链滴

Mybatis 杂记 (二)

作者：HuixiaZhang
原文链接：https://ld246.com/article/1539602701953
来源网站：链滴

Mybatis杂记:

当数据库涉及到一对多时Mybatis的处理方法:

● 例如在bean中定义如下两个类(Order 类和User类)，用户和订单是一对多关系，1个用户可能会有个订单，但是一个订单只属于一个用户。

此时你需要在那个1的地方定义一个集合，Set和List皆可!

```
public class Order {
    private long id;
    private String name;
    private double price;
    public User user;
    //Getter and Setter
    //toString
    //Constuctor
}
public class User {
    private long id;
    private String name;
    private Set<Order> orders;
     //Getter and Setter
    //toString
    //Constuctor
}
```

● 扯到一点插入语句在Mybatis中的使用！例如在User表中插入数据时:

数据库：Oracle

主键通过序列进行维护时，就可以用selectKey来引入我们的主键。

order中参数有两种:

1.AFTER:在insert之后执行

2.BEFORE:在insert之前执行

keyProperty:查询到的结果(这里即序列的值)将要赋给哪个列

resultType:查询到的结果的属性(即keyProperty中的属性)

```
<insert id="saveUser" parameterType="user">
        <selectKey order="BEFORE" keyProperty="id" resultType="long">
            select u_seq.nextval from dual
        </selectKey>
        insert into s_user(id,name) values(#{id},#{name})
    </insert>
```

● 做一些查询的demo:

1.基于User的id去查询所有的信息:

在Mapper接口中定义：User findUserAndOrders(long id);

Mapper.xml中:

同样的套路，先定义一个order的resultMap：

```
<resultMap id="order_mod1" type="order">
        <id property="id" column="ids"/>
        <result property="name" column="names"/>
        <result property="price" column="price"/>
    </resultMap>
```

随后定义User的resultMap：

一对一的时候用的是association，这里使用collection来描述“多”的这个关系，另一方面，在bean写的也是个集合！

在collection中，property参数根据之前谈过的规则(Mybatis会自动去搜寻getXX()方法，并将其以掉get，并取后面字段小写的方式来获取属性名),resultMap参数写的是刚才写的id 为order_mod1的rsultMap.

```
<resultMap id="user_mod1" type="user" >
    <id property="id" column="id"/>
    <result property="name" column="name"/>
    <!-- 表示集合的封装 property指向封装对象中的集合引用变量，引用对象 -->
    <collection property="orders" resultMap="order_mod1" />
</resultMap>
```

这样的话 select就可以这么写：

```
<select id="findUserAndOrders" parameterType="long" resultMap="user_mod1">
    select s.id,s.NAME,d.id ids,d.name names,d.PRICE,d.USER_ID
    from S_USER s, S_ORDER d
    where s.ID = d.USER_ID and s.ID = #{id}
</select>
```

2.根据用户的id去寻找所有的Order：

在Mapper中定义：Set<Order> selectOrderByUser_id(long id);

Mapper.xml中：

```
<select id="selectOrderByUser_id" parameterType="long" resultMap="order_mod1">
    select id ids,name names,PRICE
    from S_ORDER
    where USER_ID = #{id}
</select>
```

3.查询所有的用户及订单：

在Mapper接口中定义：List<User> selectUserandOrder();

Mapper.xml中:

在collection中 select中填写的是刚才2中的操作(根据用户的id去寻找所有的Order)，column属性又把ser_Id传过去，查询到的所有数据就会拼接在property的属性中，这样就完成了查询所有数据的能力！

```
<resultMap id="mod_user" type="user">
    <id property="id" column="id"/>
    <result property="name" column="name"/>
    <!-- 基于当前查询的用户 id 去 order 表找订单 -->
    <collection property="orders" column="id" select="selectOrderByUser_id"/>
</resultMap>
```

select标签：

```
<select id="selectUserandOrder"  resultMap="mod_user">
    select s.id,s.NAME,d.id ids,d.NAME names,d.PRICE,d.USER_ID
    from S_USER s, S_ORDER d
    where s.ID = d.USER_ID
</select>
```