JDBC开发存在的缺点

--需求：
从数据库当中查询所有的用户信息，将用户信息封装为用户对象，然后将用户对象放到List集合中

--编写JDBC代码
准备javabean：User

--JDBC开发的缺点：
*缺点一：重复代码太多，降低开发效率（比较繁琐，有些代码没有必要重复）例如下面对查询结果集的处理代码

```
while (rs.next()){
            ...
             //从结果集中获取数据
             --//下面的四行代码反复调用同一个方法：rs.getString("仅仅是这不同");
             int id = rs.getInt("id");
             String loginName = rs.getString("loginName");
             String loginPwd = rs.getString("loginPwd");
             String realName = rs.getString("realName");

             //将上面零散的数据封装到一个user对象中（即封装成javabean）
             //将javabean放到容器userList集合中
             User user = new User();
             //--下面这四行也是反复调用同一类型方法：user.setXxx();
             user.setId(id);
             user.setLoginName(loginName);
             user.setLoginPwd(loginPwd);
             user.setRealName(realName);
             userList.add(user);
             ...
        }
```

--上面的代码反复的从结果集中取数据，反复调用对象中给属性赋值的方法，实际开发中数据库中字段数超过30个都是很
--常见的，上面重复的调用方法就会巨繁琐，那么这些操作我们完全可以用反射机制替代
--而MyBatis框架就是别人提前写好的java代码，在mybatis框架中封装了JDBC代码
--其底层使用了反射机制，帮我们自动创建java对象，自动给java对象的属性赋值，以上代码在mybatis中就不需要写了

* 缺点二：
在JDBC开发中sql语句是编写在java程序中的，sql语句不支持配置。当SQL语句后期需要调优时，SQL语句被修改的概率是很高的。在Java程序中编写SQL语句，后期修改，就得修改Java源代码，这就导致代码需要重新编译，项目需要重新部署等操作。
修改源代码违背了开闭原则:OCP
--在互联网分布式架构的项目中，并发量大，系统需要不断优化，各方面的优化就有一方面是SQL语句的优化，SQL语句的优化是很重要的。

创建一个名为a-jdbc-test的普通java项目，导入MySQL数据库驱动，编写一个测试类JDBCTest01.java，然后在主要新建一个entity包存放实体类，实体类用于创建所查询数据对应的实体类对象，用该对象封装查询到的数据

JDBCTest01.java代码如下：

```
package com.mybatis.jdbc.test;

import com.mybatis.jdbc.entity.User;

import java.sql.*;
import java.util.ArrayList;
import java.util.List;

public class JDBCTest01 {

    public static void main(String[] args) {

        Connection conn = null;
        PreparedStatement ps = null;
        ResultSet rs = null;

        List<User> userList = new ArrayList();

        try {
            //
            Class.forName("com.mysql.jdbc.Driver");

            //
            conn =
DriverManager.getConnection("jdbc:mysql://localhost:3306/mysqlstudy?
useSSH=false","root","rong195302");

            //
            String sql = "select id,loginName,loginPwd,realName from t_user";
            ps = conn.prepareStatement(sql);

            //
            rs = ps.executeQuery();

            //
            while (rs.next()){

                //从结果集中获取数据
                int id = rs.getInt("id");
                String loginName = rs.getString("loginName");
                String loginPwd = rs.getString("loginPwd");
                String realName = rs.getString("realName");

                //将上面零散的数据封装到一个user对象中（即封装成javabean）
                //将javabean放到容器userList集合中
                User user = new User();
                user.setId(id);
                user.setLoginName(loginName);
                user.setLoginPwd(loginPwd);
                user.setRealName(realName);
                userList.add(user);
            }

        } catch (ClassNotFoundException e) {
            e.printStackTrace();
```

```
        } catch (SQLException throwables) {
            throwables.printStackTrace();
        } finally {
            if (rs != null) {
                try {
                    rs.close();
                } catch (SQLException throwables) {
                    throwables.printStackTrace();
                }
            }

            if (ps != null) {
                try {
                    ps.close();
                } catch (SQLException throwables) {
                    throwables.printStackTrace();
                }
            }

            if (conn != null) {
                try {
                    conn.close();
                } catch (SQLException throwables) {
                    throwables.printStackTrace();
                }
            }
        }

        //拿着List集合去做展示（MVC中的View）
        for (User user:
             userList) {
            System.out.println(user);
        }

    }

}
```

实体类User.java如下

```
package com.mybatis.jdbc.entity;

public class User {

    int id;
    String loginName;
    String loginPwd;
    String realName;

    public User() {
    }

    public int getId() {
        return id;
    }
```

```
    public void setId(int id) {
        this.id = id;
    }

    public String getLoginName() {
        return loginName;
    }

    public void setLoginName(String loginName) {
        this.loginName = loginName;
    }

    public String getLoginPwd() {
        return loginPwd;
    }

    public void setLoginPwd(String loginPwd) {
        this.loginPwd = loginPwd;
    }

    public String getRealName() {
        return realName;
    }

    public void setRealName(String realName) {
        this.realName = realName;
    }

    @Override
    public String toString() {
        return "User{" +
                "id=" + id +
                ", loginName='" + loginName + '\'' +
                ", loginPwd='" + loginPwd + '\'' +
                ", realName='" + realName + '\'' +
                '}';
    }
}
```