



链滴

MyBatis曾经发布过相同标题的文章三

作者: [xiaoming](#)

原文链接: <https://ld246.com/article/1470043516988>

来源网站: 链滴

许可协议: [署名-相同方式共享 4.0 国际 \(CC BY-SA 4.0\)](#)

type 属性是类的完全限定名。method 是该类中的那个方法名。

使用 @Param(“xxx”),参数应该被命名为 #{xxx}。

主键是序列,参数是@Param:毫无意义。

1. 删

```
BaseMapper.java TestSqlProvider.java
1 package cn.com.eastit.erpx.mapper;
2
3 import java.util.List;
30
31 public interface BaseMapper<T> {
32
33     @Delete(value = { "delete from erp_user where id = #{id}" })
34     @Delete(value = { "delete from erp_user where id = #{xiaoming}" })
35     @DeleteProvider(method = "delete", type = TestSqlProvider.class)
36     void delete(@Param(value = "xiaoming") Long id);
37
38
39
40
41
42
43
BaseMapper.java TestSqlProvider.java
1 package cn.com.eastit.erpx.mapper;
2
3 import java.util.Map;
4
5 import org.apache.ibatis.annotations.Param;
6 import org.apache.ibatis.jdbc.SqlBuilder;
7
8 import cn.com.eastit.erpx.entity.system.User;
9
10 @SuppressWarnings("deprecation")
11 public class TestSqlProvider extends SqlBuilder {
12
13     private static final String TABLE_NAME = "erp_user";
14
15     public String delete(Long id) {
16
17         String sql = "delete from erp_user where id = #{id}";
18         String sql = "delete from erp_user where id = #{xiaoming}";
19         String sql = "delete from erp_user where id = '"+id+"'";
20         return sql;
21
22         return new org.apache.ibatis.jdbc.SQL(){
23             DELETE_FROM(TABLE_NAME);
24             WHERE("id=#{id}");
25         }.toString();
26         return new org.apache.ibatis.jdbc.SQL(){
27             DELETE_FROM(TABLE_NAME);
28             WHERE("id=#{xiaoming}");
29         }.toString();
30         return new org.apache.ibatis.jdbc.SQL(){
31             DELETE_FROM(TABLE_NAME);
32             WHERE("id='"+id+"'");
33         }.toString();
34
35         BEGIN();
36         DELETE_FROM(TABLE_NAME);
37         WHERE("id=#{id}");
38         BEGIN();
39         DELETE_FROM(TABLE_NAME);
40         WHERE("id=#{xiaoming}");
41         BEGIN();
42         DELETE_FROM(TABLE_NAME);
43         WHERE("id='"+id+"'");
44         return sql();
45
46     }
47 }
```

