人事管理系统

1.登录代码

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.ResultSet;
import java.sql.SQLException;

class get_ID {
        private static String ID;
        public void id_set(String ID){
                this.ID = ID;
        }
        public static String id_get(){
                return ID;
        }
}//登录成功后，获取得到登录的 ID,以便进行员工个人信息界面的显示

public class login extends JFrame implements ActionListener{
        //一、界面方法
        private JLabel img = new JLabel(new ImageIcon("D:\\人事.png"));//图片
        private    JLabel IDAccount = new JLabel("请输入您的帐号");
        private    JTextField tfAccount = new JTextField();
        private    JLabel IDPassword = new JLabel("请输入您的密码");
        private    JPasswordField pfPassword = new JPasswordField();
        private JPanel p1 = new JPanel();
        private JPanel p2 = new JPanel();
        private JPanel p3 = new JPanel();
        private JPanel p4 = new JPanel();
        private JButton btLogin = new JButton("登录");
        private JButton btExit = new JButton("退出");
        private JRadioButton r1 = new JRadioButton("管理员");
        private JRadioButton r2 = new JRadioButton("员工");


        private ButtonGroup group = new ButtonGroup();//两种角色登录

        DbProcess dbProcess = new DbProcess();
//        界面初始化
        public login(){
```

```
		super("人事管理系统——登录");
//		this.setLayout(new FlowLayout());
		this.add(img,"North");//将图片放在最上边
		group.add(r1);
		group.add(r2);//将其只能选一个
		p1.setLayout(null);//清除默认布局，以便自定义
		this.add(p1);

		//将组件放入面板中
		IDAccount.setBounds(80,30,100,30);//
		p1.add(IDAccount);
		IDPassword.setBounds(80,80,100,30);
		p1.add(IDPassword);
		tfAccount.setBounds(185,30,240,30);
		p1.add(tfAccount);
		pfPassword.setBounds(185,80,240,30);
		p1.add(pfPassword);
		r1.setBounds(185,120,80,50);
		p1.add(r1);
		r2.setBounds(275,120,80,50);
		p1.add(r2);
		btLogin.setBounds(160,190,100,40);
		p1.add(btLogin);
		btExit.setBounds(290,190,100,40);
		p1.add(btExit);
		r2.setSelected(true);//设置员工处于被选中状态
//		System.out.println(r2.isSelected());

		this.setSize(550,600);
		//toCenter(this);
		this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
//		this.pack();
//		this.setResizable(true);//自己通过鼠标调整尺寸
		this.setVisible(true);//使界面可见
		this.setLocation(500,150);
//		this.setLocationRelativeTo(null);

		//添加动作事件监听
		btLogin.addActionListener(this);
		btExit.addActionListener(this);
	}
	//实现方法
	public void actionPerformed(ActionEvent e){
		if(r2.isSelected() == true)
		{
			if(e.getSource() == btLogin){
				//判断员工的 ID 和密码，实现登录，并且跳转到员工的可视界
面
				get_ID id_g = new get_ID();
				id_g.id_set(tfAccount.getText().trim());
				String ID = id_g.id_get();

				char[] password = pfPassword.getPassword();
				String Password = "";
				for(int i = 0 ; i < password.length ; i++)
				{
					Password = Password + password[i];
				}//获取输入的密码

				//判断 ID 和 Password 是否为空
				if(ID.equals(""))
				{
					JOptionPane.showMessageDialog(null,"ID 不能为空","警告
",JOptionPane.WARNING_MESSAGE);
					return;
				}
				else if(!ID.equals("")&&Password.equals(""))
				{
					JOptionPane.showMessageDialog(null,"请输入密码","警告
",JOptionPane.WARNING_MESSAGE);
					return;
				}

//				System.out.println(ID);
//				System.out.println(Password);

				//获取数据库中的员工 ID 和密码，与输入的匹配
				String sql = "select * from person ;";
				System.out.println("queryUser(). sql = " + sql);

				dbProcess.connect();
				ResultSet rs = dbProcess.executeQuery(sql);
				String sqlID = "";
				String sqlpassword="";
				try {
					while(rs.next()){
						sqlID = rs.getString("id");
```

```
                        sqlpassword = rs.getString("password");
                        if( (sqlID.equals(ID))&&(sqlpassword.equals(Password)))
                        {
                            break;
                        }//找到了则退出循环
                    }
//                      System.out.println(sqlID);
//                     System.out.println(sqlpassword);

                    if( (sqlID.equals(ID))&&(sqlpassword.equals(Password)))
                    {
                        if(Password.equals("Staff"+ID))
                        {
                            JOptionPane.showMessageDialog(null,"登录成功，
该密码为初始密码，为了更好地保护您的隐私，请您尽快更改!","登录
",JOptionPane.INFORMATION_MESSAGE);
                        }else{
                            JOptionPane.showMessageDialog(null," 登录成功
","登录",JOptionPane.INFORMATION_MESSAGE);

                        }
                        this.dispose();
                        new Staff().showAll();//打开员工个人信息界面，并且显
示出该员工的所有信息
                    }
                    else{
                        JOptionPane.showMessageDialog(null," 员工不存在或
密码错误","登录",JOptionPane.QUESTION_MESSAGE);
                    }
                } catch (SQLException throwables) {
                    throwables.printStackTrace();
                }

            }else if(e.getSource() == btExit){
                System.exit(0);//如果点击退出，则退出程序
            }
        }
        else if(r1.isSelected() == true){
            if(e.getSource() == btLogin){
                String ID = tfAccount.getText().trim();
                char[] password = pfPassword.getPassword();
                String Password = "";
                for(int i = 0 ; i < password.length ; i++)
                {
                    Password = Password + password[i];
                }//获取输入的密码

                 //判断 ID 和 Password 是否为空
                if(ID.equals(""))
                {
                    JOptionPane.showMessageDialog(null,"ID 不能为空","警告
",JOptionPane.WARNING_MESSAGE);
                    return;
                }
                else if(!ID.equals("")&&Password.equals(""))
                {
                    JOptionPane.showMessageDialog(null,"请输入密码","警告
",JOptionPane.WARNING_MESSAGE);
                    return;
                }

//                System.out.println(ID);
//                System.out.println(Password);

                //获取数据库中的管理员 ID 和密码，与输入的匹配
                String sql = "select * from person where
authority='administrator'";
                System.out.println("queryAdministator(). sql = " + sql);

                dbProcess.connect();
                ResultSet rs = dbProcess.executeQuery(sql);
                String sqlID = "";
                String sqlpassword="";
                try {
                    while(rs.next()){
                         sqlID = rs.getString("id");
                         sqlpassword = rs.getString("password");
                        if( (sqlID.equals(ID))&&(sqlpassword.equals(Password)))
                        {
                            break;
                        }//找到了则退出循环
                    }
//                       System.out.println(sqlID);
//                      System.out.println(sqlpassword);

                    if( (sqlID.equals(ID))&&(sqlpassword.equals(Password)))
                    {
                        JOptionPane.showMessageDialog(null,"登录成功","登
录",JOptionPane.INFORMATION_MESSAGE);
                        this.dispose();
                        new administrator_mian().queryAllProcess();//打开管理
员操作界面
                    }
                    else{
                        JOptionPane.showMessageDialog(null,"管理员不存在
或密码错误","登录",JOptionPane.QUESTION_MESSAGE);
                    }
                } catch (SQLException throwables) {
                    throwables.printStackTrace();
                }


