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

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

```java
package cn.com.eastit.erpx.mapper;

import java.util.List;

public interface BaseMapper<T> {

    @Delete(value = { "delete from erp_user where id = #{id}" })
    @Delete(value = { "delete from erp_user where id = #{xiaoming}" })
    @DeleteProvider(method = "delete", type = TestSqlProvider.class)
    void delete(@Param(value = "xiaoming") Long id);
```

BaseMapper.java TestSqlProvider.java

```java
package cn.com.eastit.erpx.mapper;

import java.util.Map;

import org.apache.ibatis.annotations.Param;
import org.apache.ibatis.jdbc.SqlBuilder;

import cn.com.eastit.erpx.entity.system.User;

@SuppressWarnings("deprecation")
public class TestSqlProvider extends SqlBuilder {

    private static final String TABLE_NAME = "erp_user";

    public String delete(Long id) {

        String sql = "delete from erp_user where id = #{id}";
        String sql = "delete from erp_user where id = #{xiaoming}";
        String sql = "delete from erp_user where id = '"+id+"'";
        return sql;

        return new org.apache.ibatis.jdbc.SQL(){{
            DELETE_FROM(TABLE_NAME);
            WHERE("id=#{id}");
        }}.toString();
        return new org.apache.ibatis.jdbc.SQL(){{
            DELETE_FROM(TABLE_NAME);
            WHERE("id=#{xiaoming}");
        }}.toString();
        return new org.apache.ibatis.jdbc.SQL(){{
            DELETE_FROM(TABLE_NAME);
            WHERE("id="+id);
        }}.toString();

        BEGIN();
        DELETE_FROM(TABLE_NAME);
        WHERE("id=#{id}");
        BEGIN();
        DELETE_FROM(TABLE_NAME);
        WHERE("id=#{xiaoming}");
        BEGIN();
        DELETE_FROM(TABLE_NAME);
        WHERE("id="+id);
        return SQL();

    }
```

2.改

BaseMapper.java TestSqlProvider.java

```java
package cn.com.eastit.erpx.mapper;

import java.util.List;

public interface BaseMapper<T> {

    @UpdateProvider(type = TestSqlProvider.class, method = "update")
    @Update(value = { "update erp_user set login_name=#{loginName}, name=#{name}, tel=#{tel}, email=#{email}, "
            + "password=#{password}, salt=#{salt}, department_id=#{departmentId} where id=#{id}" })
    @Update(value = { "update erp_user set login_name=#{xiaoming.loginName}, name=#{xiaoming.name}, tel=#{xiaoming.tel}, email=#{xiaoming.email}, "
            + "password=#{xiaoming.password}, salt=#{xiaoming.salt}, department_id=#{xiaoming.departmentId} where id=#{xiaoming.id}" })
    void update(@Param(value = "xiaoming") T t);
```

```java
import java.util.Map;

@SuppressWarnings("deprecation")
