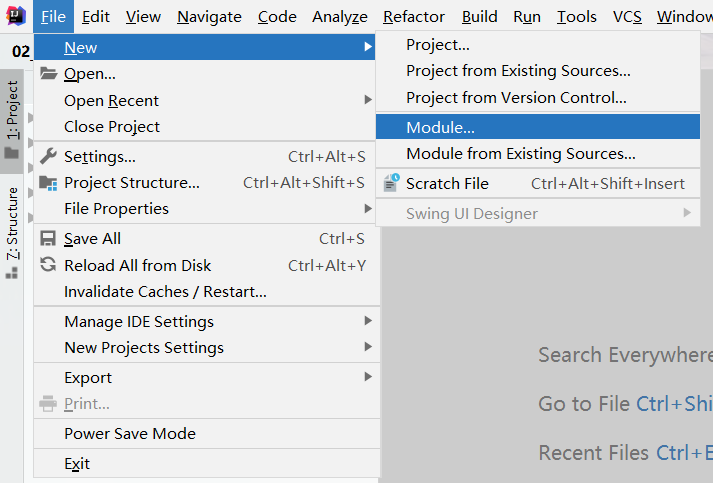
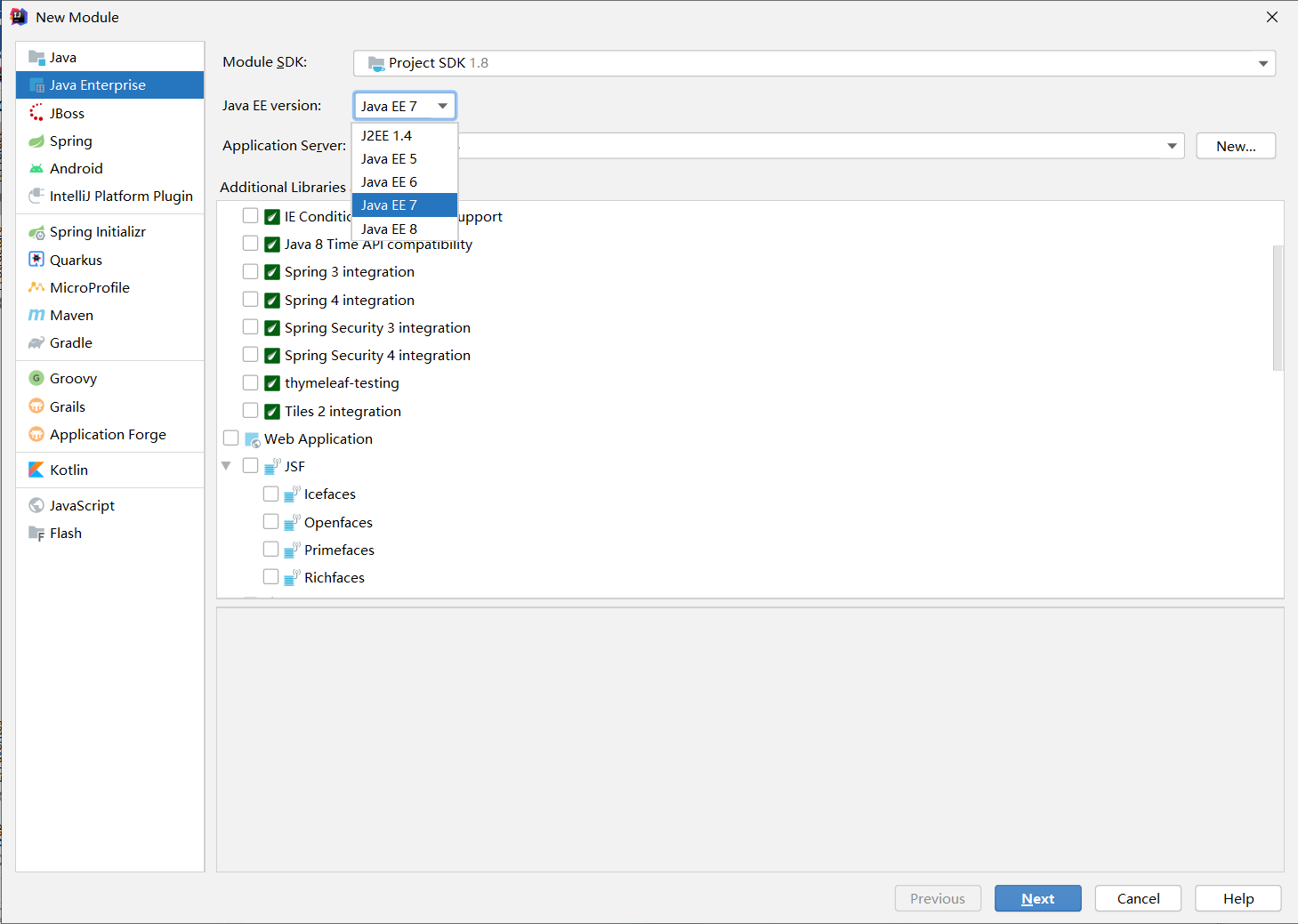
MyBatis代码说明

