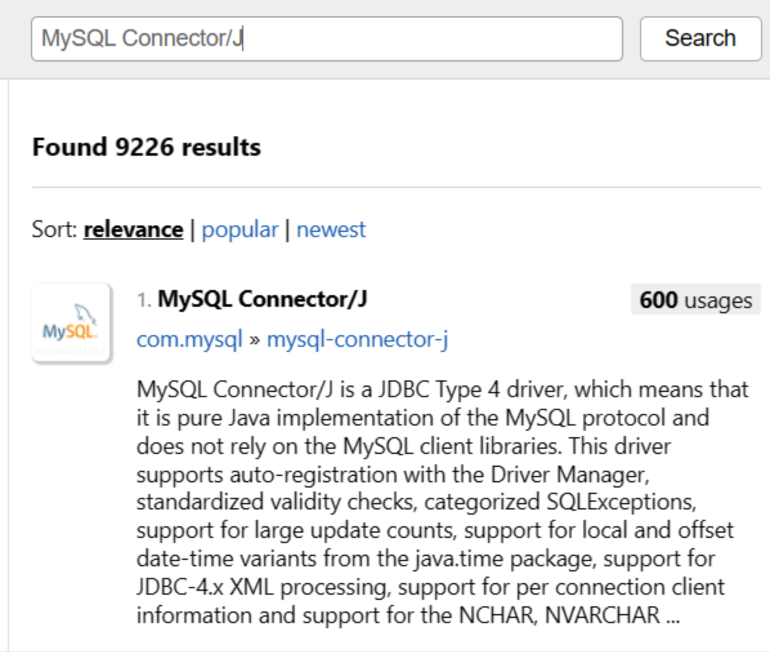
JPA PERSISTENCE

DEPENDENCIAS



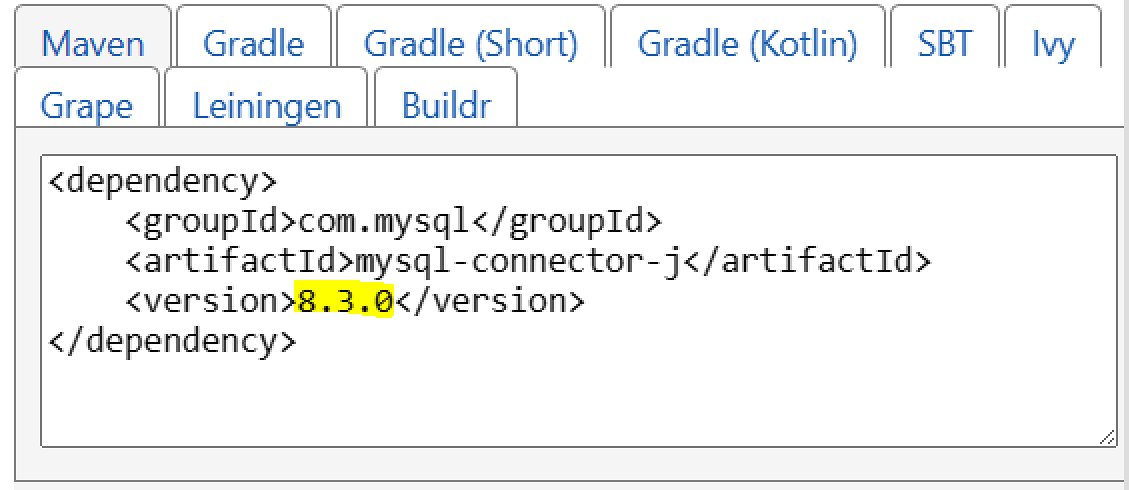
Found 9226 results

Sort: **relevance** | popular | newest

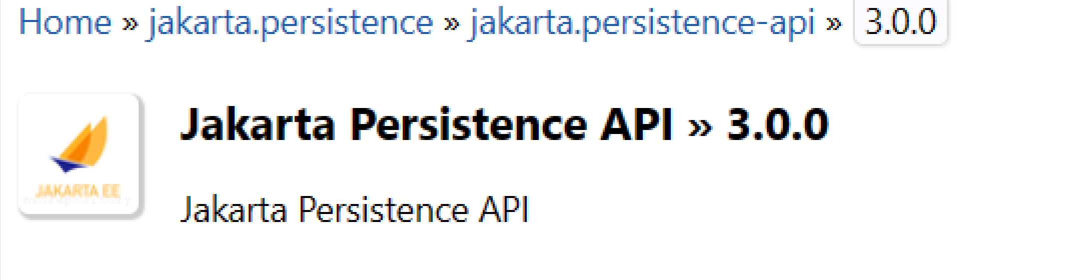
1. **MySQL Connector/J** **600** usages

com.mysql » mysql-connector-j

MySQL Connector/J is a JDBC Type 4 driver, which means that it is pure Java implementation of the MySQL protocol and does not rely on the MySQL client libraries. This driver supports auto-registration with the Driver Manager, standardized validity checks, categorized SQLExceptions, support for large update counts, support for local and offset date-time variants from the java.time package, support for JDBC-4.x XML processing, support for per connection client information and support for the NCHAR, NVARCHAR ...



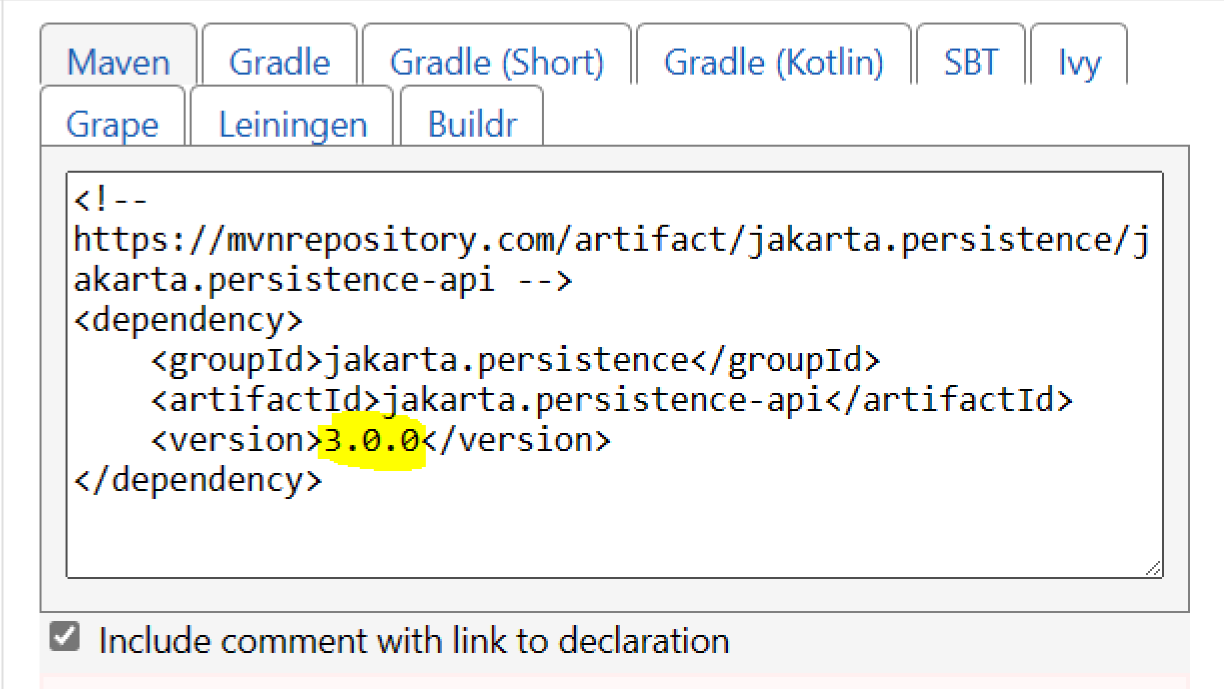
```
<dependency>
    <groupId>com.mysql</groupId>
    <artifactId>mysql-connector-j</artifactId>
    <version>8.3.0</version>
</dependency>
```