public class TestSqlProvider extends SqlBuilder {
    private static final String TABLE_NAME = "erp_user";
    public String update(User user) {
        String sql = "update erp_user set login_name=#{xiaoming.loginName}, name=#{xiaoming.name}, tel=#{xiaoming.tel}, email=#{xiaoming.email}, "
                + "password=#{xiaoming.password}, salt=#{xiaoming.salt}, department_id=#{xiaoming.departmentId} where id=#{xiaoming.id}";
        String sql = "update erp_user set login_name=#{loginName}, name=#{name}, tel=#{tel}, email=#{email}, "
                + "password=#{password}, salt=#{salt}, department_id=#{departmentId} where id=#{id}";
        String sql = "update erp_user set login_name='"+user.getLoginName()+"', name='"+user.getName()+"', tel='"+user.getTel()+"', email='"+user.getEmail()+"', "
                + "password='"+user.getPassword()+"', salt='"+user.getSalt()+"', department_id='"+user.getDepartmentId()+"' where id='"+user.getId()+"'";
        return sql;
        return new org.apache.ibatis.jdbc.SQL(){{
            UPDATE(TABLE_NAME);
            SET("login_name=#{loginName}");
            SET("name=#{name}");
            SET("tel=#{tel}");
            SET("email=#{email}");
            SET("password=#{password}");
            SET("salt=#{salt}");
            SET("department_id=#{departmentId}");
            WHERE("id=#{id}");
        }}.toString();
        return new org.apache.ibatis.jdbc.SQL(){{
            UPDATE(TABLE_NAME);
            SET("login_name=#{xiaoming.loginName}");
            SET("name=#{xiaoming.name}");
            SET("tel=#{xiaoming.tel}");
            SET("email=#{xiaoming.email}");
            SET("password=#{xiaoming.password}");
            SET("salt=#{xiaoming.salt}");
            SET("department_id=#{xiaoming.departmentId}");
            WHERE("id=#{xiaoming.id}");
        }}.toString();
        return new org.apache.ibatis.jdbc.SQL(){{
            UPDATE(TABLE_NAME);
            SET("login_name='"+user.getLoginName()+"'");
            SET("name='"+user.getName()+"'");
            SET("tel='"+user.getTel()+"'");
            SET("email='"+user.getEmail()+"'");
            SET("password='"+user.getPassword()+"'");
            SET("salt='"+user.getSalt()+"'");
            SET("department_id='"+user.getDepartmentId()+"'");
            WHERE("id='"+user.getId()+"'");
        }}.toString();
```

```java
package cn.com.eastit.erpx.mapper;

import java.util.Map;

@SuppressWarnings("deprecation")
public class TestSqlProvider extends SqlBuilder {

    private static final String TABLE_NAME = "erp_user";

    public String update(User user) {

        BEGIN();
        UPDATE(TABLE_NAME);
