UA05. ORM

5.6.3. Uno a muchos

La relación **uno a muchos** consiste simplemente en que un objeto **padre** tenga una lista **Ordenar** de otros objetos **hijo** de forma que al persistirse el objeto principal también se persista la lista de objetos **hijo**. Esta relación también suele llamarse **maestro-detalle** o **padre-hijo**.

Para el ejemplo vamos a suponer las tablas: **Alumnado** y **Email**

Usaremos el código de Alumnado anterior añadiendo la siguiente variable:

```
@OneToMany(cascade= CascadeType.ALL)
@JoinColumn(name="idAlum")
private List<Email> listaEmails;
```

El código de la clase **Email** será el siguiente:

```
@Entity
@Table(name = "Email")
public class Email implements Serializable {
    @Id
    @Column(name = "idEmail")
    private int idEmail;
    @Column(name = "email")
     private String email;
    @ManyToOne
```

```
    @JoinColumn(name="idAlum")
    private Alumnado alumnado;
```

En la clase **Alumnado**, a la variable **listaEmail** se ha añadido una anotación para indicar la relación **uno a muchos**.

- **OneToMany**: Como su nombre indica le dice a Hibernate que esta propiedad contendrá la lista de hijos.
- **cascade:** Actualizacion y borrado en cascada.
- **JoinColumn:** Indicaremos el nombre de la columna que en la tabla hija contiene la clave ajena a la tabla padre. En nuestro ejemplo es la columna de la base de datos **idAlum** que se encuentra en la tabla **Email** la cual enlaza con la tabla **Alumnado**.

A la variable **alumnado** se han añadido dos anotaciones para indicar la relación:

- **ManyToOne:** Al ser el otro lado de la relación indicamos que desde este lado es una relación mucho a uno.
- **JoinColumn:** Indicaremos el nombre de la columna que en la tabla hija contiene la clave ajena a la tabla padre. En nuestro ejemplo es la columna de la base de datos **IdAlum** que se encuentra en la tabla **Email** la cual enlaza con la tabla **Alumnado**.

Para realizar una prueba, usaremos el código:

```
Alumnado alumnado=new Alumnado(9, "Rosa", "Díaz", "Del Toro");
List<Email> email=new ArrayList<>();
email.add(new Email(3, "rosa@yahoo.com",alumnado));
email.add(new Email(2, "rosa@hotmail.com",alumnado));
email.add(new Email(1, "rosa@gmail.com",alumnado));

alumnado.setEmails(email);
```

1. En la línea 1 se crea el objeto **Alumnado**.

2. En la segunda línea se crea el objeto **ArrayList** que implementa el interfaz **List** el cual contendrá la lista de hijos.
3. Desde las líneas 3 a la 5 se crean los objetos **Email** y se añaden al **List**.
4. En la línea 7 se establece la relación entre la lista de hijos (**Email**) y el padre (**Alumnado**).
5. Finalmente, y no mostrado en este código, se guarda el objeto **Alumnado** y, automáticamente, se guardarán también sus hijos.