¿Qué es una JPA Query o consulta de JPA ? .¿Cómo funciona ? . El uso de JPA para realizar consultas a una base de datos entidad relación es de las cosas más habituales que nos podemos encontrar en nuestro día a día . Sin embargo cuando la gente comienza a manejar JPA siempre aparecen algunas dudas en cuanto a este concepto básico. Vamos a ver un ejemplo que nos ayude a clarificar las cosas. Para ello vamos a empezar viendo el contenido de la entidad que queremos consultar de la base de datos.

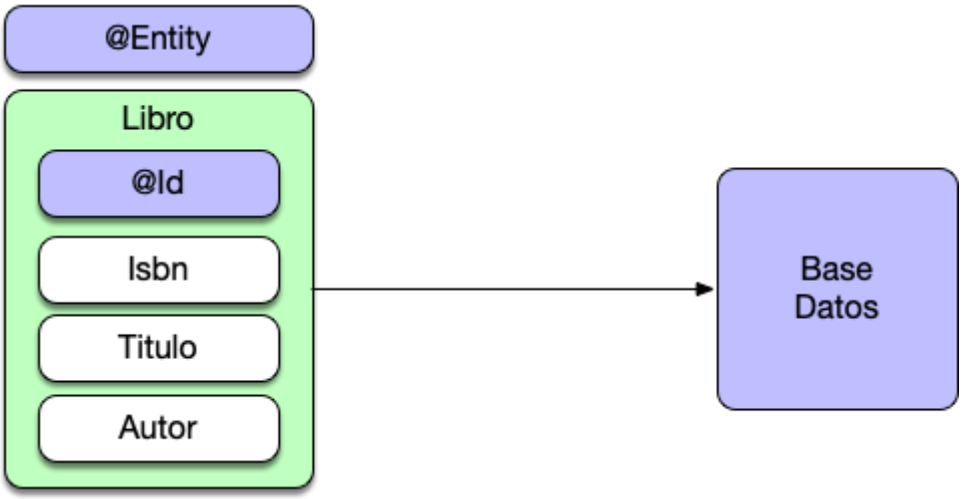
```
package com.arquitecturajava.dominio;

import java.util.Date;

import javax.persistence.Entity;
import javax.persistence.Id;
import javax.persistence.Table;

@Entity
@Table(name = "libros")
public class Libro {

	@Id
	private String isbn;
	private String titulo;
	private String autor;
	private Date fecha;
	private int precio;
	public String getIsbn() {
		return isbn;
	}
	public void setIsbn(String isbn) {
		this.isbn = isbn;
	}
```

```
	public String getTitulo() {
		return titulo;
	}
	public void setTitulo(String titulo) {
		this.titulo = titulo;
	}
	public String getAutor() {
		return autor;
	}
	public void setAutor(String autor) {
		this.autor = autor;
	}
	public Date getFecha() {
		return fecha;
	}
	public void setFecha(Date fecha) {
		this.fecha = fecha;
	}
	public int getPrecio() {
		return precio;
	}
	public void setPrecio(int precio) {
		this.precio = precio;
	}
	public Libro(String isbn, String titulo, String autor, Date
fecha, int precio) {
		super();
		this.isbn = isbn;
		this.titulo = titulo;
		this.autor = autor;
		this.fecha = fecha;
```

```
                this.precio = precio;
        }
        public Libro() {
                super();
        }
        public Libro(String isbn) {
                super();
                this.isbn = isbn;
        }
        @Override
        public int hashCode() {
                final int prime = 31;
                int result = 1;
                result = prime * result + ((isbn == null) ? 0 :
isbn.hashCode());
                return result;
        }
        @Override
        public boolean equals(Object obj) {
                if (this == obj)
                        return true;
                if (obj == null)
                        return false;
                if (getClass() != obj.getClass())
                        return false;
                Libro other = (Libro) obj;
                if (isbn == null) {
                        if (other.isbn != null)
                                return false;
                } else if (!isbn.equals(other.isbn))
                        return false;
```

```
                return true;
        }
        @Override
        public String toString() {
                return "Libro [isbn=" + isbn + ", titulo=" + titulo +
", autor=" + autor + ", fecha=" + fecha + ", precio="
                                + precio + "]";
        }
}
```

Esta entidad se encarga de persistir los objetos libro en la base de datos .Ahora bien para ello necesita hacer uso de algunas anotaciones . En primer lugar disonemos de las anotaciones @Entity y @Id . La anotación @Entity define que la entidad es persistible contra la base de datos y la anotación @Id identifica cual es la clave primaria de la tabla.