        SET("login_name=#{loginName}");
        SET("name=#{name}");
        SET("tel=#{tel}");
        SET("email=#{email}");
        SET("password=#{password}");
        SET("salt=#{salt}");
        SET("department_id=#{departmentId}");
        WHERE("id=#{id}");
        return SQL();

        BEGIN();
        UPDATE(TABLE_NAME);
        SET("login_name=#{xiaoming.loginName}");
        SET("name=#{xiaoming.name}");
        SET("tel=#{xiaoming.tel}");
        SET("email=#{xiaoming.email}");
        SET("password=#{xiaoming.password}");
        SET("salt=#{xiaoming.salt}");
        SET("department_id=#{xiaoming.departmentId}");
        WHERE("id=#{xiaoming.id}");
        return SQL();

        BEGIN();
        UPDATE(TABLE_NAME);
        SET("login_name='"+user.getLoginName()+"'");
        SET("name='"+user.getName()+"'");
        SET("tel='"+user.getTel()+"'");
        SET("email='"+user.getEmail()+"'");
        SET("password='"+user.getPassword()+"'");
        SET("salt='"+user.getSalt()+"'");
        SET("department_id='"+user.getDepartmentId()+"'");
        WHERE("id='"+user.getId()+"'");
        return SQL();
```

3.查

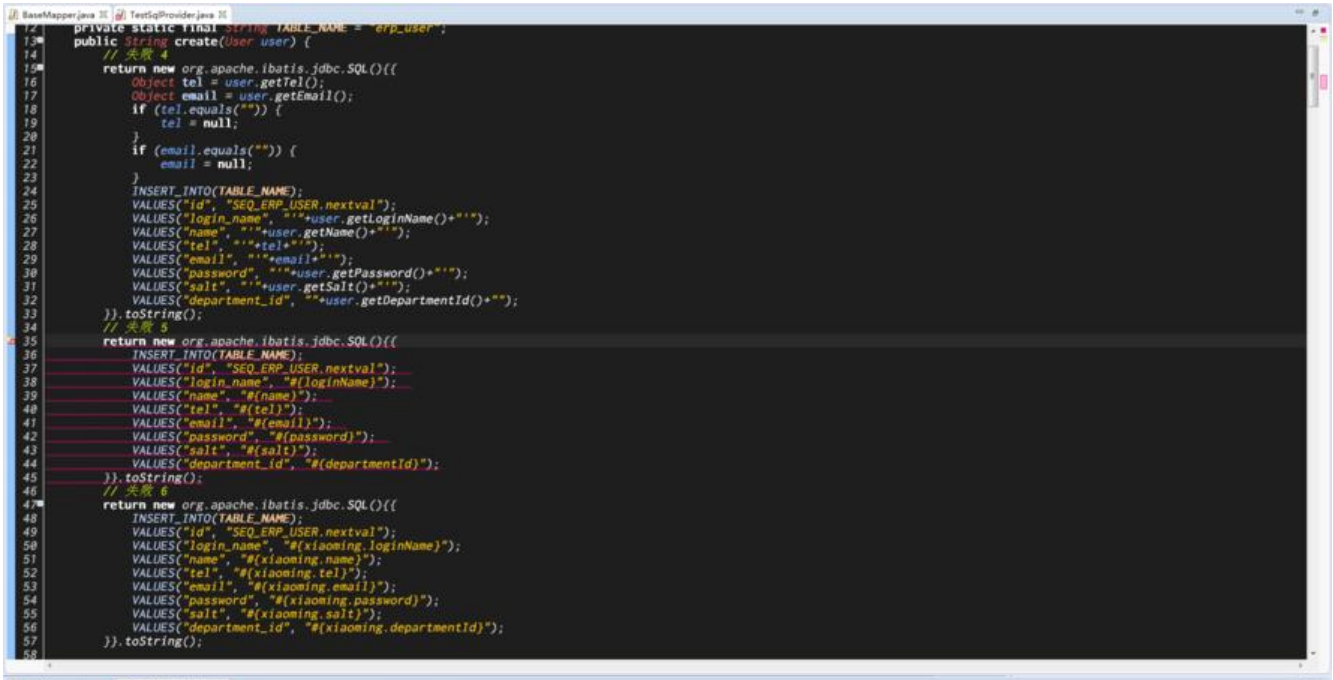
```java
package cn.com.eastit.erpx.mapper;

import java.util.List;

public interface BaseMapper<T> {

    @SelectProvider(method = "findForPage", type = TestSqlProvider.class)
    // 不可取因为这条语句是初始化参数为null
    // JDBC requires that the JdbcType must be specified for all nullable parameters.
    @Select(value = { "select id, login_name loginName, name, email, tel, department_id departmentId from erp_user where 1 = 1 and name = #{xiaoming.name} "
            + "and login_name = #{xiaoming.loginName} and department_id = #{xiaoming.departmentId}" })
    List<Map<String, Object>> findForPage(@Param(value = "xiaoming") Map<String, Object> param);
```

```java
public class TestSqlProvider extends SqlBuilder {
    private static final String TABLE_NAME = "erp_user";
    public String findForPage(Map<String, Object> parameters) {
        Object name = parameters.get("name");
        Object loginName = parameters.get("loginName");
        Object departmentId = parameters.get("departmentId");
        /* 大同小异 */
        String sql = "select id, login_name loginName, name, email, tel, department_id departmentId from erp_user where 1 = 1";
        if (name != null) {
            sql = sql + " and name = '"+name.toString().trim()+"'";
        }
        if (loginName != null) {
            sql = sql + " and login_name = '"+loginName.toString().trim()+"'";
        }
        if (departmentId != null) {
            sql = sql + " and department_id = '"+departmentId.toString().trim()+"'";
