动态SQL

1. 什么是动态SQL

- mybatis核心：对sql语句进行灵活操作，通过表达式进行判断，对sql进行灵活拼接、组装。

2. 需求

- 用户信息综合查询列表和用户信息查询列表总数这两个statement的定义使用动态sql。
- 对查询条件进行判断，如果输入参数不为空，才进行查询条件拼接。

3. mapper.xml

```
<!-- 用户信息的综合查询
 #{userCustom.sex}：取出pojo包装对象中性别的值。
 ${userCustom.username}：取出pojo包装对象中用户名称。
 -->
<select id="findUserList" parameterType="com.swxc.mybatis.po.UserQueryVo" resultType="com.swxc.mybatis.po.UserCustom">
    select * from user
    <!--
    where可以自动去掉条件中的第一个and
    -->
    <where>
        <if test="userCustom != null">
            <if test="userCustom.sex != null and userCustom.sex != ''">
                and user.sex = #{userCustom.sex}
            </if>
            <if test="userCustom.username != null and userCustom.username != ''">
                and user.username like '%${userCustom.username}%'
            </if>
        </if>
    </where>
</select>

<!-- 用户信息的综合查询总数 -->
<select id="findUserCount" parameterType="com.swxc.mybatis.po.UserQueryVo" resultType="int">
    select count(*) from user
    <where>
        <if test="userCustom != null">
            <if test="userCustom.sex != null and userCustom.sex != ''">
                and user.sex = #{userCustom.sex}
            </if>
            <if test="userCustom.username != null and userCustom.username != ''">
                and user.username like '%${userCustom.username}%'
            </if>
        </if>
    </where>
</select>
```

4. 测试

```
// 用户信息的综合查询
@Test
public void findUserListTest() throws Exception {
    SqlSession sqlSession = sqlSessionFactory.openSession();
    UserMapper userMapper = sqlSession.getMapper(UserMapper.class);
```

```
    UserQueryVo userQueryVo = new UserQueryVo();
    UserCustom userCustom = new UserCustom();
    // userCustom.setSex("g");
    userCustom.setUsername("五");
    userQueryVo.setUserCustom(userCustom);
    List list = userMapper.findUserList(userQueryVo);
    sqlSession.close();
    System.out.println(list);
}
```

5. sql片段

1. 需求

- 将以上实现的动态sql判断代码抽取出来，组成一个sql片段。其他的statement中就可以引用该sql片段。

2. 定义sql片段

```
<!-- 定义sql片段
 id：sql片段的唯一标识
 经验：是基于单表定义定义sql片段，这样可重用性才高
        在sql片段中不要包括where
 -->
<sql id="query_user_where">
    <if test="userCustom != null">
        <if test="userCustom.sex != null and userCustom.sex != ''">
            and user.sex = #{userCustom.sex}
        </if>
        <if test="userCustom.username != null and userCustom.username != ''">
            and user.username like '%${userCustom.username}%'
        </if>
    </if>
</sql>
```

3. 引用sql片段

```
<select id="findUserList" parameterType="com.swxc.mybatis.po.UserQueryVo" resultType="com.swxc.mybatis.po.UserCustom">
    select * from user
    <!--
    where可以自动去掉条件中的第一个and
    -->
    <where>
        <!-- refid：引用sql片段的id，如果refid指定的id不在本mapper文件中，需要添加namespace -->
        <include refid="query_user_where" />
        <!-- 在这里还要引用其他的sql片段 -->
    </where>
</select>
```

```
<!-- 用户信息的综合查询总数 -->
<select id="findUserCount" parameterType="com.swxc.mybatis.po.UserQueryVo" resultType="int">
    select count(*) from user
    <where>
        <include refid="query_user_where" />
    </where>
</select>
```

6. foreach

```
向sql传递数组或List，mybatis使用foreach解析。
```

1. 需求

- 在用户查询列表和查询总数的statement中增加多个id输入查询。

- sql语句如下：
  - select * from user where id=1 or id=10 or id=16
  - select * from user where id in (1,10,16)

2. 在输入参数类型中添加Listids传入多个id

- UserQueryVo类中添加属性ids，类型为List，包括其get和set方法。

3. 修改mapper.xml

- where id=1 or id=10 or id=16
- 在查询条件中，查询条件定义成一个sql片段，需要修改sql片段。

```
<sql id="query_user_where">
    <if test="userCustom != null">
        <if test="userCustom.sex != null and userCustom.sex != ''">
            and user.sex = #{userCustom.sex}
        </if>
        <if test="userCustom.username != null and userCustom.username != ''">
            and user.username like '%${userCustom.username}%'
        </if>
        <if test="ids != null">
            <!-- 使用foreach遍历传入ids
             collection：指定输入对象中集合属性
             item：每个遍历生成的对象名
             open：开始遍历时拼接的串
             close：结束遍历时拼接的串
             separator：遍历的两个对象中间需要拼接的串 or
             -->
            <foreach collection="ids" item="user_id" open="and (" close=")" separator="or">
                <!-- 每次遍历需要拼接的串 -->
                id = #{user_id}
            </foreach>
        </if>
    </if>
</sql>
```

4. 测试

```
// 用户信息的综合查询
@Test
public void findUserListTest() throws Exception {
    SqlSession sqlSession = sqlSessionFactory.openSession();
    UserMapper userMapper = sqlSession.getMapper(UserMapper.class);
    UserQueryVo userQueryVo = new UserQueryVo();
    UserCustom userCustom = new UserCustom();
    // userCustom.setSex("g");
    userCustom.setUsername("五");
    userQueryVo.setUserCustom(userCustom);
    List ids = new ArrayList();
    ids.add(1);
    ids.add(2);
    ids.add(3);
    ids.add(4);
    ids.add(5);
    userQueryVo.setIds(ids);
    List list = userMapper.findUserList(userQueryVo);
    sqlSession.close();
    System.out.println(list);
}
```