Soluciones de los ejercicios de Spring Boot Data JPA

Los ejercicios resueltos se han desarrollado en un único proyecto. Dentro del proyecto se podrán observar comentarios para distinguir qué ejercicio corresponde a cada uno.

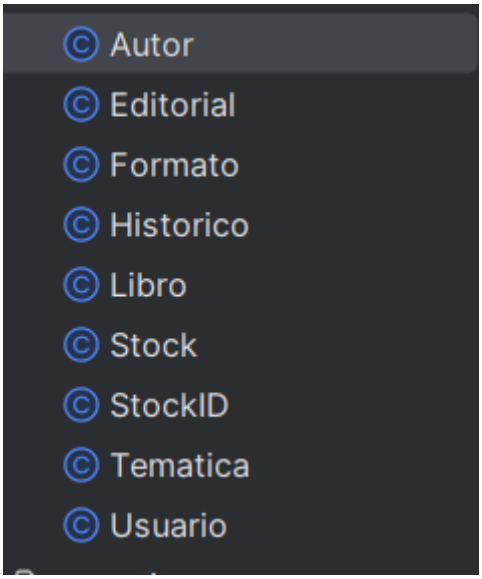
Ejercicio 1: Conexión y configuración de nuestro fichero *properties*.

En el fichero properties configuramos la conexión a la base de datos MySQL y la configuración de Hibernate.

```
spring.application.name=Spring-JPA
spring.datasource.url=jdbc:mysql://localhost:3306/biblioteca
spring.datasource.username=root


#Configuración de hibernate
# Keep the connection alive if idle for a long time (needed in production)
spring.datasource.testWhileIdle = true
spring.datasource.validationQuery = SELECT 1


spring.jpa.show-sql = true

# Hibernate ddl auto (create, create-drop, update)
spring.jpa.hibernate.ddl-auto = update

# Naming strategy
spring.jpa.hibernate.naming-strategy = org.hibernate.cfg.ImprovedNamingStrategy

# The SQL dialect makes Hibernate generate better SQL for the chosen database
spring.jpa.properties.hibernate.dialect = org.hibernate.dialect.MySQLDialect
```

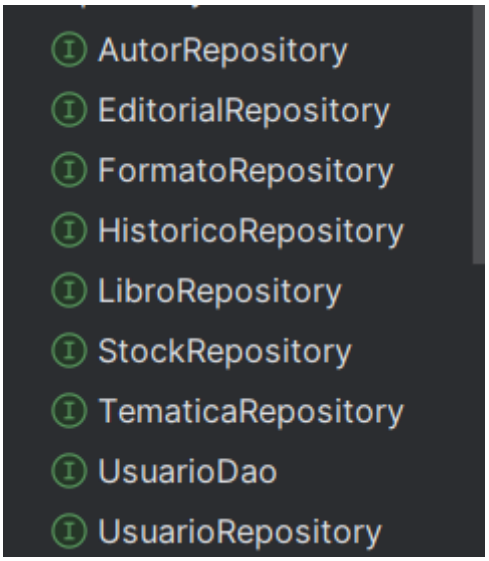
Ejercicio 2: Creación o mapeo de todas las tablas de la base de datos biblioteca.

Para el ejercicito dos creamos todas las entidades para crear o mapear la base de datos de la biblioteca:

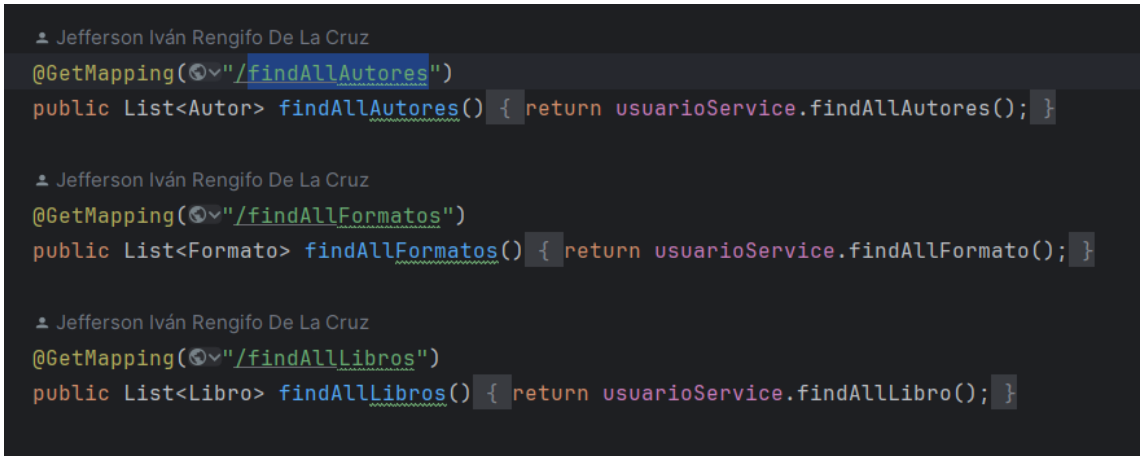


Ejercicio 3: Definición de las interfaces JPA para cada una de las entidades.

Para cada una de las entidades creamos las interfaces de tipo **JPARepository**:

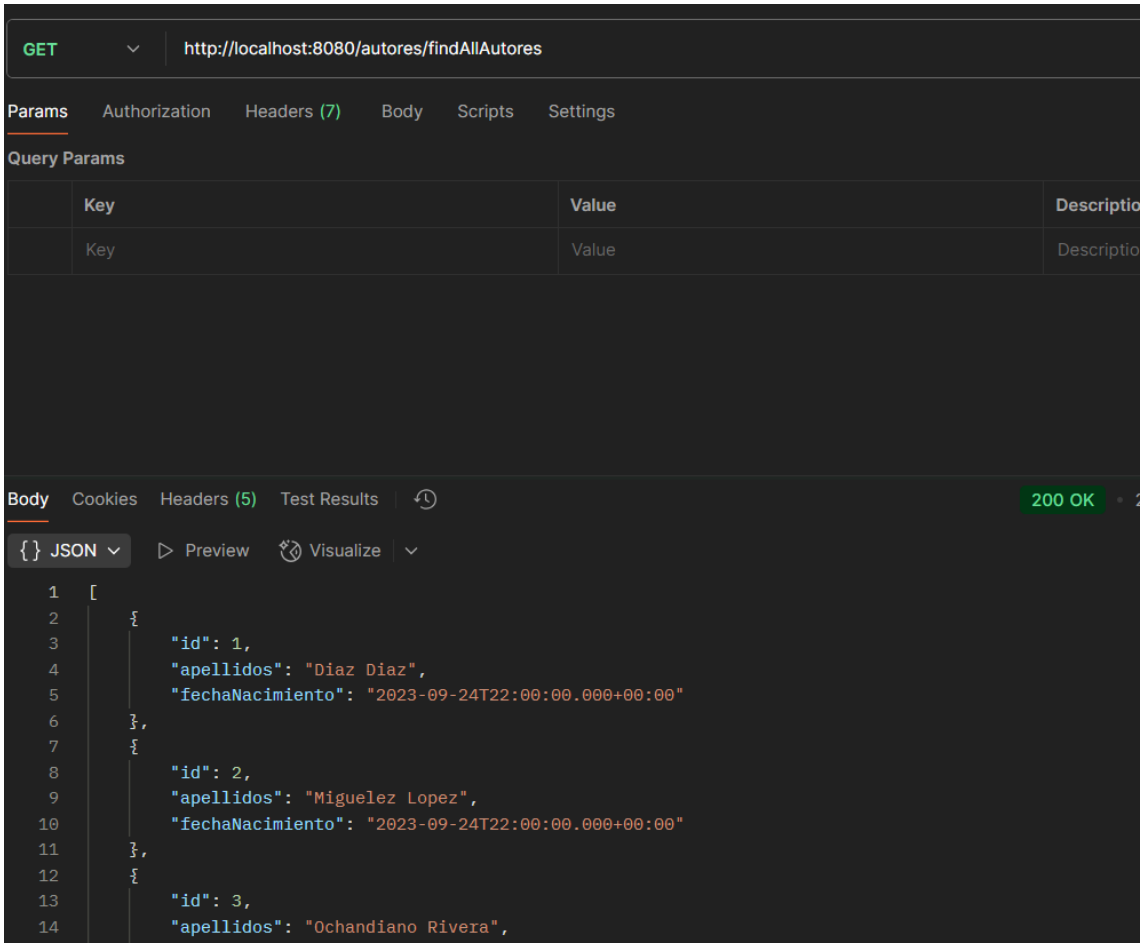


En la siguiente imagen se puede apreciar las consultas por defecto de JPA:



```
Jefferson Iván Rengifo De La Cruz
@GetMapping("/findAllAutores")
public List<Autor> findAllAutores() { return usuarioService.findAllAutores(); }

Jefferson Iván Rengifo De La Cruz
@GetMapping("/findAllFormatos")
public List<Formato> findAllFormatos() { return usuarioService.findAllFormato(); }

Jefferson Iván Rengifo De La Cruz
@GetMapping("/findAllLibros")
public List<Libro> findAllLibros() { return usuarioService.findAllLibro(); }
```

Y el resultado en postman es el siguiente:



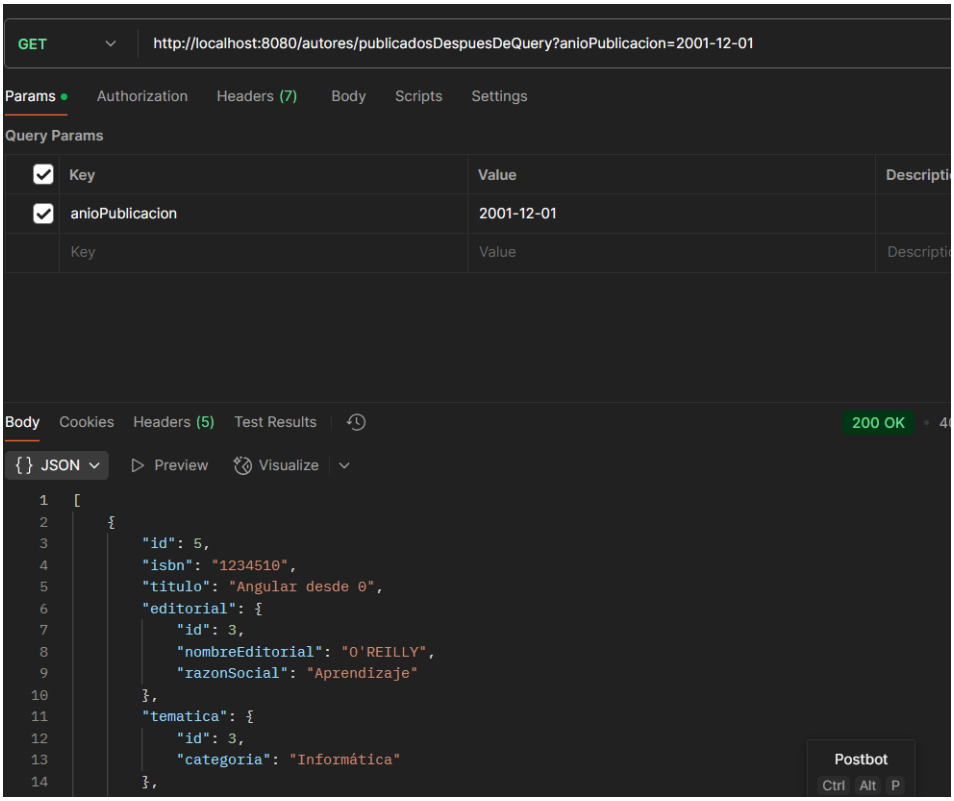
```
GET  http://localhost:8080/autores/findAllAutores

Params  Authorization  Headers (7)  Body  Scripts  Settings

Query Params

| Key | Value | Descriptio |
| Key | Value | Descriptio |

Body  Cookies  Headers (5)  Test Results  200 OK

{ } JSON  Preview  Visualize

[
    {
        "id": 1,
        "apellidos": "Diaz Diaz",
        "fechaNacimiento": "2023-09-24T22:00:00.000+00:00"
    },
    {
        "id": 2,
        "apellidos": "Miguelez Lopez",
        "fechaNacimiento": "2023-09-24T22:00:00.000+00:00"
    },
    {
        "id": 3,
        "apellidos": "Ochandiano Rivera",
```