        }else if(e.getSource() == btExit){
                System.exit(0);//如果点击退出，则退出程序
        }
    }
}
public static void main(String[] args){
    new login();
}
}
```

2.管理员登陆后主界面代码

```
import javax.swing.*;
import javax.swing.table.DefaultTableModel;

import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.ItemEvent;
import java.awt.event.ItemListener;
import java.awt.event.MouseAdapter;
import java.awt.event.MouseEvent;
import java.sql.*;
import java.util.ArrayList;
import java.util.Iterator;
import java.util.List;
import java.util.Vector;

public class administrator_mian extends JFrame implements ActionListener {
	// 定义组件
	JLabel jLPersonInfoTable = null;//表
	JLabel jLSelectQueryField = null;//选择查询字段
	JLabel jLEqual = null;//=
	JLabel jLID = null;
	JLabel jLPassword = null;
	JLabel jLAuthority = null;
	JLabel jLName = null;
	JLabel jLSex = null;
	JLabel jLBirthday = null;
	JLabel jLDepartment = null;
	JLabel jLJob = null;
	JLabel jLEdu_level = null;
	JLabel jLSpecialty = null;
	JLabel jLAdress = null;
	JLabel jLTel = null;
	JLabel jLEmail = null;
	JLabel jLState = null;
	JLabel jLRemark = null;

	JTextField jTFQueryField = null;//查询字段
	int rowCount = 0;//获取受影响的行数，如果该员工存在，则值为 1，不存在
则值为 0


	JTextField jTFID = null;
	JTextField jTFPassword = null;
	JTextField jTFAuthority = null;
	JTextField jTFName = null;
	JTextField jTFSex = null;
	JTextField jTFBirthday = null;
	JTextField jTFDepartment = null;
	JTextField jTFJob = null;
	JTextField jTFEdu_level = null;
	JTextField jTFSpecialty = null;
	JTextField jTFAdress = null;
	JTextField jTFTel = null;
	JTextField jTFEmail = null;
	JTextField jTFState = null;
	JTextField jTFRemark = null;

	JButton jBQuery = null;//查询
	JButton jBQueryAll = null;//查询所有记录
	JButton jBInsert = null;//插入
JButton jBUpdate = null;//更新
JButton jBDeleteCurrentRecord = null;//删除当前记录
JButton jBDeleteAllRecords = null;//删除所有记录
JButton back = null;//返回
JButton personnel_change = null;


//JComboBox jCBSelectQueryField = null;
JComboBox<String> jCBSelectQueryField = null;//查询字段
JPanel jP1,jP2,jP3,jP4,jP5,jP6,jP7,jP8,jP9 = null;
JPanel jPTop, jPBottom = null;
DefaultTableModel personTableModel = null;
JTable personJTable = null;
JScrollPane personJScrollPane = null;
Vector personVector = null;
Vector titleVector = null;

private static DbProcess dbProcess;
String SelectQueryFieldStr = "员工号";

// 构造函数
public administrator_mian() {
	// 创建组件
	jLPersonInfoTable = new JLabel("员工信息表");
	jLSelectQueryField = new JLabel("选择查询字段");
	jLEqual = new JLabel(" = ");
	jLID = new JLabel("员工号");
	jLPassword = new JLabel("密码");
	jLAuthority = new JLabel("权限");
	jLName = new JLabel("姓名");
	jLSex = new JLabel("性别");
	jLBirthday = new JLabel("生日");
	jLDepartment = new JLabel("部门");
	jLJob = new JLabel("职务");
	jLEdu_level = new JLabel("受教育程度");
	jLSpecialty = new JLabel("专业技能");
	jLAdress = new JLabel("家庭住址");
	jLTel = new JLabel("联系电话");
	jLEmail = new JLabel("电子邮箱");
	jLState = new JLabel("当前状态");
	jLRemark = new JLabel("备注");

	jTFQueryField = new JTextField(10);//查询字段
	jTFID = new JTextField(10);
```

```
jTFPassword = new JTextField(10);
jTFAuthority = new JTextField(10);
jTFName     = new JTextField(10);
jTFSex = new JTextField(10);
jTFBirthday = new JTextField(10);
jTFDepartment = new JTextField(10);
jTFJob = new JTextField(10);
jTFEdu_level = new JTextField(10);
jTFSpecialty = new JTextField(10);
jTFAdress = new JTextField(10);
jTFTel = new JTextField(10);
jTFEmail = new JTextField(10);
jTFState = new JTextField(10);
jTFRemark = new JTextField(10);


jBQuery = new JButton("查询");
jBQueryAll = new JButton("查询所有记录");
jBInsert = new JButton("添加");
jBUpdate = new JButton("更新");
jBDeleteCurrentRecord = new JButton("删除当前记录");
jBDeleteAllRecords = new JButton("删除所有记录");
back = new JButton("返回");
personnel_change = new JButton("人事变更");

//  设置监听
jBQuery.addActionListener(this);
jBQueryAll.addActionListener(this);
jBInsert.addActionListener(this);
jBUpdate.addActionListener(this);
jBDeleteCurrentRecord.addActionListener(this);
jBDeleteAllRecords.addActionListener(this);
back.addActionListener(this);
personnel_change.addActionListener(this);

jCBSelectQueryField = new JComboBox<String>();//查询字段
jCBSelectQueryField.addItem("员工号");
jCBSelectQueryField.addItem("密码");
jCBSelectQueryField.addItem("权限");
jCBSelectQueryField.addItem("姓名");
jCBSelectQueryField.addItem("性别");
jCBSelectQueryField.addItem("生日");
jCBSelectQueryField.addItem("部门");
jCBSelectQueryField.addItem("职务");
```

```
		jCBSelectQueryField.addItem("受教育程度");
		jCBSelectQueryField.addItem("专业技能");
		jCBSelectQueryField.addItem("家庭住址");
		jCBSelectQueryField.addItem("联系电话");
		jCBSelectQueryField.addItem("电子邮箱");
		jCBSelectQueryField.addItem("当前状态");
		jCBSelectQueryField.addItem("备注");
		jCBSelectQueryField.addItemListener(new ItemListener() {//下拉框事件监
听
			public void itemStateChanged(ItemEvent event) {
				switch (event.getStateChange()) {
					case ItemEvent.SELECTED:
						SelectQueryFieldStr = (String) event.getItem();
						System.out.println("选中：" + SelectQueryFieldStr);
						break;
					case ItemEvent.DESELECTED:
						System.out.println("取消选中：" + event.getItem());
						break;
				}
			}
		});

		personVector = new Vector();
		titleVector = new Vector();

