青岛理工大学

**《Java程序设计课程实训》**

题目 带界面的电子商城系统

姓名： 侯宇轩

学号： 201811050392

班级： 软件186

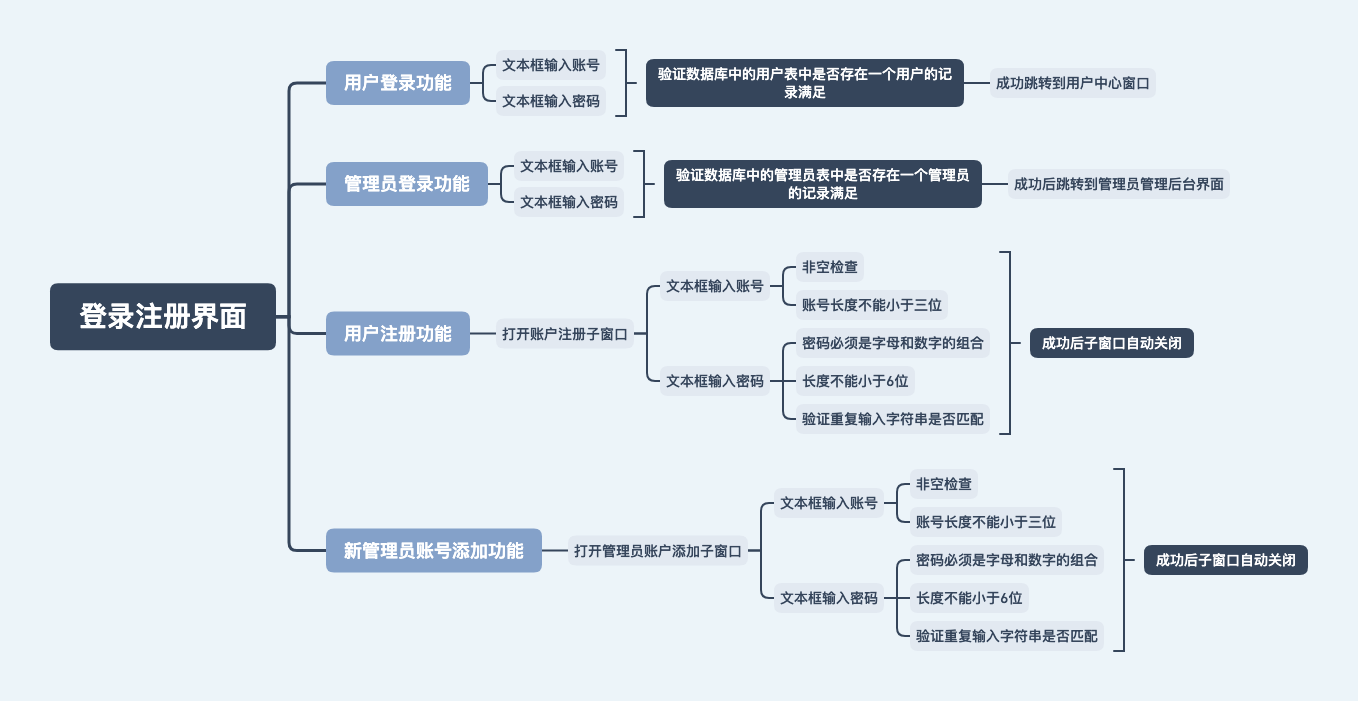
专业：软件工程（移动互联网方向）

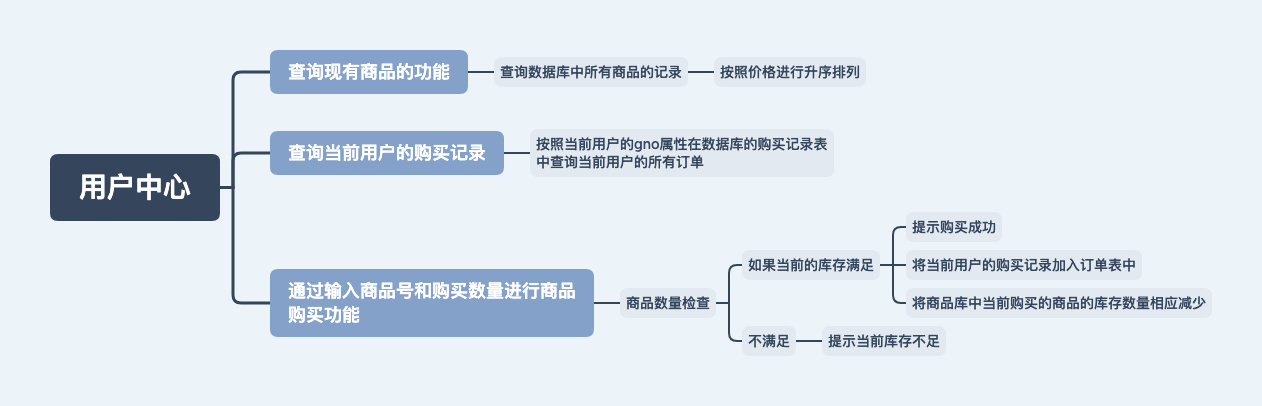
地点： 青软实训总部基地

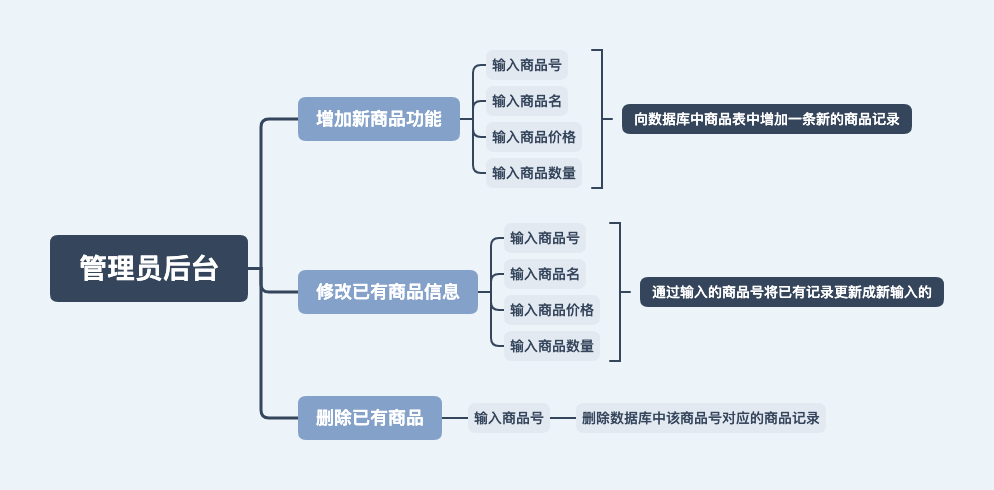
时间： 2020/6/22 至 2020/7/10

**一、项目名称**  
　　带模拟界面的电子商城系统

**二、功能要求**







**三、需求分析**

电子商城整体架构流程图：

主界面

输入账号

输入密码

选择登陆身份

注册账号

用户

管理员

账号密码验证

提示未通过原因

验证成功，进入用户中心

账号密码验证

提示未通过原因

验证成功进入商品管理中心

查询所有可以购买的商品

查询当前登入

用户的购买记录

购买商品

修改商品信息

删除商品

添加商品

添加管理员账户

查询是否有库存

库存充足

购买成功

提示库存不足

购买失败

修改当前商品数据库中此商品的库存

将当前购买记录加入购买记录表中

对商品数据库表进行相应的操作

查询商品库存

**四、设计思想**

**主要是利用windowbuilder搭建界面，然后通过mysql数据库实现用户数据、管理员数据、商品数据、订单数据的增删改查功能，通过正则表达式实现账号密码的格式检验。**

**五、具体实现**

**实现图片：**

**登录检验：**

****

**注册界面的格式检验：**

****

**注册界面的非空检验：**

****

**注册成功：**

****

**管理员用户界面：**

****

**用户中心:**

****

**代码模块：**

**注册登录界面实现功能实现的几个按钮代码：**

1. JButton register\_button = **new** JButton("注册");
2. register\_button.addActionListener(**new** ActionListener() {
3. **public** **void** actionPerformed(ActionEvent e) {
4. register regwin=**new** register();
5. regwin.setVisible(**true**);
6. table.add(regwin);
7. }
8. });

11. register\_button.setBounds(426, 417, 96, 46);
12. table.add(register\_button);

15. JButton add\_new\_admin = **new** JButton("创建新管理员");
16. add\_new\_admin.addActionListener(**new** ActionListener() {
17. **public** **void** actionPerformed(ActionEvent e) {
18. add\_admin adminwin=**new** add\_admin();
19. adminwin.setVisible(**true**);
20. table.add(adminwin);
21. }
22. });
23. add\_new\_admin.setBounds(233, 487, 126, 46);
24. table.add(add\_new\_admin);