# 一、搭建环境

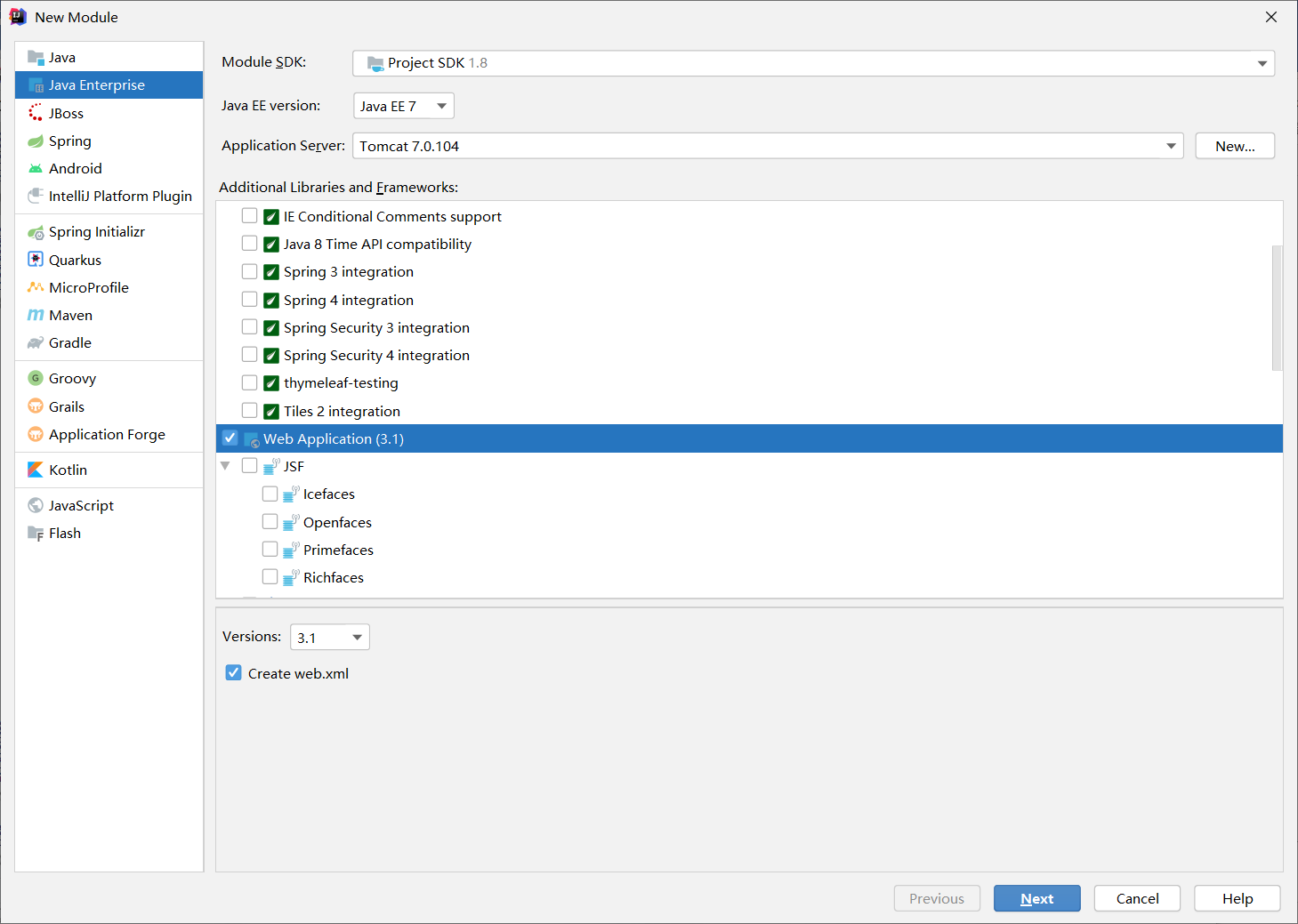
首先建立一个Module



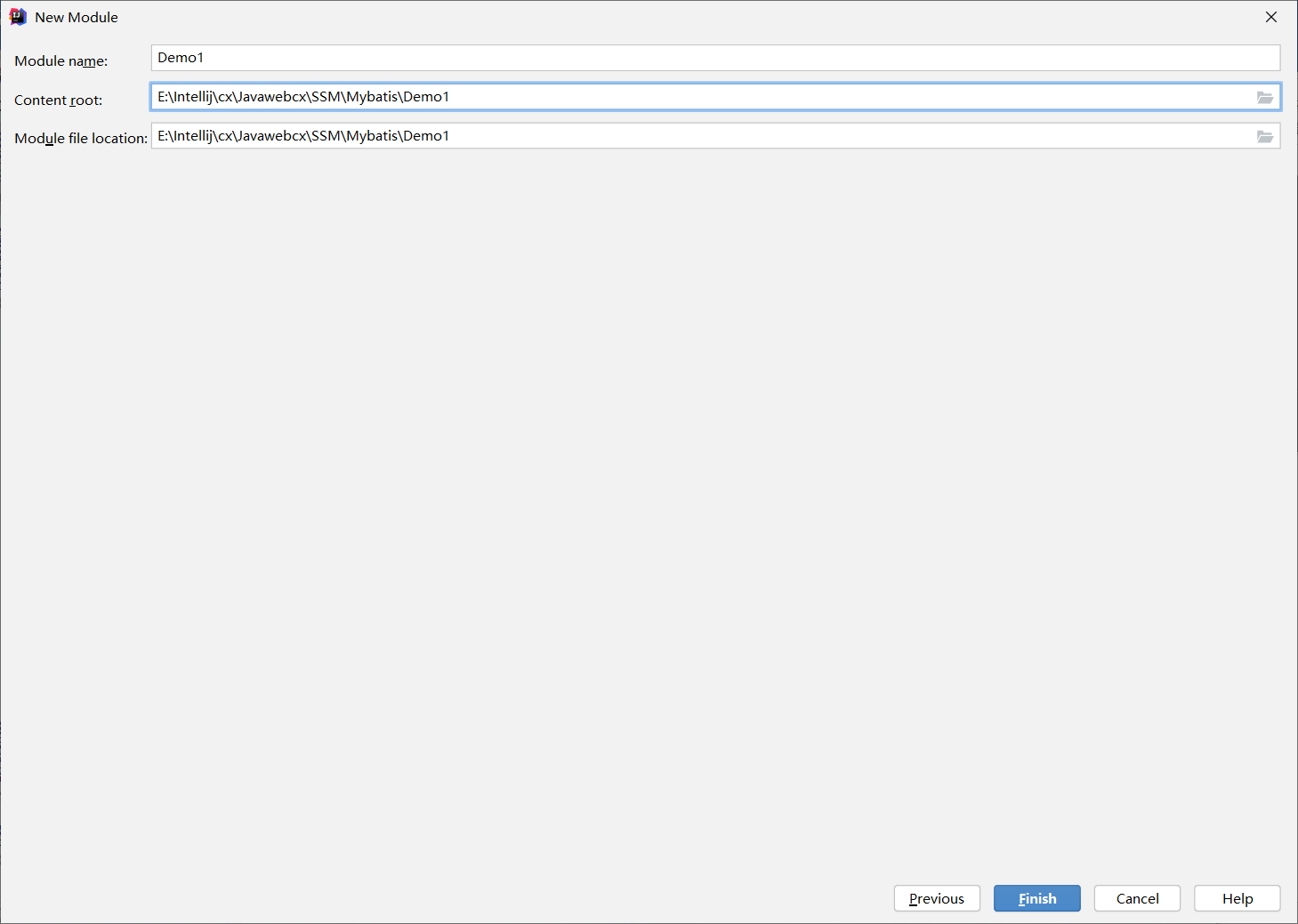
选中java enterprise—>java ee 7



再勾选Web Application(我喜欢用Java EE7，他使用的是3.1版本的xml)，勾选Create Web.xml，点击next。



给项目先命名，放在你自己想要放的文件夹路径下面，点击finish。



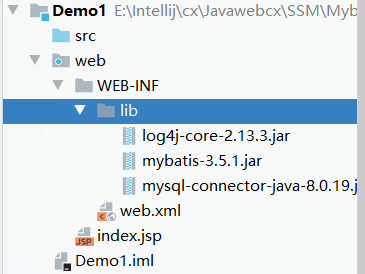
这样，一个简单地搭建就好了



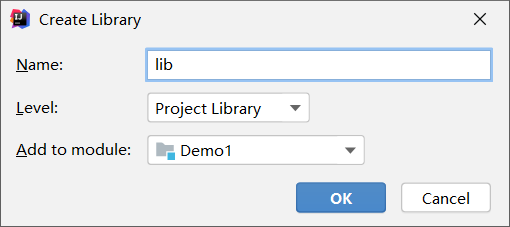
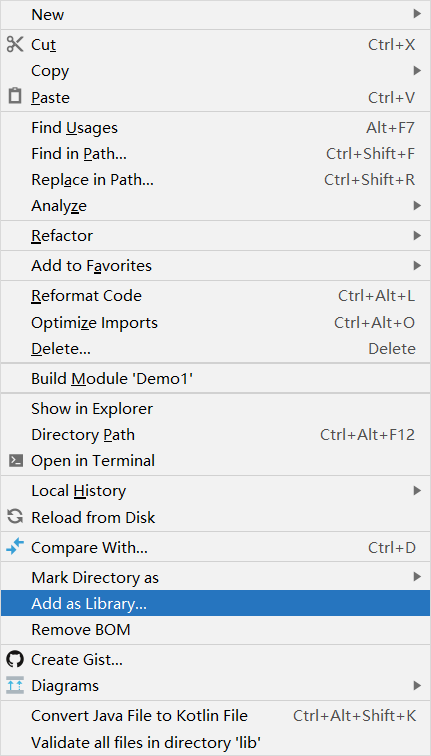
注意：本次没有使用Maven搭建依赖。

接下来我来创建文件夹，在WEB-INF文件夹下面创建lib库文件，放置需要的jar包。

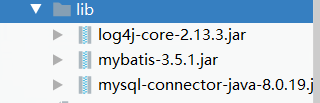
暂时添加log4j.jar，mybatis.jar，mysql-connector.jar3个包。



