





package com.kin.dao;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.HashMap;

import java.util.Map;

import org.junit.Test;

import com.kin.bean.Department;

import com.kin.bean.Employee;

import com.mysql.jdbc.Connection;

import com.mysql.jdbc.PreparedStatement;

import com.mysql.jdbc.ResultSetMetaData;

import com.mysql.jdbc.Statement;

import junit.framework.TestCase;

public class JDBCToolsTest extends TestCase {

JDBCTools jdbctools = new JDBCTools();

@Test

public void testResultSetMetaData() throws InstantiationException, IllegalAccessException{

Connection connection = null;

PreparedStatement ps =null;

ResultSet rs = null;

try {

String sql = "select department\_id departmentId,department\_name departmentName from departments where department\_id=?";

connection = jdbctools.getConection();

ps = (PreparedStatement) connection.prepareStatement(sql);

ps.setInt(1, 2001);

rs = ps.executeQuery();

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

//1.得到ResultSetMetaData对象

ResultSetMetaData rsmd = (ResultSetMetaData) rs.getMetaData();

while(rs.next()){

//2.打印每一列的列名

for(int i = 0;i<rsmd.getColumnCount();i++){

String columnLabel = rsmd.getColumnLabel(i+1);

Object columnValue = rs.getObject(columnLabel);

values.put(columnLabel, columnValue);

}

}

System.out.println(values);

Class clazz = Department.class;

Object object = clazz.newInstance();

for(Map.Entry<String, Object>entry:values.entrySet()){

String fieldName = entry.getKey();

Object fieldValue = entry.getValue();

System.out.println(fieldName+": "+fieldValue);

//ReflectionUtils.setFieldValue(entity,fieldName,value);//在反射机制那里完成！！！这一行代码还没有完成

}

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}finally{

jdbctools.releaseConnection(rs,ps,connection);

}

//ResultSetMetaData()

/\*1）what:是描述ResultSet的元数据对象，即从中可以获取到结果集中有多少列

列名是什么...

2）how: 1.得到ResultSetMetaData 对象：调用ResultSet的getMetaData()方法

2.ResultSetMetaData有哪些好用的方法：

>int getColumnCount:SQL语句中包含哪些列

>String getColumnLabel(int column):获取指定的列的别名，其中索引从1开始

\*/

}

@Test

public void testGet(){

String sql ="select \* from employees where employee\_id=?";

Employee employee = get(Employee.class, sql, 3001);

System.out.println(employee);

sql="select \* from departments where location\_id=?";

Department department = get(Department.class, sql, 1003);

System.out.println(department);

}

public <T>T get(Class<T> clazz, String sql,Object ... args){

T entity = null;

Connection connection = null;

PreparedStatement ps =null;

ResultSet rs = null;

try {

//1.得到ResultSet对象

connection = jdbctools.getConection();

ps = (PreparedStatement) connection.prepareStatement(sql);

//for循环为可变参数占位符

for(int i = 0 ; i < args.length;i++){

ps.setObject(i+1, args[i]);

}

rs = ps.executeQuery();

//2.得到ResultSetMetaData对象

ResultSetMetaData rsmd = (ResultSetMetaData) rs.getMetaData();

//3.创建一个Map<String,Object> 对象，键：SQL查询的列的别名

//值：列的值

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

//4.处理结果集，利用ResultSetMetaData填充3对应的Map对象

if(rs.next()){

for(int i = 0 ; i<rsmd.getColumnCount();i++){

String columnLabel = rsmd.getColumnLabel(i+1);

Object columnValue = rs.getObject(i+1);

values.put(columnLabel, columnValue);

}

}

//5.若Map不为空集，利用反射创建clazz对应的对象

if(values.size()>0){

entity = clazz.newInstance();

}

//6.遍历Map对象，利用反射为Class对象的对应的属性赋值。

for(Map.Entry<String, Object>entry:values.entrySet()){

String fieldName = entry.getKey();

Object value = entry.getValue();

ReflectionUtils.setFieldValue(entity,fieldName,value);//在反射机制那里完成！！！这一行代码还没有完成

}

/\*if(rs.next()){

//利用反射创建对象

entity = clazz.newInstance();

//通过解析SQL语句来判断到底选择了那些列，以及需要为entity对象

//的哪些属性赋值。

}\*/

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

} catch (InstantiationException e) {

// TODO Auto-generated catch block

e.printStackTrace();

} catch (IllegalAccessException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}finally{

jdbctools.releaseConnection(rs,ps,connection);

}

return entity;

}

public void testGetConection() throws SQLException {

System.out.println(jdbctools.getConection());

Connection connection = jdbctools.getConection();

PreparedStatement ps =null;

ResultSet rs = null;

try {

String sql = "select \* from employees where employee\_id = 3001";

ps = (PreparedStatement) connection.prepareStatement(sql);

rs = ps.executeQuery();

while(rs.next()){

System.out.println(rs.getString("last\_name"));

}

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}finally{

jdbctools.releaseConnection(rs,ps,connection);

System.out.println(connection.isClosed());

}

}

}