Home » jakarta.persistence » jakarta.persistence-api » 3.0.0

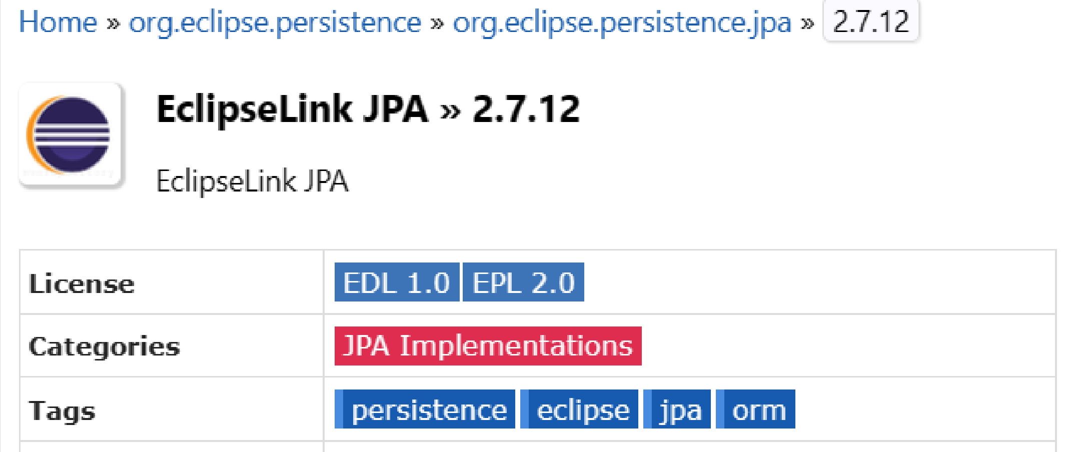
Jakarta Persistence API » 3.0.0

Jakarta Persistence API



```
<!--
https://mvnrepository.com/artifact/jakarta.persistence/j
akarta.persistence-api -->
<dependency>
    <groupId>jakarta.persistence</groupId>
    <artifactId>jakarta.persistence-api</artifactId>
    <version>3.0.0</version>
</dependency>
```

☑ Include comment with link to declaration

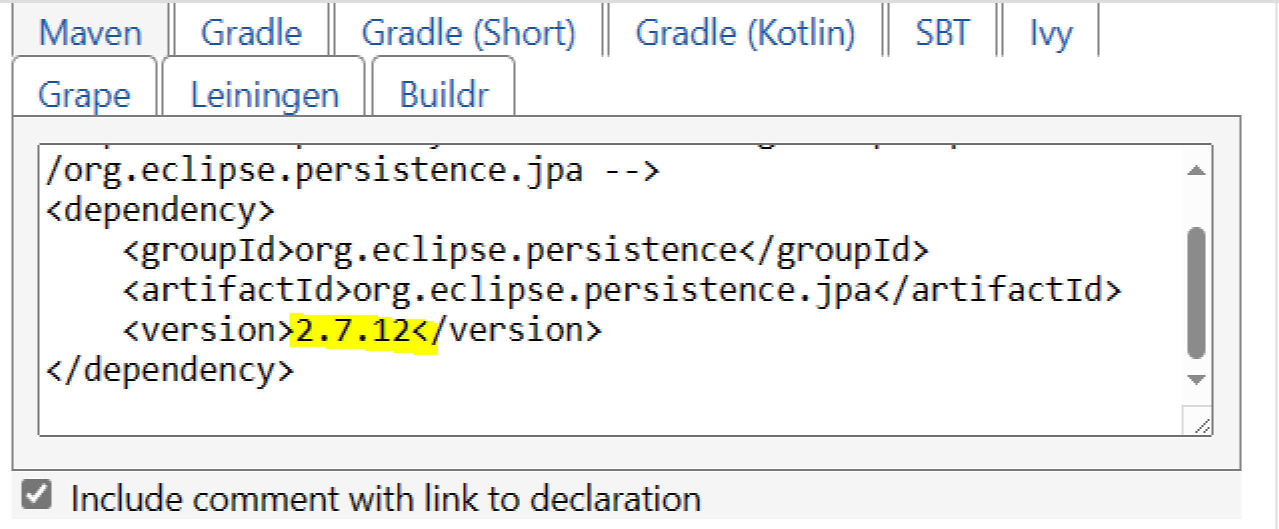


Home » org.eclipse.persistence » org.eclipse.persistence.jpa » 2.7.12

EclipseLink JPA » 2.7.12

EclipseLink JPA

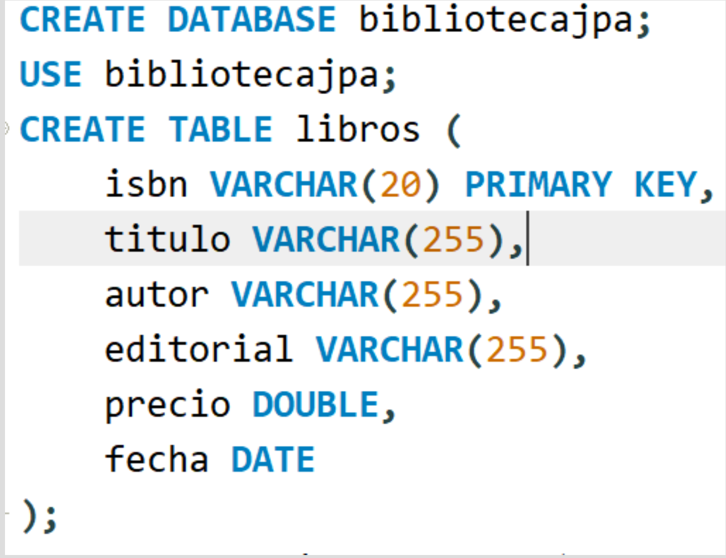
License	EDL 1.0 EPL 2.0
Categories	JPA Implementations
Tags	persistence eclipse jpa orm



