黑马程序员

《Java EE 企业级应用开发教程（SSM）（第 2 版）》

教学设计

课程名称：Java EE 企业级应用开发教程

授课年级：XXXX年级

授课学期：第 X 学期

教师姓名：某某老师

2021年6月

<table>
<tr><td>课题名称</td><td>第 3 章 动态 SQL</td><td>计划课时</td><td>5 课时</td></tr>
<tr><td>教学引入</td><td colspan="3">在实际项目的开发中，开发人员在使用 JDBC 或其他持久层框架进行开发时，经常需要根据不同的条件拼接 SQL 语句，拼接 SQL 语句时还要确保不能遗漏必要的空格、标点符号等，这种编程方式非常不方便，而 MyBatis 提供的 SQL 语句动态组装功能，恰能解决这一问题。本章将针对 MyBatis 框架的动态 SQL 进行详细讲解。</td></tr>
<tr><td>教学目标</td><td colspan="3">● 使学生掌握 MyBatis 中动态 SQL 元素的使用
● 使学生掌握 MyBatis 的条件查询操作
● 使学生掌握 MyBatis 的更新操作
● 使学生掌握 MyBatis 的复杂查询操作</td></tr>
<tr><td>教学重点</td><td colspan="3">● <choose>、<when>、<otherwise>元素
● <where>元素
● 更新操作
● <foreach>元素迭代 List
● <foreach>元素迭代 Map</td></tr>
<tr><td>教学难点</td><td colspan="3">● <foreach>元素迭代 List
● <foreach>元素迭代 Map</td></tr>
<tr><td>教学方式</td><td colspan="3">课堂教学以 PPT 讲授为主，并结合多媒体进行教学</td></tr>
<tr><td>教学过程</td><td colspan="3">
第一课时
（动态 SQL 中的元素、<if>元素、<choose><when><otherwise>元素）
一、创设情景，导入新课
在进行数据库的操作时，涉及较多的便是拼接 SQL 语句。对于拼接好的 SQL 语句，如不小心输入了空格等字符，会导致程序运行报错。本节课，将对动态 SQL 中的元素、<if>元素、<choose><when><otherwise>元素等进行详细讲解。
二、新课讲解
知识点 1-动态 SQL 中的元素
教师通过多媒体演示 PPT 内容讲解动态 SQL 中的元素。
动态 SQL 是 MyBatis 的强大特性之一，MyBatis 采用了功能强大的基于 OGNL（Object Graph Navigation Language）的表达式来完成动态 SQL。在 MyBatis 的映射文件中，开发人员可通过动态 SQL 元素灵活组装 SQL 语句，这在很大程度上避免了单一 SQL 语句的反复堆砌，提高了 SQL 语句的复用性。
MyBatis 动态 SQL 中的常用元素：
<table>
<tr><th>元素</th><th>说明</th></tr>
<tr><td><if></td><td>判断语句，用于单条件判断</td></tr>
<tr><td><choose>(<when>、<otherwise>)</td><td>用于多条件判断</td></tr>
<tr><td><where></td><td>简化 SQL 语句中的 where 的条件判断</td></tr>
<tr><td><trim></td><td>去除多余的关键字</td></tr>
</table>
</td></tr>
</table>

<set>	用于 SQL 语句的动态更新
<foreach>	循环语句，常用于 in 语句等列举条件中

知识点 2-<if>元素

教师讲解<if>元素。

在 MyBatis 中，<if>元素是最常用的判断元素，它类似于 Java 中的 if 语句，主要用于实现某些简单的条件判断。

下面通过一个具体的案例演示单条件判断下<if>元素的使用，案例具体实现步骤如下。

1. 数据库准备

在名称为 mybatis 的数据库中，创建一个 t_customer 表，并插入几条测试数据。

2. POJO 类准备

在 com.itheima.pojo 包下创建持久化类 Customer，在类中声明 id、username、jobs 和 phone 属性，及属性对应的 getter/setter 方法。

注意：持久化类 Customer 与普通的 JavaBean 并没有什么区别，只是其属性字段与数据库中的表字段相对应。

3. 创建映射文件

在项目 com.itheima.mapper 包下创建映射文件 CustomerMapper.xml，在映射文件中，根据客户姓名和年龄组合成的条件查询客户信息，使用<if>元素编写该组合条件的动态 SQL。

<if>元素的 test 属性多用于条件判断语句中，用于判断真假，大部分的场景中都是进行非空判断，有时候也需要判断字符串、数字和枚举等。

4. 修改核心配置文件

在配置文件 mybatis-config.xml 中，引入 CustomerMapper.xml 映射文件，将 CustomerMapper.xml 映射文件加载到程序中。