Es momento de ver el contenido del persistence.xml que es el fichero que nos conecta con la base de datos.

```
<?xml version="1.0" encoding="UTF-8"?>
<persistence version="2.1"
    xmlns="http://xmlns.jcp.org/xml/ns/persistence"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
```

```
    xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/persistence
http://xmlns.jcp.org/xml/ns/persistence/persistence_2_1.xsd">
    <persistence-unit name="biblioteca" transaction-
type="RESOURCE_LOCAL">
        <class>com.arquitecturajava.dominio.Libro</class>
        <properties>
                <property name= "hibernate.show_sql" value="true" />
                <!-- cuando nos conectamos desde JPA hay  q elegir el
tipo de motor -->
                        <property name="hibernate.dialect"
value="org.hibernate.dialect.MySQLDialect" />
            <property name="javax.persistence.jdbc.driver"
value="com.mysql.jdbc.Driver" />
            <property name="javax.persistence.jdbc.url"
value="jdbc:mysql://localhost/biblioteca" />
            <property name="javax.persistence.jdbc.user" value="root"
/>
            <property name="javax.persistence.jdbc.password" value=""
/>
        </properties>
    </persistence-unit>
</persistence>
```

En este caso como se puede observar simplemente configuramos la conexión a la base de datos y definimos el motor de SQL que vamos a utilizar (mysql).

```
<property name="hibernate.dialect"
value="org.hibernate.dialect.MySQLDialect" />
```