		// 定义表头
		titleVector.add("员工号");
		titleVector.add("密码");
		titleVector.add("权限");
		titleVector.add("姓名");
		titleVector.add("性别");
		titleVector.add("生日");
		titleVector.add("部门");
		titleVector.add("职务");
		titleVector.add("受教育程度");
		titleVector.add("专业技能");
		titleVector.add("家庭住址");
		titleVector.add("联系电话");
		titleVector.add("电子邮箱");
		titleVector.add("当前状态");
		titleVector.add("备注");
		//studentTableModel = new DefaultTableModel(tableTitle, 15);
		personJTable = new JTable(personVector, titleVector);
		personJTable.setPreferredScrollableViewportSize(new
Dimension(1200,240));
		personJTable.setAutoResizeMode(2);
		personJTable.setRowHeight(35);//设置行高
		personJTable.getColumnModel().getColumn(12).setPreferredWidth(120);//
设置列宽
		personJScrollPane = new JScrollPane(personJTable);
		//分别设置水平和垂直滚动条自动出现
		personJScrollPane.setHorizontalScrollBarPolicy(
				JScrollPane.HORIZONTAL_SCROLLBAR_AS_NEEDED);
		personJScrollPane.setVerticalScrollBarPolicy(
				JScrollPane.VERTICAL_SCROLLBAR_AS_NEEDED);

		//为表格添加监听器

		jP1 = new JPanel();
		jP2 = new JPanel();
		jP3 = new JPanel();
		jP4 = new JPanel();
		jP5 = new JPanel();
		jP6 = new JPanel();
		jP7 = new JPanel();
		jP8 = new JPanel();
		jP9 = new JPanel();
		jPTop = new JPanel();
		jPBottom = new JPanel();

		jP1.add(jLPersonInfoTable,BorderLayout.SOUTH);
		jP2.add(personJScrollPane);

		jP3.add(jLSelectQueryField);
		jP3.add(jCBSelectQueryField);
		jP3.add(jLEqual);
		jP3.add(jTFQueryField);
		jP3.add(jBQuery);
		jP3.add(jBQueryAll);
		jP3.setLayout(new FlowLayout(FlowLayout.CENTER));
//		jP3.setPreferredSize(new Dimension(20,20));

		jP4.add(jLID);
		jP4.add(jTFID);
		jP4.setPreferredSize(new Dimension(30,20));


		jP6.add(jBInsert);
```

```
        jP6.add(jBUpdate);
        jP6.add(jBDeleteCurrentRecord);
        jP6.add(jBDeleteAllRecords);
        jP6.add(personnel_change);
        jP6.add(back);
        jP6.setLayout(new FlowLayout(FlowLayout.CENTER));
        jP6.setPreferredSize(new Dimension(20,20));

        jPTop.add(jP1);
        jPTop.add(jP2);

        jPBottom.setLayout(new GridLayout(5, 2));
        jPBottom.add(jP3);
        jPBottom.add(jP4);
        jPBottom.add(jP6);

        this.add("North", jPTop);
        this.add("South", jPBottom);

        this.setLayout(new GridLayout(2, 1));
        this.setTitle("人事管理系统——管理员操作界面");
        this.setSize(1300, 700);
        this.setLocation(150, 50);
        this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        this.setVisible(true);
        this.setResizable(false);
        this.setLayout(null);

        dbProcess = new DbProcess();
    }

    @Override
    public void actionPerformed(ActionEvent e) {
        if(e.getActionCommand().equals("查询")
                    && !jTFQueryField.getText().isEmpty()){
            System.out.println("actionPerformed(). 查询");
            String sQueryField = jTFQueryField.getText().trim();
            queryProcess(sQueryField);
            jTFQueryField.setText("");
        }else if(e.getActionCommand().equals("查询所有记录")) {
            System.out.println("actionPerformed(). 查询所有记录");

            queryAllProcess();
        }else if((e.getActionCommand().equals("添加"))){
                try {
                        new new_staff_addition().Auto_distribution();
                } catch (SQLException throwables) {
                        throwables.printStackTrace();
                }
        }
        else if(e.getActionCommand().equals("删除当前记录")){
                if(jTFID.getText().equals(""))
                {
                        JOptionPane.showMessageDialog(null,"请在上方设置需要删除
的员工号","处理结果",JOptionPane.ERROR_MESSAGE);
                }
                else{
                        //得到要被删掉的员工号
                        try{
                                String previous_id = "";
                                String   sql1   =   "select   *   from   person   where   id   =
'"+jTFID.getText()+"';";
                                System.out.println("query_sql1 = " + sql1);
                                dbProcess.connect();
                                ResultSet rs1 = dbProcess.executeQuery(sql1);
                                rowCount = 0 ;//再次清零，下次在更新时，再获取值 1 或
0

                                while (rs1.next()) {
                                        previous_id = rs1.getString("id");
                                        rowCount++;
                                }
                              //再获取要被删除的员工 id 后，执行删除操作
                                if(rowCount != 0){
                                        System.out.println("actionPerformed(). 删除当前记录
");

                                        deleteCurrentRecordProcess();
                                }else{
                                        JOptionPane.showMessageDialog(null,
                                                    " 该 员 工 不 存 在 "," 处 理 结 果
",JOptionPane.ERROR_MESSAGE);
                                }


                                //更新 personnel 表

                                //判断在员工信息中是否该员工
                                if(rowCount != 0)
                                {
```

```
                        try {
                            //  建立查询条件
                            String sql = "select * from personnel;";
                            System.out.println("query_sql = " + sql);
                            dbProcess.connect();
                            ResultSet rs = dbProcess.executeQuery(sql);


                            int personnel_last_id = 0;
                            while (rs.next()) {
                                personnel_last_id = rs.getInt("id");

                            }

                            int insert_id = personnel_last_id + 1;//加 1

                            //2.将辞退员工的人事变更加入到 personnel 表
中


                            String    sql2    =    "insert    into    personnel
values("+insert_id+",'"+previous_id+"',2,'辞退');";
                            dbProcess.executeUpdate(sql2);

                            dbProcess.disconnect();
                        } catch (SQLException sqle) {
                            System.out.println("sqle = " + sqle);
                            JOptionPane.showMessageDialog(null,
                                    " 数 据 操 作 错 误 ",  " 错 误 ",
JOptionPane.ERROR_MESSAGE);
                        }
                    }
                }catch (SQLException throwables) {
                    throwables.printStackTrace();
                }

            }

        }else if(e.getActionCommand().equals("删除所有记录")){



            //将所有员工都辞退
            try{
                String sql = "select * from person";
                System.out.println("query_sql = " + sql);
                dbProcess.connect();
                ResultSet rs = dbProcess.executeQuery(sql);
                //循环添加辞退记录
                while (rs.next()){
                    String id = rs.getString("id");

                    try {
                        //  建立查询条件
                        String sql1 = "select * from personnel;";
                        System.out.println("query_sql = " + sql1);
                        dbProcess.connect();
                        ResultSet rs1 = dbProcess.executeQuery(sql1);


                        int personnel_last_id = 0;
                        while (rs.next()) {
                            personnel_last_id = rs.getInt("id");

                        }

                        int insert_id = personnel_last_id + 1;//加 1

                        //2.将辞退员工的人事变更加入到 personnel 表中


                        String sql2 = "insert into personnel
values("+insert_id+",'"+id+"',2,'辞退');";
                        dbProcess.executeUpdate(sql2);