右键lib添加到library库中



添加完成后，右边就会出现小三角



# 二、IDEA操作

接下来创建需要用到的包：

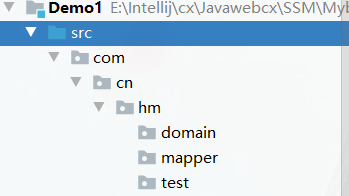
com.cn.hm表示域名+部门名

domain:实体层

mapper:放置映射文件

test:测试代码用的

初次使用暂时添加这几个包



在src文件下创建log4j.properties和mybatis-config.xml文件。



log4j.properties：配置日志信息

log4j.rootLogger=DEBUG, stdout, logfile

log4j.category.org.springframework=ERROR

log4j.category.org.apache=INFO

log4j.appender.stdout=org.apache.log4j.ConsoleAppender

log4j.appender.stdout.layout=org.apache.log4j.PatternLayout

log4j.appender.stdout.layout.ConversionPattern=%d %p [%c] - %m%n

log4j.appender.logfile=org.apache.log4j.RollingFileAppender

log4j.appender.logfile.File=${myweb.root}/WEB-INF/log/myweb.log

log4j.appender.logfile.MaxFileSize=512KB

log4j.appender.logfile.MaxBackupIndex=5

log4j.appender.logfile.layout=org.apache.log4j.PatternLayout

log4j.appender.logfile.layout.ConversionPattern=%d %p [%c] - %m%n

mybatis-config.xml：配置mybatis的信息

*<?*xml version="1.0" encoding="UTF-8" *?>*

<!DOCTYPE configuration

PUBLIC "-//mybatis.org//DTD Config 3.0//EN"

