## 一、三层架构

1.界面层、业务逻辑层、数据访问层

**(1)**界面层（表示层，视图层):主要功能是接受用户的数据，显示请求的处理结果。使用web 页面和用户交互，手机 app 也就是表示层的，用户在app中操作，业务逻辑在服务器端处理。对应：controller包(servlet)

**(2)**业务逻辑层:接收表示传递过来的数据，检查数据，计算业务逻辑，调用数据访问层获取数据。service包(XXXService类)

**(3)**数据访问层:与数据库打交道。主要实现对数据的增、删、改、查。将存储在数据库中的数据提交给业务层，同时将业务层处理的数据保存到数据库。dao包(XXXDao类)

**(4)关系：**用户使用界面层<-->业务逻辑层<-->数据库访问层(持久层)<-->数据库(mysql)

**(5)**对应的框架：

——界面层：springmvc(servlet)

——业务逻辑层:spring(service类)

——数据访问层:dao类(mybatis),代替JDBC访问Mysql数据库

## 使用JDBC的缺陷

1. 代码比较多，开发效率低。
2. 需要关注Connection、Statement、ResultSet对象创建和销毁(关闭流资源)
3. 对ResultSet查询的结果，需要自己封装为List
4. 重复的代码比较多
5. 业务代码和数据库的操作混在一起

——DB工具类(DBUtils)：一定程度上减轻了重复代码，但Mybatis的出现，极大解决了JDBC缺陷问题。

## Mybatis

1. Mybatis是SQL映射框架（数据库表中的一行数据映射为一个java对象，操作该对象，等同于操作表数据）
2. 提供创建Connection、Statement、ResultSet的能力。
3. 提供执行SQL能力。
4. 提供循环SQL，把SQL的结果转为Java对象，并最终转换为List集合。
5. 提供关闭资源的能力。

过程：开发人员提供SQL语句——Mybatis处理SQL——开发人员得到List集合或Java对象(表中的数据)

**总结：**Mybatis是一个SQL映射框架，提供数据库的操作能力。增强的JDBC，使用Mybatis让开发人员集中精神写SQL就可以了，不必关心Connection、Statement、ResultSet的创建，销毁，以及sql的执行。