                        dbProcess.disconnect();
                    } catch (SQLException sqle) {
                        System.out.println("sqle = " + sqle);
                        JOptionPane.showMessageDialog(null,
                                "数据操作错误", "错误",
JOptionPane.ERROR_MESSAGE);
                    }
                }
            }catch(SQLException throwables)
            {
                throwables.printStackTrace();
            }
```

```
                //先添加到人事表中，再删除
                System.out.println("actionPerformed(). 删除所有记录");
                deleteAllRecordsProcess();
                JOptionPane.showMessageDialog(null,"成功删除所有记录","处理结
果",JOptionPane.INFORMATION_MESSAGE);
            }
            else if(e.getActionCommand().equals("更新")){
                new update_staff();
            }
            else if(e.getActionCommand().equals("返回"))
            {
                this.dispose();
                new login();
            }
            else if(e.getActionCommand().equals("人事变更")){
                this.dispose();
                new personnel().queryAll_personnel();
                //人事变更操作界面

            }
        }

//      public static void main(String[] args) {
//          // TODO Auto-generated method stub
//          DatabaseCourseDesign getcon = new   DatabaseCourseDesign();
//      }

        public void queryProcess(String sQueryField)
        {
            try{
                // 建立查询条件
                String sql = "select * from person where ";
                String                          queryFieldStr                          =
jCBSelectQueryFieldTransfer(SelectQueryFieldStr);

                if(queryFieldStr.equals("id")){//int sAge.
                    sql = sql + queryFieldStr;
                    sql = sql + " = " + sQueryField;
                }else{
                    sql = sql + queryFieldStr;
                    sql = sql + " = ";
                    sql = sql + "'" + sQueryField + "';";
                }
                System.out.println("queryProcess(). sql = " + sql);

                dbProcess.connect();
                ResultSet rs = dbProcess.executeQuery(sql);

                // 将查询获得的记录数据，转换成适合生成 JTable 的数据形式
                personVector.clear();
                while(rs.next()){
                    Vector v = new Vector();
                    v.add(rs.getString("id"));
                    v.add(rs.getString("password"));
                    v.add(rs.getString("authority"));
                    v.add(rs.getString("name"));
                    v.add(rs.getString("sex"));
                    v.add(rs.getString("birthday"));
                    v.add(rs.getString("department"));
                    v.add(Integer.valueOf(rs.getInt("job")));
                    v.add(Integer.valueOf(rs.getInt("edu_level")));
                    v.add(rs.getString("specialty"));
                    v.add(rs.getString("address"));
                    v.add(rs.getString("tel"));
                    v.add(rs.getString("email"));
                    v.add(rs.getString("state"));
                    v.add(rs.getString("remark"));
                    personVector.add(v);
                }

                personJTable.updateUI();

                dbProcess.disconnect();
            }catch(SQLException sqle){
                System.out.println("sqle = " + sqle);
                JOptionPane.showMessageDialog(null,
                        "数据操作错误","错误",JOptionPane.ERROR_MESSAGE);
            }catch(Exception e){
                System.out.println("e = " + e);
                JOptionPane.showMessageDialog(null,
                        "数据操作错误","错误",JOptionPane.ERROR_MESSAGE);
            }
        }

        public void queryAllProcess()
        {
            try{
```

```
			// 建立查询条件
			String sql = "select * from person;";
			System.out.println("queryAllProcess(). sql = " + sql);

			dbProcess.connect();
			ResultSet rs = dbProcess.executeQuery(sql);

			// 将查询获得的记录数据，转换成适合生成 JTable 的数据形式
			personVector.clear();
			while(rs.next()){
				Vector v = new Vector();
				v.add(rs.getString("id"));
				v.add(rs.getString("password"));
				v.add(rs.getString("authority"));
				v.add(rs.getString("name"));
				v.add(rs.getString("sex"));
				v.add(rs.getString("birthday"));
				v.add(rs.getString("department"));
				v.add(Integer.valueOf(rs.getInt("job")));
				v.add(Integer.valueOf(rs.getInt("edu_level")));
				v.add(rs.getString("specialty"));
				v.add(rs.getString("address"));
				v.add(rs.getString("tel"));
				v.add(rs.getString("email"));
				v.add(rs.getString("state"));
				v.add(rs.getString("remark"));
				personVector.add(v);
			}

			personJTable.updateUI();

			dbProcess.disconnect();
		}catch(SQLException sqle){
			System.out.println("sqle = " + sqle);
			JOptionPane.showMessageDialog(null,
					"数据操作错误","错误",JOptionPane.ERROR_MESSAGE);
		}
	}

	public void deleteCurrentRecordProcess()
	{
		String id = jTFID.getText().trim();

		// 建立删除条件
		String sql = "delete from person where id = '" + id + "';";
		System.out.println("deleteCurrentRecordProcess(). sql = " + sql);
		try{
			if (dbProcess.executeUpdate(sql) < 1) {
				System.out.println("deleteCurrentRecordProcess().           delete
database failed.");
				JOptionPane.showMessageDialog(null,
						"  该  员  工  不  存  在  ","  处  理  结  果
",JOptionPane.ERROR_MESSAGE);
			}else{
				JOptionPane.showMessageDialog(null,"删除成功","处理结果
",JOptionPane.INFORMATION_MESSAGE);
			}
		}catch(Exception e){
			System.out.println("e = " + e);
			JOptionPane.showMessageDialog(null,
					"数据操作错误","错误",JOptionPane.ERROR_MESSAGE);
		}
		queryAllProcess();
	}

	public void deleteAllRecordsProcess()
	{
		// 建立删除条件
		String sql = "delete from person;";
		System.out.println("deleteAllRecordsProcess(). sql = " + sql);
		try{
			if (dbProcess.executeUpdate(sql) < 1) {
				System.out.println("deleteAllRecordsProcess().   delete   database
failed.");
			}
		}catch(Exception e){
			System.out.println("e = " + e);
			JOptionPane.showMessageDialog(null,
					"数据操作错误","错误",JOptionPane.ERROR_MESSAGE);
		}
		queryAllProcess();
	}