2. 改

```
BaseMapper.java TestSqlProvider.java
1 package cn.com.eastit.erpx.mapper;
2
3 import java.util.List;
30
31 public interface BaseMapper<T> {
32
33     @UpdateProvider(type = TestSqlProvider.class, method = "update")
34     @Update(value = { "update erp_user set login_name=#{loginName}, name=#{name}, tel=#{tel}, email=#{email}, "
35         + "password=#{password}, salt=#{salt}, department_id=#{departmentId} where id=#{id}" })
36     @Update(value = { "update erp_user set login_name=#{xiaoming.loginName}, name=#{xiaoming.name}, tel=#{xiaoming.tel}, email=#{xiaoming.email}, "
37         + "password=#{xiaoming.password}, salt=#{xiaoming.salt}, department_id=#{xiaoming.departmentId} where id=#{xiaoming.id}" })
38     void update(@Param(value = "xiaoming") T t);
39
40
41
42
43
44
45
46
47 }
```

```
BaseMapper.java TestSqlProvider.java 11
9
10 import java.util.Map;
11 @SuppressWarnings("deprecation")
12 public class TestSqlProvider extends SqlBuilder {
13     private static final String TABLE_NAME = "erp_user";
14     public String update(User user) {
15         String sql = "update erp_user set login_name=#{xiaoming.loginName}, name=#{xiaoming.name}, tel=#{xiaoming.tel}, email=#{xiaoming.email}, "
16             + "password=#{xiaoming.password}, salt=#{xiaoming.salt}, department_id=#{xiaoming.departmentId} where id=#{xiaoming.id}";
17         String sql = "update erp_user set login_name=#{loginName}, name=#{name}, tel=#{tel}, email=#{email}, "
18             + "password=#{password}, salt=#{salt}, department_id=#{departmentId} where id=#{id}";
19         String sql = "update erp_user set login_name="+user.getLoginName()+"", name="+user.getName()+"", tel="+user.getTel()+"", email="+user.getEmail()+"", "
20             + "password="+user.getPassword()+"", salt="+user.getSalt()+"", department_id="+user.getDepartmentId()+" where id="+user.getId()+"";
21         return sql;
22         return new org.apache.ibatis.jdbc.SQL(){
23             UPDATE(TABLE_NAME);
24             SET("login_name=#{loginName}");
25             SET("name=#{name}");
26             SET("tel=#{tel}");
27             SET("email=#{email}");
28             SET("password=#{password}");
29             SET("salt=#{salt}");
30             SET("department_id=#{departmentId}");
31             WHERE("id=#{id}");
32         }.toString();
33         return new org.apache.ibatis.jdbc.SQL(){
34             UPDATE(TABLE_NAME);
35             SET("login_name=#{xiaoming.loginName}");
36             SET("name=#{xiaoming.name}");
37             SET("tel=#{xiaoming.tel}");
38             SET("email=#{xiaoming.email}");
39             SET("password=#{xiaoming.password}");
40             SET("salt=#{xiaoming.salt}");
41             SET("department_id=#{xiaoming.departmentId}");
42             WHERE("id=#{xiaoming.id}");
43         }.toString();
44         return new org.apache.ibatis.jdbc.SQL(){
45             UPDATE(TABLE_NAME);
46             SET("login_name="+user.getLoginName()+"");
47             SET("name="+user.getName()+"");
48             SET("tel="+user.getTel()+"");
49             SET("email="+user.getEmail()+"");
50             SET("password="+user.getPassword()+"");
51             SET("salt="+user.getSalt()+"");
52             SET("department_id="+user.getDepartmentId()+"");
53             WHERE("id="+user.getId()+"");
54         }.toString();
55     }
56 }
```

```
BaseMapper.java TestSqlProvider.java 11
1 package cn.com.eastit.erpx.mapper;
2
3 import java.util.Map;
4
5
6 @SuppressWarnings("deprecation")
7 public class TestSqlProvider extends SqlBuilder {
8     private static final String TABLE_NAME = "erp_user";
9
10     public String update(User user) {
11
12         BEGIN();
13         UPDATE(TABLE_NAME);
14         SET("login_name=#{loginName}");
15         SET("name=#{name}");
16         SET("tel=#{tel}");
17         SET("email=#{email}");
18         SET("password=#{password}");
19         SET("salt=#{salt}");
20         SET("department_id=#{departmentId}");
21         WHERE("id=#{id}");
22         return SQL();
23
24         BEGIN();
25         UPDATE(TABLE_NAME);
26         SET("login_name=#{xiaoming.loginName}");
27         SET("name=#{xiaoming.name}");
28         SET("tel=#{xiaoming.tel}");
29         SET("email=#{xiaoming.email}");
30         SET("password=#{xiaoming.password}");
31         SET("salt=#{xiaoming.salt}");
32         SET("department_id=#{xiaoming.departmentId}");
33         WHERE("id=#{xiaoming.id}");
34         return SQL();
35
36         BEGIN();
37         UPDATE(TABLE_NAME);
38         SET("login_name="+user.getLoginName()+"");
39         SET("name="+user.getName()+"");
40         SET("tel="+user.getTel()+"");
41         SET("email="+user.getEmail()+"");
42         SET("password="+user.getPassword()+"");
43         SET("salt="+user.getSalt()+"");
44         SET("department_id="+user.getDepartmentId()+"");
45         WHERE("id="+user.getId()+"");
46         return SQL();
47     }
48 }
```