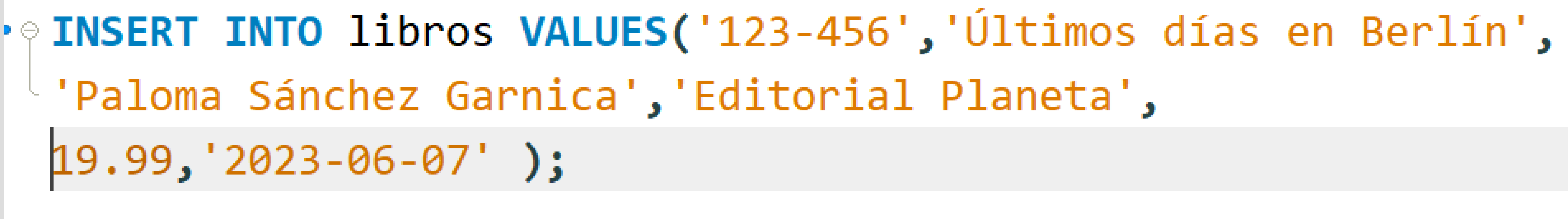
```
/org.eclipse.persistence.jpa -->
<dependency>
    <groupId>org.eclipse.persistence</groupId>
    <artifactId>org.eclipse.persistence.jpa</artifactId>
    <version>2.7.12</version>
</dependency>
```

☑ Include comment with link to declaration

BBDD

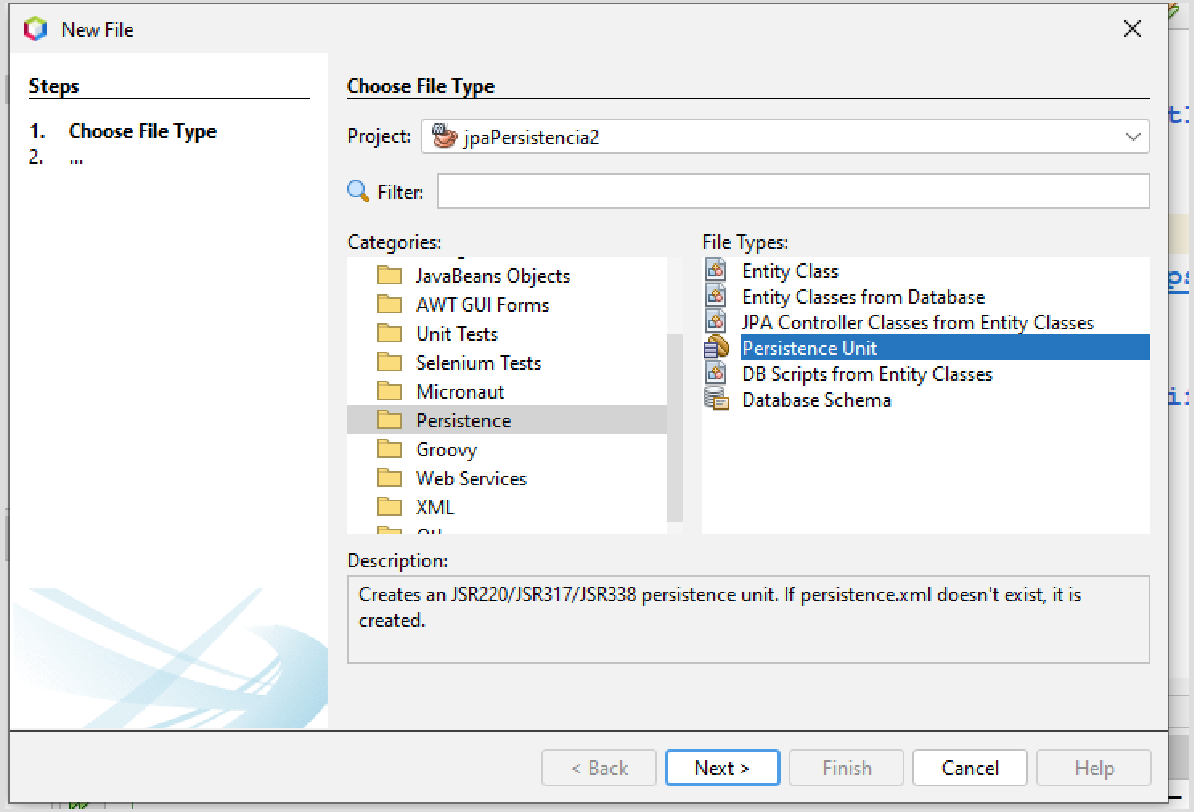


```sql
CREATE DATABASE bibliotecajpa;
USE bibliotecajpa;
CREATE TABLE libros (
    isbn VARCHAR(20) PRIMARY KEY,
    titulo VARCHAR(255),
    autor VARCHAR(255),
    editorial VARCHAR(255),
    precio DOUBLE,
    fecha DATE
);
```



```sql
INSERT INTO libros VALUES('123-456','Últimos días en Berlín',
'Paloma Sánchez Garnica','Editorial Planeta',
19.99,'2023-06-07' );
```

CREACIÓN CONEXIÓN DESDE JAVA PERSISTENCE.XML



New File

Steps

1. Choose File Type
2. ...

Choose File Type

Project: jpaPersistencia2

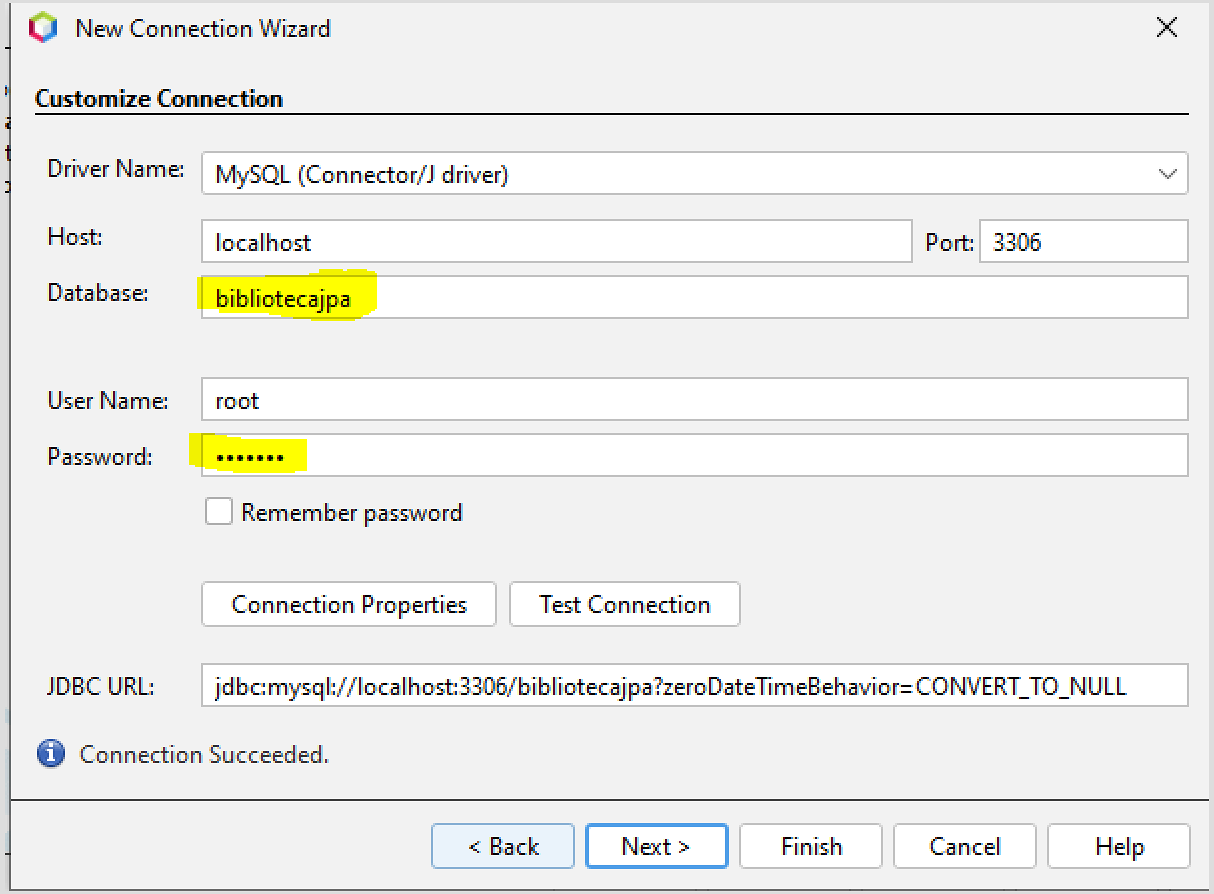
Filter:

Categories:
- JavaBeans Objects
- AWT GUI Forms
- Unit Tests
- Selenium Tests
- Micronaut
- Persistence
- Groovy
- Web Services
- XML

File Types:
- Entity Class
- Entity Classes from Database
- JPA Controller Classes from Entity Classes
- Persistence Unit
- DB Scripts from Entity Classes
- Database Schema

Description:

Creates an JSR220/JSR317/JSR338 persistence unit. If persistence.xml doesn't exist, it is created.



New Connection Wizard

Customize Connection

Driver Name: MySQL (Connector/J driver)

Host: localhost

Port: 3306

Database: bibliotecajpa

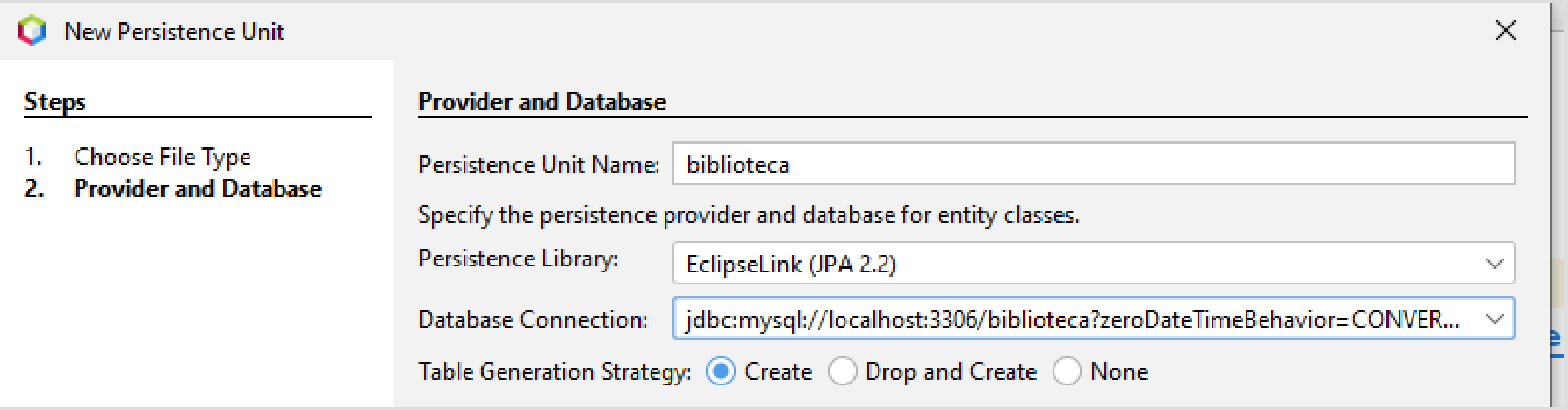
User Name: root

Password: •••••••

□ Remember password

JDBC URL: jdbc:mysql://localhost:3306/bibliotecajpa?zeroDateTimeBehavior=CONVERT_TO_NULL

Connection Succeeded.



New Persistence Unit

Steps

1. Choose File Type
2. **Provider and Database**

Provider and Database

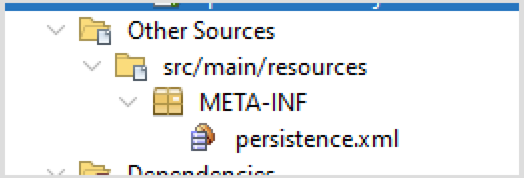
Persistence Unit Name: biblioteca

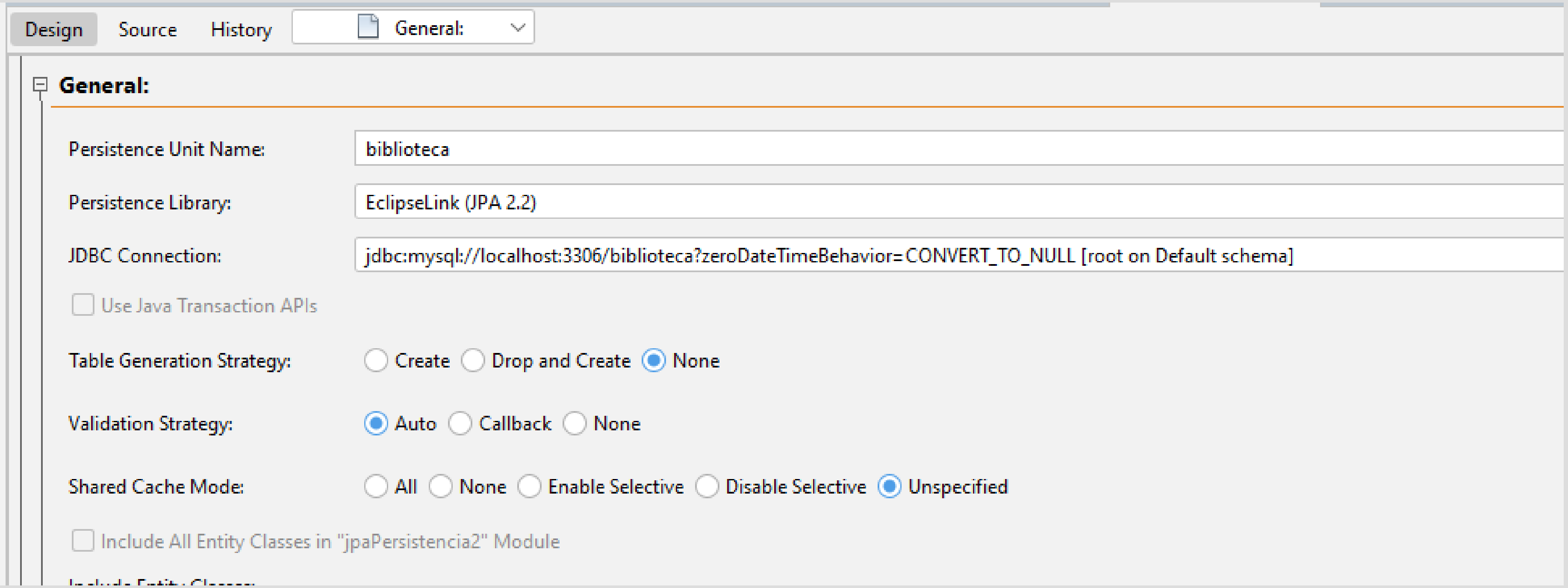
Specify the persistence provider and database for entity classes.

