# baomidou

## selectOne() 报错：查到了多条数据。

改用：com.baomidou.mybatisplus.extension.toolkit.SqlHelper#getObject

出处：com.baomidou.mybatisplus.extension.service.impl.ServiceImpl#getOne

## 如何知晓executeBatch()更新了多少条？

请见：xyz.kuilei.spring.demo.common.util.DemoSqlHelper#executeBatch

## 字段名为id时，使用Oracle sequence.NEXTVAL总是报insert null。

请见：xyz.kuilei.spring.demo.bean.entity.TGroupEntity#GROUP\_ID

## 如何让executeBatch()执行自定义的单entity sql？

请见：xyz.kuilei.spring.demo.txnservice.impl.TxnGroupServiceImpl#txnMemberBatchAdd

## executeBatch()的entity list可以是map/json list。

方法reference如下：

com.baomidou.mybatisplus.extension.toolkit.SqlHelper#executeBatch(

java.lang.Class<?>,

org.apache.ibatis.logging.Log,

java.util.Collection<E>,

int,

java.util.function.BiConsumer<org.apache.ibatis.session.SqlSession,E>

)

说明：

1. 参数1必须是entity class，试过改成Map.class或JSONObject.class直接报错；
2. 参数3（entity list）可以是map/json list。

来源：

同事不想把JSON转成entity再executeBatch()，想着能不能直接用，弄下来居然可以。

推测：

参数1是在告诉baomidou（或者说mybatis）这个数据对象有哪些key。

debug跟踪：

<I> org.apache.ibatis.session.SqlSession#insert(java.lang.String, java.lang.Object)

<C> org.apache.ibatis.session.defaults.DefaultSqlSession

insert(java.lang.String, java.lang.Object)

update(java.lang.String, java.lang.Object) // wrapCollection只处理Collection、array

<I> org.apache.ibatis.executor.Executor#update

<A> org.apache.ibatis.executor.BaseExecutor

update

doUpdate

/\*

一路追踪parameterObject。

起初我被batchResult.addParameterObject(parameterObject)、batchResultList.add(new BatchResult(ms, sql, parameterObject))这2行干扰了，后结合doFlushStatements确认：既然stmt已经executeBatch了，那么给stmt参数赋值的地方不会是这2行。

\*/

<C> com.baomidou.mybatisplus.core.executor.MybatisBatchExecutor#doUpdate

<I> org.apache.ibatis.executor.statement.StatementHandler#parameterize

<C> org.apache.ibatis.executor.statement.RoutingStatementHandler#parameterize

<C> org.apache.ibatis.executor.statement.PreparedStatementHandler#parameterize

<C> com.baomidou.mybatisplus.core.MybatisParameterHandler#setParameters

<C> org.apache.ibatis.session.Configuration#newMetaObject

<C> org.apache.ibatis.reflection.MetaObject#forObject

<C> org.apache.ibatis.reflection.MetaObject#MetaObject

结合com.baomidou.mybatisplus.core.MybatisParameterHandler#setParameters、org.apache.ibatis.reflection.MetaObject#MetaObject，推测得证。

