mybatis调用mysql存储过程（返回参数，单结果集，多结果集）

# 一、接收一个返回值

# 使用Map接收返回参数，output参数放在传入的param中

# 创建表

# DROP TABLE IF EXISTS `demo`;

# CREATE TABLE `demo` (

# `id` int(11) NOT NULL AUTO\_INCREMENT,

# `name` varchar(255) DEFAULT NULL,

# PRIMARY KEY (`id`)

# ) ENGINE=InnoDB AUTO\_INCREMENT=32 DEFAULT CHARSET=utf8;

# 

# -- ----------------------------

# -- Records of demo

# -- ----------------------------

# INSERT INTO `demo` VALUES ('1', '测试');

# 创建存储过程

# DROP PROCEDURE IF EXISTS pro;

# CREATE PROCEDURE pro (

# IN userId INT,

# OUT userCount INT

# )

# BEGIN

# 

# DECLARE user\_name VARCHAR (64);

# SELECT NAME FROM demo WHERE id = userId INTO user\_name;

# INSERT INTO demo (NAME) VALUES (user\_name);

# SELECT COUNT(\*) FROM demo INTO userCount;

# 

# END;

# mapper.xml

# <select id="procedureTest" parameterMap="pm" statementType="CALLABLE">

# CALL pro(#{userId,mode=IN},#{userCount,mode=OUT,jdbcType=INTEGER});

# </select>

# 

# <parameterMap id="pm" type="java.util.Map">

# <parameter property="userId" jdbcType="INTEGER" mode="IN"></parameter>

# <parameter property="userCount" jdbcType="INTEGER" mode="OUT"></parameter>

# </parameterMap>

# service

# @Override

# public void procedureTest() {

# Map<String, Object> param = new HashMap<>();

# param.put("userId", "1");

# demoMapper.procedureTest(param);

# System.out.println("输出结果是："+param.get("userCount"));

# }

# 输出结果

# 参数名不一定和存储过程一样，只和传参顺有关，使用下图配置同样可以获取结果

# System.out.println("输出结果是："+param.get("userCount\_2"));

# 注意事项：

# （来源：https://blog.csdn.net/hj7jay/article/details/75334581?utm\_source=copy）

# 1、  存储过程的参数和名称无关，只和顺序有关系

# 2、  存储过程的output参数，只能通过传入的map获取

# 3、  存储过程返回的结果集可直接用返回的map接收

# 4、  存储过程的return结果需要使用?=call procName(?,?)的第一个参数接收，需要指定对应的mode为OUT类型

# 5、  存储过程对应的数据类型为枚举类型，需要使用大写，如VARCHAR

# output是在存储过程中的参数的返回值(输出参数)，而ReturnValue是存储过程返回的值(使用return关键字)，一个存储过程可以有任意多个依靠参数返回的值，但只有一个ReturnValue。

# 

# 存储过程主要分成三类：

# 1、返回记录集的存储过程:执行结果是一个记录集，例如，从数据库中检索出符合某一个或几个条件的记录。

# 2、返回数值得的存储过程(也可称为标量存储过程)，其执行完后返回一个值，例如数据库中执行一个有返回值的函数或命令。

# 3、行为存储过程，用来实现数据库的某个功能，而没有返回值，例如在数据库中的更新和删除操作。

# 二、接收list结果集

# 使用list接收

# @Override

# public void procedureTest() {

# Map<String, Object> param = new HashMap<>();

# param.put("userId", "1");

# List<?> list = demoMapper.procedureTest(param);

# System.out.println("输出结果是：" + param.get("userCount\_2"));

# System.out.println("输出集合：" + list.size());

# }

# <select id="procedureTest" parameterMap="pm" statementType="CALLABLE" resultMap="rm">

# CALL pro(#{userId,mode=IN},#{userCount\_2,mode=OUT,jdbcType=INTEGER});

# </select>

# 

# <parameterMap id="pm" type="java.util.Map">

# <parameter property="userId" jdbcType="INTEGER" mode="IN"></parameter>

# <parameter property="userCount\_2" jdbcType="INTEGER" mode="OUT"></parameter>

# </parameterMap>

# 

# <resultMap id="rm" type="com.csstj.srm.entity.procedureTestEntity">

# <result column="id" property="id"></result>

# <result column="name" property="userName"></result>

# </resultMap>

# 三、返回多个结果集

# 四、第二种配置也可以

# CALL s\_generatePlan\_new (

# #{organizationId, mode=IN,jdbcType=VARCHAR},

# #{gradeId, mode=IN,jdbcType=VARCHAR},

# #{semesterId, mode=IN,jdbcType=VARCHAR},

# #{className, mode=IN,jdbcType=VARCHAR},

# #{employeeCode, mode=IN,jdbcType=VARCHAR},

# #{courseName, mode=IN,jdbcType=VARCHAR},

# #{classroomName, mode=IN,jdbcType=VARCHAR},

# #{approverId, mode=IN,jdbcType=VARCHAR},

# #{createBy, mode=IN,jdbcType=VARCHAR},

# #{total,mode=OUT ,jdbcType=VARCHAR})

# ---------------------

# 作者：阿瑟与非

# 来源：CSDN

# 原文：https://blog.csdn.net/cs373616511/article/details/82890306

# 版权声明：本文为博主原创文章，转载请附上博文链接！