27. //添加监视器，并进行判断合法后跳转到用户界面
28. JButton usr\_login\_button = **new** JButton("用户登录");
29. usr\_login\_button.addActionListener(**new** ActionListener() {
30. **public** **void** actionPerformed(ActionEvent e) {
31. **if**(!check\_input(get\_usrname.getText()) || !check\_input(get\_password.getText())){ //不为空检查
32. JOptionPane.showMessageDialog(**null**,"账号密码不可为空","",JOptionPane.WARNING\_MESSAGE);
33. get\_usrname.setText(**null**);
34. }
35. **else** **if**(check\_usr.Check\_(get\_usrname.getText(),get\_password.getText()) ){
36. //用户通过验证后，进入商城主界面
37. dispose();
38. usrcenter usrwin=**new** usrcenter();
39. usrwin.get\_id(get\_usrname.getText());
40. usrwin.setVisible(**true**);
42. }
43. **else** {
44. JOptionPane.showMessageDialog(**null**,"账号不存在","",JOptionPane.WARNING\_MESSAGE);
45. get\_usrname.setText(**null**);
46. get\_password.setText(**null**);
47. }
48. }
49. });
50. usr\_login\_button.setBounds(84, 417, 96, 46);
51. table.add(usr\_login\_button);

54. //  添加监视器，并进行判断合法后跳转到管理员后台
55. JButton admin\_login\_button = **new** JButton("管理员登录");
56. admin\_login\_button.addActionListener(**new** ActionListener() {
57. **public** **void** actionPerformed(ActionEvent e) {
58. //  非空检查
59. **if**(!check\_input(get\_usrname.getText()) || !check\_input(get\_password.getText())){
60. JOptionPane.showMessageDialog(**null**,"账号不可为空","",JOptionPane.WARNING\_MESSAGE);
61. get\_usrname.setText(**null**);
62. get\_password.setText(**null**);
63. }
64. **else** **if**(check\_admin.Check(get\_usrname.getText(),get\_password.getText())){
65. //  管理员用户通过验证后，进入商城主界面
66. dispose();
67. admincenter adminwin=**new** admincenter();
68. adminwin.get\_id("123");
69. adminwin.setVisible(**true**);
70. }
71. **else** {
72. JOptionPane.showMessageDialog(**null**,"不存在管理员账户","",JOptionPane.WARNING\_MESSAGE);
73. get\_usrname.setText(**null**);
74. get\_password.setText(**null**);
75. }
76. }
77. });
78. admin\_login\_button.setBounds(246, 417, 96, 46);
79. table.add(admin\_login\_button);

注册子窗口实现的输入检验：

1. JButton register\_button = **new** JButton("注册");
2. register\_button.addActionListener(**new** ActionListener() {
3. **public** **void** actionPerformed(ActionEvent e) {
4. String uname=get\_usrname.getText();
5. String pas=get\_password.getText();
6. String rep\_pas= get\_repeat\_password.getText();
7. /\*账号、密码非空检验\*/
8. **if**(!check\_input(uname)) { //不为空检查
9. JOptionPane.showMessageDialog(**null**,"账号不可为空","",JOptionPane.WARNING\_MESSAGE);
10. get\_usrname.setText(**null**);
11. }
12. **else** **if**(!check\_input(pas)){
13. JOptionPane.showMessageDialog(**null**,"密码不可为空","",JOptionPane.WARNING\_MESSAGE);
14. get\_password.setText(**null**);
15. }
17. /\*账号长度检验\*/
18. **else** **if**(uname.length() < 3) {
19. JOptionPane.showMessageDialog(**null**,"账号不能小于三位","",JOptionPane.WARNING\_MESSAGE);
20. get\_usrname.setText(**null**);
21. }
23. /\*密码长度检验、正则表达式对格式的检验\*/
24. **else** **if**(!Pattern.matches(regex,pas) || pas.length() < 6 ) {
25. JOptionPane.showMessageDialog(**null**,"密码必须是字母和数字的组合，长度不能小于6位","",JOptionPane.WARNING\_MESSAGE);
26. get\_password.setText(**null**);
27. get\_repeat\_password.setText(**null**);
28. }
30. /\*密码的重复输入匹配检验\*/
31. **else** **if**(!pas.equals(rep\_pas)) {
33. JOptionPane.showMessageDialog(**null**,"密码重复不符","",JOptionPane.WARNING\_MESSAGE);
34. get\_password.setText(**null**);
35. get\_repeat\_password.setText(**null**);
36. }
37. **else** {
38. //用户通过验证后，将用户数据插入到数据库之中
39. insert\_usr.Insert\_usr(uname,pas);
40. JOptionPane.showMessageDialog(**null**,"注册成功","",JOptionPane.WARNING\_MESSAGE);
41. dispose();
42. }
43. }
45. });
46. register\_button.setBounds(169, 290, 119, 39);
47. getContentPane().add(register\_button);
48. }
49. **private** **boolean** check\_input(String str) {
50. **boolean** ok=**true**;
51. **if**(str ==**null** || str.length()==0)
52. ok=**false**;
53. **return** ok;
54. }
55. }