Mybatis官方中文文档地址：<https://mybatis.org/mybatis-3/zh/index.html>

练习：

学生表student：

**CREATE TABLE `student` (**

**`id` int NOT NULL,**

**`name` varchar(255) DEFAULT NULL,**

**`email` varchar(255) DEFAULT NULL,**

**`age` int NULL DEFAULT NULL,**

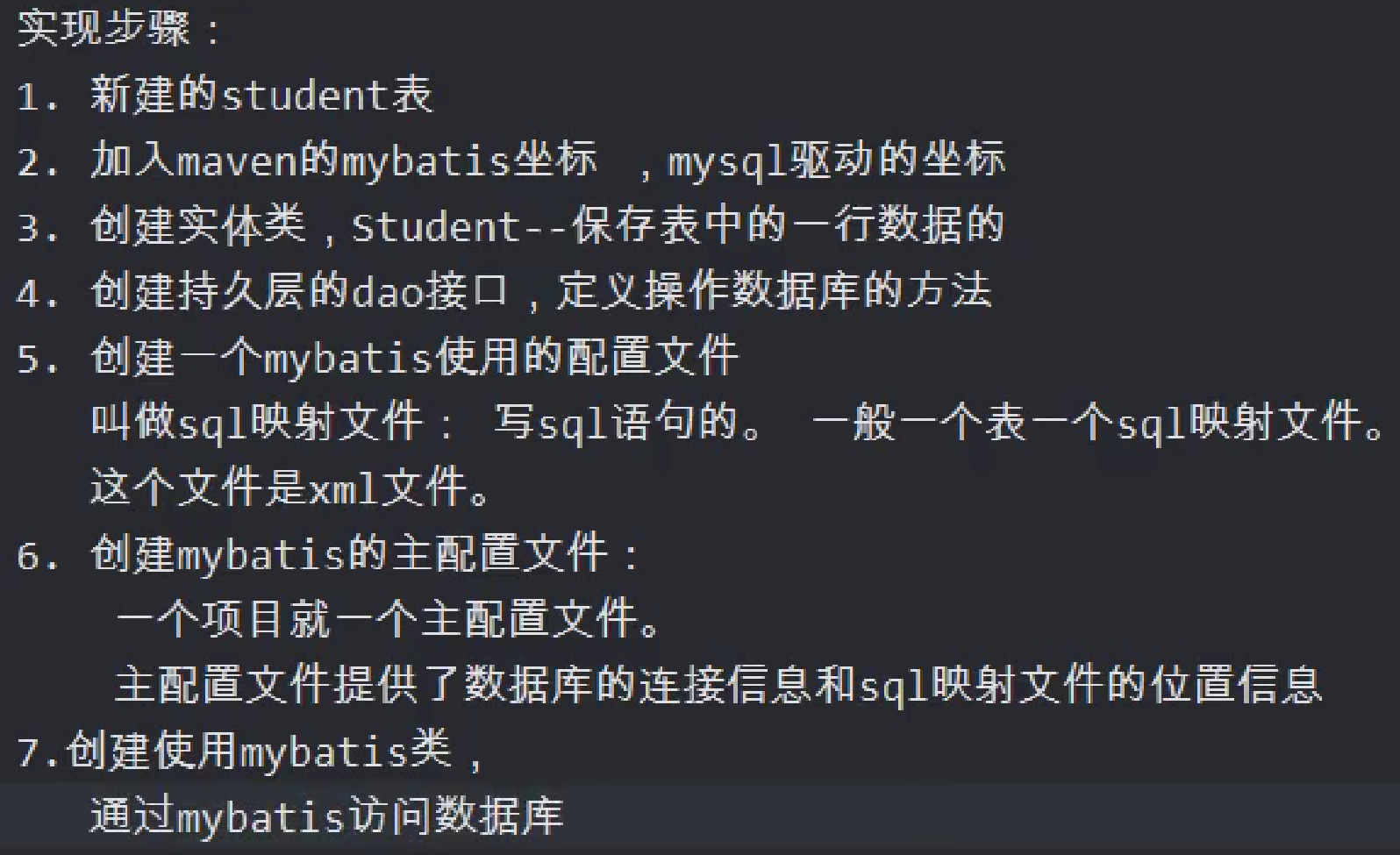
**PRIMARY KEY (`id`) USING BTREE**

**) ENGINE = InnoDB CHARACTER SET = utf8;**

插入数据：

**insert into student values(100,'张松','1210762604@qq.com','24');**

**insert into student [values(101,'李四','2393579405@qq.com','23');](mailto:values(101,'李四','2393579405@qq.com','23');)**



pom.xml引入maven依赖包

<!-- mybatis依赖 -->

<dependency>

<groupId>org.mybatis</groupId>

<artifactId>mybatis</artifactId>

<version>3.5.1</version>

</dependency>

<!-- mysql驱动依赖 -->

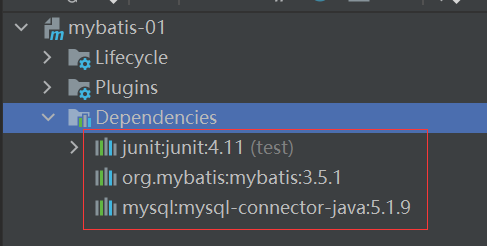
<dependency>

<groupId>mysql</groupId>

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

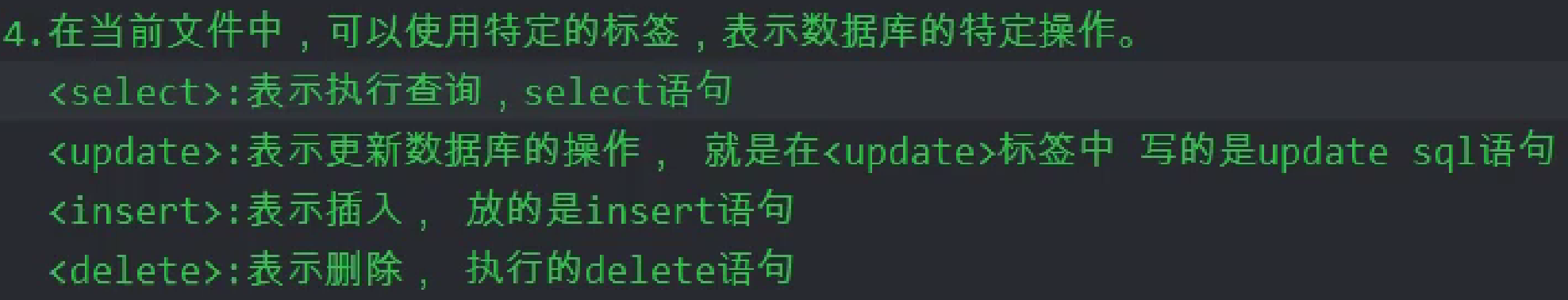
<version>5.1.9</version>

</dependency>



Mybatis映射文件：





<?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">

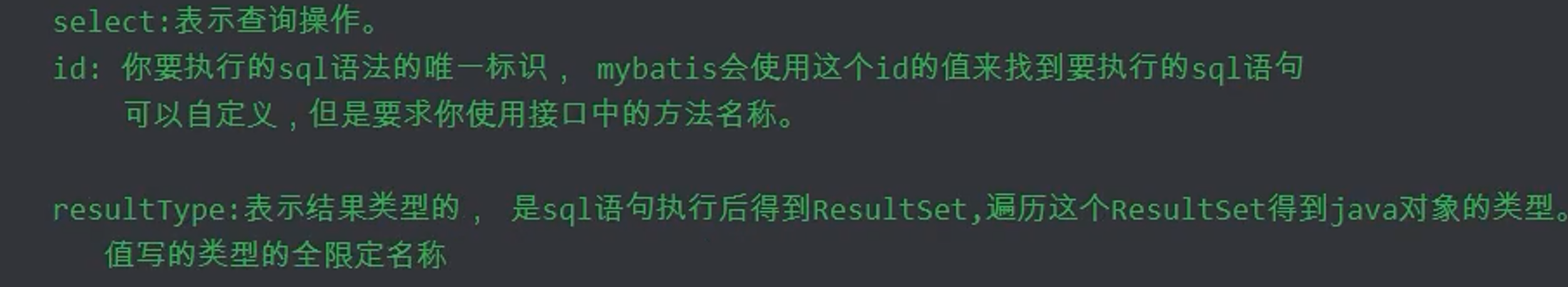
<mapper namespace="com.zsitking.dao.StudentDao">

<select id="selectAllData" resultType="com.zsitking.entity.Student">

select id,name,email,age from student order by id;

</select>

</mapper>



Mybaits连接数据库

在resources目录下新建mybatis-config.xml

<?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">