Persistence Library: EclipseLink (JPA 2.2)

Database Connection: jdbc:mysql://localhost:3306/biblioteca?zeroDateTimeBehavior=CONVER...

Table Generation Strategy: ● Create ○ Drop and Create ○ None





General:

Persistence Unit Name: biblioteca

Persistence Library: EclipseLink (JPA 2.2)

JDBC Connection: jdbc:mysql://localhost:3306/biblioteca?zeroDateTimeBehavior=CONVERT_TO_NULL [root on Default schema]

□ Use Java Transaction APIs

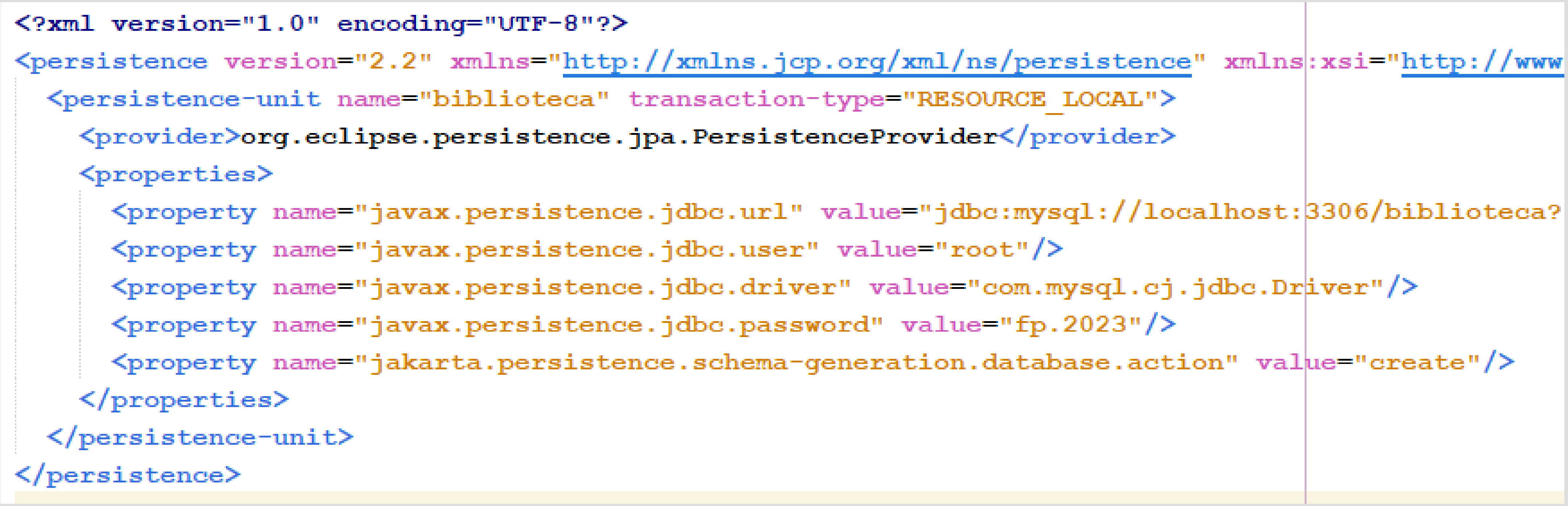
Table Generation Strategy: ○ Create ○ Drop and Create ● None

Validation Strategy: ● Auto ○ Callback ○ None

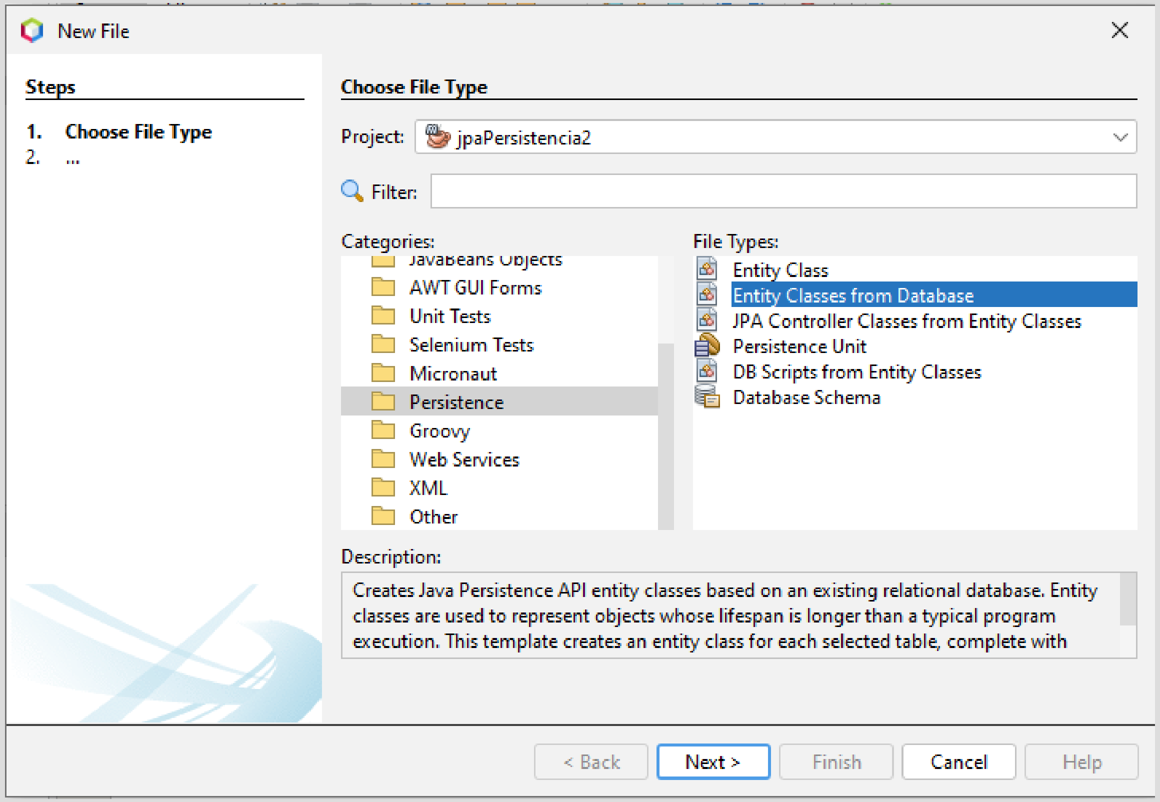
Shared Cache Mode: ○ All ○ None ○ Enable Selective ○ Disable Selective ● Unspecified