图1-1 MybatisParameterHandler

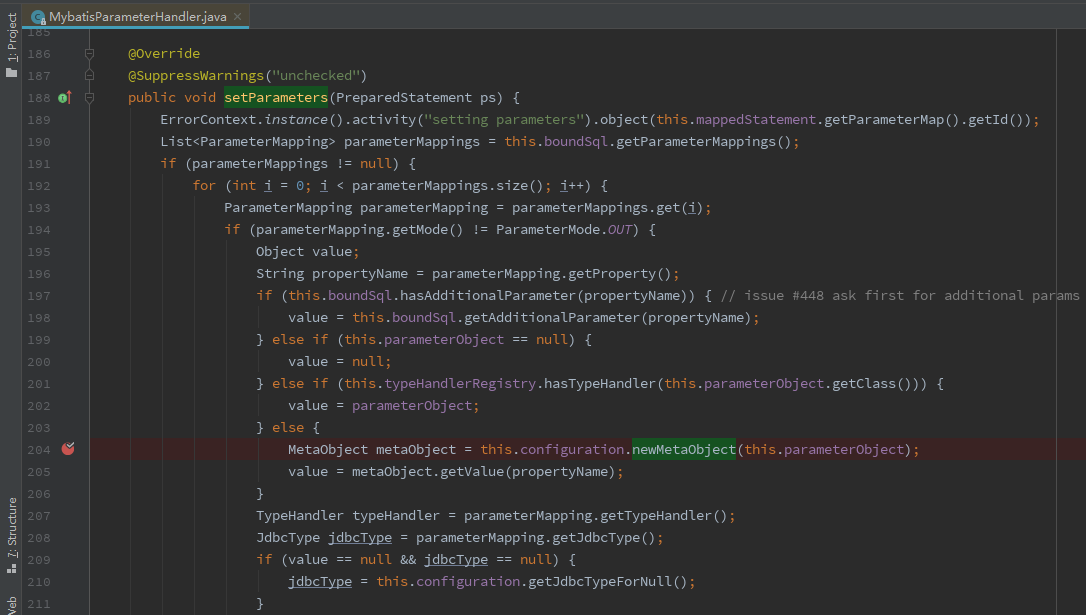


图1-2 MetaObject



# 事务

## 先判断重复性再insert，前者放入@Transactional方法是否必要？

背景：

1. 要insert的记录，除了id字段之外其他一致。
2. 除了id字段，没有其他unique、key列，也就是说用于判断重复性的字段不是unique、key。
3. Oracle的事务级别是默认的READ COMMIT（关系型数据库一般如此）。

期望：

判断与insert一体化。

回答：

没有必要。

原因：

如下情形就会重复：2个线程都已经判断过了重复性，但是因为对方都还没insert，所以都认为没有重复，此时继续往下执行，就插入重复记录了。

衍生问题1：

先insert再判断重复性？

回答1：

不行，因为事务的隔离性，对方在没有commit之前，自己是看不到的，此时判断重复性自然是false。

哪怕“判断、insert、判断”也还是不行。

衍生问题2：

insert语句加where用一个sql？

回答2：

不行，@Transactional方法是一个事务，一个sql也是一个事务，二者在事务隔离性上没有区别。

衍生问题3：

如何避免？

回答3：

1. 把字段声明为unique或key。
2. 事务级别改成READ UNCOMMITTED＋先insert再判断重复性，几乎不会这么做。

因为有脏读的问题，所以会出现两边都判断重复性是true的情况，以至于都rollback，最后没有insert任何记录。

1. 事务级别改成Serializable，几乎不会这么做。
2. NOTE: 我目前还不知道……字段不是unique或key的话，是否有办法一体化避免这个问题。

结论1：

在对重复性要求不高的场景，判断有必要而且要在insert之前，可以不用放入@Transactional方法。

## 读写分离，service与txnservice

背景：

1. spring在方法中首次调用mybatis对象时，才会初始化连接，连接的事务性取决于方法是否被声明为@Transactional。
2. 事务在完成或取消之前将一直占用资源。如果数据库中同时存在多个长时事务，那么数据库的压力会很大，更不用说长时写入事务。

分析：

1. 事务的持续时间要尽量的短。
2. 要用到事务的地方多是写入。
3. 只有读的事务是在浪费资源。

结论：

1. 读写分离，读放到service，写放到txnservice，service需要写的时候再调用txnservice。
2. 对于要写多次的事务，可以把所有记录先准备好。写一次准备一次会占用时间。

# mapper.xml

## 手动分页

|  |
| --- |
| <!-- 列出群体 --> <sql id="groupList\_with"> with tmp1 as (  select \*  from T\_GROUP  where "NAME" = #{vo.name} ) </sql>  <select id="groupList" parameterType="xyz.kuilei.spring.demo.bean.vo.PageVo" resultType="xyz.kuilei.spring.demo.bean.entity.TGroupEntity"> <include refid="groupList\_with"/>  select  t2.GROUP\_ID as groupId,  t2."NAME" as "name",  t2.CREATE\_TIME as createTime,  t2.UPDATE\_TIME as updateTime from (  <!-- 序号 -->  select  t1.\*,  rownum as ROW\_NUM  from (  <!-- 排序 -->  select \* from tmp1  order by CREATE\_TIME desc  ) t1 ) t2 where <![CDATA[  t2.ROW\_NUM > (${vo.pageNo} - 1) \* ${vo.pageSize} and t2.ROW\_NUM <= ${vo.pageNo} \* ${vo.pageSize} ]]> </select>  <select id="groupListTotal" parameterType="xyz.kuilei.spring.demo.bean.vo.PageVo" resultType="java.lang.Integer"> <include refid="groupList\_with"/>  select *count*(1) from tmp1 </select> |

## 手动批量insert

|  |
| --- |
| <insert id="memberBatchAdd" parameterType="java.util.List"> begin <foreach collection="list" item="item" index="index" separator=";" close=";"> insert into T\_GROUP\_MEMBER (  GROUP\_MEMBER\_ID,  GROUP\_ID,  "TYPE",  IDENTIFIER ) values (  #{item.groupMemberId},  #{item.groupId},  #{item.type},  #{item.identifier} ) </foreach> end; </insert> |

## 删除分区表

对于oracle，用execute immediate语句。不能放到<update>、<insert>标签，可以放到<select>标签。

# Oracle

1. to\_number转换失败直接报错，不支持失败取默认值；
2. join会打乱排序，先order再join时要注意。