"http://mybatis.org/dtd/mybatis-3-config.dtd"*>*

<configuration>

<environments default="development">

<environment id="development">

<transactionManager type="JDBC"/>

<dataSource type="POOLED">

<property name="driver" value="com.mysql.cj.jdbc.Driver"/>

<property name="url" value="jdbc:mysql://localhost:3306/test?serverTimezone=GMT%2B8"/>

<property name="username" value="root"/>

<property name="password" value="123456"/>

</dataSource>

</environment>

</environments>

<mappers>

*<!--<mapper resource="org/mybatis/examp1e/BlogMapper.xm1"/>-->*

<mapper resource="com/cn/hm/mapper/StudentMapper.xml"/>

</mappers>

</configuration>

搭建完成后就开始编写代码了

首先在domain层里新建一个Student类

public class Student {

private String id;

private String name;

private Integer age;

public String getId() {

return id;

}

public void setId(String id) {

this.id = id;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public Integer getAge() {

return age;

}

public void setAge(Integer age) {

this.age = age;

}

@Override

public String toString() {

return "Student{" +

"id='" + id + '\'' +

", name='" + name + '\'' +

", age=" + age +

'}';

}

}

在mapper里面新建一个学生的映射StudentMapper.xml

*<?*xml version="1.0" encoding="UTF-8" *?>*

<!DOCTYPE mapper

PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"

"http://mybatis.org/dtd/mybatis-3-mapper.dtd"*>*

*<!-- namespace:命名空间*

*不同的mapper映射文件使用namespace来做区分*

*不同的mapper映射文件所使用的namespace的命名不允许出现重复*

*使用命名空间 .sqlId的形式来找到我们想要执行的sql语句*

*test1.getById1*

*-->*

<mapper namespace="test1">

*<!--*

*sql语句必须要写在相应的的标签中*

*<insert>在标签中写insert开头的语句，处理添加操作*

*<update>在标签中写update开头的语句，处理修改操作*

*<delete>在标签中写delete开头的语句，处理删除操作*

*<select>在标签中写select开头的语句，处理查询操作*

*parameterType:为sql语句传递的参数类型*

*-->*

<select id="getById" parameterType="java.lang.String" resultType="com.cn.hm.domain.Student">

select *\** from tbl\_student where id =#{id}

</select>

*<!--*

*如果返回的是多条记录，那么resultType返回值类型，应该写为集合的泛型*

*-->*

<select id="getAll" resultType="com.cn.hm.domain.Student">

select *\** from tbl\_student

</select>

*<!--*

*注意：*

*在未来实际项目开发中*

*所有的标签都必须要写id属性*

*<select>标签中parameterType可以省略不写*

*resultType属性必须得写*

*对于<insert><update><delete>这3个标签*

*通常只写id属性，其他属性一概不写*

*-->*

<insert id="save" parameterType="com.cn.hm.domain.Student">

insert into tbl\_student(id,name,age) values(#{id},#{name},#{age})

</insert>

<update id="update">

update tbl\_student set name=#{name},age=#{age} where id=#{id}

</update>

<delete id="delete">

delete from tbl\_student where id=#{id}

</delete>

</mapper>

在test里面创建一个Test1测试类

import org.apache.ibatis.io.Resources;

import org.apache.ibatis.session.SqlSession;

import org.apache.ibatis.session.SqlSessionFactory;

import org.apache.ibatis.session.SqlSessionFactoryBuilder;

import java.io.IOException;

import java.io.InputStream;

public class Test1 {

public static void main(String[] args) {

String resource = "mybatis-config.xml";

*//输入流*

InputStream inputStream = null;

try {

*//通过加载Mybatis的主配置文件mybatis-config.xml，创建输入对象*

inputStream = Resources.*getResourceAsStream*(resource);

}catch (IOException e){

e.printStackTrace();

}

*/\**

*SqlSessionFactoryBuilder：SqlSessionFactory的创建者*

*通过该建造者对象调用建造方法，为我们创建一个SqlSessionFactory对象*

*SqlSessionFactory对象唯一的作用技术为我们创建SqlSession对象*

*\*/*

SqlSessionFactory sqlSessionFactory =new SqlSessionFactoryBuilder().build(inputStream);