<!--配置数据库映射参数-->

<configuration>

<!--

环境配置：数据库的连接信息

default:必须和某个environment的id值一样。

告诉mybatis使用哪个数据库的连接信息。也就是访问哪个数据库

-->

<environments default="mydev">

<!-- 单个环境参数配置

transactionManager : mybatis的事务类型

type: JDBC(表示使用jdbc中的Connection对象的commit , rollback做事务处理)

-->

<environment id="mydev">

<transactionManager type="JDBC"/>

<!--

dataSource:表示数据源,连接数据库的

type:表示数据源的类型，POOLED表示使用连接池

-->

<dataSource type="POOLED">

<!-- 数据库的驱动类名 -->

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

<!-- 连接数据库的url字符串 -->

<property name="url" value="jdbc:mysql://localhost:3306/mybatis"/>

<!-- 数据库连接用户名你 -->

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

<!-- 数据库连接密码 -->

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

</dataSource>

</environment>

<!-- 其它环境参数配置 -->

</environments>

<mappers>

<mapper resource="com/zsitking/dao/StudentDao.xml"/>

</mappers>

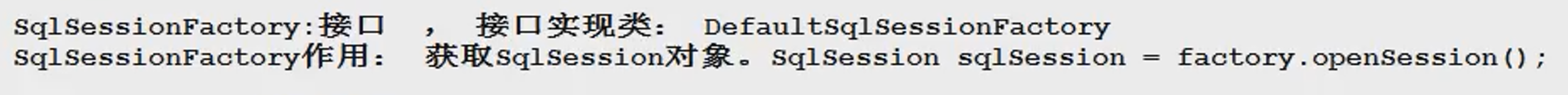
</configuration>

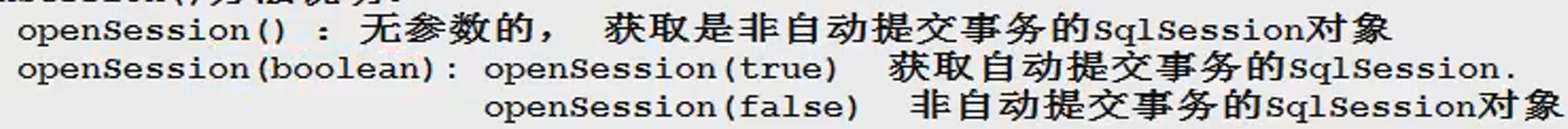
主要类介绍：

**（1）Resources**：mybatis中的一个类，负责读取主配置文件。例如——InputStream in = Resources.getResourceAsStream(“mybatis-config.xml”);

**（2）SqlSessionFactoryBuilder：**主要用于创建SqlSessionFactoryBuilder对象，使用其中的build方法来引用SqlSessionFactory类。

**（3）SqlSessionFactory：**重量级对象，程序创建一个对象耗时比较长，使用资源比较多。

**（4）openSession()方法说明**：



无参的**openSession()需要使用sqlSession.commit()提交事务。有参openSession(true)不需要sqlSession.commit().**

**（5）SqlSession：**

