- BeanHandler：将结果集解析为Java Bean

```
QueryRunner queryRunner = new QueryRunner();
Connection connection = JDBCTools.getConnection();
String sql = "select * from user where id = ?";
try {
    User user = (User) queryRunner.query(connection,sql,new
BeanHandler(User.class),11);
    System.out.println(user);
} catch (SQLException e) {
    e.printStackTrace();
}finally {
    JDBCTools.release(connection);
}
```

- BeanListHandler：将结果集解析为List

```
QueryRunner queryRunner = new QueryRunner();
Connection connection = JDBCTools.getConnection();
String sql = "select * from user";
try {
    List<User> list = (List<User>) queryRunner.query(connection,sql,new
BeanListHandler<>(User.class));
    for (User user:list){
        System.out.println(user);
    }
} catch (SQLException e) {
    e.printStackTrace();
}finally {
    JDBCTools.release(connection);
}
```

- BeanMapHandler：将结果集解析为Map

```
QueryRunner queryRunner = new QueryRunner();
Connection connection = JDBCTools.getConnection();
String sql = "select * from user where id = ?";
try {
    Map<String,User> map = queryRunner.query(connection,sql,new
BeanMapHandler<>(User.class),11);
} catch (SQLException e) {
    e.printStackTrace();
}finally {
    JDBCTools.release(connection);
}
```

- MapListHandler：将结果集解析为List

```
QueryRunner queryRunner = new QueryRunner();
Connection connection = JDBCTools.getConnection();
String sql = "select * from user";
try {
    List<Map<String,Object>> list =
(List<Map<String,Object>>)queryRunner.query(connection,sql,new
MapListHandler());
    for (Map<String,Object> map : list){
        System.out.println(map);
    }
} catch (SQLException e) {
    e.printStackTrace();
}finally {
    JDBCTools.release(connection);
}
```

```
//新增
QueryRunner queryRunner = new QueryRunner();
Connection connection = JDBCTools.getConnection();
String sql = "insert into user(name,age,money) values(?,?,?)";
try {
    int row = queryRunner.update(connection,sql,"哈哈",33,500);
    System.out.println(row);
} catch (SQLException e) {
    e.printStackTrace();
} finally {
    JDBCTools.release(connection);
}
```

```java
//修改
QueryRunner queryRunner = new QueryRunner();
Connection connection = JDBCTools.getConnection();
String sql = "update user set name = ?,age = ?,money = ? where id = ?";
try {
    int row = queryRunner.update(connection,sql,"呵呵",30,600,20);
    System.out.println(row);
} catch (SQLException e) {
    e.printStackTrace();
} finally {
    JDBCTools.release(connection);
}
```

```java
//删除
QueryRunner queryRunner = new QueryRunner();
Connection connection = JDBCTools.getConnection();
String sql = "delete from user where id = ?";
try {
    int row = queryRunner.update(connection,sql,20);
} catch (SQLException e) {
    e.printStackTrace();
} finally {
    JDBCTools.release(connection);
}
```

DBUtils工具：以查询为例，工具内部根据外部传来的Connection和SQL语句去执行命令，得到ResultSet，根据外部传来的类的模版将结果集解析为对应的JavaBean进行返回。

反射，JDBC

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,int... id){
        PreparedStatement preparedStatement = null;
        ResultSet resultSet = null;
        Object object = null;
        try {
            preparedStatement = connection.prepareStatement(sql);
            if(id.length > 1){
                preparedStatement.setInt(1,id[0]);
            }
            resultSet = preparedStatement.executeQuery();
            object = resultSetHandler.handle(resultSet);
        } catch (SQLException e) {
            e.printStackTrace();
        } finally {
            try {
                if(preparedStatement != null){
                    preparedStatement.close();
                }
                if(resultSet != null){
                    resultSet.close();
                }
            } catch (SQLException e) {
                e.printStackTrace();
            }
        }
        return object;
    }
}
```

MyBeanHandler

```
package com.southwind.util;

import org.apache.commons.dbutils.ResultSetHandler;

import java.lang.reflect.Constructor;
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 {
        ResultSetMetaData resultSetMetaData = resultSet.getMetaData();
        int count = resultSetMetaData.getColumnCount();
        Object object = null;
        try {
            Constructor constructor = aClass.getConstructor();
            object = constructor.newInstance();
        } catch (NoSuchMethodException e) {
            e.printStackTrace();
        } catch (InstantiationException e) {
            e.printStackTrace();
        } catch (IllegalAccessException e) {
            e.printStackTrace();
        } catch (InvocationTargetException e) {
            e.printStackTrace();
        }

        while(resultSet.next()){
            for(int i = 1; i <= count; i++){
                String columnClassName =
resultSetMetaData.getColumnClassName(i);
                String columnLabel = resultSetMetaData.getColumnLabel(i);
                Object value = null;
                String methodName =
"set"+columnLabel.substring(0,1).toUpperCase()+columnLabel.substring(1);
                Method method = null;
                try {
                    switch (columnClassName){
                        case "java.lang.Integer":
                            value = resultSet.getInt(columnLabel);
                            method =
aClass.getMethod(methodName,int.class);
                            break;
                        case "java.lang.String":
                            value = resultSet.getString(columnLabel);
                            method =
aClass.getMethod(methodName,String.class);
                            break;
                    }
                    method.invoke(object,value);
                } catch (NoSuchMethodException e) {
                    e.printStackTrace();
                } catch (InvocationTargetException e){
                    e.printStackTrace();
                } catch (IllegalArgumentException e){
                    e.printStackTrace();
                } catch (IllegalAccessException e){
                    e.printStackTrace();
                }
            }
        }

        return object;
    }
}
```

MyBeanListHandler

```
package com.southwind.util;

import org.apache.commons.dbutils.ResultSetHandler;

import java.lang.reflect.Constructor;
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 List handle(ResultSet resultSet) throws SQLException {
        ResultSetMetaData resultSetMetaData = resultSet.getMetaData();
        int count = resultSetMetaData.getColumnCount();
        List list = new ArrayList();
        Object object = null;
        while(resultSet.next()){
            try {
                Constructor constructor = aClass.getConstructor();
                object = constructor.newInstance();
            } catch (NoSuchMethodException e) {
                e.printStackTrace();
            } catch (InstantiationException e) {
                e.printStackTrace();
```

```java
            } catch (IllegalAccessException e) {
                e.printStackTrace();
            } catch (InvocationTargetException e) {
                e.printStackTrace();
            }
            for(int i = 1; i <= count; i++){
                String columnClassName =
resultSetMetaData.getColumnClassName(i);
                String columnLabel = resultSetMetaData.getColumnLabel(i);
                Object value = null;
                String methodName =
"set"+columnLabel.substring(0,1).toUpperCase()+columnLabel.substring(1);
                Method method = null;
                try {
                    switch (columnClassName){
                        case "java.lang.Integer":
                            value = resultSet.getInt(columnLabel);
                            method =
aClass.getMethod(methodName,int.class);
                            break;
                        case "java.lang.String":
                            value = resultSet.getString(columnLabel);
                            method =
aClass.getMethod(methodName,String.class);
                            break;
                    }
                    method.invoke(object,value);
                } catch (NoSuchMethodException e) {
                    e.printStackTrace();
                } catch (InvocationTargetException e){
                    e.printStackTrace();
                } catch (IllegalArgumentException e){
                    e.printStackTrace();
                } catch (IllegalAccessException e){
                    e.printStackTrace();
                }
            }
            list.add(object);
        }
        return list;
    }
}
```