用户中心实现的功能按钮：

1. JButton query\_all = **new** JButton("查询所有商品");
2. query\_all.addActionListener(**new** ActionListener() {
3. **public** **void** actionPerformed(ActionEvent e) {
4. //实现查商品表中的所有商品
5. ArrayList<String>list = Query\_all.get\_queryall();
6. show.setText(**null**);
7. **for**(String record: list) {
8. show.append(record+'\n');
9. //需要按照价格从低到高排序显示
10. }
11. }
12. });
13. query\_all.setBounds(90, 522, 117, 64);
14. contentPane.add(query\_all);
16. JButton query\_buy\_records = **new** JButton("查询购买记录");
17. query\_buy\_records.addActionListener(**new** ActionListener() {
18. **public** **void** actionPerformed(ActionEvent e) {
19. //查询购买记录表中该用户的购买记录
20. ArrayList<String>list = Query\_buy\_records.get\_buy\_records(uno); // 函数未修改完成
21. show.setText(**null**);
22. **for**(String record: list) {
23. show.append(record+'\n');
24. }
25. }
26. });
27. query\_buy\_records.setBounds(262, 522, 117, 64);
28. contentPane.add(query\_buy\_records);

31. JButton buy\_button = **new** JButton("购买");
32. buy\_button.addActionListener(**new** ActionListener() {
33. **public** **void** actionPerformed(ActionEvent e) {
34. //购买输入的id的商品，向
35. String gno=goods\_id.getText();
36. String g\_num=goods\_buy\_count.getText();
37. **int** num=Integer.parseInt(g\_num);
38. **int** num\_exist=check\_goods\_count.get(gno);
39. **if**(num\_exist>=num) {
40. String rec=insert\_buy\_records.Insert\_records(uno,gno,g\_num);
41. **int** t=num\_exist-num;
42. String now=String.valueOf(t);
43. subtract\_goods\_count.change(gno,now);
44. show.setText(rec);
45. JOptionPane.showMessageDialog(**null**,"购买成功","",JOptionPane.WARNING\_MESSAGE);
46. }
47. **else** {
48. JOptionPane.showMessageDialog(**null**,"库存不足！请重新选择购买的商品数量","",JOptionPane.WARNING\_MESSAGE);
49. }
50. }
51. });

管理员后台实现的功能按钮：

1. JButton change\_button = **new** JButton("修改商品信息");
2. change\_button.addActionListener(**new** ActionListener() {
3. **public** **void** actionPerformed(ActionEvent e) {
4. String no=change\_gno\_in.getText();
5. String name=change\_gname\_in.getText();
6. String price=change\_gprice\_in.getText();
7. String count=change\_gcount\_in.getText();
8. change\_goods\_message.getchange(no, name, price, count);
9. JOptionPane.showMessageDialog(**null**,"修改商品信息成功","",JOptionPane.WARNING\_MESSAGE);
10. change\_gno\_in.setText(**null**);
11. change\_gname\_in.setText（**null**）;
12. change\_gprice\_in.setText（**null**）;
13. change\_gcount\_in.setText（**null**）;
14. }
15. }）;
16. change\_button.setBounds（177 ，  778 ，  139 ，  48 ）;
17. contentPane.add（change\_button）;
19. delete\_gno\_in =  **新的** JTextField（）;
20. delete\_gno\_in.setColumns（10 ）;
21. delete\_gno\_in.setBounds（394 ，  499 ，  265 ，  37 ）;
22. contentPane.add（delete\_gno\_in）;
24. JButton delete\_button =  **new** JButton（“删除商品” ）;
25. delete\_button.addActionListener（**new** ActionListener（）{
26. **公共无效** actionPerformed（ActionEvent e）{
27. 字符串no = delete\_gno\_in.getText（）;
28. delete\_goods.getdelete（no）; //从当前数据库中删除
29. JOptionPane.showMessageDialog（**null**，“删除商品成功” ，“” ，JOptionPane.WARNING\_MESSAGE）;
30. delete\_gno\_in.setText（**null**）;
31. }
32. }）;
33. delete\_button.setBounds（693 ，  494 ，  139 ，  48 ）;
34. contentPane.add（delete\_button）;
36. show =  **new** JTextArea（）;
37. show.setBounds（266 ，  57 ，  714 ，  370 ）;
38. contentPane.add（显示）;
40. query =  **new** JButton（“查询库存” ）;
41. query.addActionListener（**new** ActionListener（）{
42. **公共无效** actionPerformed（ActionEvent e）{
43. ArrayList <String> list = Query\_all.get\_queryall（）;
44. show.setText（**null**）;
45. **对于**（字符串记录：列表）{
46. show.append（record + '\ n' ）;
47. //需要按照价格从低到高排序显示
48. }
49. }
50. }）;
51. query.setBounds（68 ，  242 ，  127 ，  62 ）;
52. contentPane.add（query）;
53. }
54. **公共无效的** get\_id（String id）{
55. ano = id;
56. }