JPA Query y su uso

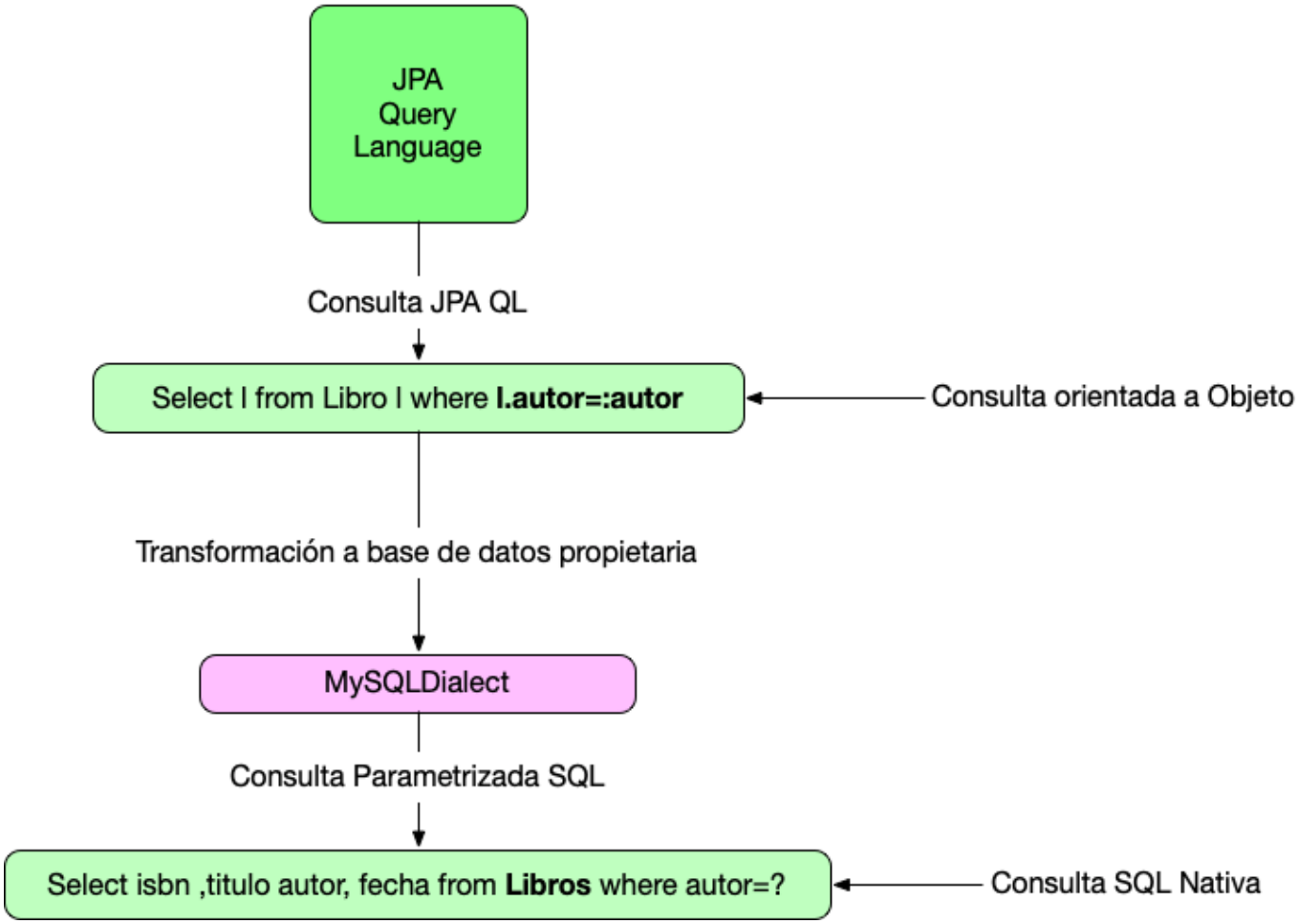
La consulta más sencilla que se puede hacer es la de buscar un Libro:

```
package com.arquitecturajava.main;

import java.util.List;

import javax.persistence.EntityManager;
import javax.persistence.EntityManagerFactory;
import javax.persistence.Persistence;
import javax.persistence.TypedQuery;

import com.arquitecturajava.dominio.Libro;

public class Principal001Where {

	public static void main(String[] args) {
		// unidad de persistencia
		EntityManagerFactory emf=
Persistence.createEntityManagerFactory("biblioteca");
		EntityManager em= emf.createEntityManager();
		// dise□o una primera consulta orientada a objeto con
where
		TypedQuery<Libro> consulta=em.createQuery("select l
from Libro l where l.autor=:autor", Libro.class);
		consulta.setParameter("autor", "cecilio");
		List<Libro> lista=consulta.getResultList();
		for (Libro unLibro:lista) {
			System.out.println(unLibro);
		}

	}

}
```

En este caso estamos buscando un Libro por Autor . Muchas personas cuando empiezan con JPA se pierden un poco ya que cuesta entender como Java Persistence API construye la consulta. En este caso la consulta es :

```
"select l from Libro l where l.autor=:autor"
```

Parece una consulta SQL pero no lo es , se trata de una consulta JPA QL (Java Persistence Query Language) . Es un lenguaje de Consulta orientado a objeto que permite generar neutralidad entre las diferentes bases de datos. JPA se encargará de transformar estos datos en una consulta nativa de SQL.



Para realizar esa transformación se apoyará en el dialecto de JPA seleccionado en este caso MySQLDialect. De esta forma es como un framework de Persistencia funciona.

Entidades vs Tablas

Muchas personas me preguntan si es correcto poner :

```
"select l from Libro l where l.autor=:autor"
```

Cuando la tabla en la base de datos se denomina Libros . Si que es correcta ya que la consulta de JPA nunca hace referencia a una tabla de la base de datos sino a la Entidad del dominio que en nuestro caso de denomina “Libro” por lo tanto es lo que tenemos que poner. Eso sí el framework se encargará del mapeo de forma automática ya que la tabla en la base de datos se denomina Libros . Este mapeo lo realiza la anotación @Table de la entidad

```
@Entity
@Table(name = "libros")
public class Libro {
...
}
```

- ¿JPA vs Hibernate?
- Ejemplo de JPA (Java Persistence API)
- JPA Orphan Removal y como usarlo
- Java Persistence API
- Spring Boot JPA