**Mybatis02 使用ModelMapper替代dao**

**前置一致：jar,c3p0配置文件，log配置文件，model类**

**后置需要：modelMapper接口，modelMapper配置文件，mybatis全局配置文件**

**需要注意的点：**

1. **ModelMapper接口 + ModelMapper.xml**
2. **ModelMapper.xml中的<mapper >标签的属性namespace ，需要设置为ModelMapper接口的包路径**

**示例：**

**<mapper namespace="cn.cxd.modelMapper.UserMapper">**

1. **ModelMapper接口的方法 和 ModelMapper.xml的标签中的方法，在参数、返回值、id上要能对应的上，其中id就是接口中的方法名称。在id是唯一的情况下，我们似乎在这种情况下想多态似乎成为了问题。但是可以在查询条件上有想法，因为有<if>标签的帮助，参考一.4.a)【parameType】。**
2. **ModelMapper.xml中标签的parameterType , resultType 也许会是一个Model对象，鉴于每次都要输入一串很长的model路径，我们可以选择对全局配置文件下功夫，参考二.1全局配置文件的【为类设置别名】。**
3. **ModelMapper接口有时涉及的方法并不单是要处理一个表中的数据，因此要为ModelMapper.xml考虑参数传入设置和返回值的接收问题。其中：**
   1. **parameType只能接收一个对象，但是我们可以将若干信息封装成一个Map或是一个更复杂的一个Vo对象（其中包含了Model类对象可作为参数使用）。**

**参数会作为查询条件where的重要成分。由于存在<if>标签，因此我们可以在这里考虑多态：需要将各种可能需要出现的参数聚合到一个对象上，聚合的方式：**

* + - 1. **可以是简单不大好的单层的聚合，就是单纯的将所有参数堆砌到一个对象上，**
      2. **也可以是更好一些的分层次将一些信息聚合到一个子对象上，再由这个子对象与其他子对象聚合成一个大的目标对象。**

**这样的多态，实现要依赖于<if>标签，<if>标签可以多层次的出现，test属性作为判断条件，可以使用and, or关联起多个条件。对于条件可以做一个值的判断，使用 != / == 。对于聚合了model类或是集合的更复杂一点的类作为本次CURD的条件：**

* + - * 1. **Model assciation: #{modelName.propertyName} 代表参数类parameterType的model类的 propertyName成员变量**
        2. **List Collection: 使用<foreach>标签，存在属性**

**Collection : 参数类parameterType的CollectionName**

**Item : 每一次遍历取值而赋的变量名，将被用到<foreach>标签体中，#{itemName}**

**Open : 为了构造完整的 用于使用集合元素 的sql语句而开头的若干个词语**

**Close : 为了构造完整的 用于使用集合元素 的sql语句而结尾的若干个词语**

**Separator : 为了构造完整的 用于使用集合元素 的sql语句而用于连接各个元素的若干个词语**

**Index : 索引，可以直接设为”index”**

**示例：**

**<if test="idsList != null">**

**<foreach collection="idsList" item="id" open="id in (" close=")" separ ator="," index="index">**

**#{id}**

**</foreach>**

**</if>**

* 1. **resultType 也是仅能接收一个对象，我们可以构造一个用于接受结果的类，当然这个类也许也能承担传递多个参数的任务，而这个类在接收查询结果上需要借助resultMap的帮助，完成结合了集合和Model对象的返回值接收上的问题。其中resultMap：**
     + 1. **type 属性：代表这个ResultMap映射到那个类上，也就是说，这个resultMap并不是凭空捏造的，只是一种映射，它需要一个实际存在的类去存储所需要的值。**
       2. **id 用于标识唯一的值，也就是说当设置的id一致时，认为是同一个对象，将查询到的结果放入到映射的对象中。其中id是重中之重，如果没有设置id，那么数据库查询出来的结果对于这个映射类而言没有id，每一条数据就对应一个对象，那么一对多和多对多的关系就没有办法使用映射类表现出来了。**
       3. **association 用于为在映射类中的model类的成员变量设置内容，再将这些设置好了的成员对象返回为一个Model类，需要属性JavaType:指定model类，具备id result标签可以使用。**
       4. **collection 用于为映射类中的model List集合设置各个Model的成员变量内容，再返回为若干个model类对象的集合，需要属性ofType:指定集合中的model类，具备id result标签可以使用。如果是普通类型的集合，则设置ofType后即可。**
       5. **其中各个标签的属性property代表这一层标签的父标签的type类中的成员变量，而column则代表sql查询出来的表的各个列的名称。当property和column并列时，代表将sql查询出来的值映射到property指定的成员变量中。当property和type并列时，代表将type指定的类映射到property指定的成员变量中，明显这个成员变量也是一个类。**

