

实验 2 员工管理系统实验程序

一、环境准备（工具及版本）

- 1、Java 版本：JDK 1.8
- 2、构建工具：Maven 3.6.3
- 3、开发工具：IntelliJ IDEA Ultimate 2022.3.3
- 4、数据库：MySQL 5.7+ 以上
- 5、数据库工具：Navicat 10+以上

二、Mybatis 环境搭建（略，可参考第一章实验）

搭建过程省略，遇问题请参考第一章的搭建过程。

三、Maven 项目工程创建（略，可参考第一章实验）

创建过程省略，遇问题请参考第一章的入门实验。

工程名称为：employee

Group: com.xhxcg

Version: Java EE 8

四、员工管理系统实验案例

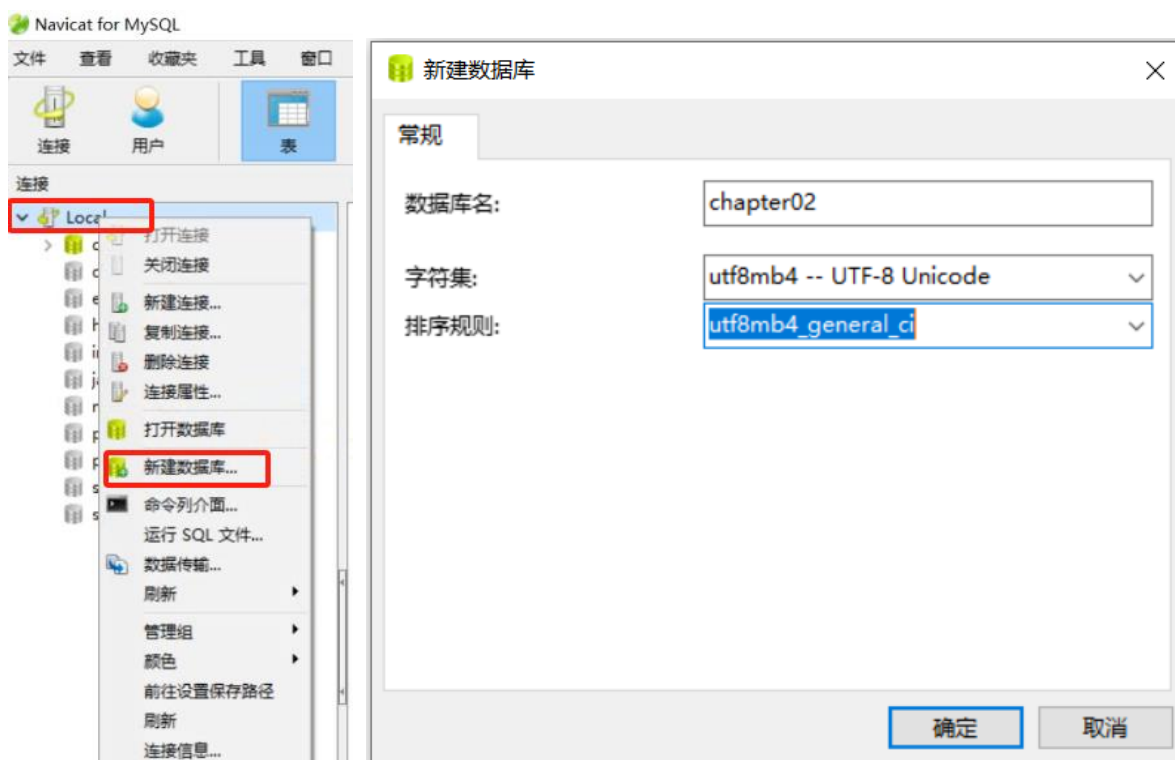
实验目标：完成一个员工管理系统，能够实现如下功能：根据 id 查询、新增、修改、删除员工信息。

第 1 步：数据库开发

先使用 MySQL 数据库创建一个 chapter02 数据库，然后创建一个用户表 t_employee，方法如下：

MySQL 安装过程，这里省略。安装完成后，我们通过使用 Navicat 工具，连接上你安装的 MySQL 数据库，输入用户名和密码进行登录。假设建立的 MySQL 连接名为 Local，数据库登录用户名和密码为：root/123456。登录后，即可手动创建数据库，我们需要创建的数据