5. 创建获取 SqlSession 对象的工作类

本案例使用 2.3.7 节的 MyBatisUtils 类作为获取 SqlSession 对象的工具类。首先在项目的 src/main/java 目录下创建一个 com.itheima.utils 包，然后将 2.3.7 节的 MyBatisUtils 类复制到该包下即可。

6. 修改测试类

在测试类 MyBatisTest 中，编写测试方法 findCustomerByNameAndJobsTest()，该方法用于根据客户姓名和职业组合条件查询客户信息列表。

7. 查看运行结果

执行 MyBatisTest 测试类的 findCustomerByNameAndJobsTest()方法。

知识点 3-<choose><when><otherwise>元素

教师讲解<choose><when><otherwise>元素。

在使用<if>元素时，只要 test 属性中的表达式为 true，就会执行元素中的条件语句，但是在实际应用中，有时只需要从多个选项中选择一个去执行。

例如下面的场景：

“当客户名称不为空，则只根据客户名称进行客户筛选；

当客户名称为空，而客户职业不为空，则只根据客户职业进行客户筛选。

当客户名称和客户职业都为空，则要求查询出所有电话不为空的客户信息。”

此种情况下，使用<if>元素进行处理是非常不合适的。针对上面情况，MyBatis 提供了<choose>、<when>、<otherwise>元素进行处理，这三个元素往往组合在一起使用，作用相当于 Java 语言中的 if…else if…else。

接下来，我们演示如何使用<choose>、<when>、<otherwise>元素组合去实现上面场景出现的情况，具体如下。

(1) 在映射文件 CustomerMapper.xml 中，添加使用<choose>、<when>、<otherwise>元素执行上述情况的动态 SQL。

(2) 在测试类 MyBatisTest 中，编写测试方法 findCustomerByNameOrJobsTest()，该方法用于根据客户姓名或职业查询客户信息列表。

三、归纳总结

教师回顾本节课所讲的内容，并通过测试题的方式引导学生解答问题并给予指导。

四、布置作业

教师通过高校教辅平台（http://tch.ityxb.com）布置本节课作业以及下节课的预习作业。

第二课时

（<where>元素、<trim>元素、更新操作、<foreach>元素中的属性）

一、复习巩固

教师通过上节课作业的完成情况，对学生吸收不好的知识点进行再次巩固讲解。

二、通过直接导入的方式导入新课

掌握了动态 SQL 中的元素以及条件查询操作的部分元素外，接下来继续学习条件查询操作中的剩余元素以及更新操作等。

三、新课讲解

知识点 1-<where>元素

教师讲解<where>元素。

在前两节的案例中，映射文件中编写的 SQL 后面都加入了“where 1=1”的条件，而加入了条件“1=1”后，既保证了 where 后面的条件成立，又避免了 where 后面第一个词是 and 或者 or 之类的关键字。如果将 CustomerMapper.xml 文件中 where 后“1=1”的条件去掉，MyBatis 所拼接出来的 SQL 语句如下所示。

```
select * from t_customer
   where and username like concat('%',?, '%')
      and
    jobs = #{jobs}
```

上述 SQL 语句中，where 后直接跟的是 and，这在运行时会报 SQL 语法错误，针对这种情况，可以使用 MyBatis 提供的<where>元素和<trim>元素进行处理。

例如下面的代码片段。

```
    <!-- <where>元素-->
    <select id="findCustomerByNameAndJobs"
            parameterType="com.itheima.pojo.Customer"
```

```
            resultType="com.itheima.pojo.Customer">
        select * from t_customer
        <where>
            <if test="username !=null and username !=''">
                and username like concat('%',#{username}, '%')
            </if>
            <if test="jobs !=null and jobs !=''">
                and jobs= #{jobs}
            </if>
        </where>
    </select>
```

上述配置代码中，使用<where>元素对“where 1=1”条件进行了替换，<where>元素会自动判断由组合条件拼装的 SQL 语句，只有<where>元素内的某一个或多个条件成立时，才会在拼接 SQL 中加入 where 关键字，否则将不会添加；即使 where 之后的内容有多余的“AND”或“OR”，<where>元素也会自动将他们去除。

知识点 2-<trim>元素

教师讲解<trim>元素。

<trim>元素用于删除多余的关键字，它可以直接实现<where>元素的功能。<trim>元素包含 4 个属性，具体如下表所示。

属性	说明
prefix	指定给 SQL 语句增加的前缀
prefixOverrides	指定 SQL 语句中要去掉的前缀字符串
suffix	指定给 SQL 语句增加的后缀
suffixOverrides	指定 SQL 语句中要去掉的后缀字符串