*//我们未来所有的操作，使用的都是SqlSession对象*

*//例如增删改查，处理事务等等，都是统一使用session来完成*

SqlSession session = sqlSessionFactory.openSession();

*/\**

*需求：根据id查单条*

*如果取得的是单条记录，我们请用select方法*

*参数1：根据命名空间 .sqlId的形式找到我们需要使用的sql语句*

*参数2：我们要为sql语句中传递的参数*

*\*/*

*/\* Student student = session.selectOne("test1.getById","A0001");*

*System.out.println(student);*

*session.close();\*/*

*//查询学生表中的所有数据信息*

*/\*List<Student> studentlist = session.selectList("test1.getAll");*

*for (Student student:studentlist){*

*System.out.println(studentlist);*

*}*

*session.close();\*/*

*/\**

*MyBatis默认情况下是手动提交事务*

*添加、修改、删除都需要提交事务*

*\*/*

*//添加操作*

*/\*Student student = new Student();*

*student.setId("A0006");*

*student.setName("cxk");*

*student.setAge(29);*

*session.insert("test1.save",student);*

*session.commit();//手动提交事务*

*session.close();\*/*

*//修改操作*

*/\*Student student = new Student();*

*student.setId("A0006");*

*student.setName("cxk1");*

*student.setAge(30);*

*session.update("test1.update",student);*

*session.commit();*

*session.close();\*/*

*//删除操作*

session.delete("test1.delete","A0005");

session.commit();

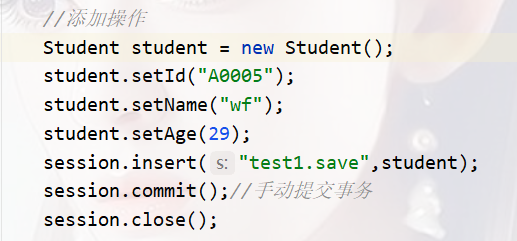
session.close();

