HIBERNATE

Sesión 3: Relaciones entre objetos

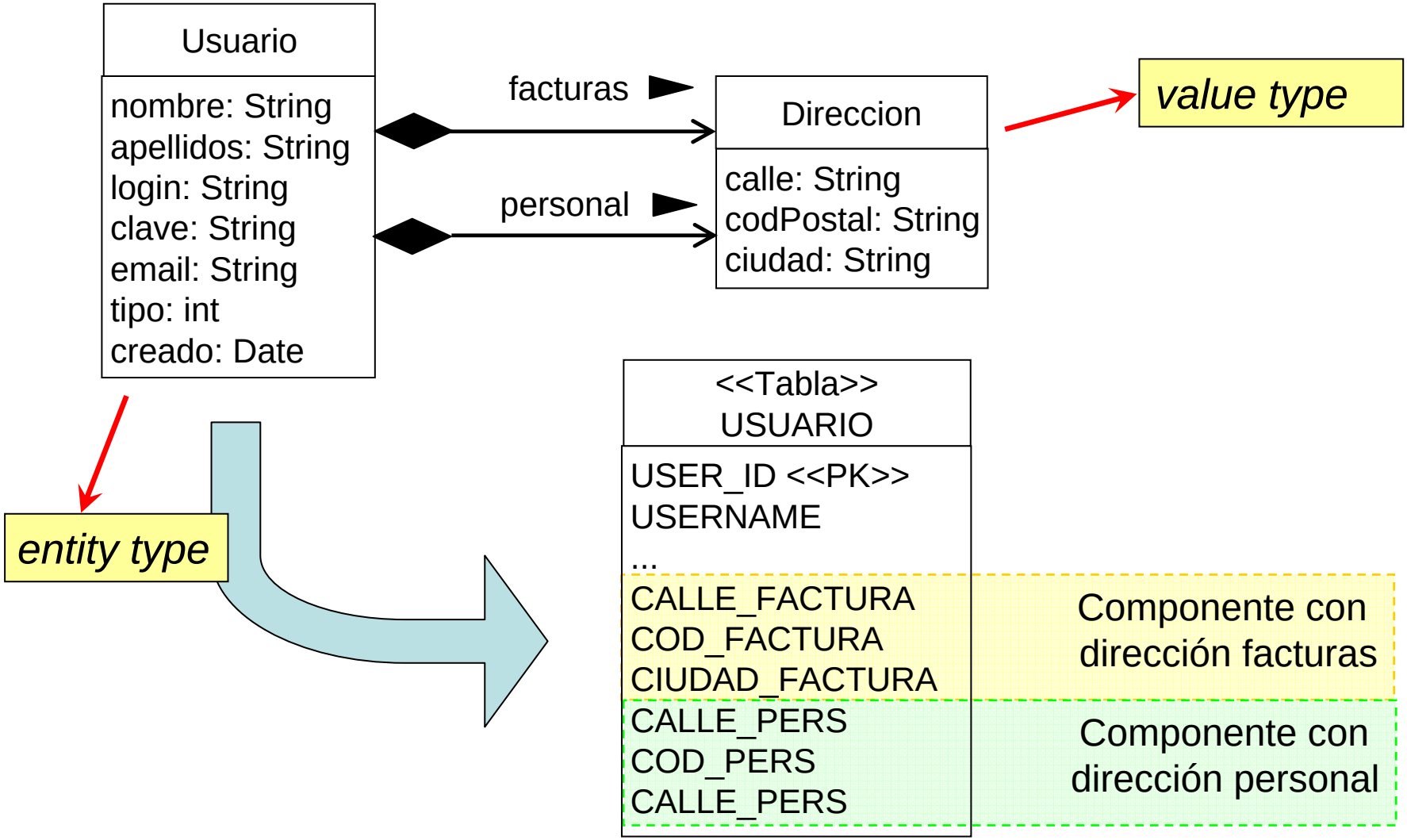
Puntos a tratar

- Persistencia transitiva
- Composición
- Herencia
- Asociaciones

Persistencia transitiva

- Nos permite propagar la persistencia a subgrafos *transient* o *detached* de forma automática
  - Trabajamos con grafos de objetos debido a las relaciones entre clases
- Hibernate NO navega (por defecto) a través de las asociaciones
- Para "activar" la persistencia transitiva en asociaciones:
  - Asignamos un valor a cascade distinto de "none"
  - cascade="all,delete-orphan" $\Rightarrow$ Relación padre/hijo