在实现 where 条件判断时，通常使用 prefix 和 prefixOverrides 属性。使用<trim>元素替换 where 1=1 条件，如下代码片段所示。

```
    <!-- <trim>元素-->
    <select id="findCustomerByNameAndJobs"
            parameterType="com.itheima.pojo.Customer"
            resultType="com.itheima.pojo.Customer">
        select * from t_customer
        <trim prefix="where" prefixOverrides="and">
            <if test="username !=null and username !=''">
                and username like concat('%',#{username}, '%')
            </if>
            <if test="jobs !=null and jobs !=''">
                and jobs= #{jobs}
            </if>
        </trim>
    </select>
```

上述配置代码中，使用<trim>元素对“where 1=1”条件进行了替换，<trim>

元素的作用是去除一些多余的前缀字符串，它的 prefix 属性代表的是语句的前缀（where），而 prefixOverrides 属性代表的是需要去除的前缀字符串（SQL 中的“AND”或“OR”）。

知识点 3-更新操作

教师讲解更新操作。

在实际应用中，大多数情况下都是更新某一个或几个字段。如果更新的每一条数据都要将其所有的属性都更新一遍，那么执行效率是非常差的。为了解决更新数据的效率问题，MyBatis 提供了<set>元素。

<set>元素主要用于更新操作，它可以在动态 SQL 语句前输出一个 SET 关键字，并将 SQL 语句中最后一个多余的逗号去除。<set>元素与<if>元素结合可以只更新需要更新的字段。

下面通过一个案例演示如何使用<set>元素更新数据库的信息，案例具体步骤如下。

(1) 在映射文件 CustomerMapper.xml 中，添加使用<set>元素执行更新操作的动态 SQL。

(2) 为了验证上述配置，可以在测试类中编写测试方法 updateCustomerBySetTest()。

注意：

在映射文件中使用<set>元素和<if>元素组合进行 update 语句动态 SQL 组装时，如果<set>元素内包含的内容都为空，则会出现 SQL 语法错误。因此，在使用<set>元素进行字段信息更新时，要确保传入的更新字段不能都为空。

除了使用<set>元素外，还可以通过<trim>元素来实现更新操作。

知识点 4-<foreach>元素中的属性

教师讲解<foreach>元素中的属性。

<foreach>元素主要用于遍历，能够支持数组、List 或 Set 接口的集合。在实际开发中，<foreach>元素通常和 SQL 语句中的 in 关键字结合使用。

接下来讲<foreach>元素包含的几个属性：

属性	说明
item	表示集合中每一个元素进行迭代时的别名。该属性为必选
index	在 List 和数组中，index 是元素的序号，在 Map 中，index 是元素的 key。该属性可选
open	表示 foreach 语句代码的开始符号，一般和 close=")"合用。常用在 in 条件语句中。该属性可选
separator	表示元素之间的分隔符，该属性可选
close	表示 foreach 语句代码的关闭符号，一般和 open="("合用。常用在 in 条件语句中。该属性可选
collection	用于指定遍历参数的类型。注意，

	该属性必须指定，不同情况下，该属性的值是不一样的。

collection 属性取值的三种情况：

(1) 若入参为单参数且参数类型是一个 List，collection 属性值为 list。

(2) 若入参为单参数且参数类型是一个数组，collection 属性值为 array。

若传入参数为多参数，就需要把参数封装为一个 Map 进行处理，collection 属性值为 Map。

四、归纳总结

教师回顾本节课所讲的内容，并通过测试题的方式引导学生解答问题并给予指导。

五、布置作业

教师通过高校教辅平台（http://tch.ityxb.com）布置本节课作业以及下节课的预习作业。

第三课时

（<foreach>元素迭代数组、<foreach>元素迭代 List、<foreach>元素迭代 Map、案例：学生信息查询系统）

一、复习巩固

教师通过上节课作业的完成情况，对学生吸收不好的知识点进行再次巩固讲解。

二、通过直接导入的方式导入新课

上节课掌握了条件查询操作的剩余元素以及更新操作等，接下来学习复杂的查询操作，如<foreach>元素的迭代操作。

三、新课讲解

知识点 1-<foreach>元素迭代数组

教师讲解<foreach>元素迭代数组。

<foreach>元素可以实现数组类型的输入参数的遍历。例如，要从数据表 t_customer 中查询出 id 为 1、2、3 的客户信息，如果采用单条查询的方式，势必会造成资源的浪费，此时就可以利用数组作为参数，存储 id 的属性值 1、2、3，并通过<foreach>元素迭代数组完成客户信息的批量查询操作。

