【编程新技术实务】实验一 Java语言编程（JDBC封装）

原创

就算过了一载春秋 于 2019-11-04 22:01:12 发布 3263 收藏 30

分类专栏： \/\/Java \/\/编程新技术实务 文章标签： JDBC 封装 Java SQL

本文链接：https://blog.csdn.net/qq_40889820/article/details/102742406

\/\/Java 同时被 2 个专栏收录

7 篇文章 0 订阅

订阅专栏

\/\/编程新技术实务

5 篇文章 1 订阅

订阅专栏

目录

实验目的

安装、配置好Java编程环境、数据库环境，使用Java 进行编程以及数据库编程。

实验对应知识点

Java编译、运行，path、classpath环境变量，规范注释。数据库的JDBC驱动。

实验前任务

学习Java的基本语法以及数据库的理论基础。

实验要求及步骤

1、Java环境的安装、配置

略。懒得装新的，用的JDK8。
这是以前安装时记录的->Windows10搭建Java环境

2、安装开发工具Eclipse IDE for JavaEE或者MyEclipse

略。
这是以前装Eclipse时记录的->Eclipse的下载、安装与汉化
现在改用IDEA了，方便是真的方便。

3、数据库的安装配置

略。装的MySQL 8.0。
（数据库这门课下学期学，这学期就搞关于数据库的实验，这排课排的有点水平。所以目前对数据库的理解非常浅显。）

4、使用Java编程完成如下任务

（1）创建数据库表

> 创建数据库表users，字段分别为
> **username**(主键，varchar(10))、
> **pass** (varchar(8))；
> 数据库表person，字段分别为
> **username**(varchar(10)，对应于users表的username)、
> **name**(主键，varchar(20))、
> **age**(int，可以为空)、
> **teleno**(char(11)，可以为空)；
> 如表users中username则表person中也不能有相应的username的数据。

（这实验应该是上古实验，上面最后一句不知道说的啥，算了，反正我也不想知道
可能是要用外键约束吧）

（2）插入数据

> 在表users中插入4行数据，数据分别是
> (ly,123456)、(liming,345678)、(test, 11111)、(test1,12345)，
> 在表person中插入3行数据，数据分别为
> (ly,雷力)、(liming,李明,25)、(test,测试用户,20,13388449933)

（3）插入数据and查询数据

> 在person表中插入5行数据，分别为
> (ly,王五)、(test2,测试用户2)、(test1,测试用户1,33)、
> (test,张三,23,18877009966)、(admin,admin)。
> 对于表中已有的**username**，则根据最新的数据修改其相应字段值；
> 如该**username**不存在，则首先在表users中插入该**username**，
> 默认的password为888888，然后才能将数据插入至person表。

（4）删除数据

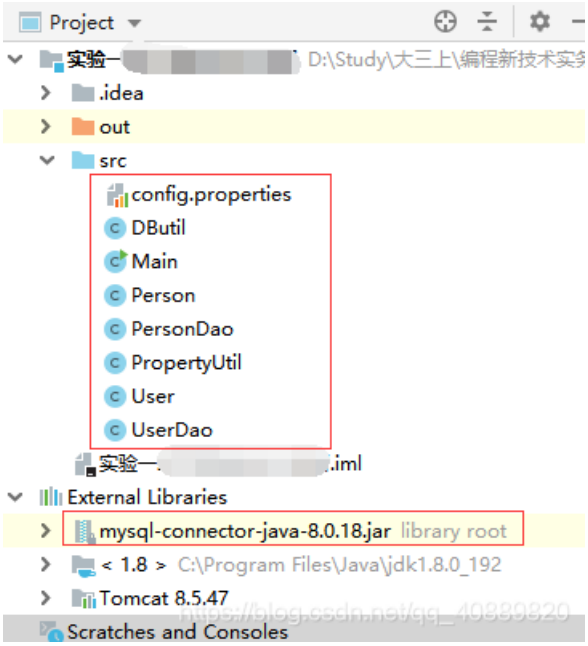
> 删除users表中test打头的**username**，同时按照规则一并删除person表相应的数据