Composición

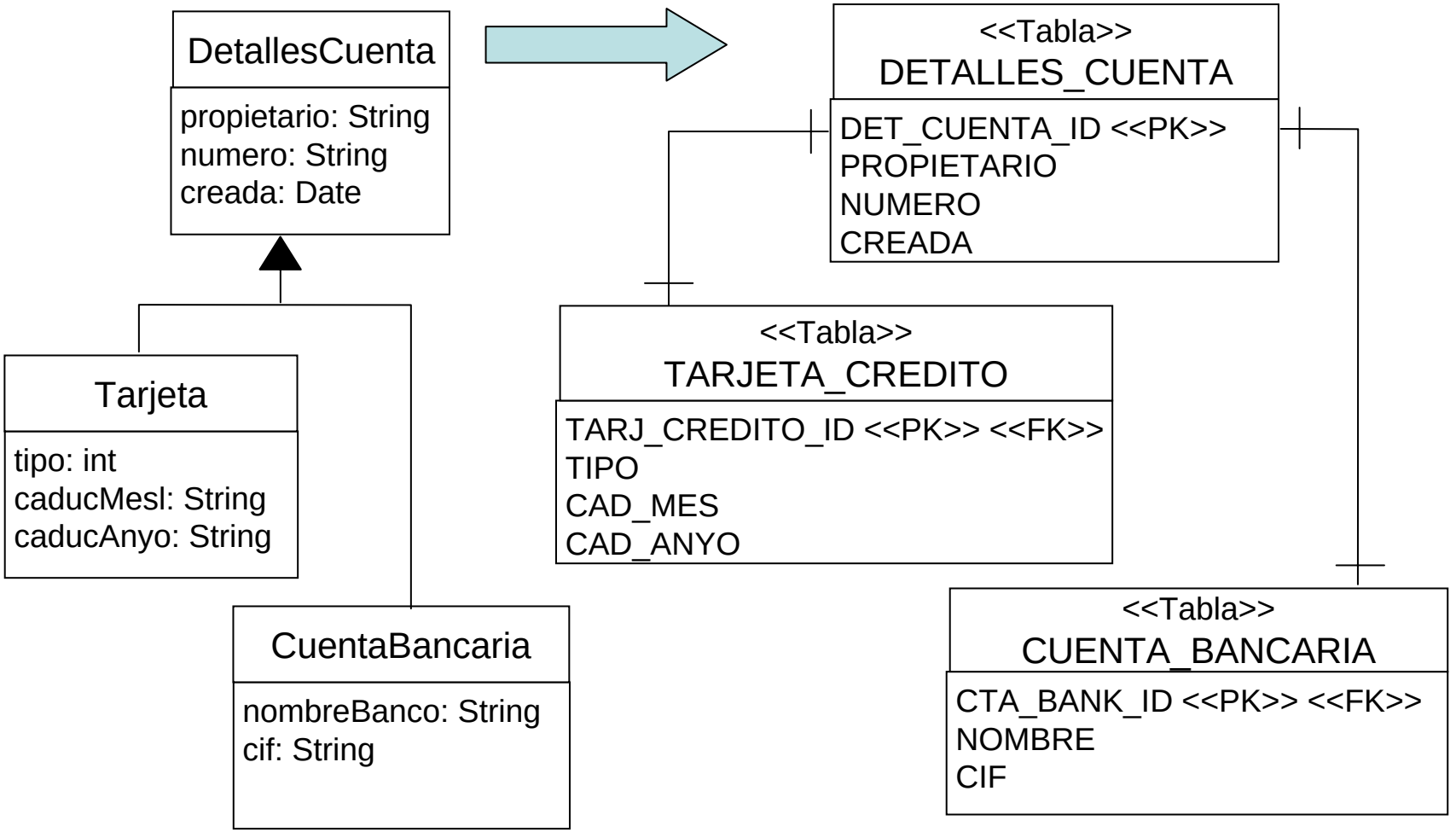


Mapeado para la composición

```
<class name="Usuario" table="USUARIO">
    <id name="id" column="USER_ID" type="long">
        <generator class="native"/> </id>
    <property name="nombre" column="USERNAME"
            type="string"/>
    <component name="direccionPersonal" class="Direccion">
        <property name="calle" type="string"
                column="CALLE_PERS" not-null="true"/>
        <property name="ciudad" type="string"
                column="CIUDAD_PERS" not-null="true"/>
        <property name="codPostal" type="short"
                column="COD_PERS" not-null="true"/>
    </component>

    <component name="direccionFacturas" class="Direccion">
     ...
    </component>
</class>
```