}

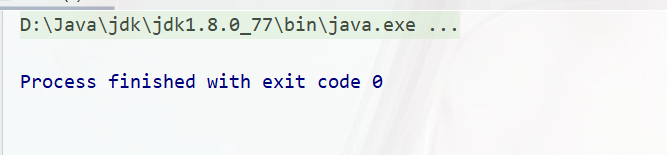
}

运行后的结果

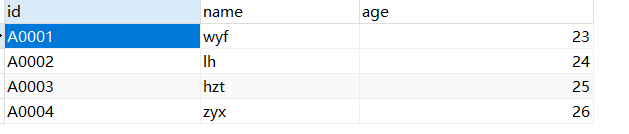
运行代码：



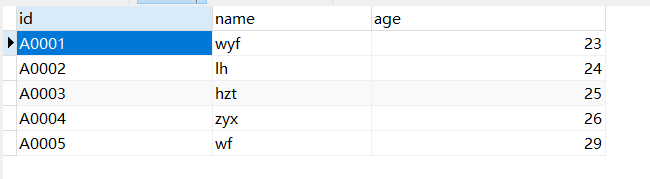
控制台显示：



数据库初始数据：



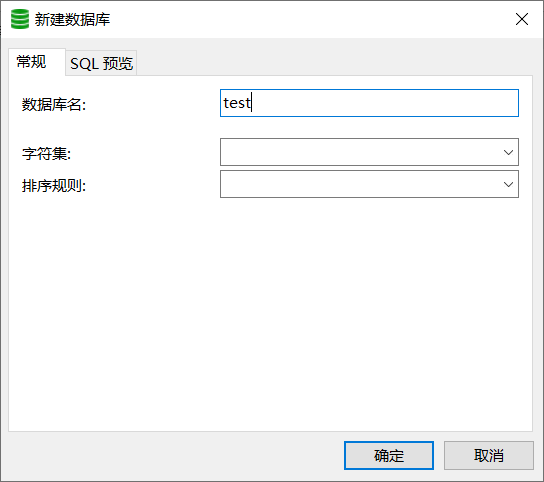
数据库添加后的数据：

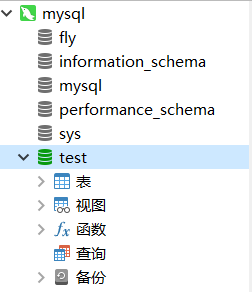