库名为：chapter02，如图：选中 Local 连接名，右键选择【新建数据库】



数据库创建好后，创建表 **t_employee**，表结构如下。并按要求加入几条测试数据。

```
USE chapter02;
CREATE TABLE t_employee(
    eid INT PRIMARY KEY AUTO_INCREMENT,
    ename VARCHAR(50),
    eage INT,
    eposition varchar(50)
);
INSERT INTO t_employee(ename,eage,eposition) VALUES('Lucy',25,'职员');
INSERT INTO t_employee(ename,eage,eposition) VALUES('Lili',20,'经理');
INSERT INTO t_employee(ename,eage,eposition) VALUES('Jim',20,'部长');
```

全部创建完成后，效果如下：

t_employee @employee (Local) - 表

文件 编辑 查看 窗口 帮助

导入向导 导出向导 筛选向导 网

eid	ename	eage	eposition
1	Lucy	25	职员
2	Lili	20	经理
3	Jim	20	部长
6	胡要胡	36	员工

至此，数据库部分开发完毕。

第 2 步：程序代码开发

1、创建 POJO 实体

打开前面创建的 Mybatis 项目工程, 工程名称是 **employee**。创建代码包为: com.xhcg.pojo,

并且创建 **Employee** 类

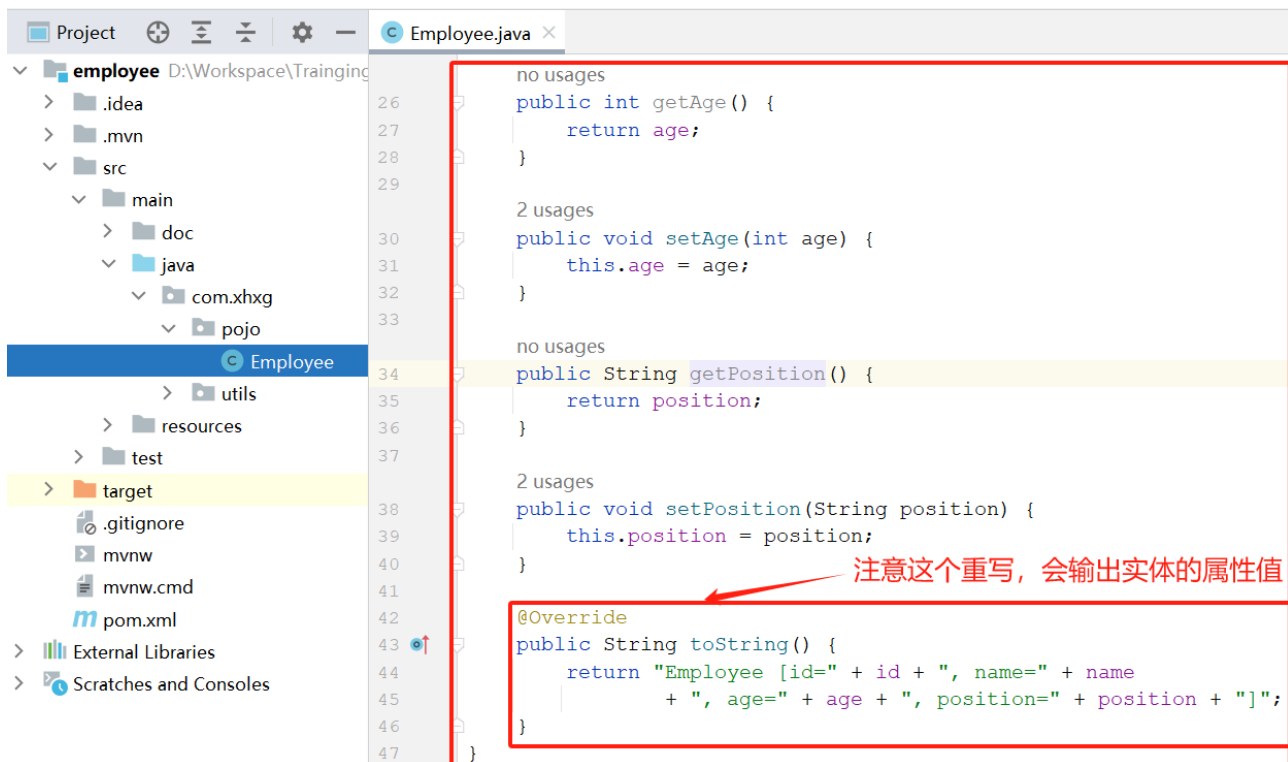
Project ▾

employee D:\Workspace\Trainging\07.java