Herencia


DetallesCuenta
propietario: String
numero: String
creada: Date
<<Tabla>>
DETALLES_CUENTA
DET_CUENTA_ID <<PK>>
PROPIETARIO
NUMERO
CREADA
Tarjeta
tipo: int
caducMesl: String
caducAnyo: String
<<Tabla>>
TARJETA_CREDITO
TARJ_CREDITO_ID <<PK>> <<FK>>
TIPO
CAD_MES
CAD_ANYO
CuentaBancaria
nombreBanco: String
cif: String
<<Tabla>>
CUENTA_BANCARIA
CTA_BANK_ID <<PK>> <<FK>>
NOMBRE
CIF

Mapeado para la herencia

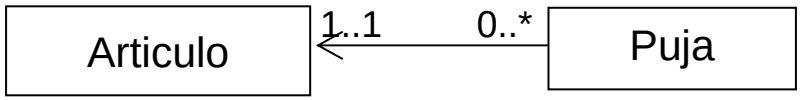
```
<hibernate-mapping>
   <class name="DetallesCuenta" table="DETALLES_CUENTA">
       <id name="id" column="DET_CUENTA_ID" type="long">
           <generator class="native"/> </id>
       <property name="propietario" column="PROPIETARIO"
                 type="string"/>
       <joined-subclass name="Tarjeta" table="TARJETA_CREDITO">
              <key column="TARJ_CRED_ID"/>
              <property name="tipo" column="TIPO"/>
        ...
       </joined-subclass>
       <joined-subclass name="CuentaBancaria" table="CUENTA_BANCARIA">
       ...
       </joined-subclass>
    </class>
<hibernate-mapping>
```

Asociaciones

- Las asociaciones en Hibernate son inherentemente **unidireccionales**
- Hibernate NO gestiona asociaciones persistentes
  - Si una asociación es bidireccional, no importa la multiplicidad, se deben considerar ambos extremos de la relación

Asociación many-to-one (unidireccional)

Implementación Java




```
public class Puja {
    ...
   private Articulo articulo;

  public void setArticulo(Articulo articulo)
     { this.articulo = articulo; }

  public Articulo getArticulo()
    { return articulo; }
  ...
}
```

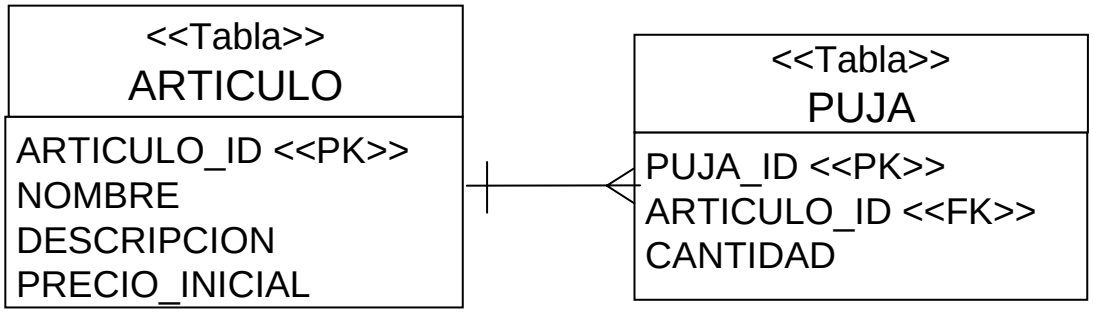
```
public class Articulo {
      private Long id;
      ...

      public Long getId()
        { return id; }

      public void setId(Long id)
        { this.id = id; }
     ...
}
```

Asociación many-to-one (unidireccional)

Mapeado Hibernate




```
<class name="Puja" table="PUJA">
    <id name="id" column="PUJA_ID"> <generator class="native"> </id>
     ...
    <many-to-one name="articulo" column="ARTICULO_ID" class="Articulo"
                 not-null="true"/>
</class>
```

```
<class name="Articulo" table="ARTICULO">
    <id name="id" column="ARTICULO_ID">
       <generator class="native"/> </id>
    ...
</class>
```

Asociación many-to-one (one-to-many) bidireccional

Implementación Java

```
public class Puja {
    ...
   private Articulo articulo;

  public void setArticulo(Articulo articulo)
     { this.articulo = articulo; }

  public Articulo getArticulo()
    { return articulo; }
  ...
}
```

```
public class Articulo {  ...
    private Set pujas = new HashSet();
    public void setPujas(Set pujas)
      { this.pujas = pujas; }
    public Set getPujas()
      { return pujas; }
    public void addPuja(Puja puja) {
      puja.setArticulo(this);
      pujas.add(puja);
    }
    ...
}
```