<foreach>元素迭代数组的实现具体如下。

(1) 在映射文件 CustomerMapper.xml 中，添加使用<foreach>元素迭代数组执行批量查询操作的动态 SQL。

(2) 为了验证上述配置，可以在测试类 MyBatisTest 中，编写测试方法 findByArrayTest()。

知识点 2-<foreach>元素迭代 List

教师讲解<foreach>元素迭代 List。

在上一节中，使用<foreach>元素完成了客户信息的批量查询操作，方法参数为一个数组，现在我们更改参数类型，传入一个 List 类型的参数来实现同样的需求。

<foreach>元素迭代 List 的实现步骤具体如下。

(1) 在映射文件 CustomerMapper.xml 中，添加使用<foreach>元素迭代 List 集合执行批量查询操作的动态 SQL。

(2) 为了验证上述配置，可以在测试类 MyBatisTest 中，编写测试方法

findByListTest()。

知识点 3-<foreach>元素迭代 Map

教师讲解<foreach>元素迭代 Map。

在前两节中，使用<foreach>元素完成了客户信息的批量查询操作，MyBatis 入参均为一个参数，如果入参为多个参数，例如，需要查询出性别是男性并且职业为老师的所有客户信息。这时，就需要把这些参数封装成一个 Map 集合进行处理。

下面通过一个案例演示如何使用<foreach>元素迭代 Map 集合，实现多参数入参查询操作，案例具体实现步骤如下。

(1) 在映射文件 CustomerMapper.xml 中，添加使用<foreach>元素迭代 Map 集合执行批量查询操作的动态 SQL。

(2) 为了验证上述配置，在测试类 MyBatisTest 中，编写测试方法 findByMapTest()。

知识点 4-案例：学生信息查询系统

教师讲解案例：学生信息查询系统。

案例目标：

案例要求利用本章所学知识完成一个学生信息查询系统，该系统要求实现 2 个以下功能。

（1）多条件查询

当用户输入的学生姓名不为空，则只根据学生姓名进行学生信息的查询；

当用户输入的学生姓名为空，而学生专业不为空，则只根据学生专业进行学生的查询；

当用户输入的学生姓名和专业都为空，则要求查询出所有学号不为空的学生信息。

（2）单条件查询查询出所有 id 值小于 5 的学生的信息。

实现步骤：多条件查询

1. 项目搭建

创建一个名称为 mybatis-demo03 的项目，项目的具体搭建过程请参考 1.3 节。

2. 数据准备

在名称为 mybatis 的数据库中，创建一个 dm_student 表，并插入几条测试数据。

3. POJO 类准备

在项目 src/main/java 目录下创建 com.itheima.pojo 包，在 com.itheima.pojo 包下创建持久化类 Student，在类中声明 id、name、major 和 sno 属性，以及属性对应的 getter/setter 方法。

4. 创建映射文件

在项目 src/main/java 目录下创建 com.itheima.mapper 包，在 com.itheima.mapper 包下创建映射文件 StudentMapper.xml，在映射文件中，编写根据学生姓名和专业组合成的条件查询学生信息的动态 SQL。

5. 修改 mybatis-config.xml 核心配置文件

在 mybatis-config.xml 映射文件的<mappers>元素下添加 StudentMapper

.xml 映射文件路径的配置，用于将 StudentMapper.xml 映射文件加载到程序中。

6. 编写 MyBatisUtils 工具类

在项目 src/main/java 目录下创建 com.itheima.utils 包，在 com.itheima.utils 包下创建 MyBatisUtils 工具类，该类用于封装读取配置文件信息的代码。

7. 编写测试方法

为了验证上述配置，可以在测试类 MyBatisTest 中，编写测试方法 findStudentByNameOrMajorTest()，该方法用于根据学生姓名或专业查询学生信息。

实现步骤：单条件查询

1. 修改映射文件

在映射文件 StudentMapper.xml 中的<mapper>元素下，编写查询所有 id 值小于 5 的学生信息的动态 SQL。

2. 编写测试方法

为了验证上述配置，可以在测试类 MyBatisTest 中，编写测试方法 findByListTest()。

四、归纳总结

教师回顾本节课所讲的内容，并通过测试题的方式引导学生解答问题并给予指导。

五、布置作业

教师通过高校教辅平台（http://tch.ityxb.com）布置本节课作业以及下节课的预习作业。

第四、五课时（上机练习）

上机一：（考察知识点为动态 SQL 中的元素、<if>元素、<choose><when><otherwise>元素、<where>元素、<trim>元素、更新操作、<foreach>元素中的属性、<foreach>元素迭代数组、<foreach>元素迭代 List、<foreach>元素迭代 Map、案例：学生信息查询系统）

形式：单独完成

题目：

要求利用本章所学知识完成一个学生信息查询系统，该系统要求实现 2 个以下功能。

（1）多条件查询

当用户输入的学生姓名不为空，则只根据学生姓名进行学生信息的查询；

当用户输入的学生姓名为空，而学生专业不为空，则只根据学生专业进行学生的查询；

当用户输入的学生姓名和专业都为空，则要求查询出所有学号不为空的学生信息。

（2）单条件查询查询出所有 id 值小于 5 的学生的信息。

注意命名要求：创建的项目名称为 mybatis-demo03；

教学后记	