（5）类的设计要求：

> 需要将数据库的连接、操作进行封装，以便在后续实验中进行重用。此项为扣分项，没有进行封装的实验分会相应扣减。
> 自此实验结束，按照要求提交源代码进行验收。
> 要求每个处理阶段均要在控制台打印出处理完成后的结果，格式按照制表方式输出，如：
> 表users
> 字段名xx 字段名xx ….
> xx xx
> 表person
> 字段名xx 字段名xx ….
> xx xx

实验代码

项目结构图



其中，config.properties为配置文件，PropertyUtil类加载配置文件，DButil类封装了链接数据库、增删改查等操作，User类是数据库中users表中一行数据的映射，Person类是数据库中person表中一行数据的映射，UserDao类实现对users表的具体操作，PersonDao类实现对person表的具体操作。

以下代码仅供参考

config.properties文件

```
url=jdbc:mysql://localhost:3306/test?serverTimezone=GMT%2B8&useUnicode=true&characterEncoding=utf-8
driverClass=com.mysql.cj.jdbc.Driver
user=root
password=********
```

这里只让配置文件提供了链接数据库时所需的url、驱动、数据库账号和密码。

PropertyUtil类

```java
package lab1;
import java.io.*;
import java.util.Properties;
public class PropertyUtil {
    public static String getValue(String key){
        String ret = null;
        try{
            InputStream in = PropertyUtil.class.getResourceAsStream("config.properties");
            Properties properties = new Properties();
            properties.load(in);
            ret = properties.getProperty(key);
            in.close();
        }catch (Exception e){
            e.printStackTrace();
        }
        return ret;
    }
}
```

例：PropertyUtil.getValue(“user”) = “root”
即加载配置文件中的数据

DButil类

```java
import java.util.*;
import java.sql.*;
public class DButil {
    private String url = PropertyUtil.getValue("url");
    private String user = PropertyUtil.getValue("user");
    private String password = PropertyUtil.getValue("password");
    private String driverClass = PropertyUtil.getValue("driverClass");
    private Connection con = null;
    public DButil(){
        con = getConnection();
    }
    public Connection getConnection() {
        Connection conn = null;
        try{
            System.out.println("正在连接数据库...");
            Class.forName(driverClass);
            conn = DriverManager.getConnection(url,user,password);
            System.out.println("数据库连接成功！");
        }catch (Exception e){
            e.printStackTrace();
        }
        return conn;
    }
    //赠、删、改等调用这个函数
    public int executeUpdate(String sql,Object... params){
        int rlt = 0;
        try{
            PreparedStatement pstmt = null;
            pstmt = con.prepareStatement(sql);
            putParams(pstmt,params);
            rlt = pstmt.executeUpdate();
            pstmt.close();
        }catch (Exception e){
            e.printStackTrace();
```

```
        }
        return rlt;
    }
    //放置参数
    private void putParams(PreparedStatement pstmt, Object[] params) throws SQLException{
        if(params!=null){
            for(int i=0;i<params.length;++i){
                //pstmt.setObject(i+1,params);
                if(params[i] instanceof String) pstmt.setString(i+1,(String)params[i]);
                else if(params[i] instanceof Integer) pstmt.setInt(i+1,(Integer)params[i]);
                else if(params[i] == null) pstmt.setNull(i+1, Types.INTEGER);
            }
        }
    }
    //查询调用这个函数
    public List<Map<String,Object>> query(String sql,Object... params){
        PreparedStatement pstmt = null;
        List<Map<String,Object>> list = null;
        try{
            pstmt = con.prepareStatement(sql);
            putParams(pstmt,params);
            ResultSet rs = pstmt.executeQuery();
            ResultSetMetaData rsmd = rs.getMetaData();
            String[] keys = new String[rsmd.getColumnCount()];
            for(int i=1;i<=rsmd.getColumnCount();++i){
                keys[i-1] = rsmd.getColumnLabel(i);
            }
            list = new ArrayList<Map<String,Object>>();
            while(rs.next()){
                Map<String,Object> map = new HashMap<String,Object>();
                for(int i=0;i<keys.length;++i){
                    map.put(keys[i],rs.getObject(keys[i]));
                }
                list.add(map);
            }
            rs.close();
            pstmt.close();
        }catch (Exception e){
            e.printStackTrace();
        }
        return list;
    }
    public void close(){
        if(this.con!=null){
            try{
                this.con.close();
                System.out.println("已关闭接口..");
            }catch (Exception e){
                e.printStackTrace();
            }
        }
    }
}
```