根据商品号查询商品信息的方法：

1. **package** method;
3. **import** java.sql.SQLException;
5. **import** com.GetDBConnection;
7. **public** **class** change\_goods\_message {
8. **public** **static** **void** main(String args[]) {
9. getchange("201811050392","侯宇轩","信控学院","软件186");
10. }
12. **public** **static** **void** getchange(String gno,String gname,String gprice,String gcount){
13. java.sql.Connection con;
14. java.sql.PreparedStatement preSql;  //预处理语句对象
15. con = GetDBConnection.connectDB();
16. **if**(con == **null** ) **return**;
17. String sqlStr ="update database.goods set gno=?,gname=?,gprice=?,gcount=? where (gno=?)";
18. **try** {
19. preSql = con.prepareStatement(sqlStr);
20. preSql.setString(5,gno);
21. preSql.setString(1,gno);
22. preSql.setString(2,gname);
23. preSql.setString(3,gprice);
24. preSql.setString(4,gcount);
25. **int** ok = preSql.executeUpdate();       //预处理语句调用方法执行
26. con.close();
27. }
28. **catch**(SQLException e) {
29. System.out.println("记录中Sno值不能重复"+e);
30. }
31. }
32. }

管理员账号的存在检验：

1. **package** method;
3. **import** java.sql.ResultSet;
4. **import** java.sql.SQLException;
5. **import** java.util.ArrayList;
7. **import** com.GetDBConnection;
9. **public** **class** check\_admin {
11. **public** **static** **void** main(String args[]) {
12. Check("201811050392","侯宇轩");
13. }
14. **public** **static** **boolean** Check(String adminid,String password){//static 可以直接调用函数，不用创建对象
15. java.sql.Connection con;
16. java.sql.Statement sql;
17. ResultSet rs;
18. **boolean** ok=**false**;
19. con = GetDBConnection.connectDB();
20. **try** {
21. sql=con.createStatement();
23. rs = sql.executeQuery("select \* from database.admin where ano = '"+adminid+"' and apassword = '"+password+"'");
24. **if**(rs.next())
25. ok=**true**;
26. con.close();
27. }
28. **catch**(SQLException e) {}
29. **return** ok;
30. }
31. }

根据商品号查询当前商品的数量：

1. **package** method;
3. **import** java.sql.ResultSet;
4. **import** java.sql.SQLException;
6. **import** com.GetDBConnection;
8. **public** **class** check\_goods\_count {
9. **public** **static** **int** get(String gno){//static 可以直接调用函数，不用创建对象
10. java.sql.Connection con;
11. java.sql.Statement sql;
12. ResultSet rs;
13. **int** cnt=0;
14. con = GetDBConnection.connectDB();
15. **try** {
16. sql=con.createStatement();
17. rs = sql.executeQuery("select gcount from database.goods where gno=\""+gno+"\"");
18. **while**(rs.next()) {
19. 字符串tmp = rs.getString（1 ）;
20. cnt = Integer.parseInt（tmp）;
21. }
22. con.close（）;
23. }
24. **捕获**（SQLException e）{}
26. **返回** cnt;
27. }
28. }

用户账号的存在检验：

1. **package** method;
3. **import** java.sql.ResultSet;
4. **import** java.sql.SQLException;
6. **import** com.GetDBConnection;
8. **public** **interface** check\_usr {
9. **public** **static** **void** main(String args[]) {
10. Check\_("201811050392","侯宇轩");
11. }
12. **public** **static** **boolean** Check\_(String uno,String password){//static 可以直接调用函数，不用创建对象
13. java.sql.Connection con;
14. java.sql.Statement sql;
15. ResultSet rs;
16. **boolean** ok=**false**;
17. con = GetDBConnection.connectDB();
18. **try** {
19. sql=con.createStatement();
20. rs = sql.executeQuery("select \* from database.usr where uno = '"+uno+"' and upassword = '"+password+"'");
21. **if**(rs.next())
22. ok=**true**;
23. con.close();
24. }
25. **catch**(SQLException e) {}
26. **return** ok;
27. }
28. }

