UA05. ORM

5.6.1. Uno a uno (unidireccional)

La relación uno a uno en Hibernate consiste simplemente en que un objeto tenga una referencia a otro objeto de forma que al persistirse el primer objeto también se persista el segundo.

En este caso, la relación va a ser unidireccional es decir que la relación **uno a uno** va a ser en un único sentido.

Ejemplo

Supongamos dos tablas, **Alumnado** y **Dirección**, donde se recogen los datos del alumnado y sus domicilios.

Estas dos clases van a tener una relación uno a uno.

```
public class Alumnado implements Serializable  {
     private int idAlum;
     private String nombre;
     private String ape1;
     private String ape2;
     private Direccion direccion;

     public Alumnado(){
     }
```

```
    public Alumnado(int idAlum, String nombre, String ape1, String a
        this.idAlum = idAlum;
        this.nombre = nombre;
        this.ape1 = ape1;
        this.ape2 = ape2;
    }
}
```

```
public class Direccion implements Serializable  {
    private int idAlum;
    private String calle;
    private int numero;
    private String poblacion;
    private String provincia;

    public Direccion(){
    }

    public Direccion(int idAlum, String calle, int numero, String pc
        this.idAlum = idAlum;
        this.calle = calle;
        this.numero = numero;
        this.poblacion = poblacion;
        this.provincia = provincia;
    }
}
```

Observa que **Alumnado** tiene una variable de tipo **Dirección** mientras que la clase **Direccion** no posee ninguna referencia a **Alumnado** ya que hemos definido una direccionalidad desde **Alumnado hacia Direccion** pero no al revés.

Anotaciones

Al ser en un único sentido, el código fuente de la clase **Alumnado** quedará así:

```
import java.io.Serializable;
import javax.persistence.*;
@Entity
@Table(name="Alumnado")
public class Alumnado implements Serializable  {
    @Id
    @Column(name="IdAlum")
    private int idAlum;

    @Column(name="nombre")
    private String nombre;

    @Column(name="ape1")
    private String ape1;

    @Column(name="ape2")
    private String ape2;

    @OneToOne(cascade=CascadeType.ALL)
    @PrimaryKeyJoinColumn
    private Direccion direccion;
}
```

```
import java.io.Serializable;
import javax.persistence.*;
@Entity
@Table(name="Direccion")
public class Direccion implements Serializable {
    @Id
```

```
    @Column(name = "idAlum")
    private int idAlum;
    @Column(name="calle")
    private String calle;
    @Column(name="numero")
    private int numero;
    @Column(name="poblacion")
    private String poblacion;
    @Column(name="provincia")
    private String provincia;
```

A la propiedad **direccion** se han añadido dos anotaciones para indicar la relación **uno a uno** y que ésta relación se implemente mediante la clave primaria.

- **@OneToOne(cascade=CascadeType.ALL):** Esta anotación indica la relación uno a uno de las 2 tablas. Tambien indicamos el valor de cascade al igual que en el fichero de hibernate.
- **@PrimaryKeyJoinColumn:** Indicamos que la relación entre las dos tablas se realiza mediante la clave primaria.

En el código de **Direccion** no es necesario indicar nada.

Código necesario

Ahora que ya tenemos preparadas las clase Java para que puedan persistirse veamos el código necesario para persistirlas.

```
Direccion direccion=new Direccion(1, "Plaza del ayuntamiento", 8, "Xa
Alumnado alumnado=new Alumnado(1, "Juan", "Perez", "García");
alumnado.setDireccion(direccion);

Session session=sessionFactory.openSession();
session.beginTransaction();
```

```
    session.save(alumnado);

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

Como podemos ver no hay nada nuevo en el código Java para persistir una relación **uno a uno**, simplemente creamos las 2 clases (Líneas 1 y 2) y establecemos la relación entre ambas asignando al objeto **Alumnado** la referencia al objeto **Direccion** (Línea 3). Por último, simplemente persistimos la clase **Alumnado** tal y como se ha explicado anteriormente.

Al ejecutar el ejemplo Hibernate vemos cómo se han creado las filas en las tablas **Alumnado** y **Direccion** mientras que desde Java sólo se ha persistido la clase **Alumnado**.

Si vemos el **log** que se genera al persistir los 2 objetos, podemos ver que se realiza primero una orden **SELECT** contra la tabla **Direccion** para comprobar si ya existe la dirección en la base de datos. Ésto lo realiza **hibernate** ya que, si ya existe, no es necesario insertar la fila de la dirección pero, si los datos son distintos, hibernate lanzará un **UPDATE** para modificarlos.

La clave primaria de **Direccion** y **Alumnado** debe ser la misma para que se establezca correctamente la relación.