User类

```java
public class User {
    private String username;//primary key
    private String password;
    public User(String username,String password){
        this.username = username;
        this.password = password;
    }
    public String getUsername() {
        return username;
    }

    public String getPassword() {
        return password;
    }

    public void setUsername(String username) {
        this.username = username;
    }

    public void setPassword(String password) {
        this.password = password;
    }
}
```

Person类

```java
public class Person{
    private String username;
    private String name;//Primary key
    private Integer age;
    private String teleno;
    public Person(String username,String name,Integer age,String teleno){
        this.username = username;
        this.name = name;
        this.age = age;
        this.teleno = teleno;
    }
    public Person(String username,String name){
        this(username,name,null,"");
    }
    public Person(String username,String name,Integer age){
        this(username,name,age,"");
    }
    public String getUsername(){
        return username;
    }
    public String getName(){
        return name;
    }
    public Integer getAge(){
        return age;
    }
    public String getTeleno(){
        return teleno;
    }
    public void setUsername(String username){
        this.username = username;
    }
    public void setName(String name){
        this.name = name;
    }
    public void setAge(Integer age){
        this.age = age;
    }
    public void setTeleno(String teleno){
        this.teleno = teleno;
    }
}
```

UserDao类

```java
/*
这个类中不要出现和SQL相关的类比如Connection、Statement、PreparedStatement、ResultSet等
*/
import java.util.*;
public class UserDao {
    public void createTable(DButil dbutil) throws Exception {
        String sql = "create table users"
                + "( "
                + "username varchar(10) not null,"
                + "pass varchar(8) not null,"
                + "primary key (username)"
                + ")";
        dbutil.executeUpdate(sql);
    }

    public void addUser(User u,DButil dbutil) throws Exception {
        String sql = "insert into users(username,pass) values(?, ?)";
        Object[] obj = {u.getUsername(),u.getPassword()};
        dbutil.executeUpdate(sql,obj);
    }

    public void delUserOnUsername(String username,DButil dbutil) throws Exception {
        String sql = "delete from users where username like ?";
        dbutil.executeUpdate(sql,username);
    }

    public void dropTable(DButil dbutil) throws Exception{
        String sql = "drop table users";
        dbutil.executeUpdate(sql);
    }

    public List<User> queAll(DButil dbutil) throws Exception {
        String sql = "select * from users";
        List<Map<String,Object>> list = dbutil.query(sql);
        List<User> userList = new ArrayList<>();
        User user = null;
        for(Map<String,Object> map:list){
            user = new User((String)map.get("username"),(String)map.get("pass"));
            //System.out.println(map.get("username")+" "+map.get("pass"));
            userList.add(user);
        }
        return userList;
    }
}
```