- > .idea
- > .mvn
- src
 - main
 - doc
 - java
 - com.xhxcg
 - pojo
 - Employee
 - utils
 - resources
 - test
 - target
 - .gitignore
 - mvnw
 - mvnw.cmd
 - pom.xml
- External Libraries
- Scratches and Consoles

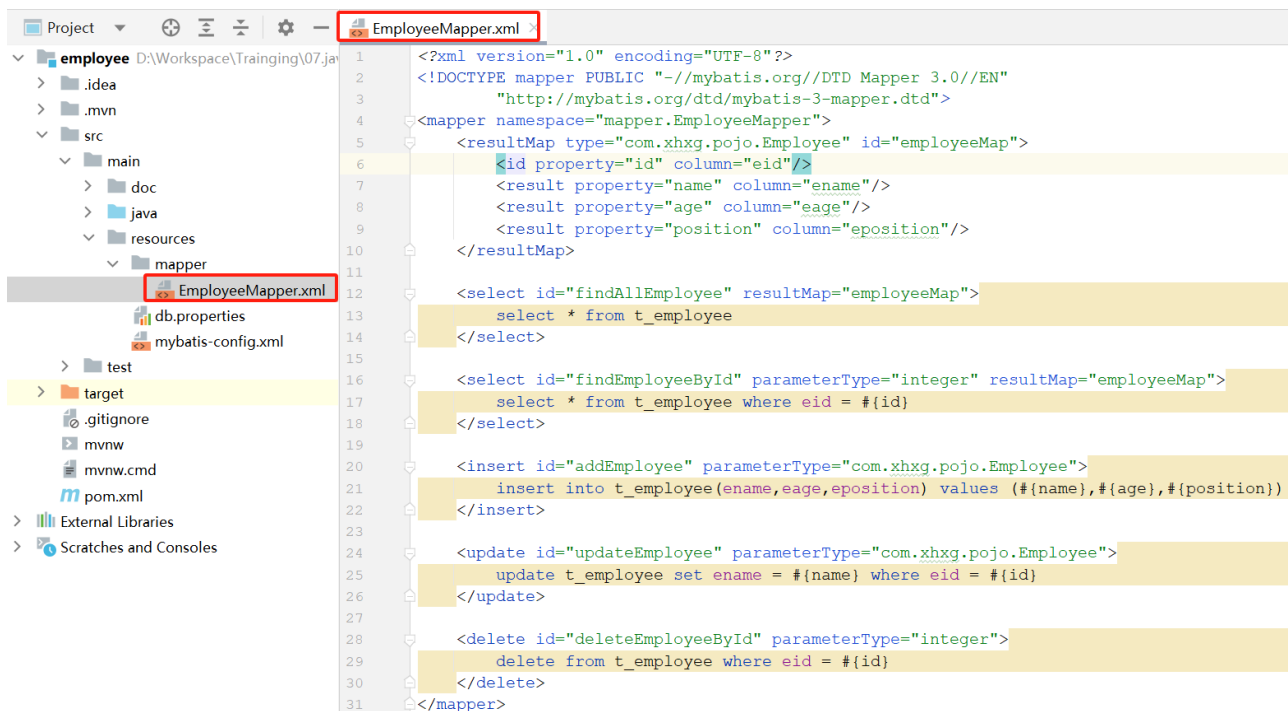
Employee.java x

```
1 package com.xhxcg.pojo;
2
3 12 usages
4 public class Employee {
5     3 usages
6     private int id;
7     3 usages
8     private String name;
9
10    3 usages
11    private int age;
12    3 usages
13    private String position;
14
15    no usages
16    public int getId() {
17        return id;
18    }
19
20    1 usage
21    public void setId(int id) {
22        this.id = id;
23    }
24
25    public String getName() {
26        return name;
27    }
28
29    2 usages
30    public void setName(String name) {
31        this.name = name;
32    }
33 }
```