Ejercicio 4: Creación de la primera Query en la interfaz JPA.

```java
2 usages
@Repository
public interface LibroRepository extends JpaRepository<Libro, Long> {


    //Consulta ejercicio 4
    1 usage
    @Query("SELECT l FROM Libro l WHERE l.anioPublicacion > :fecha")
    List<Libro> findLibrosPublicadosDespuesDe(@Param("fecha") Date fecha);
```

Creamos la Query para obtener los libros cuya fecha de publicación es mayor que 01/12/2001

Y cómo resultado obtenemos los siguientes datos en postman:



La petición está en el controller:

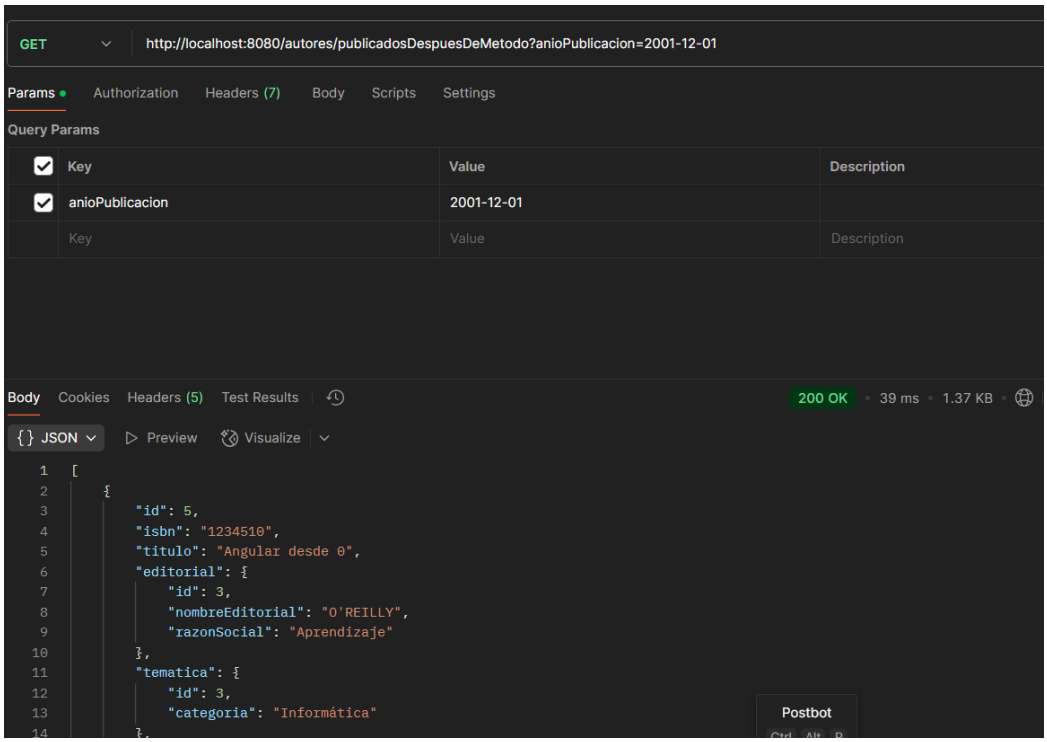
```java
//Ejercicio 4
Jefferson Iván Rengifo De La Cruz *
@GetMapping("/publicadosDespuesDeQuery")
public List<?> obtenerLibrosPublicadosDespuesDe(@Valid @RequestParam @DateTimeFormat(pattern = "yyyy-MM-dd") Date anioPu
    return usuarioService.obtenerLibrosPublicadosDespuesDeQuery(anioPublicacion);
}
```

Ejercicio 5: Creación de la notación basada en nombre del método.

```
//Consulta Ejercicio 5 con método
1 usage
List<Libro> findByAnioPublicacionAfter(Date fecha);
```

Con la notación basada en nombre del método no hace falta la Query para obtener el mismo resultado del ejercicio 4.