PersonDao类

```java
/*
这个类中不要出现和SQL相关的类比如Connection、Statement、PreparedStatement、ResultSet等
*/
import java.util.ArrayList;
import java.util.List;
import java.util.Map;
public class PersonDao{
    public void createTable(DButil dbutil) throws Exception{
        String sql = "create table person"
                + "("
                + "username varchar(10) not null,"
                + "name varchar(20) not null,"
```

```
                + "age int default 18,"
                + "teleno char(11) default '18570253175',"
                + "primary key (name)"
                + ")";
        dbutil.executeUpdate(sql);
    }

    public void addPerson(Person p,DButil dbutil) throws Exception{
        String sql = "insert into person(username,name,age,teleno) values(?, ?, ?, ?)";
        Object[] obj = {p.getUsername(),p.getName(),p.getAge(),p.getTeleno()};
        dbutil.executeUpdate(sql,obj);
    }

    public void addOrModifyPerson(Person p,DButil dbutil) throws Exception{
        String sql_1 = "select * from person where username=?";
        List<Map<String,Object>> userlist,personlist;
        personlist = dbutil.query(sql_1,p.getUsername());
        if(personlist.isEmpty()){//person表中不存在该username
            String sql_2 = "select * from users where username=?";
            userlist = dbutil.query(sql_2,p.getUsername());
            if(userlist.isEmpty()){//user表中也不存在该username,
                String sql_3 = "insert into users(username,pass) values(?,?)";
                Object[] obj = {p.getUsername(), "888888"};//默认的passwaod为888888
                dbutil.executeUpdate(sql_3,obj);
            }
            String sql_4 = "insert into person(username,name,age,teleno) values(?, ?, ?, ?)";
            Object[] obj = {p.getUsername(),p.getName(),p.getAge(),p.getTeleno()};
            dbutil.executeUpdate(sql_4,obj);
        }
        else{
            String sql_5 = "update person set name=?,age=?,teleno=? where username=?";
            Object[] obj = {p.getName(),p.getAge(),p.getTeleno(),p.getUsername()};
            dbutil.executeUpdate(sql_5,obj);
        }
    }
    public void delPersonOnUsername(String username,DButil dbutil) throws Exception{
        String sql = "delete from person where username like ?";
        dbutil.executeUpdate(sql,username);
    }

    public void dropTable(DButil dbutil) throws Exception{
        String sql = "drop table person";
        dbutil.executeUpdate(sql);
    }

    public List<Person> queAll(DButil dbutil) throws Exception{
        String sql = "select * from person";
        List<Map<String,Object>> list = dbutil.query(sql);
        List<Person> personList = new ArrayList<>();
        Person person = null;
        for(Map<String,Object>map:list){
            person = new Person((String)map.get("username"),(String)map.get("name"),(Integer)map.get("age"),(Str
ing)map.get("teleno"));
            personList.add(person);
        }
        return personList;
    }
}
```

Main类

```
import java.util.*;
public class Main {
    DButil dbutil = new DButil();
    UserDao userDao = new UserDao();
    PersonDao personDao = new PersonDao();
    public void step_1()throws Exception{
        userDao.createTable(dbutil);//创建数据库表users
        personDao.createTable(dbutil);//创建数据库表person
        //打印结果
        userDao.queAll(dbutil);
        personDao.queAll(dbutil);
        show();
    }
    public void step_2() throws Exception{
        //在表users中插入4行数据
        userDao.addUser(new User("ly","123456"),dbutil);
        userDao.addUser(new User("liming","345678"),dbutil);
        userDao.addUser(new User("test","11111"),dbutil);
        userDao.addUser(new User("test1","12345"),dbutil);
        //在表person中插入3行数据
        personDao.addPerson(new Person("ly","雷力"),dbutil);
        personDao.addPerson(new Person("liming","李明",25),dbutil);
        personDao.addPerson(new Person("test","测试用户",20,"13388449933"),dbutil);
        //打印结果
        userDao.queAll(dbutil);
        personDao.queAll(dbutil);
        show();
    }
    public void step_3() throws Exception{
        //在person表中插入5行数据，对于表中已有的username，根据最新的数据修改；如不存在，首先在user表中插入该username，再插入person表
        personDao.addOrModifyPerson(new Person("ly","王五"),dbutil);
        personDao.addOrModifyPerson(new Person("test2","测试用户2"),dbutil);
        personDao.addOrModifyPerson(new Person("test1","测试用户1",33),dbutil);
        personDao.addOrModifyPerson(new Person("test","张三",23,"18877009966"),dbutil);
        personDao.addOrModifyPerson(new Person("admin","admin"),dbutil);
        //打印结果
        userDao.queAll(dbutil);
        personDao.queAll(dbutil);
        show();
    }
    public void step_4() throws Exception{
        //删除users表中test打头的username
        userDao.delUserOnUsername("test",dbutil);
        //删除person表中test打头的usename
        personDao.delPersonOnUsername("test%",dbutil);
        //打印结果
        userDao.queAll(dbutil);
        personDao.queAll(dbutil);
        show();
    }
    public void show() throws Exception{//输出格式就不搞的很好了
        System.out.println("表users:");
        System.out.println("+********************************************************************************+");
        System.out.println("|字段名 username                    字段名 pass                                  |");
        System.out.println("+--------------------------------------------------------------------------------+");
        List<User> list = userDao.queAll(dbutil);
        for(User user:list){
```

```
        for(User user:list){
            String s1 = user.getUsername();
            String s2 = user.getPassword();
            System.out.printf("%20s"+"\t\t"+"%s"+"\t\t\n",s1,s2);
            System.out.println("+------------------------------------------------------------------------------+");
        }
        List<Person> personList = personDao.queAll(dbutil);
        System.out.println("表person:");
        System.out.println("+**************************************************************************+");
        System.out.println("|字段名 username        字段名 name        字段名age         字段名 teleno|");
        System.out.println("+------------------------------------------------------------------------------+");
        for(Person person:personList){
            String s1 = person.getUsername();
            String s2 = person.getName();
            Integer s3 = person.getAge();
            String s4 = person.getTeleno();
            System.out.printf("%10s"+"\t\t"+"%10s"+"\t\t",s1,s2);
            if(s3!=null) System.out.print(s3);
            System.out.println("\t\t"+s4);
            System.out.println("+------------------------------------------------------------------------------+");
        }
    }
    public static void main(String[] argc)throws Exception{
        Main m = new Main();
        m.userDao.dropTable(m.dbutil);
        m.personDao.dropTable(m.dbutil);
        System.out.println("第一步: ");
        m.step_1();
        System.out.println("第二步: ");
        m.step_2();
        System.out.println("第三步: ");
        m.step_3();
        System.out.println("第四步: ");
        m.step_4();
        m.dbutil.close();
    }
}
```