	public String jCBSelectQueryFieldTransfer(String InputStr)
	{
		String outputStr = "";
		System.out.println("jCBSelectQueryFieldTransfer(). InputStr = " + InputStr);
		if(InputStr.equals("员工号")){
			outputStr = "id";
		}else if(InputStr.equals("密码")){
			outputStr = "password";
		}else if(InputStr.equals("权限")){
			outputStr = "authority";
		}else if(InputStr.equals("姓名")){
			outputStr = "name";
		}else if(InputStr.equals("性别")){
			outputStr = "sex";
		}else if(InputStr.equals("生日")){
			outputStr = "birthday";
		}else if(InputStr.equals("部门")){
			outputStr = "department";
		}else if(InputStr.equals("职务")){
			outputStr = "job";
		}else if(InputStr.equals("受教育程度")){
			outputStr = "edu_level";
		}else if(InputStr.equals("专业技能")){
			outputStr = "specialty";
		}else if(InputStr.equals("家庭住址")){
			outputStr = "address";
		}else if(InputStr.equals("联系电话")){
			outputStr = "tel";
		}else if(InputStr.equals("电子邮箱")){
			outputStr = "email";
		}else if(InputStr.equals("当前状态")){
			outputStr = "state";
		}else if(InputStr.equals("备注")){
			outputStr = "remark";
		}
		System.out.println("jCBSelectQueryFieldTransfer().	outputStr	=	"	+
outputStr);
		return outputStr;
	}
}
```

3.数据库连接代码

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


public class DbProcess{
	Connection connection = null;
	ResultSet rs = null;

	//mysql 数据库 url
	String userMySql="root";
	String passwordMySql="aa617100";
	String urlMySql = "jdbc:mysql://localhost:3306/personnel_data?user="
			+userMySql+"&password="+passwordMySql                    +
"&useUnicode=true&characterEncoding=gbk";

	//sqlserver 数据库 url
	//String                    urlSqlServer                    =
"jdbc:sqlserver://localhost:1433;integratedSecurity=true;DatabaseName=InfoDb";

	public DbProcess() {
		try {
			//mysql 数据库设置驱动程序类型
			Class.forName("com.mysql.cj.jdbc.Driver");
			System.out.println("mysql 数据库驱动加载成功");

			//sqlserver 数据库设置驱动程序类型
			//Class.forName("com.microsoft.sqlserver.jdbc.SQLServerDriver");
			//System.out.println("sqlserver 数据库驱动加载成功");

		}
		catch(java.lang.ClassNotFoundException e) {
			e.printStackTrace();
		}
	}


	public void connect(){
		try{
			//mysql 数据库
			connection = DriverManager.getConnection(urlMySql);
		//sqlserver 数据库
		//connection = DriverManager.getConnection(urlSqlServer);


		if(connection!=null){
			System.out.println("数据库连接成功");
		}
	}
	catch(Exception e){
		e.printStackTrace();
	}
}

public void disconnect(){
	try{
		if(connection != null){
			connection.close();
			connection = null;
		}
	}
	catch(Exception e){
		e.printStackTrace();
	}
}


public ResultSet executeQuery(String sql) {
	try {
		System.out.println("executeQuery(). sql = " + sql);

		PreparedStatement pstm = connection.prepareStatement(sql);
		// 执行查询
		rs = pstm.executeQuery();
	}
	catch(SQLException ex) {
		ex.printStackTrace();
	}
	return rs;
}

//插入
//executeUpdate 的返回值是一个整数，指示受影响的行数（即更新计数）。
//executeUpdate 用于执行 INSERT、UPDATE 或 DELETE 语句
//以及 SQL DDL（数据定义语言）语句，例如 CREATE TABLE 和 DROP TABLE。
```

```
	//执行增、删、改语句的方法
	public int executeUpdate(String sql) {
		int count = 0;
		connect();
		try {
			Statement stmt = connection.createStatement();
			count = stmt.executeUpdate(sql);
		}
		catch(SQLException ex) {
			System.err.println(ex.getMessage());
		}
		disconnect();
		return count;
	}
}
```