        }
        return sql;
        /* 等价 */
        return new org.apache.ibatis.jdbc.SQL(){{
            SELECT("id, login_name loginName, name, email, tel, department_id departmentId");
            FROM(TABLE_NAME);
            if (name != null) {
                WHERE("name = #{name}");
            }
            if (loginName != null) {
                WHERE("login_name = #{loginName}");
            }
            if (departmentId != null) {
                WHERE("department_id = #{departmentId}");
            }
        }}.toString();
        /* 类比 */
        BEGIN();
        SELECT("id, login_name loginName, name, email, tel, department_id departmentId");
        FROM(TABLE_NAME);
        if (name != null) {
            WHERE("name = #{name}");
        }
        if (loginName != null) {
            WHERE("login_name = #{loginName}");
        }
        if (departmentId != null) {
            WHERE("department_id = #{departmentId}");
        }
        return SQL();
```

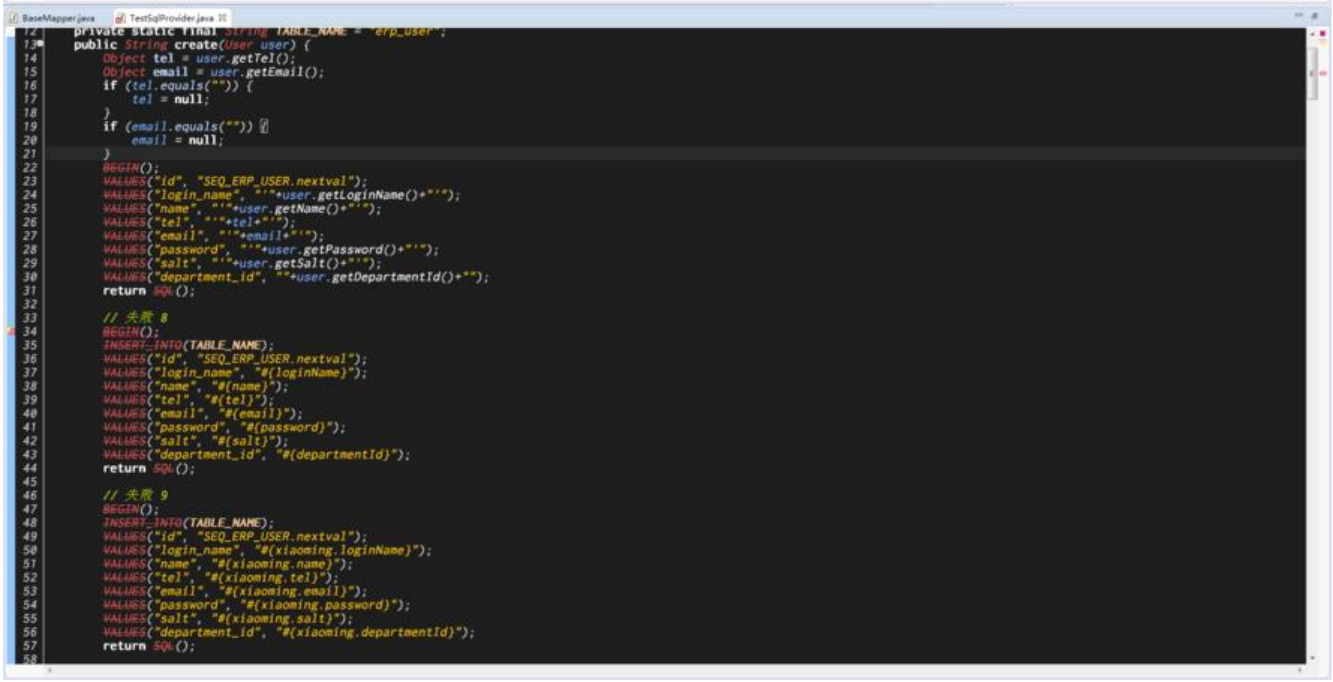
4.增

```java
package cn.com.eastit.erpx.mapper;

import java.util.List;

public interface BaseMapper<T> {

    @InsertProvider(type = TestSqlProvider.class,method = "create")
    @Insert(value = { "insert into erp_user(id, login_name, name, tel, email, password, salt, department_id) "
            + "values(#{id}, #{loginName}, #{name}, #{tel}, #{email}, #{password}, #{salt}, #{departmentId})" })
    @SelectKey(before = true, keyProperty = "id", resultType = Long.class, statement = { "select SEQ_ERP_USER.nextval from dual" }, statementType = StatementType.PREPARED)
    @Insert(value = { "insert into erp_user(id, login_name, name, tel, email, password, salt, department_id) "
            + "values(#{id}, #{xiaoming.loginName}, #{xiaoming.name}, #{xiaoming.tel}, #{xiaoming.email}, #{xiaoming.password}, #{xiaoming.salt}, #{xiaoming.departmentId})" })
    @SelectKey(before = true, keyProperty = "id", resultType = Long.class, statement = { "select SEQ_ERP_USER.nextval from dual" }, statementType = StatementType.PREPARED)
    void create(@Param(value = "xiaoming") T t);
```

```java
package cn.com.eastit.erpx.mapper;

import java.util.Map;

@SuppressWarnings("deprecation")
public class TestSqlProvider extends SqlBuilder {

    private static final String TABLE_NAME = "erp_user";


    public String create(User user) {
        Object tel = user.getTel();
        Object email = user.getEmail();
        if (tel.equals("")) {
            tel = null;
        }
        if (email.equals("")) {
            email = null;
        }

        // 原因
        // 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.