□ Include All Entity Classes in "jpaPersistencia2" Module

Include Entity Classes:



```
<?xml version="1.0" encoding="UTF-8"?>
<persistence version="2.2" xmlns="http://xmlns.jcp.org/xml/ns/persistence" xmlns:xsi="http://www
  <persistence-unit name="biblioteca" transaction-type="RESOURCE_LOCAL">
    <provider>org.eclipse.persistence.jpa.PersistenceProvider</provider>
    <properties>
      <property name="javax.persistence.jdbc.url" value="jdbc:mysql://localhost:3306/biblioteca?
      <property name="javax.persistence.jdbc.user" value="root"/>
      <property name="javax.persistence.jdbc.driver" value="com.mysql.cj.jdbc.Driver"/>
      <property name="javax.persistence.jdbc.password" value="fp.2023"/>
      <property name="jakarta.persistence.schema-generation.database.action" value="create"/>
    </properties>
  </persistence-unit>
</persistence>
```



Steps

1. **Choose File Type**
2. ...

Choose File Type

Project: jpaPersistencia2

Filter:

Categories:

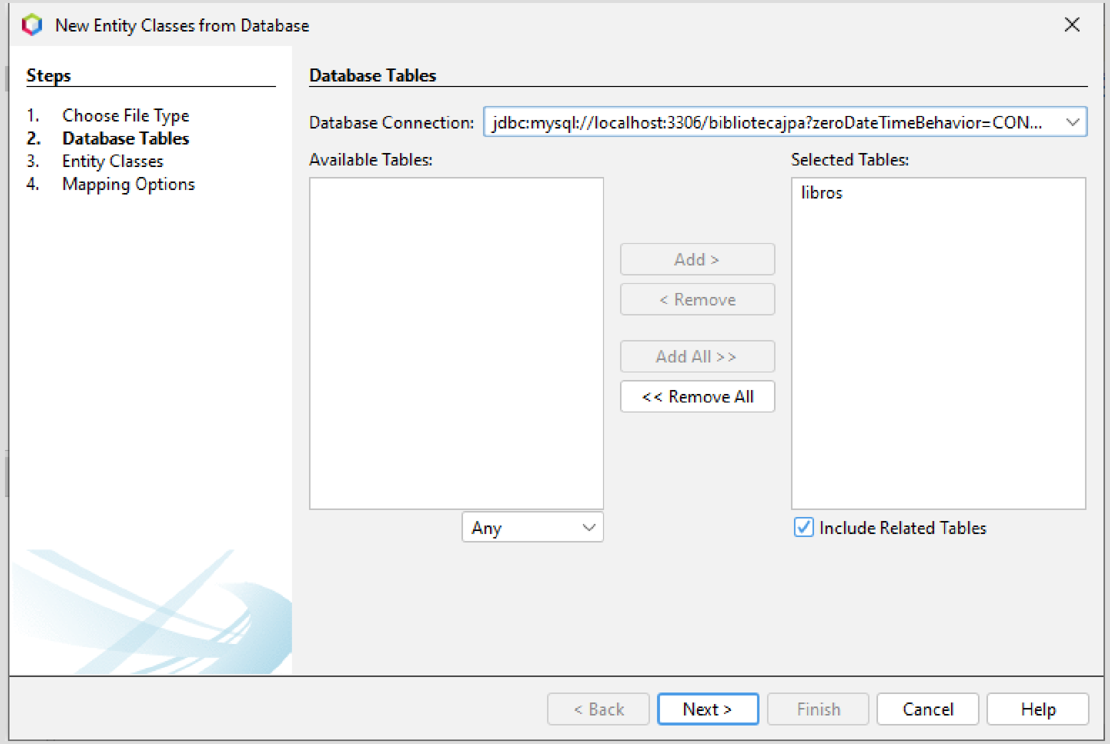
- JavaBeans Objects
- AWT GUI Forms
- Unit Tests
- Selenium Tests
- Micronaut
- Persistence
- Groovy
- Web Services
- XML
- Other

File Types:

- Entity Class
- Entity Classes from Database
- JPA Controller Classes from Entity Classes
- Persistence Unit
- DB Scripts from Entity Classes
- Database Schema

Description:

Creates Java Persistence API entity classes based on an existing relational database. Entity classes are used to represent objects whose lifespan is longer than a typical program execution. This template creates an entity class for each selected table, complete with



Steps

1. Choose File Type
2. **Database Tables**
3. Entity Classes
4. Mapping Options

Database Tables

Database Connection: jdbc:mysql://localhost:3306/bibliotecajpa?zeroDateTimeBehavior=CON...

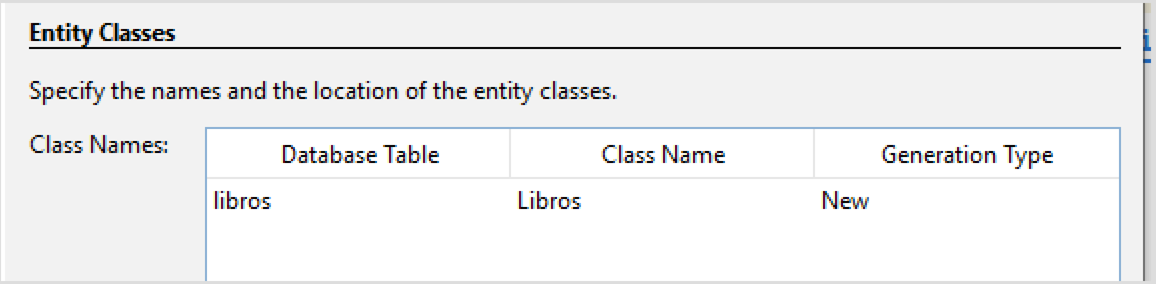
Available Tables:

Any

Selected Tables:

libros

☑ Include Related Tables

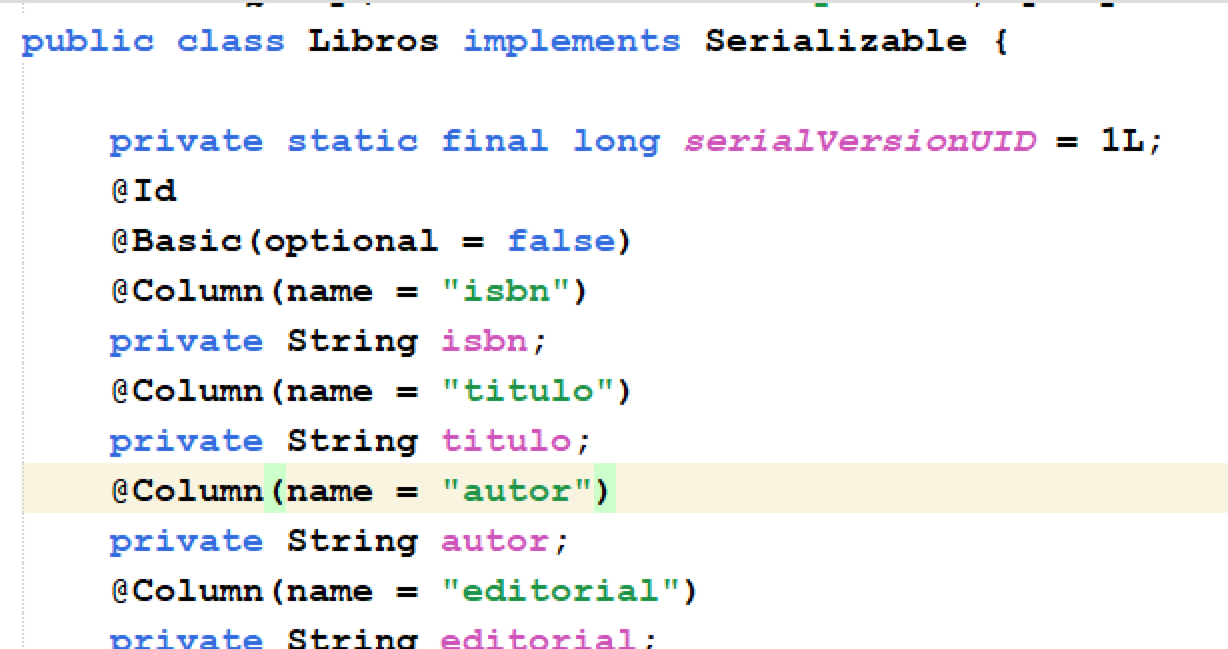


Entity Classes

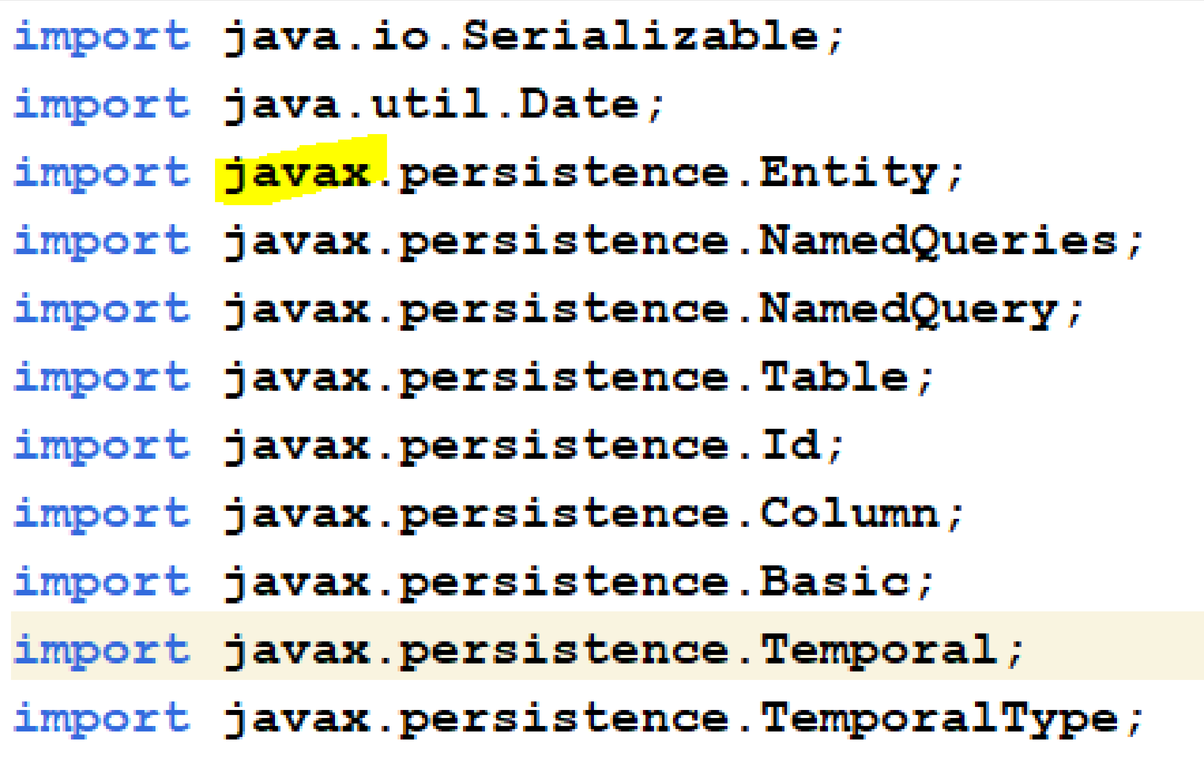
Specify the names and the location of the entity classes.

Class Names:

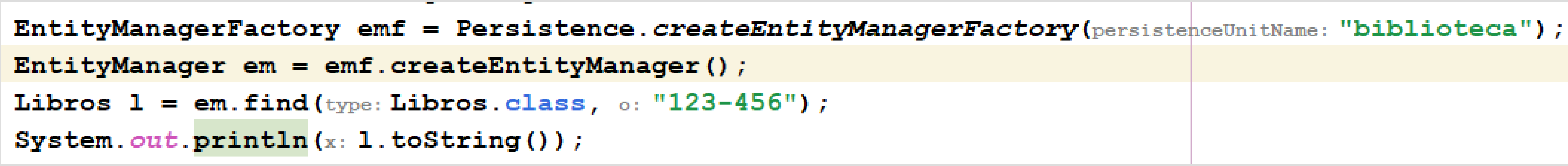
Database Table	Class Name	Generation Type
libros	Libros	New



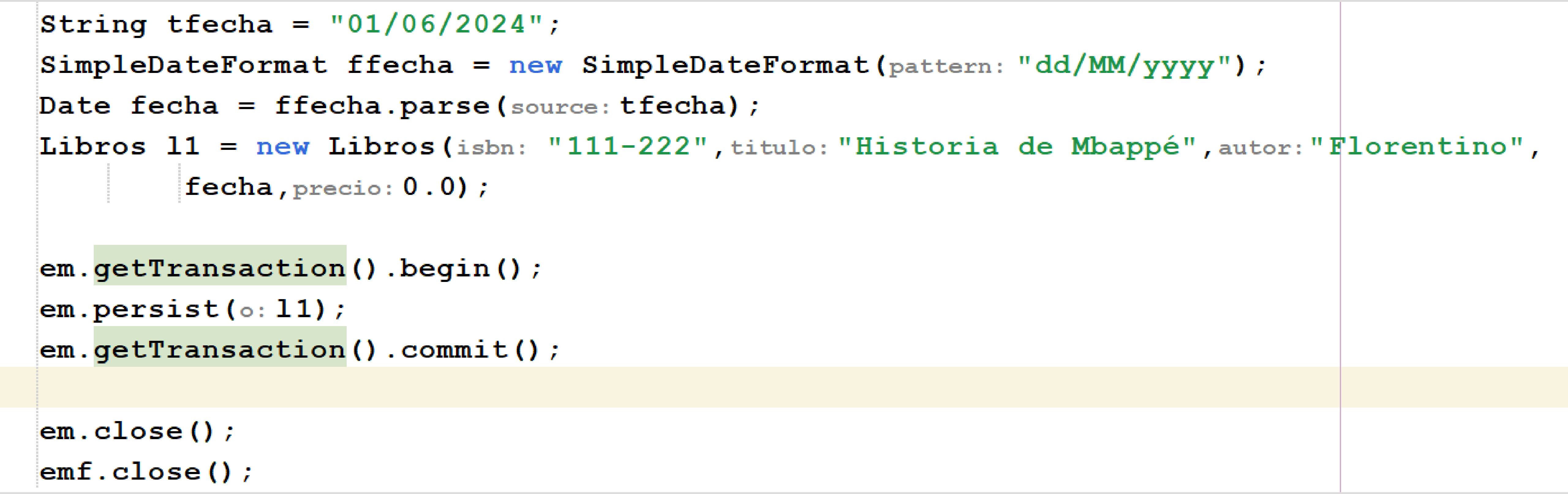
```java
public class Libros implements Serializable {

    private static final long serialVersionUID = 1L;
    @Id
    @Basic(optional = false)
    @Column(name = "isbn")
    private String isbn;
    @Column(name = "titulo")
    private String titulo;
    @Column(name = "autor")
    private String autor;
    @Column(name = "editorial")
    private String editorial;
```