4.修改信息界面代码

```
import javax.swing.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.ResultSet;
import java.sql.SQLException;

public class update_staff extends JFrame implements ActionListener {
	JLabel jLID = new JLabel("员工号");
	JLabel jLPassword = new JLabel("密码");
	JLabel jLAuthority =new JLabel("权限");
	JLabel jLName = new JLabel("姓名");
	JLabel jLSex = new JLabel("性别");
	JLabel jLBirthday = new JLabel("生日");
	JLabel jLDepartment = new JLabel("部门");
	JLabel jLJob = new JLabel("职务");
	JLabel jLEdu_level = new JLabel("受教育程度");
	JLabel jLSpecialty = new JLabel("专业技能");
	JLabel jLAdress = new JLabel("家庭住址");
	JLabel jLTel = new JLabel("联系电话");
	JLabel jLEmail = new JLabel("电子邮箱");
	JLabel jLState = new JLabel("当前状态");
	JLabel jLRemark = new JLabel("备注");

	JTextField text_ID = new JTextField();
```

```
    JTextField text_Password = new JTextField();
    JTextField text_Authority =new JTextField();
    JTextField text_Name = new JTextField();
    JTextField text_Sex = new JTextField();
    JTextField text_Birthday = new JTextField();
    JTextField text_Department = new JTextField();
    JTextField text_Job = new JTextField();
    JTextField text_Edu_level = new JTextField();
    JTextField text_Specialty = new JTextField();
    JTextField text_Adress = new JTextField();
    JTextField text_Tel = new JTextField();
    JTextField text_Email = new JTextField();
    JTextField text_State = new JTextField();
    JTextField text_Remark = new JTextField();

    JButton update = new JButton("更新该员工信息");
    DbProcess dbProcess = new DbProcess();

     int a     = 2;//受影响的行数
     String department ="";//获取更新的员工的原来的部门
     int job = 100;//获取以前的职位
     int rowCount = 0;//获取受影响的行数，如果该员工存在，则值为 1，不存在
则值为 0

    public update_staff(){
        super("更新");
        setBounds(500,80,500,700);
        this.setVisible(true);
        this.setLayout(null);
        jLID.setBounds(20,30,60,24);
        this.add(jLID);
        text_ID.setBounds(90,30,240,24);
        this.add(text_ID);
        jLPassword.setBounds(20,60,60,24);
        this.add(jLPassword);
        text_Password.setBounds(90,60,240,24);
        this.add(text_Password);
        jLAuthority.setBounds(20,90,60,24);
        this.add(jLAuthority);
        text_Authority.setBounds(90,90,240,24);
        this.add(text_Authority);
        jLName.setBounds(20,120,60,24);
        this.add(jLName);
        text_Name.setBounds(90,120,240,24);
```

```
this.add(text_Name);
jLSex.setBounds(20,150,60,24);
this.add(jLSex);
text_Sex.setBounds(90,150,240,24);
this.add(text_Sex);

jLBirthday.setBounds(20,180,60,24);
this.add(jLBirthday);
text_Birthday.setBounds(90,180,240,24);
this.add(text_Birthday);

jLDepartment.setBounds(20,210,60,24);
this.add(jLDepartment);
text_Department.setBounds(90,210,240,24);
this.add(text_Department);

jLJob.setBounds(20,240,60,24);
this.add(jLJob);
text_Job.setBounds(90,240,240,24);
this.add(text_Job);

jLEdu_level.setBounds(20,270,60,24);
this.add(jLEdu_level);
text_Edu_level.setBounds(90,270,240,24);
this.add(text_Edu_level);

jLSpecialty.setBounds(20,300,60,24);
this.add(jLSpecialty);
text_Specialty.setBounds(90,300,240,24);
this.add(text_Specialty);

jLAdress.setBounds(20,330,60,24);
this.add(jLAdress);
text_Adress.setBounds(90,330,240,24);
this.add(text_Adress);

jLTel.setBounds(20,360,60,24);
this.add(jLTel);
text_Tel.setBounds(90,360,240,24);
this.add(text_Tel);

jLEmail.setBounds(20,390,60,24);
this.add(jLEmail);
text_Email.setBounds(90,390,240,24);
```

```
		this.add(text_Email);

		jLState.setBounds(20,420,60,24);
		this.add(jLState);
		text_State.setBounds(90,420,240,24);
		this.add(text_State);

		jLRemark.setBounds(20,450,60,24);
		this.add(jLRemark);
		text_Remark.setBounds(90,450,240,24);
		this.add(text_Remark);

		update.setBounds(120,550,150,40);
		this.add(update);

		update.addActionListener(this);
	}
	public void actionPerformed(ActionEvent e){
		if(e.getSource() == update)
		{
			if((e.getActionCommand().equals("更新该员工信息")
				&& !text_ID.getText().isEmpty())
				&&( !text_Password.getText().isEmpty()
				|| !text_Authority.getText().isEmpty()
				|| !text_Name.getText().isEmpty()
				|| !text_Sex.getText().isEmpty()
				|| !text_Birthday.getText().isEmpty()
				|| !text_Department.getText().isEmpty()
				|| !text_Job.getText().isEmpty()
				|| !text_Edu_level.getText().isEmpty()
				|| !text_Specialty.getText().isEmpty()
				|| !text_Adress.getText().isEmpty()
				|| !text_Tel.getText().isEmpty()
				|| !text_Email.getText().isEmpty()
				|| !text_State.getText().isEmpty()
				|| !text_Remark.getText().isEmpty())){
			System.out.println("actionPerformed(). 更新员工信息");
			try {

				//得到该员工先前的部门和职位
				String sql1 = "select * from person where id =
'"+text_ID.getText()+"';";
				System.out.println("query_sql1 = " + sql1);
				dbProcess.connect();
```

```
                    ResultSet rs1 = dbProcess.executeQuery(sql1);
                    rowCount = 0 ;//再次清零，下次在更新时，再获取值 1 或 0
                    while (rs1.next()) {
                        department = rs1.getString("department");
                        job = Integer.valueOf(rs1.getString("job"));
                        rowCount++;
                    }
                    if( rowCount != 0){
                        updateProcess();
                        JOptionPane.showMessageDialog(null,"更新成功","处理结
果",JOptionPane.INFORMATION_MESSAGE);
                    }else{
                        JOptionPane.showMessageDialog(null,
                                "该员工不存在！","处理结果
",JOptionPane.ERROR_MESSAGE);
                    }

                    //更新 personnel 表

                    //1.如果更新时有职务调整，则插入其中
                    if(!text_Job.getText().isEmpty())
                    {
                        //判断在员工信息中是否该员工
                        if(rowCount != 0)
                        {
                            try {
                                // 建立查询条件
                                String sql = "select * from personnel;";
                                System.out.println("query_sql = " + sql);
                                dbProcess.connect();
                                ResultSet rs = dbProcess.executeQuery(sql);


                                int personnel_last_id = 0;
                                while (rs.next()) {
                                    personnel_last_id = rs.getInt("id");

                                }

                                int insert_id = personnel_last_id + 1;//加 1

                                //2.将更新的员工的人事变更加入到 personnel
表中
```

```
                                        ResultSet rs2 = dbProcess.executeQuery(sql1);
                                        String now_department = "";
                                        int now_job = 100;
                                        while (rs2.next()) {
                                                now_department                                    =
