仿写DBUtils

MyQueryRunner

```java
package com.southwind.util;

import org.apache.commons.dbutils.ResultSetHandler;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;

public class MyQueryRunner {
    public Object query(Connection connection, String sql, ResultSetHandler resultSetHandler,Object... params){
        PreparedStatement preparedStatement = null;
        ResultSet resultSet = null;
        Object result = null;
        try {
            preparedStatement = connection.prepareStatement(sql);
            if(params.length > 0){
                for (int i = 0; i < params.length; i++){
                    Object object = params[i];
                    if(object instanceof Integer){
                        preparedStatement.setInt(i+1,(Integer) object);
                    }
                    if(object instanceof String){
                        preparedStatement.setString(i+1,(String) object);
                    }
                }
            }
            resultSet = preparedStatement.executeQuery();
            result = resultSetHandler.handle(resultSet);
        } catch (SQLException e) {
            e.printStackTrace();
        } finally {
            JDBCTools.release(null,preparedStatement,resultSet);
        }
        return result;
    }
}
```

四种完成解析的实现类

MyBeanHandler

```
package com.southwind.util;

import org.apache.commons.dbutils.ResultSetHandler;

import java.lang.reflect.InvocationTargetException;
import java.lang.reflect.Method;
import java.sql.ResultSet;
import java.sql.ResultSetMetaData;
import java.sql.SQLException;

public class MyBeanHandler implements ResultSetHandler {

    private Class aClass;

    public MyBeanHandler(Class aClass){
        this.aClass = aClass;
    }

    @Override
    public Object handle(ResultSet resultSet) throws SQLException {
        //结果集的结构（字段信息）
        ResultSetMetaData resultSetMetaData = resultSet.getMetaData();
        int size = resultSetMetaData.getColumnCount();
        Object object = null;
        try {
            object = aClass.getConstructor().newInstance();
            if (resultSet.next()){
                for(int i = 1; i <= size; i++){
                    String className =
resultSetMetaData.getColumnClassName(i);
                    String columnName = resultSetMetaData.getColumnName(i);
                    String methodName =
"set"+columnName.substring(0,1).toUpperCase()+columnName.substring(1);
                    Method method = null;
                    switch (className){
                        case "java.lang.String":
                            String valueStr =
resultSet.getString(columnName);
                            method =
aClass.getMethod(methodName,String.class);
                            method.invoke(object,valueStr);
                            break;
                        case "java.lang.Integer":
                            int valueInt = resultSet.getInt(columnName);
                            method =
aClass.getMethod(methodName,int.class);
                            method.invoke(object,valueInt);
                            break;
                    }
                }
            }
        } catch (InstantiationException e) {
            e.printStackTrace();
        } catch (IllegalAccessException e) {
            e.printStackTrace();
        } catch (InvocationTargetException e) {
            e.printStackTrace();
        } catch (NoSuchMethodException e) {
            e.printStackTrace();
        }
        return object;
    }
}
```

MyBeanListHandler

```
package com.southwind.util;

import org.apache.commons.dbutils.ResultSetHandler;

import java.lang.reflect.InvocationTargetException;
import java.lang.reflect.Method;
import java.sql.ResultSet;
import java.sql.ResultSetMetaData;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;

public class MyBeanListHandler implements ResultSetHandler {

    private Class aClass;

    public MyBeanListHandler(Class aClass){
        this.aClass = aClass;
    }

    @Override
    public Object handle(ResultSet resultSet) throws SQLException {
        ResultSetMetaData resultSetMetaData = resultSet.getMetaData();
        List list = new ArrayList();
        int size = resultSetMetaData.getColumnCount();
        try {
            Object object = null;
            while(resultSet.next()){
                object = aClass.getConstructor().newInstance();
                for(int i = 1; i <= size+1; i++){
                    String className =
resultSetMetaData.getColumnClassName(i);
                    String columnName = resultSetMetaData.getColumnName(i);
                    String methodName =
"set"+columnName.substring(0,1).toUpperCase()+columnName.substring(1);
                    Method method = null;
                    Object value = null;
                    switch (className){
                        case "java.lang.String":
                            value = resultSet.getString(i);
                            method =
aClass.getMethod(methodName,String.class);
                            break;
                        case "java.lang.Integer":
                            value = resultSet.getInt(i);
                            method =
aClass.getMethod(methodName,int.class);
                            break;
                    }
                    method.invoke(object,value);
                }
                list.add(object);
            }
        } catch (InstantiationException e) {
            e.printStackTrace();
        } catch (IllegalAccessException e) {
            e.printStackTrace();
        } catch (InvocationTargetException e) {
            e.printStackTrace();
        } catch (NoSuchMethodException e) {
            e.printStackTrace();
        }
        return list;
    }
}
```

