# JDBC Pasos para implementarlo

## Depenencias

<!-- MySQL -->

<dependency>

<groupId>mysql</groupId>

<artifactId>mysql-connector-java</artifactId>

<version>8.0.33</version>

</dependency>

<!-- Eithers -->

<dependency>

<groupId>io.vavr</groupId>

<artifactId>vavr</artifactId>

<version>0.10.4</version>

</dependency>

<!-- Logger -->

<dependency>

<groupId>org.apache.logging.log4j</groupId>

<artifactId>log4j-core</artifactId>

<version>2.17.2</version>

</dependency>

<!-- Lombok -->

<dependency>

<groupId>org.projectlombok</groupId>

<artifactId>lombok</artifactId>

<version>1.18.22</version>

</dependency>

<!-- Para parsear JSON -->

<dependency>

<groupId>com.fasterxml.jackson.dataformat</groupId>

<artifactId>jackson-dataformat-yaml</artifactId>

<version>2.13.2</version>

</dependency>

<dependency>

<groupId>com.fasterxml.jackson.datatype</groupId>

<artifactId>jackson-datatype-jsr310</artifactId>

<version>2.13.2</version>

</dependency>

<!-- JAXB (XML) -->

<dependency>

<groupId>org.glassfish.jaxb</groupId>

<artifactId>jaxb-runtime</artifactId>

<version>4.0.0</version>

</dependency>

<!-- inyectar dependencias -->

<dependency>

<groupId>org.jboss.weld.se</groupId>

<artifactId>weld-se-core</artifactId>

<version>4.0.3.Final</version>

</dependency>

<!-- Jakarta -->

<dependency>

<groupId>jakarta.xml.bind</groupId>

<artifactId>jakarta.xml.bind-api</artifactId>

<version>4.0.0</version>

</dependency>

<dependency>

<groupId>jakarta.platform</groupId>

<artifactId>jakarta.jakartaee-web-api</artifactId>

<version>9.1.0</version>

</dependency>

<dependency>

<groupId>jakarta.platform</groupId>

<artifactId>jakarta.jakartaee-api</artifactId>

<version>8.0.0</version>

</dependency>

## Creando la clase DBConnection

Establecemos los cimientos para iniciar la comunicación entre el servidor sql y nuestro proyecto:

public class DBConnection {

private final Configuration config;

private Connection connection;

@Inject

public DBConnection(Configuration config) {

this.config = config;

}

@PreDestroy

public void closePool() {

// Cerrar el pool de conexiones antes ce que se destruya el objeto

try {

close();

} catch (SQLException e) {

log.error(e);

}

}

public Connection getConnection() throws SQLException {

if (connection == null || connection.isClosed()) {

connection = DriverManager.getConnection(config.getProperty("urlDB"), config.getProperty("user\_name"), config.getProperty("password"));

}

return connection;

}

public void close() throws SQLException {

if (connection != null && !connection.isClosed()) {

connection.close();

}

}

}