rs2.getString("department");
                                              now_job = Integer.valueOf(rs2.getInt("job"));
                                        }
                                        String    sql2    =    "insert    into    personnel
values("+insert_id+",'"+text_ID.getText().trim()+"',1,'"+department+"——>"+now_de
partment+"，"+job+"——>"+now_job+"');";
                                        dbProcess.executeUpdate(sql2);

                                        dbProcess.disconnect();
                                  } catch (SQLException sqle) {
                                        System.out.println("sqle = " + sqle);
                                        JOptionPane.showMessageDialog(null,
                                                    " 数 据 操 作 错 误 ", " 错 误 ",
JOptionPane.ERROR_MESSAGE);
                                  }
                            }
                      }
                      text_ID.setText("");
                      text_Job.setText("");

                } catch (SQLException throwables) {
                      throwables.printStackTrace();
                }
          }else if(e.getActionCommand().equals("更新该员工信息")
                      && text_ID.getText().isEmpty()){
                JOptionPane.showMessageDialog(null,"ID 不 能 为 空 "," 处 理 结 果
",JOptionPane.ERROR_MESSAGE);///必须有 ID
          }else if(e.getActionCommand().equals("更新该员工信息")
                      && (!text_ID.getText().isEmpty())
                              && text_Password.getText().isEmpty()
                              && text_Authority.getText().isEmpty()
                              && text_Name.getText().isEmpty()
                              && text_Sex.getText().isEmpty()
                              && text_Birthday.getText().isEmpty()
                              && text_Department.getText().isEmpty()
                              && text_Job.getText().isEmpty()
                              && text_Edu_level.getText().isEmpty()
                              && text_Specialty.getText().isEmpty()
```

```
						&& text_Adress.getText().isEmpty()
						&& text_Tel.getText().isEmpty()
						&& text_Email.getText().isEmpty()
						&& text_State.getText().isEmpty()
						&& text_Remark.getText().isEmpty()){
					JOptionPane.showMessageDialog(null,"请设置需要更新的数据！","处理结果",JOptionPane.WARNING_MESSAGE);
				}//如果只输入 ID
		}
	}
	public void updateProcess() throws SQLException {
		String id = text_ID.getText().trim();
		String password = text_Password.getText().trim();
		String authority = text_Authority.getText().trim();
		String name = text_Name.getText().trim();
		String sex = text_Sex.getText().trim();
		String birthday = text_Birthday.getText().trim();
		String department = text_Department.getText().trim();
		String job = text_Job.getText().trim();
		String edu_level = text_Edu_level.getText().trim();
		String specialty = text_Specialty.getText().trim();
		String address = text_Adress.getText().trim();
		String tel = text_Tel.getText().trim();
		String email = text_Email.getText().trim();
		String state = text_State.getText().trim();
		String remark = text_Remark.getText().trim();


		// 建立更新条件
		String sqlx = "update person set ";
		String sqly = " WHERE id = '" + id + "';";
		String sql1 = null;
		String sql2 = null;
		String sql3 = null;
		String sql4 = null;
		String sql5 = null;
		String sql6 = null;
		String sql7 = null;
		String sql8 = null;
		String sql9 = null;
		String sql10 = null;
		String sql11 = null;
		String sql12 = null;
		String sql13 = null;
```

```
String sql14 = null;

try {


        if (!password.equals("")) {
                sql1 = sqlx + "password='" + password + "'" + sqly;
                System.out.println("updateProcess(). sql1 = " + sql1);
                a = dbProcess.executeUpdate(sql1);
                text_Password.setText("");
        }
        if (!authority.equals("")) {
                sql2 = sqlx + "authority='" + authority + "'" + sqly;
                System.out.println("updateProcess(). sql2 = " + sql2);
                a = dbProcess.executeUpdate(sql2);
                text_Authority.setText("");
        }
        if (!name.equals("")) {
                sql3 = sqlx + "name='" + name + "'" + sqly;
                System.out.println("updateProcess(). sql3 = " + sql3);
                a = dbProcess.executeUpdate(sql3);
                text_Name.setText("");
        }
        if (!sex.equals("")) {
                sql4 = sqlx + "sex='" + sex + "'" + sqly;
                System.out.println("updateProcess(). sql4 = " + sql4);
                a = dbProcess.executeUpdate(sql4);
                text_Sex.setText("");
        }
        if (!birthday.equals("")) {
                sql5 = sqlx + "birthday='" + birthday + "'" + sqly;
                System.out.println("updateProcess(). sql5 = " + sql5);
                a = dbProcess.executeUpdate(sql5);
                text_Birthday.setText("");
        }
        if (!department.equals("")) {
                sql6 = sqlx + "department='" + department + "'" + sqly;
                System.out.println("updateProcess(). sql6 = " + sql6);
                a = dbProcess.executeUpdate(sql6);
                text_Department.setText("");
        }
        if (!job.equals("")) {
                sql7 = sqlx + "job=" + job + sqly;
                System.out.println("updateProcess(). sql7 = " + sql7);
```

```
		a = dbProcess.executeUpdate(sql7);
	}
	if (!edu_level.equals("")) {
		sql8 = sqlx + "edu_level=" + edu_level + sqly;
		System.out.println("updateProcess(). sql8 = " + sql8);
		a = dbProcess.executeUpdate(sql8);
		text_Edu_level.setText("");
	}
	if (!specialty.equals("")) {
		sql9 = sqlx + "specialty='" + specialty + "'" + sqly;
		System.out.println("updateProcess(). sql9 = " + sql9);
		a = dbProcess.executeUpdate(sql9);
		text_Specialty.setText("");
	}
	if (!address.equals("")) {
		sql10 = sqlx + "address='" + address + "'" + sqly;
		System.out.println("updateProcess(). sql10 = " + sql10);
		a = dbProcess.executeUpdate(sql10);
		text_Adress.setText("");
	}
	if (!tel.equals("")) {
		sql11 = sqlx + "tel='" + tel + "'" + sqly;
		System.out.println("updateProcess(). sql11 = " + sql11);
		a = dbProcess.executeUpdate(sql11);
		text_Tel.setText("");
	}
	if (!email.equals("")) {
		sql12 = sqlx + "email='" + email + "'" + sqly;
		System.out.println("updateProcess(). sql12 = " + sql12);
		a = dbProcess.executeUpdate(sql12);
		text_Email.setText("");
	}
	if (!state.equals("")) {
		sql13 = sqlx + "state='" + state + "'" + sqly;
		System.out.println("updateProcess(). sql13 = " + sql13);
		a = dbProcess.executeUpdate(sql13);
		text_State.setText("");
	}
	if (!remark.equals("")) {
		sql14 = sqlx + "remark='" + remark + "'" + sqly;
		System.out.println("updateProcess(). sql14 = " + sql14);
		a = dbProcess.executeUpdate(sql14);
		text_Remark.setText("");
	}
```