El resultado en postman es el siguiente:



En el controller está la petición:

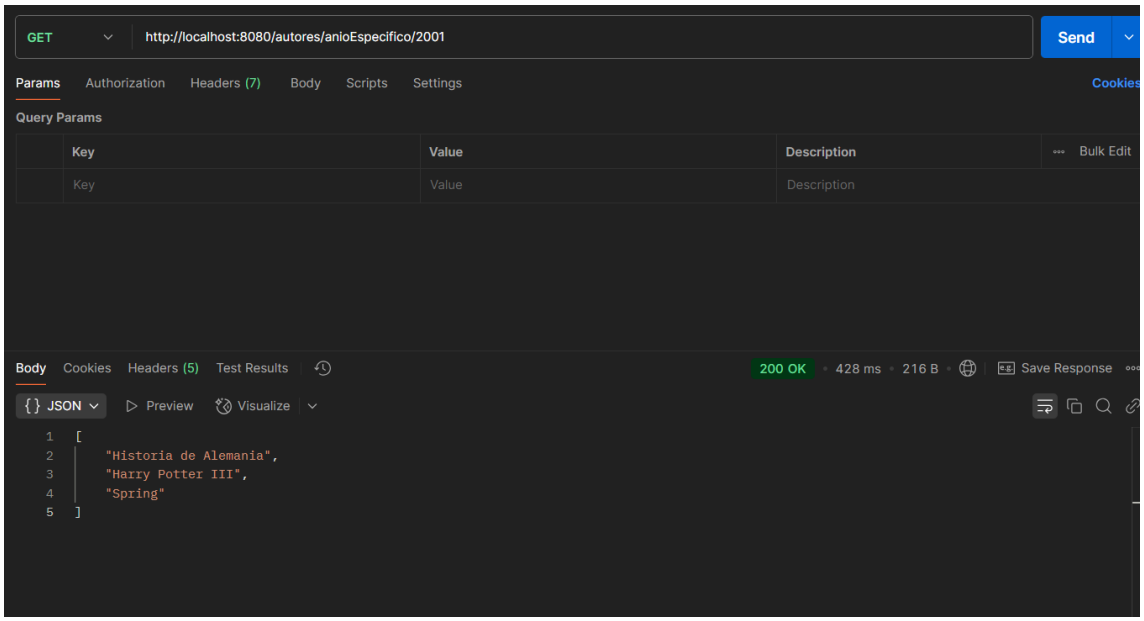
```
//Ejercicio 5
Jefferson Iván Rengifo De La Cruz *
@GetMapping("/publicadosDespuesDeMetodo")
public List<?> obtenerLibrosPublicadosDespuesDeMetodo(@Valid @RequestParam @DateTimeFormat(pattern = "yyyy-MM-dd") Date
    return usuarioService.obtenerLibrosPublicadosDespuesDeMetodo(anioPublicacion);
}
```