MyMapHandler

```
package com.southwind.util;

import org.apache.commons.dbutils.ResultSetHandler;

import java.lang.reflect.InvocationTargetException;
import java.lang.reflect.Method;
import java.sql.ResultSet;
import java.sql.ResultSetMetaData;
import java.sql.SQLException;
import java.util.HashMap;
import java.util.Map;

public class MyMapHandler implements ResultSetHandler {

    private Class aClass;

    public MyMapHandler(Class aClass){
        this.aClass = aClass;
    }

    @Override
    public Map<Integer,Object> handle(ResultSet resultSet) throws
SQLException {
        ResultSetMetaData resultSetMetaData = resultSet.getMetaData();
        int size = resultSetMetaData.getColumnCount();
        Map<Integer,Object> map = new HashMap<>();
        Object object = null;
        int id = 0;
        try {
            if(resultSet.next()){
                object = aClass.getConstructor().newInstance();
                for(int i = 1; i <= size; i++){
                    String className =
resultSetMetaData.getColumnClassName(i);
                    String columName = resultSetMetaData.getColumnName(i);
                    if(columName.equals("id")){
                        id = resultSet.getInt("id");
                    }
                    Object value = null;
                    String methodName =
"set"+columName.substring(0,1).toUpperCase()+columName.substring(1);
                    Method method = null;
                    switch (className){
                        case "java.lang.String":
                            value = resultSet.getString(i);
                            method =
aClass.getMethod(methodName,String.class);
                            break;
                        case "java.lang.Integer":
                            value = resultSet.getInt(i);
                            method =
aClass.getMethod(methodName,int.class);
                            break;
                    }
                    method.invoke(object,value);
                }
            }
        } catch (InstantiationException e) {
            e.printStackTrace();
        } catch (IllegalAccessException e) {
            e.printStackTrace();
        } catch (InvocationTargetException e) {
            e.printStackTrace();
        } catch (NoSuchMethodException e) {
            e.printStackTrace();
        }
        map.put(id,object);
        return map;
    }
}
```

MyMapListHandler

```
package com.southwind.util;

import org.apache.commons.dbutils.ResultSetHandler;

import java.sql.ResultSet;
import java.sql.ResultSetMetaData;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

public class MyMapListHandler implements ResultSetHandler {

    @Override
    public Object handle(ResultSet resultSet) throws SQLException {
        ResultSetMetaData resultSetMetaData =  resultSet.getMetaData();
        int size = resultSetMetaData.getColumnCount();
        List<Map<String,Object>> list = new ArrayList<>();
        while(resultSet.next()){
            Map<String,Object> map = new HashMap<>();
            for(int i = 1; i <= size; i++){
                String columnClass =
resultSetMetaData.getColumnClassName(i);
                String columnName = resultSetMetaData.getColumnName(i);
                Object object = null;
                switch (columnClass){
                    case "java.lang.String":
                        object = resultSet.getString(i);
                        break;
                    case "java.lang.Integer":
                        object = resultSet.getInt(i);
                        break;
                }
                map.put(columnName,object);
                list.add(map);
            }
        }
        return list;
    }
}
```

Test

```
package com.southwind.test;

import com.southwind.entity.User;
import com.southwind.util.JDBCTools;
import com.southwind.util.MyMapHandler;
import com.southwind.util.MyMapListHandler;
import com.southwind.util.MyQueryRunner;

import java.sql.Connection;
import java.util.List;
import java.util.Map;

public class Test {
    public static void main(String[] args) {

        MyQueryRunner myQueryRunner = new MyQueryRunner();
        Connection connection = JDBCTools.getConnection();

        String sql = "select * from user where id = ?";
        myQueryRunner.query(connection,sql,new
MyBeanHandler(User.class),21);

        String sql = "select * from course where id = ?";
        Course course = (Course) myQueryRunner.query(connection,sql,new
MyBeanHandler(Course.class),1);
        System.out.println(course);

        String sql = "select * from user where name = ? and age = ?";
        User user = (User) myQueryRunner.query(connection,sql,new
MyBeanHandler(User.class),"李四",22);
        System.out.println(user);

        String sql = "select * from user";
        List<User> list = (List<User>)
myQueryRunner.query(connection,sql,new BeanListHandler(User.class));
        for(User user:list){
            System.out.println(user);
        }

        String sql = "select * from course";
        List<Course> list = (List<Course>)
myQueryRunner.query(connection,sql,new BeanListHandler(Course.class));
        for(Course course:list){
            System.out.println(course);
        }

        String sql = "select * from user where id = ?";
        Map<Integer,User> map = (Map<Integer, User>)
myQueryRunner.query(connection,sql,new MyMapHandler(User.class),21);
        System.out.println(map);

        String sql = "select * from user";
        List<Map<String,Object>> list = (List<Map<String,Object>>)
myQueryRunner.query(connection,sql,new MyMapListHandler());
        System.out.println(list);

        JDBCTools.release(connection,null,null);
    }
}
```