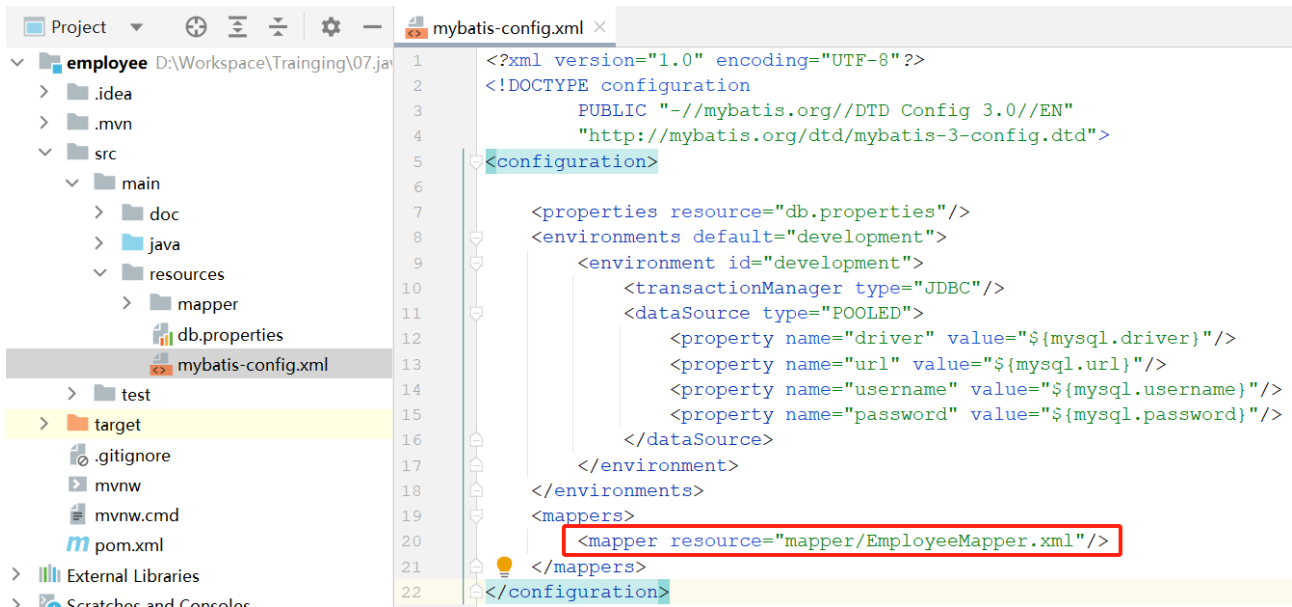
2、创建映射文件 EmployeeMapper.xml

在 src/main/resources 目录下创建目录 mapper，然后创建一个 EmployeeMapper.xml 文件，内容如下。



3、创建 mybatis-config.xml 配置文件

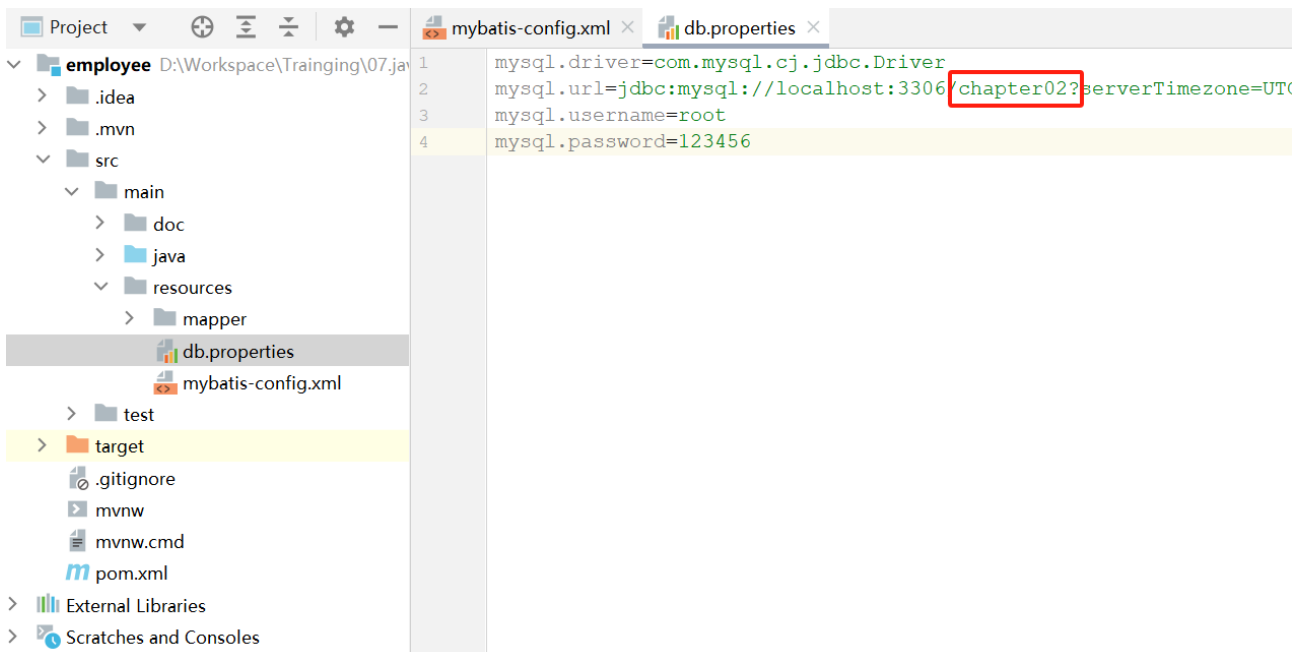
在 src/main/resources 目录下创建一个 mybatis-config.xml 文件，内容如下：



里面的配置项，需要创建另一个配置文件 db.properties 进行配置，所以我们跟着创建。

4、创建 db.properties 配置文件

在 src/main/resources 目录下创建一个 db.properties 文件，内容如下：



5、创建一个 Mybatis 连接工具类 JDBCUtils

由于执行测试时，每个方法执行都需要读取配置文件，并根据配置文件的信息构建 SqlSessionFactory 对象、创建 SqlSession 对象、释放资源，这导致了大量的重复代码。为了

简化开发，我们可以将读取配置文件和释放资源的代码封装到一个工具类 JDBCUtils 中（需要创建一个 utils 的 package 包，类路径：**com.xhxcg.utils.JDBCUtils**），然后通过工具类（JDBCUtils）创建 SqlSession 对象。代码如下：