Ejercicio 6: Consultas desde JPA a nuestra base de datos

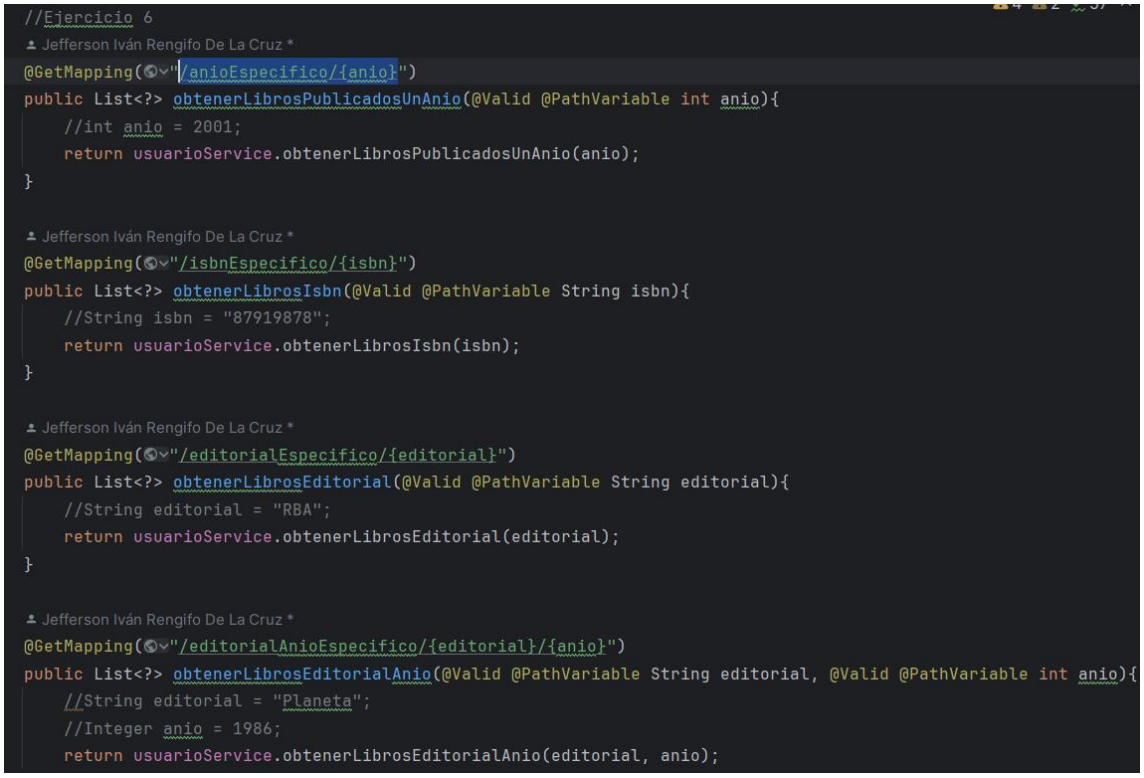
Para el ejercicio 6 las Queries creadas son de la siguiente forma:

```java
//Consultas del Ejercicio 6
//Buscar libros según un año en específico
1 usage
@Query("SELECT l FROM Libro l where year(l.anioPublicacion)=:anio")
List<Libro> findLibrosPublicadosUnAnio(@Param("anio") int anio);

//Buscar libros según un isb en específico
1 usage
@Query("SELECT l FROM Libro l where l.isbn=:isbn")
List<Libro> findLibrosIsbn(@Param("isbn") String isbn);

//Buscar libros según una editorial(RBA) en específico
1 usage
@Query("SELECT l FROM Libro l JOIN Editorial e on l.editorial.id = e.id AND e.nombreEditorial=:nombreEditorial")
List<Libro> findLibrosEditorial(@Param("nombreEditorial") String nombreEditorial);

//Buscar libros según una editorial(RBA) y año(1986) en específico
1 usage
@Query("SELECT l FROM Libro l WHERE l.editorial.nombreEditorial = :nombreEditorial " +
        "AND YEAR(l.anioPublicacion) = :anio")
List<Libro> findLibrosEditorialAnio(@Param("nombreEditorial") String nombreEditorial,@Param("anio") Integer anio);
```

Estas Queries mostrarán los siguientes resultados en postman:



Las peticiones se recogen en el controller:



```
//Ejercicio 6
Jefferson Iván Rengifo De La Cruz *
@GetMapping("/anioEspecifico/{anio}")
public List<?> obtenerLibrosPublicadosUnAnio(@Valid @PathVariable int anio){
    //int anio = 2001;
    return usuarioService.obtenerLibrosPublicadosUnAnio(anio);
}

Jefferson Iván Rengifo De La Cruz *
@GetMapping("/isbnEspecifico/{isbn}")
public List<?> obtenerLibrosIsbn(@Valid @PathVariable String isbn){
    //String isbn = "87919878";
    return usuarioService.obtenerLibrosIsbn(isbn);
}

Jefferson Iván Rengifo De La Cruz *
@GetMapping("/editorialEspecifico/{editorial}")
public List<?> obtenerLibrosEditorial(@Valid @PathVariable String editorial){
    //String editorial = "RBA";
    return usuarioService.obtenerLibrosEditorial(editorial);
}

Jefferson Iván Rengifo De La Cruz *
@GetMapping("/editorialAnioEspecifico/{editorial}/{anio}")
public List<?> obtenerLibrosEditorialAnio(@Valid @PathVariable String editorial, @Valid @PathVariable int anio){
    //String editorial = "Planeta";
    //Integer anio = 1986;
    return usuarioService.obtenerLibrosEditorialAnio(editorial, anio);
```