DBUNIT

```java
public class AlumnoDAOIT {
    private IDatabaseTester databaseTester;
    private IDatabaseConnection connection;
    @BeforeEach
    public void setUp() throws Exception {

        databaseTester = new MiJdbcDatabaseTester(,cadena_conexionDB, "root", "ppss");
        //obtenemos la conexión con la BD
        connection = databaseTester.getConnection();
    }
    @Test
    public void testB1() throws Exception {
        //Inicializamos el dataSet con los datos iniciales de la tabla
        IDataSet dataSet = new FlatXmlDataFileLoader().load("/tabla2.xml");
        //Inyectamos el dataset en el objeto databaseTester
        databaseTester.setDataSet(dataSet);
        //inicializamos la base de datos con los contenidos del dataset
        databaseTester.onSetup();
        //invocamos a nuestro SUT
        Assertions.assertDoesNotThrow(()->new FactoriaDAO().getAlumnoDAO().delAlumno("11111111A"));

        //recuperamos los datos de la BD después de invocar al SUT
        IDataSet databaseDataSet = connection.createDataSet();
        //Recuperamos los datos de la tabla alumnos
        ITable actualTable = databaseDataSet.getTable("alumnos");

        //creamos el dataset con el resultado esperado
        IDataSet expectedDataSet = new FlatXmlDataFileLoader().load("/tabla4.xml");
        ITable expectedTable = expectedDataSet.getTable("alumnos");

        Assertion.assertEquals(expectedTable, actualTable);
    }
    @Test
    public void testRetrieve() throws Exception {
        //Inicializamos el dataSet con los datos iniciales de la tabla cliente
        IDataSet dataSet = new FlatXmlDataFileLoader().load("/cliente-esperado.xml");
        //Inyectamos el dataset en el objeto databaseTester
        databaseTester.setDataSet(dataSet);
        //inicializamos la base de datos con los contenidos del dataset
        databaseTester.onSetup();
        //invocamos a nuestro SUT
        Cliente cl= Assertions.assertDoesNotThrow(()->clienteDAO.retrieve(1));
        Assertions.assertAll(
                ()->Assertions.assertEquals(cl.getId(),1),
                ()->Assertions.assertEquals(cl.getNombre(),"John"),
                ()->Assertions.assertEquals(cl.getApellido(),"Smith"),
                ()->Assertions.assertEquals(cl.getDireccion(),"1 Main Street"),
                ()->Assertions.assertEquals(cl.getCiudad(),"Anycity")
        );
        //recuperamos los datos de la BD después de invocar al SUT
        IDataSet databaseDataSet = connection.createDataSet();
        //Recuperamos los datos de la tabla cliente
        ITable actualTable = databaseDataSet.getTable("cliente");

        //creamos el dataset con el resultado esperado
        IDataSet expectedDataSet = new FlatXmlDataFileLoader().load("/cliente-esperado.xml");
        ITable expectedTable = expectedDataSet.getTable("cliente");

        Assertion.assertEquals(expectedTable, actualTable);
    }
}
```

WebDriver

```java
public class LoginPage {
    WebDriver driver;
    @FindBy(name="uid") WebElement userID;
    @FindBy(name="password") WebElement password;
    @FindBy(cssSelector="h1 > p") WebElement login;
    @FindBy(className="barone") WebElement loginTitle;

    public LoginPage(WebDriver driver){
        this.driver = driver;
        this.driver.get("http://demo.guru99.com/V4");
    }
    //nos logueamos
    public ManagerPage login(String user, String pass) {
        userID.sendKeys("user");
        password.sendKeys("pass");
        login.click();
        //retur de la pagina destino
        return PageFactory.initElements(driver,ManagerPage.class);
    }
    //cogemos el titulo
    public String getTile(){
        return driver.getTitle();
    }
}

public class TestLoginPage {
    WebDriver driver;
    LoginPage poLogin;
    ManagerPage poManagerPage;

    @BeforeEach
    public void setup(){
        driver = new ChromeDriver();
        poLogin = PageFactory.initElements(driver, LoginPage.class);
    }
    @Test
    public void test_Login_Correct(){
        //cogemos el
        String loginPageTitle = poLogin.getTile();
        Assertions.assertEquals("guru99 bank",loginPageTitle);
        //nos loguemos y obtenemos la case MAnage
        poManagerPage = poLogin.login("mngr34733", "AbEvydU");
        //obtenemos el nombre de usuario
        Assertions.assertEquals("manger id : mngr34733",poManagerPage.getUserName());

    }
    @AfterEach
    public void setup(){
        driver.close();
    }
}
```

