一、动态SQL查询回顾

需求：当用户选择1，查询数据库中的computer表中的所有数据；当用户选择2的时候，查询数据库中users表中所有的数据。

使用的标签：choose、when、otherwise

1，局部配置文件中的配置信息：

```
<select id="IFELSE" parameterType="map" resultType="map">
        <choose>
            <!-- 如果test中的条件是true的话，执行when中的语句，
  执行的顺序是从上到下，先执行第一个when，再执行第二个when，
  如果还有when继续执行，否则执行otherwise中的语句，
  但是注意：一次执行只能进入一个when或者进入otherwise-->
            <when test="type==1">
                select *from computer;
            </when>
            <when test="type==2">
                select *from users;
            </when>
    <otherwise>
            select *from users;
        </otherwise>
        </choose>
    </select>
```

java中Junit测试demo：

```
@Test
    public void IFELSE(){
        Map<String, Object> map = new HashMap<String, Object>();
        map.put("type", 2);
        List<Map<String, Object>> maps =
session.selectList("cn.dao.userDaoImpl.userDao.IFELSE",map);
        for (Map<String, Object> map2 : maps) {
            System.out.println(map2);
        }
    }
```

二，SQL动态添加

1，局部配置文件的配置信息：

```
<!-- sql动态修改，注意修改、删除、添加没有返回值属性，默认返回的是一个整数 -->
    <!-- 当map中什么数据都不传递的时候，就会报错的解决办法：
    在set标签中所有判断的最后加上where后的字段名=#{where后的字段名}，这样的
话，
    就相当于"update computer set where后字段名=null where where后字段名=null;"
    这样就有利于解决当一个参数都不进行传递的时候SQL语句报错的问题。
     -->
```

```
    <update id="dynamicUpdate" parameterType="map">
        update computer
        <set>
            <!-- set标签用于嵌套if标签， 如果最后一个为空，可以去掉多余的逗号 “，”
  set标签会自动去掉倒数第二条记录后面的逗号 “，”
  尽量不要传递四个都为空的状态，这样的目前处理不了。 -->
            <if test="computerName!=null">
                computerName = #{computerName},
            </if>
            <if test="brand!=null">
                brand = #{brand},
            </if>
            <if test="runMem!=null">
                <!-- 在test中，不需要使用#{}就可以获取到参数的值 -->
                runMem = #{runMem},
            </if>
            <if test="price!=null">
                price =#{price},
            </if>
            id=#{id}
        </set>
        where id = #{id};
    </update>
```

2，Junit的测试demo代码：

```
@Test
    public void dynamicUpdate(){
        Map<String, Object> map = new HashMap<String, Object>();
//          map.put("computerName", "华硕0-1-1");
//          map.put("brand", "华硕");
//          map.put("runMem", 32);
//          map.put("price", 66666);
            Integer integer = session.update("cn.dao.userDaoImpl.userDao.dynamicUpdate", map);
            //注意mybatis插入、删除、修改数据的时候，一定要提交事务，否则相应的数据不会对数据库进行更改。
            session.commit();
            System.out.println(integer);
    }
```