MyBatis 传入参数为集合 list 数组 map 写法

作者：xunxiake
原文链接：https://ld246.com/article/1498701873695
来源网站：链滴
许可协议：署名-相同方式共享 4.0 国际 (CC BY-SA 4.0)

foreach的主要用在构建in条件中，它可以在SQL语句中进行迭代一个集合。foreach元的属性主要有item，index，collection，open，separator，close。item表示集合中每一个元素进迭代时的别名，index指定一个名字，用于表示在迭代过程中，每次迭代到的位置，open表示该语句什么开始，separator表示在每次进行迭代之间以什么符号作为分隔符，close表示以什么结束，在使用oreach的时候最关键的也是最容易出错的就是collection属性，该属性是必须指定的，但是在不同情下，该属性的值是不一样的，主要有一下3种情况：

如果传入的是单参数且参数类型是一个List的时候，collection属性值为list .</l>
如果传入的是单参数且参数类型是一个array数组的时候，collection的属性值为array .</pan>
如果传入的参数是多个的时候，我们就需要把它们封装成一个Map了，当然单参数也可封装成map，实际上如果你在传入参数的时候，在MyBatis里面也是会把它封装成一个Map的，map key就是参数名，所以这个时候collection属性值就是传入的List或array对象在自己封装的map里面的ey.

下面我们通过代码实践:

数据表:
采用Oracle的HR.Employees表
        实体:Employees

public class Employees {
    private Integer employeeId
    private String firstName;
    private Strig lastName;
    private String email;
    priate String phoneNumber;
    private Date hireDate;
 &bsp; private String jobId;
    private BigDecimal salary;<br /   private BigDecimal commissionPct;
    private nteger managerId;
    private Short departmentId;
} &nsp;
映射文件:

    <!--List:forech中的collectio属性类型是List,collection的值必须是:list,item的值可以随意,Dao接口中参数名字随意 -->
nbsp;   <select id="getEmployeesListParams" resultType="Employees">
        select *
   &nsp;   from EMPLOYEES e
        where e.EMPLOYEE_ID in
        &lt foreach collection="list" item="employeeId" index="index"
            open="(" close=")" separator=",">
           #{employeed}
        </foreach>
 &bsp; </select>

    <!--Array:forech中的collectio属性类型是array,collection的值必须是:list,item的值可以随意,Dao接口中参数名字随意 --><br /   <select id="getEmployeesArrayParams" resultType="Employees"&gt
        select *
        from EMPLOYEES e
     nbsp; where e.EMPLOYEE_ID in
      &nbp; <foreach collection="array" item="employeeId" index="index"
  &nbp;   open="(" close=")" separator=",>
           #{mployeeId}
       </foreach>
nbsp;   </select>

    <!--Map:不单单forec中的collection属性是map.key,其它所有属性都是map.key,比如下面的departmentId -->
nbsp;   <select id="getEmployeesMapParams" resultType="Employees"><r />   select *
   &bsp;   from EMPLOYEES e
     &nbp; <where>
       &nb

```
p;   &lt;if test="departmentId!=null and departmentId!=''"&gt;<br /> &n
sp;            &n
sp; e.DEPARTMENT_ID=#{departmentId}<br />      &
bsp;     &lt;/if&gt;<br />      
      &lt;if test="employeeIdsArray!=null and employeeIdsArr
y.length!=0"&gt;<br />         &nb
p;      AND e.EMPLOYEE_ID in<br />   &
bsp;            &lt;forea
h collection="employeeIdsArray" item="employeeId"<br />    &nb
p;            &nb
p;  index="index" open="(" close=")" separator=","&gt;<br />   &n
sp;            &n
sp;   #{employeeId}<br />       &n
sp;        &lt;/foreach&gt;<br />  
nbsp;         &lt;/if&gt;<br /> &nb
p;      &lt;/where&gt;<br />    &lt;/selec
&gt;<br /><br /></strong>Mapper类:<br />public interface EmployeesMapper { <br
><br />    List&lt;Employees&gt; getEmployeesListParams(List&lt;String
gt; employeeIds);<br /><br />    List&lt;Employees&gt; getEmployeesArr
yParams(String[] employeeIds);<br /><br />    List&lt;Employees&gt; get
mployeesMapParams(Map&lt;String,Object&gt; params);<br />}</div>
<div> </div>
<div>http://blog.csdn.net/small____fish/article/details/8029030</div>
```