实验结果

（改前效果，格式什么的自己慢慢调吧）

```
第一步:
表users:
+**************************************************************************+
|字段名 username                    字段名 pass                            |
+------------------------------------------------------------------------+
表person:
+**************************************************************************+
|字段名 username        字段名 name        字段名age         字段名 teleno|
+------------------------------------------------------------------------+
```



```
第二步:
表users:
+**************************************************************************+
```

字段名 username	字段名 pass
liming	345678
ly	123456
test	11111
test1	12345

表person:

字段名 username	字段名 name	字段名age	字段名 teleno
liming	李明	25	
test	测试用户	20	13388449933
ly	雷力		

第三步:

表users:

字段名 username	字段名 pass
admin	888888
liming	345678
ly	123456
test	11111
test1	12345
test2	888888

表person:

字段名 username	字段名 name	字段名age	字段名 teleno
admin	admin		
test	张三	23	18877009966

```
|      liming          李明                  25
+-----------------------------------------------------------------------+
|      test1           测试用户1              33
+-----------------------------------------------------------------------+
|      test2           测试用户2
+-----------------------------------------------------------------------+
|      ly              王五
+-----------------------------------------------------------------------+
```


第四步：

表users：

```
+***********************************************************************+
|字段名 username                  字段名 pass                           |
+-----------------------------------------------------------------------+
|      admin                      888888
+-----------------------------------------------------------------------+
|      liming                     345678
+-----------------------------------------------------------------------+
|      ly                         123456
+-----------------------------------------------------------------------+
```

表person：

```
+***********************************************************************+
|字段名 username        字段名 name        字段名age         字段名 teleno|
+-----------------------------------------------------------------------+
|      admin            admin
+-----------------------------------------------------------------------+
|      liming           李明                  25
+-----------------------------------------------------------------------+
|      ly               王五
+-----------------------------------------------------------------------+
```


主要参考资料

【1】https://www.cnblogs.com/zxwen/p/9832385.html
【2】https://blog.csdn.net/kpchen_0508/article/details/41287137
【3】https://www.runoob.com/w3cnote/jdbc-use-guide.html
【4】https://blog.csdn.net/tangyuanzong/article/details/78346979

总结

这个实验是为后面做**Java Web开发**做准备的，此前没接触过此类知识，感觉挺有趣的，但前前后后也花了挺久，评分经历了从B到B-到B+，最后将查询操作改好就没再去验收了。感谢为我提供帮助的同学们以及老师。
这里列出老师提过的几点：

- 少用静态变量，能不用就不用
- 接口用一个
- 不要把Connection等暴露出来，不安全（网上有相当一部分此类代码）
- DButil类不要用单例模式

知识浅薄，若有错误、不足之处欢迎私信或留言。