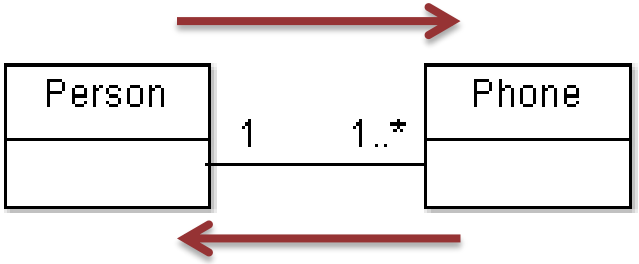
ASOCIACIONES UNO A MUCHOS BIDIRECCIONALES




```java
@Entity
public class Person {

    @Id
    @GeneratedValue
    private long id;

    @OneToMany(mappedBy = "person", cascade = Cascac
    private List<Phone> phones = new ArrayList<>();
```

```java
@Entity
public class Phone {

    @Id
    @GeneratedValue
    private long id;

    private String number;

    @ManyToOne
    private Person person;
```

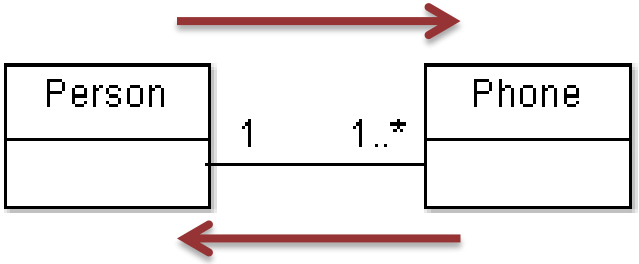
ASOCIACIONES UNO A MUCHOS BIDIRECCIONALES




```
public void addPhone(Phone phone) {
    phones.add(phone);
    phone.setPerson(this);
}

public void removePhone(Phone phone) {
    phones.remove(phone);
    phone.setPerson(null);
}
```

Métodos **HELPER**

ASOCIACIONES UNO A MUCHOS BIDIRECCIONALES




```
Person person = new Person("Pepe");
Phone phone1 = new Phone("954000000");
Phone phone2 = new Phone("600000000");

person.addPhone(phone1);
person.addPhone(phone2);
em.persist(person);
em.flush();

person.removePhone(phone1);
```