到此，使用mybatis操作数据库的增删改查就完成了。

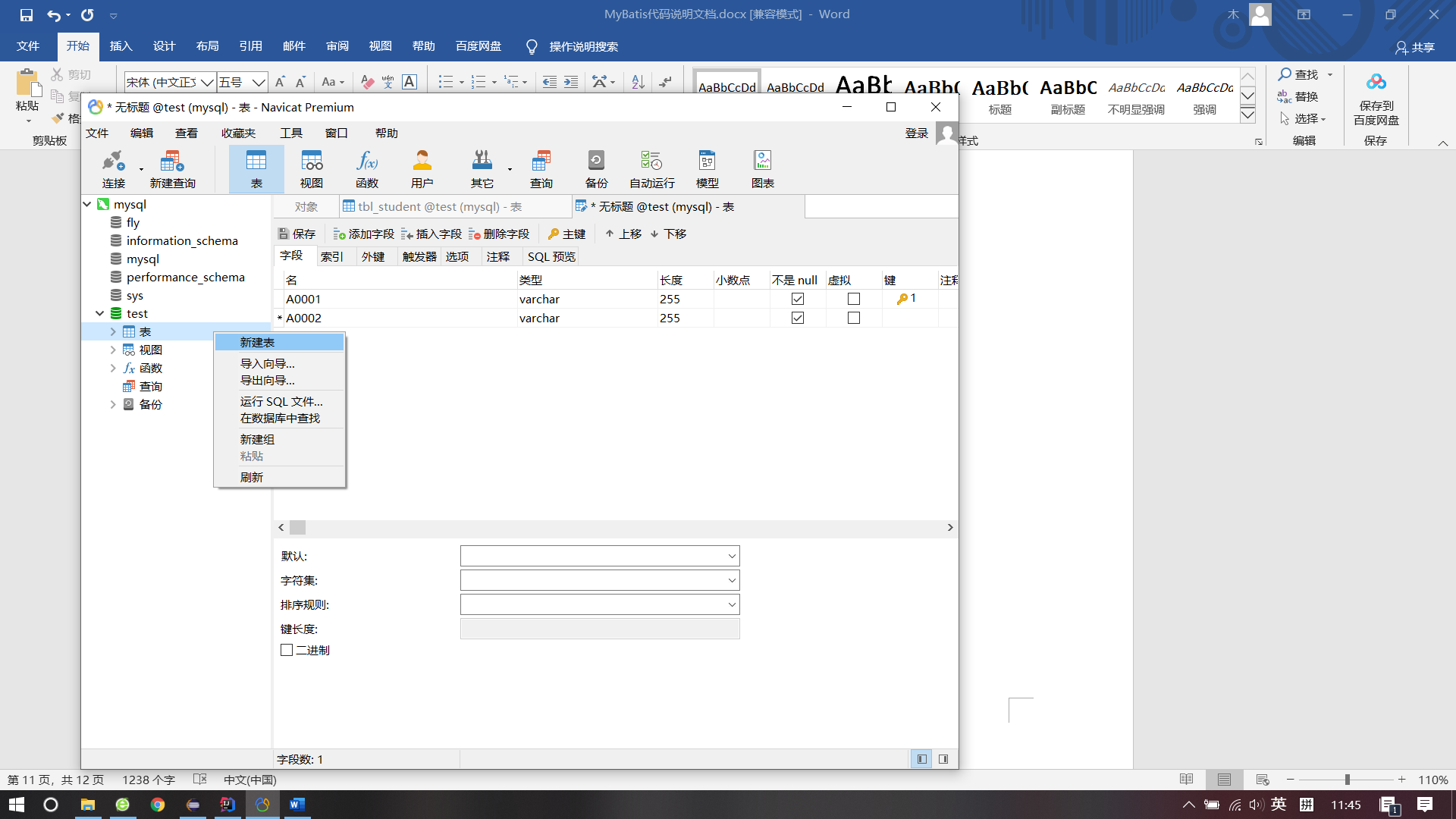
# 三、数据库操作

数据库的建立：建立test数据库并连接

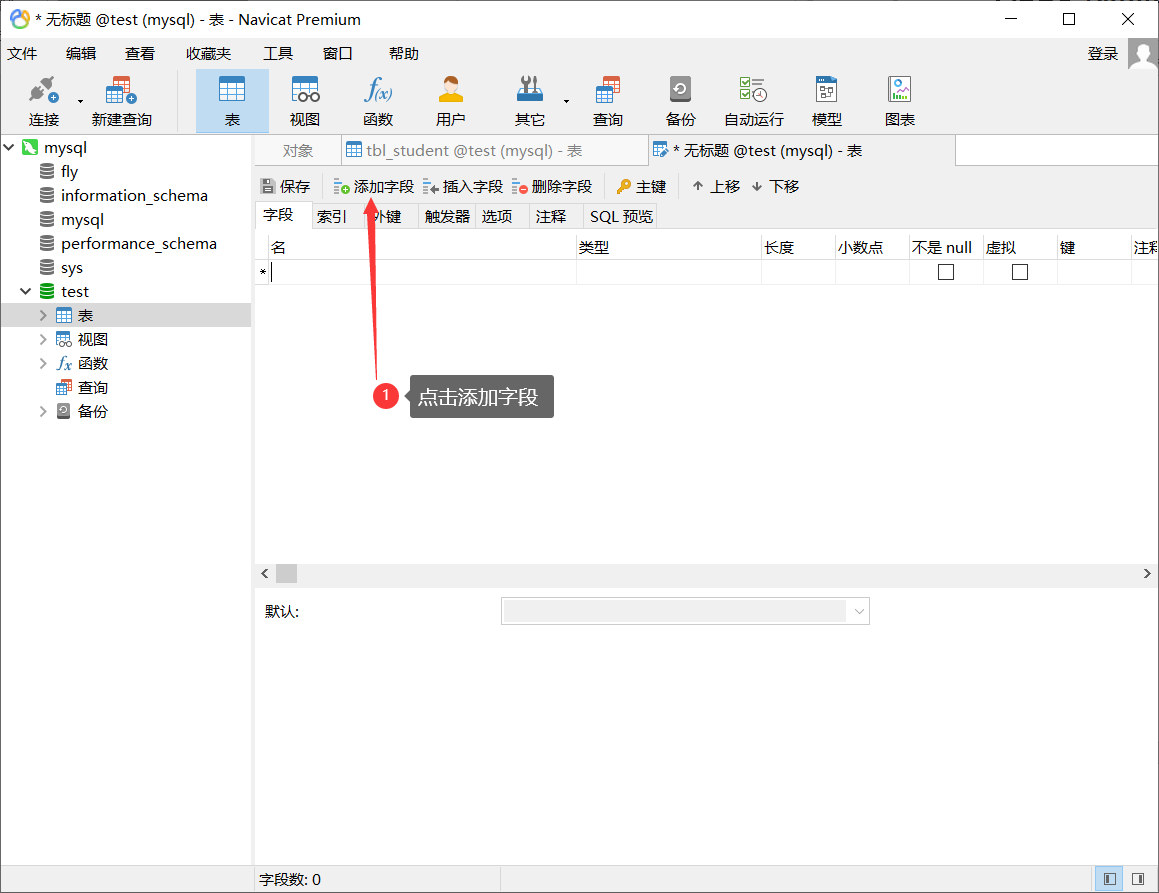




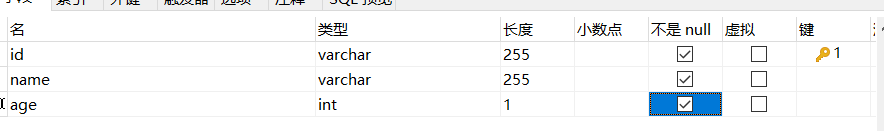
数据表的建立

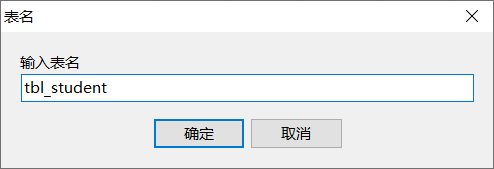


添加信息



设置属性点击保存





打开表，添加信息