在商品表中删除商品的实现：

1. **package** method;
3. **import** java.sql.SQLException;
5. **import** com.GetDBConnection;
7. **public** **class** delete\_goods {
8. **public** **static** **void** main(String args[]) {
9. getdelete("201811050392");
10. }
12. **public** **static** **void** getdelete(String no) {
13. java.sql.Connection con;
14. java.sql.Statement sql;
15. con = GetDBConnection.connectDB();
16. **try** {
17. sql=con.createStatement();
18. sql.executeUpdate("delete from database.goods where gno=\""+no+"\"");
19. con.close();
20. }
21. **catch**(SQLException e) {}
22. }
23. }

获取当前数据库中订单表的数量：

1. **package** method;
3. **import** java.sql.ResultSet;
4. **import** java.sql.SQLException;
5. **import** java.util.ArrayList;
7. **import** com.GetDBConnection;
9. **public** **class** get\_order\_num {
10. **public** **static** **void** main(String args[]) {
11. get();
12. }
13. **public** **static** **int** get(){//static 可以直接调用函数，不用创建对象
14. java.sql.Connection con;
15. java.sql.Statement sql;
16. ResultSet rs;
17. **int** number=0;
18. con = GetDBConnection.connectDB();
19. **try** {
20. sql=con.createStatement();
21. rs = sql.executeQuery("select count(\*) from database.records");
22. **while**(rs.next()) {
23. String tmp=rs.getString(1);
24. number=Integer.parseInt(tmp);
25. }
26. con.close();
27. }
28. **catch**(SQLException e) {}
29. **return** number;
30. }
31. }

获取购买一定数量的某一商品的总价格

1. **package** method;
3. **import** java.sql.ResultSet;
4. **import** java.sql.SQLException;
5. **import** java.util.ArrayList;
7. **import** com.GetDBConnection;
9. **public** **class** get\_total\_price {
10. **public** **static** **void** main(String args[]) {
11. get(0,"201811050392 侯宇轩");
12. }
13. **public** **static** **int** get(**int** cnt,String gno){//static 可以直接调用函数，不用创建对象
14. java.sql.Connection con;
15. java.sql.Statement sql;
16. ResultSet rs;
17. **int** single=0;
18. con = GetDBConnection.connectDB();
19. **try** {
20. sql=con.createStatement();
21. rs = sql.executeQuery("select gprice from database.goods where gno=\""+gno+"\"");
22. **while**(rs.next()) {
23. String tmp=rs.getString(1);
24. single=Integer.parseInt(tmp);
25. }
26. con.close();
27. }
28. **catch**(SQLException e) {}
29. **int** tot = single \* cnt;
30. **return** tot;
31. }
33. }

在用户表中插入一个新的用户：

1. **package** method;
3. **import** java.sql.ResultSet;
4. **import** java.sql.SQLException;
6. **import** com.GetDBConnection;
8. **public** **class** insert\_admin {
9. **public** **static** **void** main(String args[]) {
10. Insert\_admin("201811050392","侯宇轩");
11. }
12. **public** **static** **void** Insert\_admin(String ano,String apassword) {
13. java.sql.Connection con;
14. java.sql.PreparedStatement preSql;  //预处理语句对象
15. con = GetDBConnection.connectDB();
17. **if**(con == **null** ) **return**;

20. //插入一个新的用户
21. String sqlStr ="insert into database.admin values(?,?)";
22. **try** {
23. preSql = con.prepareStatement(sqlStr);
24. preSql.setString(1,ano);
25. preSql.setString(2,apassword);
26. **int** ok = preSql.executeUpdate();       //预处理语句调用方法执行
27. con.close();
28. }
29. **catch**(SQLException e) {
30. System.out.println("记录中ano值不能重复"+e);
31. }
32. }
33. }

在购买记录表中插入新的购买记录：

1. **package** method;
3. **import** java.sql.ResultSet;
4. **import** java.sql.SQLException;
5. **import** java.util.ArrayList;
7. **import** com.GetDBConnection;
9. **public** **class** insert\_buy\_records {
10. **public** **static** **void** main(String args[]) {
11. Insert\_records("201811050392","侯宇轩","0");
12. }
13. **public** **static** String Insert\_records(String uno,String gno,String bcount) {
14. java.sql.Connection con;
15. java.sql.Statement sql;
16. ResultSet rs;
17. java.sql.PreparedStatement preSql;  //预处理语句对象
18. con = GetDBConnection.connectDB();