3.查

```
BaseMapper.java TestSqlProvider.java 11
1 package cn.com.eastit.erpx.mapper;
2
3 import java.util.List;
4
5
6 public interface BaseMapper<T> {
7
8     @SelectProvider(method = "findForPage", type = TestSqlProvider.class)
9     // 不可取因为这类语句最初时参数为null
10     // JDBC requires that the jdbcType must be specified for all nullable parameters.
11     @Select(value = ("select id, login_name loginName, name, email, tel, department_id departmentId from erp_user where 1 = 1 and name = #{xiaoming.name} "
12         + "and login_name = #{xiaoming.loginName} and department_id = #{xiaoming.departmentId}"))
13     List<Map<String, Object>> FindForPage(@Param(value = "xiaoming") Map<String, Object> param);
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48 }
```

```

11 public class TestSqlProvider extends SqlBuilder {
12     private static final String TABLE_NAME = "erp_user";
13     public String findForPage(Map<String, Object> parameters) {
14         Object name = parameters.get("name");
15         Object loginName = parameters.get("loginName");
16         Object departmentId = parameters.get("departmentId");
17         /* 大小写 */
18         String sql = "select id, login_name loginName, name, email, tel, department_id departmentId from erp_user where 1 = 1";
19         if (name != null) {
20             sql = sql + " and name = '"+name.toString().trim()+"'";
21         }
22         if (loginName != null) {
23             sql = sql + " and login_name = '"+loginName.toString().trim()+"'";
24         }
25         if (departmentId != null) {
26             sql = sql + " and department_id = '"+departmentId.toString().trim()+"'";
27         }
28         return sql;
29         /* 等价 */
30         return new org.apache.ibatis.jdbc.SQL(){
31             SELECT("id, login_name loginName, name, email, tel, department_id departmentId");
32             FROM(TABLE_NAME);
33             if (name != null) {
34                 WHERE("name = #{name}");
35             }
36             if (loginName != null) {
37                 WHERE("login_name = #{loginName}");
38             }
39             if (departmentId != null) {
40                 WHERE("department_id = #{departmentId}");
41             }
42         }.toString();
43         /* 对比 */
44         BEGIN();
45         SELECT("id, login_name loginName, name, email, tel, department_id departmentId");
46         FROM(TABLE_NAME);
47         if (name != null) {
48             WHERE("name = #{name}");
49         }
50         if (loginName != null) {
51             WHERE("login_name = #{loginName}");
52         }
53         if (departmentId != null) {
54             WHERE("department_id = #{departmentId}");
55         }
56         return SQL();

```

4.增

```

1 package cn.com.eastit.erp.mapper;
2
3 import java.util.List;
4
5 public interface BaseMapper<T> {
6
7     @InsertProvider(type = TestSqlProvider.class, method = "create")
8     @Insert(value = ("insert into erp_user(id, login_name, name, tel, email, password, salt, department_id) "
9         + "values(#{id}, #{loginName}, #{name}, #{tel}, #{email}, #{password}, #{salt}, #{departmentId})"))
10     @SelectKey(before = true, keyProperty = "id", resultType = Long.class, statement = ("select SEQ_ERP_USER.nextval from dual"), statementType = StatementType.PREPARED)
11     @Insert(value = ("insert into erp_user(id, login_name, name, tel, email, password, salt, department_id) "
12         + "values(#{id}, #{xiaoming.loginName}, #{xiaoming.name}, #{xiaoming.tel}, #{xiaoming.email}, #{xiaoming.password}, #{xiaoming.salt}, #{xiaoming.departmentId})"))
13     @SelectKey(before = true, keyProperty = "id", resultType = Long.class, statement = ("select SEQ_ERP_USER.nextval from dual"), statementType = StatementType.PREPARED)
14     void create(@Param(value = "xiaoming") T t);
15 }
16
17 package cn.com.eastit.erp.mapper;
18
19 import java.util.Map;
20
21 @SuppressWarnings("deprecation")
22 public class TestSqlProvider extends SqlBuilder {
23     private static final String TABLE_NAME = "erp_user";
24
25     public String create(User user) {
26         Object tel = user.getTel();
27         Object email = user.getEmail();
28         if (tel.equals("")) {
29             tel = null;
30         }
31         if (email.equals("")) {
32             email = null;
33         }
34         // 原因
35         // 无效的列类型: getLong not implemented for class oracle.jdbc.driver.T4CRowIdAccessor, 实体是Long类型,数据库是NUMBER()类型,参数是String类型.
36         // 失败 1
37         String sql1 = "insert into erp_user(id, login_name, name, tel, email, password, salt, department_id) "
38             + "values(SEQ_ERP_USER.nextval, '"+user.getLoginName()+"', '"+user.getName()+"', '"+tel+"', '"+email+"', "
39             + "'"+user.getPassword()+"', '"+user.getSalt()+"', '"+user.getDepartmentId()+"'";
40         // 失败 2
41         String sql2 = "insert into erp_user(id, login_name, name, tel, email, password, salt, department_id) "
42             + "values(SEQ_ERP_USER.nextval, #{loginName}, #{name}, #{tel}, #{email}, #{password}, #{salt}, #{departmentId}) ";
43         // 失败 3
44         String sql3 = "insert into erp_user(id, login_name, name, tel, email, password, salt, department_id) "
45             + "values(SEQ_ERP_USER.nextval, #{xiaoming.loginName}, #{xiaoming.name}, #{xiaoming.tel}, #{xiaoming.email}, "
46             + " #{xiaoming.password}, #{xiaoming.salt}, #{xiaoming.departmentId}) ";
47
48
49
50
51
52