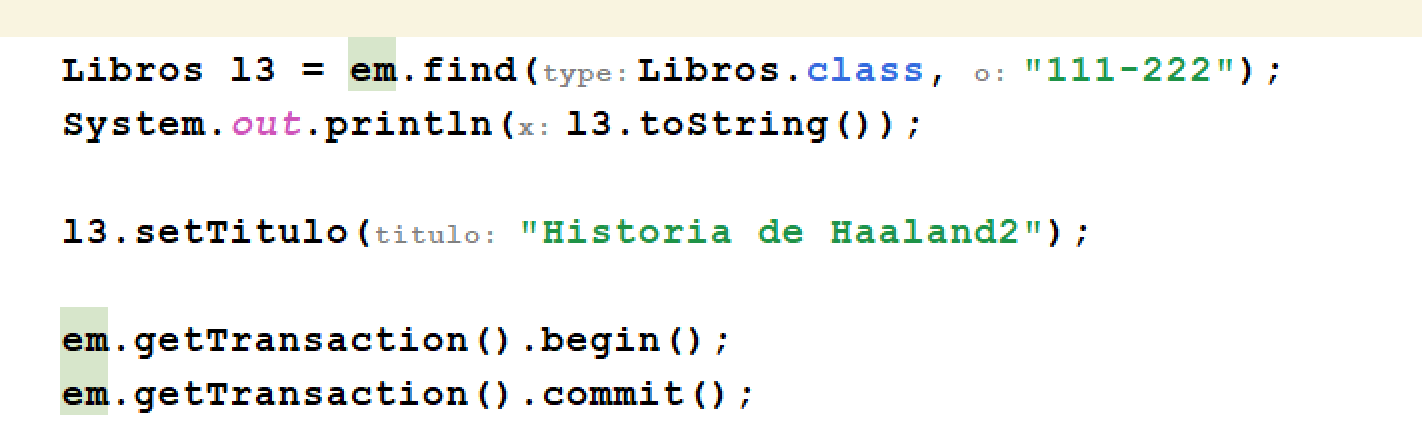
```java
import java.io.Serializable;
import java.util.Date;
import javax.persistence.Entity;
import javax.persistence.NamedQueries;
import javax.persistence.NamedQuery;
import javax.persistence.Table;
import javax.persistence.Id;
import javax.persistence.Column;
import javax.persistence.Basic;
import javax.persistence.Temporal;
import javax.persistence.TemporalType;
```



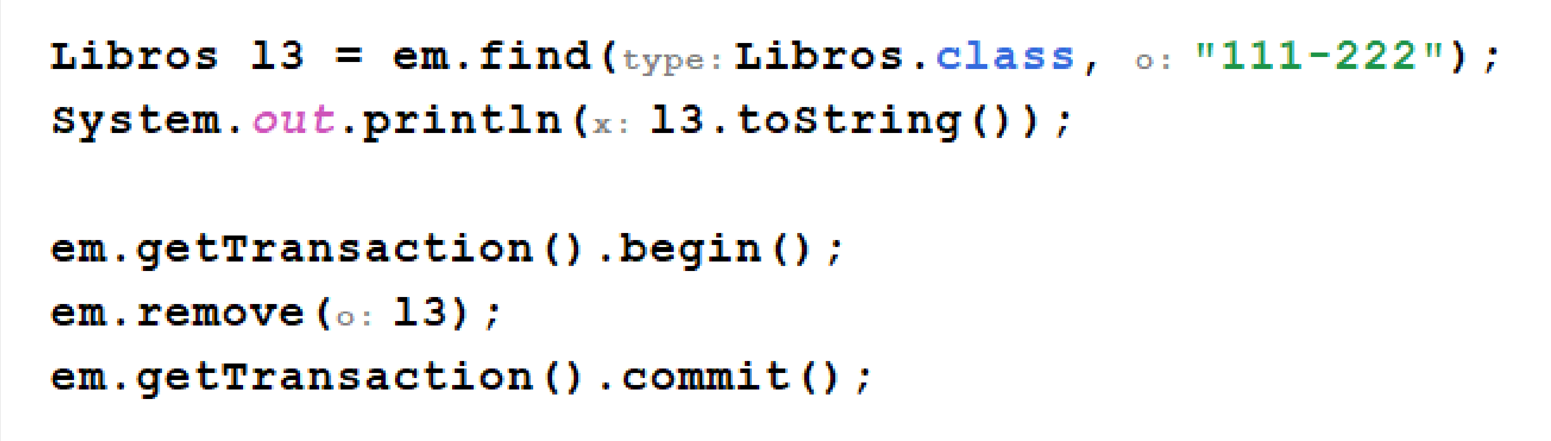
```
EntityManagerFactory emf = Persistence.createEntityManagerFactory(persistenceUnitName: "biblioteca");
EntityManager em = emf.createEntityManager();
Libros l = em.find(type: Libros.class, o: "123-456");
System.out.println(x: l.toString());
```



```
String tfecha = "01/06/2024";
SimpleDateFormat ffecha = new SimpleDateFormat(pattern: "dd/MM/yyyy");
Date fecha = ffecha.parse(source: tfecha);
Libros l1 = new Libros(isbn: "111-222",titulo: "Historia de Mbappé",autor: "Florentino",
        fecha,precio: 0.0);

em.getTransaction().begin();
em.persist(o: l1);
em.getTransaction().commit();

em.close();
emf.close();
```



```
Libros l3 = em.find(type: Libros.class, o: "111-222");
System.out.println(x: l3.toString());

l3.setTitulo(titulo: "Historia de Haaland2");

em.getTransaction().begin();
em.getTransaction().commit();
```



```
Libros l3 = em.find(type: Libros.class, o: "111-222");
System.out.println(x: l3.toString());

em.getTransaction().begin();
em.remove(o: l3);
em.getTransaction().commit();
```