Asociación many-to-one (one-to-many) bidireccional

Mapeado Hibernate

```
<class name="Puja" table="PUJA">
    <id name="id" column="PUJA_ID">
    <generator class="native"> </id>
    ...
   <many-to-one name="articulo"
                column="ARTICULO_ID"
                class="Articulo"
                not-null="true"/>
</class>
```

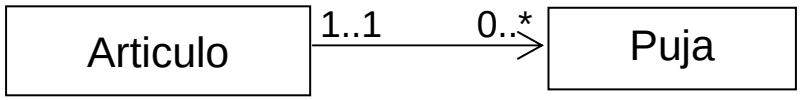
```
<class name="Articulo"
        table="ARTICULO">
    ...
   <set name="pujas"
        inverse="true"
        cascade="all-delete-orphan">
        <key column="ARTICULO_ID"/>
        <one-to-many class="Puja"/>
   <set/>
</class>
```

Asociaciones bidireccionales

- Todas las asociaciones bidireccionales necesitan que uno de sus extremos sea marcado como ***inverse***.
  - En many-to-one/one-to-many se marcará el extremo many
  - En many-to-many podemos elegir cualquiera
- Los cambios realizados solamente en el extremo de la asociación marcado como *inverse* **no** se convierten en persistentes

Asociación one-to-many (unidireccional)

Mapeado Hibernate




```
<class name="Articulo"
       table="ARTICULO">
  <id name="id"
      column="ARTICULO_ID">
  <generator class="native"/> </id>
  ...
  <set name="pujas">
     <key column="ARTICULO_ID"
          not-null="true"/>
     <one-to-many class="Puja"/>
  </set>
</class>
```

```
<class name="Puja" table="PUJA">
    <id name="id"
        column="PUJA_ID">
       <generator class="native">
    </id>
</class>
```

Asociación one-to-one (unidireccional)

Mapeado Hibernate

```
<class name="Direccion"
       table="DIRECCION">
    <id name="id"
        column="DIRECC_ID">
        <generator class="native"/>
    </id>
    <property name="calle"/>
    <property name="ciudad"/>
    <property name="codPostal"/>
</class>
```

```
<class name="Usuario"
       table="USUARIO">
    <id name="id" column="USER_ID">
    ...
    <many-to-one name="direccionFacturas"
          class="Direccion"
          column="DIRECC_ID"
          cascade="all"
          unique="true"/>
</class>
```

Asociación one-to-one (bidireccional)

Mapeado Hibernate

```
<class name="Direccion"
       table="DIRECCION">
    <id name="id"
        column="DIRECC_ID">
      <generator class="native">
    </id>
    <one-to-one name="usuario"
          class="Usuario"
          property-ref=
             "direccionFacturas"/>
    ...
</class>
```

```
<class name="Usuario"
       table="USUARIO">
    <id name="id" column="USER_ID">
    ...
    <many-to-one name="direccionFacturas"
          class="Direccion"
          column="DIRECC_ID"
          cascade="all"
          unique="true"/>
</class>
```

Asociación many-to-many (unidireccional)

Mapeado Hibernate

```
<class name="Categoria">
    <id name="id"
        column="CATEGORIA_ID">
      <generator class="native"/>
    </id>
    <set name="elementos"
         table="CategElement">
      <key column="CATEGORIA_ID"/>
      <many-to-many
        column="ELEMENTO_ID"
        class="Elemento"/>
    </set>
</class>
```

```
<class name="Elemento">
    <id name="id"
        column="ELEMENTO_ID">
     <generator class="native"/>
    </id>
</class>
```

Asociación many-to-many (bidireccional)

Mapeado Hibernate

```
<class name="Categoria">
    <id name="id"
        column="CATEGORIA_ID">
      <generator class="native"/>
    </id>
    <set name="elementos"
         table="CategElement">
      <key column="CATEGORIA_ID"/>
      <many-to-many
          column="ELEMENTO_ID"
          class="Elemento"/>
    </set>
</class>
```

```
<class name="Elemento">
    <id name="id"
        column="ELEMENTO_ID">
      <generator class="native"/>
    </id>
    <set name="categorias"
         inverse="true"
         table="CategElement">
      <key column="ELEMENTO_ID"/>
      <many-to-many
          column="CATEGORIA_ID"
          class="Categoria"/>
    </set>
</class>
```

```
cat.getElementos().add(elem);
elem.getCategorias().add(cat);
```

¿Preguntas...?