21. //获得当前存在的订单数量
23. **int** number=get\_order\_num.get();
24. **int** cnt=Integer.parseInt(bcount);
25. **int** tprice=get\_total\_price.get(cnt,gno);
26. String total\_price=String.valueOf(tprice);
27. **int** num\_=number+1;
28. String num=String.valueOf(num\_);
29. String tmp="购买成功！\n 订单号:   "+ num + "    商品编号:   " + gno +"    商品数量:   " + bcount + "    订单总金额:   " +total\_price;
30. String sqlStr ="insert into database.records values(?,?,?,?,?)";
31. **try** {
32. preSql = con.prepareStatement(sqlStr);
33. preSql.setString(1,num);
34. preSql.setString(2,uno);
35. preSql.setString(3,gno);
36. preSql.setString(4,total\_price);
37. preSql.setString(5,bcount);
38. **int** ok = preSql.executeUpdate();       //预处理语句调用方法执行
39. con.close();
40. System.out.println(num+"   "+ uno + "   " + gno + "   " + total\_price + "   " + bcount);
41. }
42. **catch**(SQLException e) {
43. System.out.println("记录中rno值不能重复"+e);
44. }
45. **return** tmp;
46. }
47. }

添加新的商品：

1. **package** method;
3. **import** java.sql.SQLException;
5. **import** com.GetDBConnection;
7. **public** **class** insert\_goods {
8. **public** **static** **void** main(String args[]) {
9. Insert\_goods("201811050392","侯宇轩","信控学院","0");
10. }
11. **public** **static** **void** Insert\_goods(String no,String name,String price,String count) {
12. java.sql.Connection con;
13. java.sql.PreparedStatement preSql;  //预处理语句对象
14. con = GetDBConnection.connectDB();
15. **if**(con == **null** ) **return**;
16. String sqlStr ="insert into database.goods values(?,?,?,?)";
17. **try** {
18. preSql = con.prepareStatement(sqlStr);
19. preSql.setString(1,no);
20. preSql.setString(2,name);
21. preSql.setString(3,price);
22. preSql.setString(4,count);
23. **int** ok = preSql.executeUpdate();       //预处理语句调用方法执行
24. con.close();
25. }
26. **catch**(SQLException e) {
27. System.out.println("记录中gno值不能重复"+e);
28. }
29. }
30. }

插入新的用户：

1. **package** method;
3. **import** java.sql.ResultSet;
4. **import** java.sql.SQLException;
6. **import** com.GetDBConnection;
8. **public** **class** insert\_usr {
9. **public** **static** **void** main(String args[]) {
10. Insert\_usr("201811050392","侯宇轩");
11. }
12. **public** **static** **void** Insert\_usr(String uno,String upassword) {
13. java.sql.Connection con;
14. java.sql.PreparedStatement preSql;  //预处理语句对象
15. con = GetDBConnection.connectDB();
17. **if**(con == **null** ) **return**;

20. //插入一个新的用户
21. String sqlStr ="insert into database.usr values(?,?)";
22. **try** {
23. preSql = con.prepareStatement(sqlStr);
24. preSql.setString(1,uno);
25. preSql.setString(2,upassword);
26. **int** ok = preSql.executeUpdate();       //预处理语句调用方法执行
27. con.close();
28. }
29. **catch**(SQLException e) {
30. System.out.println("记录中usrid值不能重复"+e);
31. }
32. }
33. }

查询数据库中的所有商品：

1. **package** method;
3. **import** java.beans.Statement;
4. **import** java.math.BigInteger;
5. **import** java.sql.\*;
6. **import** java.util.ArrayList;
7. **import** java.util.Arrays;
8. **import** java.util.Comparator;
10. **import** com.GetDBConnection;
12. **public** **class** Query\_all {
13. **public** **static** **void** main(String args[]){
14. get\_queryall();
15. }
17. **public** **static** ArrayList<String> get\_queryall() {//static 可以直接调用函数，不用创建对象
18. ArrayList<String>list=**new** ArrayList<String>();
19. java.sql.Connection con;
20. java.sql.Statement sql;
21. ResultSet rs;
22. String title="商品号"+"\t"+"商品名" + "\t" + "商品价格" + "\t" + "商品数量";
23. list.add(title);
24. con = GetDBConnection.connectDB();
25. **try** {
26. sql=con.createStatement();
27. // 查询结果按照商品价格排序
28. rs = sql.executeQuery("select \* from database.goods order by gprice+0");
29. **while**(rs.next()) {
30. String gno=rs.getString(1);
31. String gname=rs.getString(2);
32. String gprice=rs.getString(3);
33. String gcount=rs.getString(4);
34. String record = gno +"\t" + gname + "\t" + gprice+ "\t" + gcount;
35. list.add(record);
36. }
37. con.close();
38. }
39. **catch**(SQLException e) {}
41. **return** list;
42. }
43. }