```

```
BaseMapper.java TestSqlProvider.java
17 private static final String TABLE_NAME = "erp_user";
18 public String create(User user) {
19     // 失败 4
20     return new org.apache.ibatis.jdbc.SQL(){
21         Object tel = user.getTel();
22         Object email = user.getEmail();
23         if (tel.equals("")) {
24             tel = null;
25         }
26         if (email.equals("")) {
27             email = null;
28         }
29         INSERT_INTO(TABLE_NAME);
30         VALUES("id", "SEQ_ERP_USER.nextval");
31         VALUES("login_name", ""+user.getLoginName()+"");
32         VALUES("name", ""+user.getName()+"");
33         VALUES("tel", ""+tel+"");
34         VALUES("email", ""+email+"");
35         VALUES("password", ""+user.getPassword()+"");
36         VALUES("salt", ""+user.getSalt()+"");
37         VALUES("department_id", ""+user.getDepartmentId()+"");
38     }).toString();
39     // 失败 5
40     return new org.apache.ibatis.jdbc.SQL(){
41         INSERT_INTO(TABLE_NAME);
42         VALUES("id", "SEQ_ERP_USER.nextval");
43         VALUES("login_name", "#{loginName}");
44         VALUES("name", "#{name}");
45         VALUES("tel", "#{tel}");
46         VALUES("email", "#{email}");
47         VALUES("password", "#{password}");
48         VALUES("salt", "#{salt}");
49         VALUES("department_id", "#{departmentId}");
50     }).toString();
51     // 失败 6
52     return new org.apache.ibatis.jdbc.SQL(){
53         INSERT_INTO(TABLE_NAME);
54         VALUES("id", "SEQ_ERP_USER.nextval");
55         VALUES("login_name", "#{xiaoming.loginName}");
56         VALUES("name", "#{xiaoming.name}");
57         VALUES("tel", "#{xiaoming.tel}");
58         VALUES("email", "#{xiaoming.email}");
59         VALUES("password", "#{xiaoming.password}");
60         VALUES("salt", "#{xiaoming.salt}");
61         VALUES("department_id", "#{xiaoming.departmentId}");
62     }).toString();
63 }
```

```
BaseMapper.java TestSqlProvider.java
17 private static final String TABLE_NAME = "erp_user";
18 public String create(User user) {
19     Object tel = user.getTel();
20     Object email = user.getEmail();
21     if (tel.equals("")) {
22         tel = null;
23     }
24     if (email.equals("")) {
25         email = null;
26     }
27     BEGIN();
28     VALUES("id", "SEQ_ERP_USER.nextval");
29     VALUES("login_name", ""+user.getLoginName()+"");
30     VALUES("name", ""+user.getName()+"");
31     VALUES("tel", ""+tel+"");
32     VALUES("email", ""+email+"");
33     VALUES("password", ""+user.getPassword()+"");
34     VALUES("salt", ""+user.getSalt()+"");
35     VALUES("department_id", ""+user.getDepartmentId()+"");
36     return SQL();
37     // 失败 8
38     BEGIN();
39     INSERT_INTO(TABLE_NAME);
40     VALUES("id", "SEQ_ERP_USER.nextval");
41     VALUES("login_name", "#{loginName}");
42     VALUES("name", "#{name}");
43     VALUES("tel", "#{tel}");
44     VALUES("email", "#{email}");
45     VALUES("password", "#{password}");
46     VALUES("salt", "#{salt}");
47     VALUES("department_id", "#{departmentId}");
48     return SQL();
49     // 失败 9
50     BEGIN();
51     INSERT_INTO(TABLE_NAME);
52     VALUES("id", "SEQ_ERP_USER.nextval");
53     VALUES("login_name", "#{xiaoming.loginName}");
54     VALUES("name", "#{xiaoming.name}");
55     VALUES("tel", "#{xiaoming.tel}");
56     VALUES("email", "#{xiaoming.email}");
57     VALUES("password", "#{xiaoming.password}");
58     VALUES("salt", "#{xiaoming.salt}");
59     VALUES("department_id", "#{xiaoming.departmentId}");
60     return SQL();
61 }
```