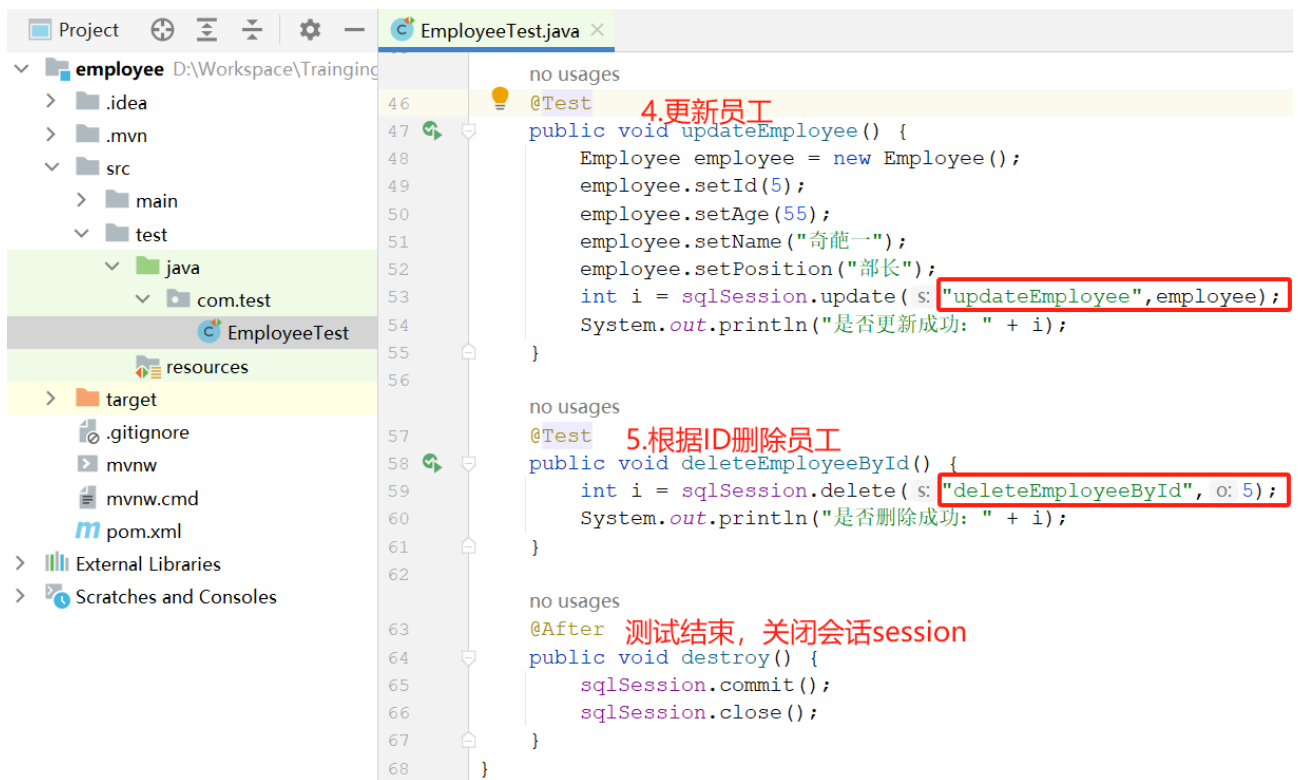
6、编写测试类

以上代码和配置创建好后，就可以编写测试类进行验证测试。

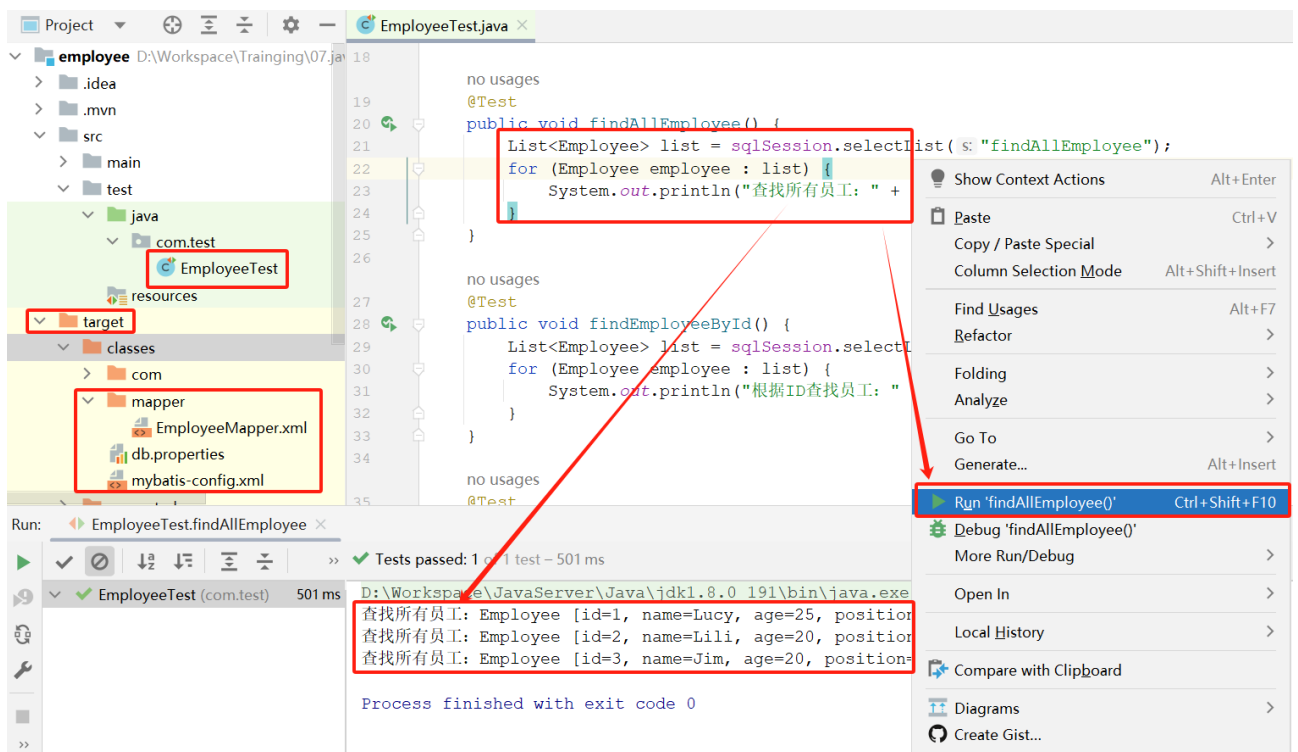
我们在 **src/test/java** 目录下，建议测试包，路径为：**com.test**，然后创建测试类 **EmployeeTest**，测试的方法如下：

```
1 package com.test;
2
3 import com.xhxc.pojo.Employee;
4 import com.xhxc.utils.JDBCUtils;
5 import org.apache.ibatis.session.SqlSession;
6 import org.junit.After;
7 import org.junit.Before;
8 import org.junit.Test;
9
10 import java.util.List;
11
12 no usages
13 public class EmployeeTest {
14     8 usages
15     private SqlSession sqlSession = null;
16     no usages
17     @Before 初始化, 获得会话session
18     public void init() {
19         sqlSession = JDBCUtils.getSqlSession();
20     }
21
22     no usages
23     @Test 1. 查询所有员工
24     public void findAllEmployee() {
25         List<Employee> list = sqlSession.selectList("findAllEmployee");
26         for (Employee employee : list) {
27             System.out.println("查找所有员工: " + employee);
28         }
29     }
30 }
```

```
27 no usages
28 @Test 2. 根据ID查询员工
29 public void findEmployeeById() {
30     List<Employee> list = sqlSession.selectList("findEmployeeById, 0, 1);
31     for (Employee employee : list) {
32         System.out.println("根据ID查找员工: " + employee);
33     }
34 }
35
36 no usages
37 @Test 3. 添加员工
38 public void addEmployee() {
39     Employee employee = new Employee();
40     employee.setAge(36);
41     employee.setName("胡要胡");
42     employee.setPosition("员工");
43
44     int i = sqlSession.insert("addEmployee", employee);
45     System.out.println("是否新增成功: " + i);
46 }
```