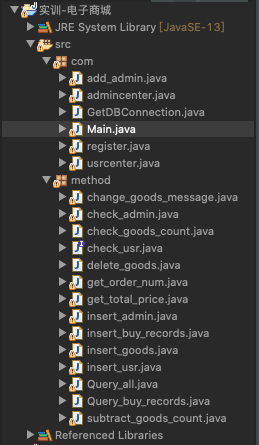
查询购买记录：

1. **package** method;
3. **import** java.sql.ResultSet;
4. **import** java.sql.SQLException;
5. **import** java.util.ArrayList;
7. **import** com.GetDBConnection;
9. **public** **class** Query\_buy\_records {
10. **public** **static** **void** main(String args[]){
11. get\_buy\_records("201811050392");
12. }
13. **public** **static** ArrayList<String> get\_buy\_records(String no){//static 可以直接调用函数，不用创建对象
14. ArrayList<String>list=**new** ArrayList<String>();
15. java.sql.Connection con;
16. java.sql.Statement sql;
17. ResultSet rs;
18. String rec="订单号" +"\t" + "用户号" + "\t" + "商品号" +"\t" + "订单总金额" +"\t"+ "商品总数量";
19. list.add(rec);
20. con = GetDBConnection.connectDB();
21. **try** {
22. sql=con.createStatement();
24. //查询结果按照订单编号排序
25. rs = sql.executeQuery("select \* from database.records where uno=\""+no+"\"" + "order by rno+0");
26. **while**(rs.next()) {
27. String rno=rs.getString(1);
28. String uno=rs.getString(2);
29. String gno=rs.getString(3);
30. String tprice=rs.getString(4);
31. String bcount=rs.getString(5);
32. String record = rno +"\t" + uno + "\t" + gno +"\t" + tprice +"\t"+ bcount;
33. list.add(record);
34. }
35. con.close();
36. }
37. **catch**(SQLException e) {}
38. **return** list;
39. }
40. }

在购买完成后数据库商品表中中减去相应的商品数量：

1. **package** method;
3. **import** java.sql.SQLException;
5. **import** com.GetDBConnection;
7. **public** **class** subtract\_goods\_count {
8. **public** **static** **void** main(String args[]) {
9. change("201811050392","侯宇轩");
10. }
12. **public** **static** **void** change(String gno,String new\_cnt){
13. java.sql.Connection con;
14. java.sql.PreparedStatement preSql;  //预处理语句对象
15. con = GetDBConnection.connectDB();
16. **if**(con == **null** ) **return**;
17. String sqlStr ="update database.goods set gcount=? where (gno=?)";
18. **try** {
19. preSql = con.prepareStatement(sqlStr);
20. preSql.setString(2,gno);
21. preSql.setString(1,new\_cnt);
22. **int** ok = preSql.executeUpdate();       //预处理语句调用方法执行
23. con.close();
24. }
25. **catch**(SQLException e) {
26. System.out.println("记录中gno值不能重复"+e);
27. }
28. }
29. }

**六、附录**

­­­­

**成绩评分表**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 序号 | 评价内容 | | 评价等级 | | | | |
| 1 | 设计思路阐述清晰，格式符合要求 | | 优 | 良 | 中 | 差 | 不及格 |
| 2 | 结构严谨，逻辑性强，语言表达准确 | | 优 | 良 | 中 | 差 | 不及格 |
| 3 | 文字通顺，技术用语准确，基本概念清楚 | | 优 | 良 | 中 | 差 | 不及格 |
| 4 | 基础理论知识扎实，回答问题有理论根据 | | 优 | 良 | 中 | 差 | 不及格 |
| 5 | 有较强的计算机应用能力 | | 优 | 良 | 中 | 差 | 不及格 |
| 6 | 功能齐全完善，能正确处理实验数据 | | 优 | 良 | 中 | 差 | 不及格 |
| 7 | 系统设计与实现方法有技巧性、新有创新意识 | | 优 | 良 | 中 | 差 | 不及格 |
| 8 | 有一定的理论或应用价值，设计思路新颖，对问题有较深刻的认识 | | 优 | 良 | 中 | 差 | 不及格 |
| 9 | 学习态度端正，主动参与性强 | | 优 | 良 | 中 | 差 | 不及格 |
| 10 | 与指导教师及时沟通，学风严谨务实，按期圆满完成规定的任务 | | 优 | 良 | 中 | 差 | 不及格 |
| 备注 |  | | | | | | |
| 总分 |  | 教师签字 |  | | | |  |