        // 失败 1
        String sql1 = "insert into erp_user(id, login_name, name, tel, email, password, salt, department_id) "
                + "values(SEQ_ERP_USER.nextval, '"+user.getLoginName()+"', '"+user.getName()+"', '"+tel+"', '"+email+"', "
                + "'"+user.getPassword()+"', '"+user.getSalt()+"', "+user.getDepartmentId()+") ";

        // 失败 2
        String sql2 = "insert into erp_user(id, login_name, name, tel, email, password, salt, department_id) "
                + "values(SEQ_ERP_USER.nextval, #{loginName}, #{name}, #{tel}, #{email}, #{password}, #{salt}, #{departmentId}) ";

        // 失败 3
        String sql3 = "insert into erp_user(id, login_name, name, tel, email, password, salt, department_id) "
                + "values(SEQ_ERP_USER.nextval, #{xiaoming.loginName}, #{xiaoming.name}, #{xiaoming.tel}, #{xiaoming.email}, "
                + "#{xiaoming.password}, #{xiaoming.salt}, #{xiaoming.departmentId}) ";
```



```
BaseMapper.java  TestSqlProvider.java
    private static final String TABLE_NAME = "erp_user";
    public String create(User user) {
        // 失败 4
        return new org.apache.ibatis.jdbc.SQL(){{
            Object tel = user.getTel();
            Object email = user.getEmail();
            if (tel.equals("")) {
                tel = null;
            }
            if (email.equals("")) {
                email = null;
            }
            INSERT_INTO(TABLE_NAME);
            VALUES("id", "SEQ_ERP_USER.nextval");
            VALUES("login_name", "'"+user.getLoginName()+"'");
            VALUES("name", "'"+user.getName()+"'");
            VALUES("tel", "'"+tel+"'");
            VALUES("email", "'"+email+"'");
            VALUES("password", "'"+user.getPassword()+"'");
            VALUES("salt", "'"+user.getSalt()+"'");
            VALUES("department_id", ""+user.getDepartmentId()+"");
        }}.toString();
        // 失败 5
        return new org.apache.ibatis.jdbc.SQL(){{
            INSERT_INTO(TABLE_NAME);
            VALUES("id", "SEQ_ERP_USER.nextval");
            VALUES("login_name", "#{loginName}");
            VALUES("name", "#{name}");
            VALUES("tel", "#{tel}");
            VALUES("email", "#{email}");
            VALUES("password", "#{password}");
            VALUES("salt", "#{salt}");
            VALUES("department_id", "#{departmentId}");
        }}.toString();
        // 失败 6
        return new org.apache.ibatis.jdbc.SQL(){{
            INSERT_INTO(TABLE_NAME);
            VALUES("id", "SEQ_ERP_USER.nextval");
            VALUES("login_name", "#{xiaoming.loginName}");
            VALUES("name", "#{xiaoming.name}");
            VALUES("tel", "#{xiaoming.tel}");
            VALUES("email", "#{xiaoming.email}");
            VALUES("password", "#{xiaoming.password}");
            VALUES("salt", "#{xiaoming.salt}");
            VALUES("department_id", "#{xiaoming.departmentId}");
        }}.toString();
```



```
BaseMapper.java  TestSqlProvider.java
    private static final String TABLE_NAME = "erp_user";
    public String create(User user) {
        Object tel = user.getTel();
        Object email = user.getEmail();
        if (tel.equals("")) {
            tel = null;
        }
        if (email.equals("")) {
            email = null;
        }
        BEGIN();
        VALUES("id", "SEQ_ERP_USER.nextval");
        VALUES("login_name", "'"+user.getLoginName()+"'");
        VALUES("name", "'"+user.getName()+"'");
        VALUES("tel", "'"+tel+"'");
        VALUES("email", "'"+email+"'");
        VALUES("password", "'"+user.getPassword()+"'");
        VALUES("salt", "'"+user.getSalt()+"'");
        VALUES("department_id", ""+user.getDepartmentId()+"");
        return SQL();

        // 失败 8
        BEGIN();
        INSERT_INTO(TABLE_NAME);
        VALUES("id", "SEQ_ERP_USER.nextval");
        VALUES("login_name", "#{loginName}");
        VALUES("name", "#{name}");
        VALUES("tel", "#{tel}");
        VALUES("email", "#{email}");
        VALUES("password", "#{password}");
        VALUES("salt", "#{salt}");
        VALUES("department_id", "#{departmentId}");
        return SQL();

        // 失败 9
        BEGIN();
        INSERT_INTO(TABLE_NAME);
        VALUES("id", "SEQ_ERP_USER.nextval");
        VALUES("login_name", "#{xiaoming.loginName}");
        VALUES("name", "#{xiaoming.name}");
        VALUES("tel", "#{xiaoming.tel}");
        VALUES("email", "#{xiaoming.email}");
        VALUES("password", "#{xiaoming.password}");
        VALUES("salt", "#{xiaoming.salt}");
        VALUES("department_id", "#{xiaoming.departmentId}");
        return SQL();
```

原文链接：MyBatis曾经发布过相同标题的文章三