POM Integración DBunit

```xml
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>ppss</groupId>
  <artifactId>matriculacion</artifactId>
  <version>1.0-SNAPSHOT</version>
  <!--modulos del pryecto-->
    <modules>
        <module>matriculacion-comun</module>
        <module>matriculacion-dao</module>
        <module>matriculacion-proxy</module>
        <module>matriculacion-bo</module>
    </modules>
    <!---->
    <packaging>pom</packaging><!--pakage para prouectos multimodulo-->
  <name>matriculacion</name>
    <properties>
        <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
        <maven.compiler.source>11</maven.compiler.source>
        <maven.compiler.target>11</maven.compiler.target>
    </properties>
    <dependencies>
        <dependency>
            <groupId>org.junit.jupiter</groupId>
            <artifactId>junit-jupiter-engine</artifactId>
            <version>5.8.2</version>
            <scope>test</scope>
        </dependency>
    </dependencies>
    <build>
        <plugins>
            <plugin>
                <groupId>org.apache.maven.plugins</groupId>
                <artifactId>maven-compiler-plugin</artifactId>
                <version>3.9.0</version>
            </plugin>
            <plugin>
                <groupId>org.apache.maven.plugins</groupId>
                <artifactId>maven-surefire-plugin</artifactId>
                <version>3.0.0-M5</version>
            </plugin>
            <!--Plugin failsafe para la integracion-->
            <plugin>
                <groupId>org.apache.maven.plugins</groupId>
                <artifactId>maven-failsafe-plugin</artifactId>
                <version>3.0.0-M5</version>
                <executions>
                    <execution>
                        <!--Asignamos la goal a la fase integration-test y verify-->
                        <goals>
                            <goal>integration-test</goal>
                            <goal>verify</goal>
                        </goals>
                    </execution>
                </executions>
            </plugin>
            <!---->
        </plugins>
    </build>
</project>
```

```xml
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
    <!--Padre del proyecto-->
    <parent>
        <artifactId>matriculacion</artifactId>
        <groupId>ppss</groupId>
        <version>1.0-SNAPSHOT</version>
    </parent>
    <!---->
    <modelVersion>4.0.0</modelVersion>
    <artifactId>matriculacion-dao</artifactId>
    <dependencies>
        <!--dependencias del pryecto-->
        <dependency>
            <groupId>${project.groupId}</groupId>
            <artifactId>matriculacion-comun</artifactId>
            <version>${project.version}</version>
        </dependency>
        <!---->
        <dependency>
            <groupId>mysql</groupId>
            <artifactId>mysql-connector-java</artifactId>
            <version>8.0.27</version>
            <scope>test</scope>
        </dependency>
        <dependency>
            <groupId>org.slf4j</groupId>
            <artifactId>slf4j-simple</artifactId>
            <version>1.7.36</version>
        </dependency>
        <dependency>
            <groupId>org.dbunit</groupId>
            <artifactId>dbunit</artifactId>
            <version>2.7.3</version>
            <scope>test</scope>
        </dependency>
    </dependencies>
    <build>
        <plugins>
        <!--plugin SQL-->
            <plugin>
                <groupId>org.codehaus.mojo</groupId>
                <artifactId>sql-maven-plugin</artifactId>
                <version>1.5</version>
                <dependencies>
                    <dependency>
                        <groupId>mysql</groupId>
                        <artifactId>mysql-connector-java</artifactId>
                        <version>8.0.27</version>
                    </dependency>
                </dependencies>
                <configuration><!--Cadena de coneion-->
                    <driver>com.mysql.cj.jdbc.Driver</driver>
                    <url>jdbc:mysql://localhost:3306/?useSSL=false</url>
                    <username>root</username>
                    <password>ppss</password>
                </configuration>
                <executions>
                    <!--añadimos la goal execute a la fase pre-integration-test para ejecutar comando SQL -->
                    <execution>
                        <id>create-alumnos-table</id>
                        <phase>pre-integration-test</phase>
                        <goals>
                            <goal>execute</goal>
                        </goals>
                        <configuration>
                            <srcFiles>
                                <srcFile>src/test/resources/sql/matriculacion.sql</srcFile>
                            </srcFiles>
                        </configuration>
                    </execution>
                    <!---->
                </executions>
            </plugin>
        </plugins>
        <!--plugin SQL-->
    </build>
</project>
```

Plugin de JACOCO

```xml
<plugin>
    <groupId>org.jacoco</groupId>
    <artifactId>jacoco-maven-plugin</artifactId>
    <version>0.8.7</version>
    <!--asociamos las goals al plugin-->
    <executions>
        <execution>
            <id>default-prepare-agent</id>
            <goals>
                <goal>prepare-agent</goal>
            </goals>
        </execution>
        <execution>
            <id>default-report</id>
            <goals>
                <goal>report</goal>
            </goals>
        </execution>
        <!--Aqui se define las reglas de las cobertura-->
        <execution>
            <id>default-check</id>
            <goals>
                <goal>check</goal>
            </goals>
            <configuration>
                <rules>
                <!--BUNDLE PACKAGE CLASS METHOD-->
                    <rule>
                        <!--nivel de app-->
                        <element>BUNDLE</element>
                        <limits>
                            <limit>
                                <!--COMPLEXITY, INSTRUCTION, CLASS,LINE,BRANCH-->
                                <counter>COMPLEXITY</counter>
                                <!--TOTALCOUNT, COVEREDCOUNT, MISSEDCOUNT, COVEREDRATIO, MISSEDRATIO-->
                                <value>COVEREDRATIO</value>
                                <minimum>0.85</minimum>
                            </limit>
                            <limit>
                                <counter>INSTRUCTION</counter>
                                <value>COVEREDRATIO</value>
                                <minimum>0.80</minimum>
                            </limit>
                        </limits>
                    </rule>
                    <!--excluimos una clase-->
                    <rule>
                        <element>CLASS</element>
                        <excludes>
                            <exclude>
                                ppss.ejercicio2.MyClass
                            </exclude>
                        </excludes>
                        <limits>
                            <limit>
                                <counter>LINE</counter>
                                <value>COVEREDRATIO</value>
                                <minimum>0.75</minimum>
                            </limit>
                        </limits>
                    </rule>
                </rules>
            </configuration>
        </execution>
        <!---->
    </executions>
    <!---->
</plugin>
```