在不同的单元测试方法中，执行不同的单元测试。如在查询所有员工的测试方法中测试时，正确会输出查询结果，如下图：



至此，本实验圆满成功，建立了 Java 代码与数据库的关联，验证了通过 Java 调用，正确获取数据库中的记录。

五、常见问题

有时候 Maven 不能正确的复制配置文件到 target 目录中, 此时运行测试程序就会报如下异常: Could not find resource mybatis-config.xml

解决办法, 就是手动将 src/main/resources/目录下的所有配置文件 (包含目录) 复制到 target/classes/目录下即可解决。

The screenshot illustrates the problem and its solution in an IDE environment. On the left, the Project Explorer shows the source code structure: `src/main/resources` contains `mapper` (with `UserMapper.xml`), `db.properties`, and `mybatis-config.xml`. The `target` directory contains `classes` and `xhxcg`. A red arrow points from the `resources` folder to the `classes` folder in `target`, with a note: "需要copy这些配置文件到classes目录下" (Need to copy these configuration files to the classes directory). The main editor shows `UserTest.java` with a test method `userFindByIdTest()` that attempts to load `mybatis-config.xml` using `Resources.getResourceAsStream()`. The Run window at the bottom shows a failed test run for `UserTest.userFindByIdTest` with a stack trace indicating the exception: `java.io.IOException: Could not find resource mybatis-config.xml`. The stack trace includes the following lines:

```
java.io.IOException: Could not find resource mybatis-config.xml
    at org.apache.ibatis.io.Resources.getResourceAsStream(Resources.java:114)
    at org.apache.ibatis.io.Resources.getResourceAsStream(Resources.java:100)
    at org.apache.ibatis.io.Resources.getResourceAsReader(Resources.java:160)
    at com.test.UserTest.userFindByIdTest(UserTest.java:20) <24 internal lines>
```

The exception is labeled "异常信息" (Exception Information). Below the stack trace, another exception is partially visible: `java.lang.NullPointerException: Create breakpoint at org.apache.ibatis.session.SqlSessionFactoryBuilder.build(SqlSessionFactoryBuilder...`.