**示例：**

1. <resultMap id="UserByIdRecordOrders" type="cn.cxd.modelVo.UserVo">
2. <id property="id" column="u\_id"></id>
3. <association property="user" javaType="user">
4. <id property="id" column="u\_id"></id>
5. <result property="username" column="u\_username"></result>
6. <result property="password" column="u\_password"></result>
7. </association>
8. <collection property="idsList" ofType="Integer"></collection>
9. <collection property="ordersList" ofType="order">
10. <id property="id" column="o\_id"></id>
11. <result property="userId" column="u\_id"></result>
12. <result property="productName" column="product"></result>
13. <result property="price" column="price"></result>
14. </collection>
15. </resultMap>
16. **Mapper.xml中可以使用<sql>标签将where子句抽取出来，作为可复用的一个模块，在需要使用这个条件模块时，使用<where>标签将这个模块，设置好refid引用即可。**
17. **Mapper.xml中的方法中的返回值resultType / resultMap 在于 modelMapper接口中声明的返回值类型不一致的情况下，返回值会是resultType / resultMap，这说明调用的还是mapper.xml中的方法，只是在main方法的方法接收的返回值还是modelMapper接口的返回值，这可能就是我们认为的阳奉阴违。**

**示例：**

1. List<Integer> userByArrayUnionList = mapper.findUserByArrayInListRecordIdsToArray(userVo);
2. System.out.println("result after: "+userByArrayUnionList);
4. result after: [User{id=1, username='yotu', password='123', age=0, kda=0},
5. User{id=3, username='leiyy', password='123', age=0, kda=0}]
6. **全局配置文件xml**

**1. 为ModelMapperXml需要使用的model类取别名，以方便使用，不必每次都要使用包名。**

**a. 需要使用<typeAlias>标签，在这个内部再使用<typeAlia>标签，其中type代表类，alias代表别名。**

**b. 也可以在<typeAlias>标签里直接使用<package>包标签，代表包下的类都取别名：首字母小写，其余字符一致。**

**2. 全局配置文件在<configuration>下的标签声明是有顺序的，如果乱放会Error**

**properties,**

**settings,**

**typeAliases,**

**typeHandlers,**

**objectFactory,**

**objectWrapperFactory,**

**reflectorFactory,**

**plugins,**

**environments,**

**databaseIdProvider,**

**mappers".**

**3. 全局配置文件，设置<mappers>标签读取局部配置xml文件，有4种方式：**

1. <mapper resource="cn/cxd/modelMapper/userMapper.xml"/>
2. <mapper **class**="cn.cxd.modelMapper.UserMapper"></mapper>
3. <**package** name="cn.cxd.modelMapper"/>

**4. <mapper url>**

**需要注意：其中的<package>包标签的配置仅适用于Mapper，不使用于Dao**

**在加载了全局配置文件，会读入<mapper>标签中的ModelMapper.xml，之后在getMapper(xxx.class)时，就会使用这个namespace在已读入的modelMapper.xml中配对上就能使用了，使用getMapper时，获取到实现了modelMapper接口的代理类，使用接口方法时，就会用上已经找到的配对好的modelMapper.xml来协助完成接口方法的调用。**