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

		try{
			if (a < 1) {
				System.out.println("updateProcess(). update database failed.");
				JOptionPane.showMessageDialog(null,
					" 该 员 工 不 存 在 ！ "," 处 理 结 果
",JOptionPane.ERROR_MESSAGE);
			}
		}catch(Exception e){
			System.out.println("e = " + e);
			JOptionPane.showMessageDialog(null,
				"更新失败","处理结果",JOptionPane.ERROR_MESSAGE);
		}
	}


}
```

5.新员工加入代码

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.Vector;

public class new_staff_addition extends JFrame implements ActionListener {
	JLabel jLID = new JLabel("员工号");
	JLabel jLPassword = new JLabel("密码");
	JLabel jLAuthority =new JLabel("权限");
	JLabel jLName = new JLabel("姓名");
	JLabel jLSex = new JLabel("性别");
	JLabel jLBirthday = new JLabel("生日");
	JLabel jLDepartment = new JLabel("部门");
	JLabel jLJob = new JLabel("职务");
	JLabel jLEdu_level = new JLabel("受教育程度");
	JLabel jLSpecialty = new JLabel("专业技能");
JLabel jLAdress = new JLabel("家庭住址");
JLabel jLTel = new JLabel("联系电话");
JLabel jLEmail = new JLabel("电子邮箱");
JLabel jLState = new JLabel("当前状态");
JLabel jLRemark = new JLabel("备注");

JTextField text_ID = new JTextField();
JTextField text_Password = new JTextField();
JTextField text_Authority =new JTextField();
JTextField text_Name = new JTextField();
JTextField text_Sex = new JTextField();
JTextField text_Birthday = new JTextField();
JTextField text_Department = new JTextField();
JTextField text_Job = new JTextField();
JTextField text_Edu_level = new JTextField();
JTextField text_Specialty = new JTextField();
JTextField text_Adress = new JTextField();
JTextField text_Tel = new JTextField();
JTextField text_Email = new JTextField();
JTextField text_State = new JTextField();
JTextField text_Remark = new JTextField();

JButton insert = new JButton("添加该新员工");
DbProcess dbProcess = new DbProcess();

public new_staff_addition(){
	super("添加");
	setBounds(500,80,500,700);
	this.setVisible(true);
	this.setLayout(null);
	jLID.setBounds(20,30,60,24);
	this.add(jLID);
	text_ID.setBounds(90,30,240,24);
	this.add(text_ID);
	jLPassword.setBounds(20,60,60,24);
	this.add(jLPassword);
	text_Password.setBounds(90,60,240,24);
	this.add(text_Password);
	jLAuthority.setBounds(20,90,60,24);
	this.add(jLAuthority);
	text_Authority.setBounds(90,90,240,24);
	this.add(text_Authority);
	jLName.setBounds(20,120,60,24);
	this.add(jLName);
```

```
text_Name.setBounds(90,120,240,24);
this.add(text_Name);
jLSex.setBounds(20,150,60,24);
this.add(jLSex);
text_Sex.setBounds(90,150,240,24);
this.add(text_Sex);

jLBirthday.setBounds(20,180,60,24);
this.add(jLBirthday);
text_Birthday.setBounds(90,180,240,24);
this.add(text_Birthday);

jLDepartment.setBounds(20,210,60,24);
this.add(jLDepartment);
text_Department.setBounds(90,210,240,24);
this.add(text_Department);

jLJob.setBounds(20,240,60,24);
this.add(jLJob);
text_Job.setBounds(90,240,240,24);
this.add(text_Job);

jLEdu_level.setBounds(20,270,60,24);
this.add(jLEdu_level);
text_Edu_level.setBounds(90,270,240,24);
this.add(text_Edu_level);

jLSpecialty.setBounds(20,300,60,24);
this.add(jLSpecialty);
text_Specialty.setBounds(90,300,240,24);
this.add(text_Specialty);

jLAdress.setBounds(20,330,60,24);
this.add(jLAdress);
text_Adress.setBounds(90,330,240,24);
this.add(text_Adress);

jLTel.setBounds(20,360,60,24);
this.add(jLTel);
text_Tel.setBounds(90,360,240,24);
this.add(text_Tel);

jLEmail.setBounds(20,390,60,24);
this.add(jLEmail);
```

```
		text_Email.setBounds(90,390,240,24);
		this.add(text_Email);

		jLState.setBounds(20,420,60,24);
		this.add(jLState);
		text_State.setBounds(90,420,240,24);
		this.add(text_State);

		jLRemark.setBounds(20,450,60,24);
		this.add(jLRemark);
		text_Remark.setBounds(90,450,240,24);
		this.add(text_Remark);

		insert.setBounds(120,550,150,40);
		this.add(insert);

		insert.addActionListener(this);
	}

	//自动分配员工号和密码
	public void Auto_distribution() throws SQLException{
		String sql = "select * from person;";
		String queryid = "";
		dbProcess.connect();
		ResultSet rs = dbProcess.executeQuery(sql);
		while(rs.next()){
			queryid = rs.getString("id");
		}
		if(queryid.equals(""))
		{
			queryid = "4300";
		}
		int id = Integer.parseInt(queryid) + 1;//加 1
		String last_id = String.valueOf(id);

		text_ID.setEnabled(false);//设置其不可改
		text_Password.setEnabled(false);

//		text_ID.setBackground(Color.RED);
//		text_Password.setBackground(Color.RED);

		text_ID.setForeground(Color.RED);
		text_Password.setForeground(Color.RED);
```

```
        text_ID.setText(last_id);//分配员工号和设置初始密码
        text_Password.setText("Staff"+last_id);
}



public void actionPerformed(ActionEvent e){

        if(e.getSource()== insert)
        {
                if(!text_ID.getText().isEmpty()
                     && !text_Authority.getText().isEmpty()
                     && !text_Name.getText().isEmpty()
                     && !text_Sex.getText().isEmpty()
                     && !text_Birthday.getText().isEmpty()
                     && !text_Department.getText().isEmpty()
                     && !text_Job.getText().isEmpty()
                     && !text_Edu_level.getText().isEmpty()
                     && !text_Specialty.getText().isEmpty()
                     && !text_Adress.getText().isEmpty()
                     && !text_Tel.getText().isEmpty()
                     && !text_Email.getText().isEmpty()
                     && !text_Password.getText().isEmpty()
                     && !text_State.getText().isEmpty()
                     && !text_Remark.getText().isEmpty()) {


              System.out.println("actionPerformed(). 添加");
                     boolean flag = insertProcess();
                     //更新 personnel 表

                     //1.获取 personnel 最后一行的 id
                     if(flag == true) {
                            try {
                                   // 建立查询条件
                                   String sql = "select * from personnel;";
                                   System.out.println("query_sql = " + sql);
                                   DbProcess dbProcess = new DbProcess();
                                   dbProcess.connect();
                                   ResultSet rs = dbProcess.executeQuery(sql);

                                   int personnel_last_id = 0;
                                   while (rs.next()) {
                                          personnel_last_id = rs.getInt("id");
```

```
                                        }

                                        int insert_id = personnel_last_id + 1;//加 1

                                        //2.将新加入员工的人事变更加入到 personnel 表中
                                        String sql2 = "insert into personnel values(" + insert_id
+ ",'" + text_ID.getText().trim() + "',0,'新员工加入')";
                                        dbProcess.executeUpdate(sql2);

                                        Auto_distribution();//要在人事表更新完之后

                                        dbProcess.disconnect();
                                    } catch (SQLException sqle) {
                                        System.out.println("sqle = " + sqle);
                                        JOptionPane.showMessageDialog(null,
                                                "数据操作错误","错误",
JOptionPane.ERROR_MESSAGE);
                                    }
                                }


//                              text_ID.setText("");
//                              text_Password.setText("");



                        }else{
                            JOptionPane.showMessageDialog(null,"请设置所有数据！","结
果",JOptionPane.WARNING_MESSAGE);
                        }

                }
        }
        public boolean insertProcess()
        {
            String id = text_ID.getText().trim();
            String password = text_Password.getText().trim();
            String authority = text_Authority.getText().trim();
            String name = text_Name.getText().trim();
            String sex = text_Sex.getText().trim();
            String birthday = text_Birthday.getText().trim();
            String department = text_Department.getText().trim();
            String job = text_Job.getText().trim();
            String edu_level = text_Edu_level.getText().trim();
```

```
		String specialty = text_Specialty.getText().trim();
		String address = text_Adress.getText().trim();
		String tel = text_Tel.getText().trim();
		String email = text_Email.getText().trim();
		String state = text_State.getText().trim();
		String remark = text_Remark.getText().trim();

		// 建立插入条件
		String sql = "insert into person values('";
		sql = sql + id + "','";
		sql = sql + password + "','";
		sql = sql + authority + "','";
		sql = sql + name + "','";
		sql = sql + sex + "','";
		sql = sql + birthday + "','";
		sql = sql + department + "',";
		sql = sql + job + ",";
		sql = sql + edu_level + ",'";
		sql = sql + specialty + "','";
		sql = sql + address + "','";
		sql = sql + tel + "','";
		sql = sql + email + "','";
		sql = sql + state + "','";
		sql = sql + remark + "');";

		System.out.println("insertProcess(). sql = " + sql);
		try{

			//部门编号必须是部门表中的部门

if(!(department.equals("001")||department.equals("002")||department.equals("00
3")||department.equals("004")||department.equals("005"))){
				JOptionPane.showMessageDialog(null," 没 有 该 部 门 "," 结 果
",JOptionPane.ERROR_MESSAGE);
				return false;
			}
			int result = dbProcess.executeUpdate(sql);
			if (result < 1) {
				System.out.println("insertProcess(). insert database failed.");
				return false;
			}else{
				JOptionPane.showMessageDialog(null," 添 加 成 功 "," 结 果
",JOptionPane.INFORMATION_MESSAGE);
```

```
                    text_Authority.setText("");
                    text_Name.setText("");
                    text_Sex.setText("");
                    text_Birthday.setText("");
                    text_Department.setText("");
                    text_Job.setText("");
                    text_Edu_level.setText("");
                    text_Specialty.setText("");
                    text_Adress.setText("");
                    text_Tel.setText("");
                    text_Email.setText("");
                    text_State.setText("");
                    text_Remark.setText("");
                }
            }catch(Exception e){
                System.out.println("e = " + e);
                JOptionPane.showMessageDialog(null,
                        "数据操作错误","错误",JOptionPane.ERROR_MESSAGE);
            }
            return true;
        }
}
```

6.人事自动变更界面代码

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.Vector;

public class personnel extends JFrame implements ActionListener {
    static String number = "";//获取记录编号
    private JLabel per_info = new JLabel("人事变更表");
    private JButton back = new JButton("返回上一页");
    private JButton query = new JButton("查询");//通过 id 查询某个变更记录
    private JButton queryall = new JButton("查询所有记录");
    private JLabel through_id = new JLabel("输入记录编号：");
    private JTextField input_id = new JTextField();
    private JLabel through_Staff_id = new JLabel("输入员工号：");
```

```
	private JTextField input_Staff_id = new JTextField();
	private JLabel through_change_code = new JLabel("输入变更类型代码：");
	private JTextField input_change_code = new JTextField();

	private JTable personnelTable = null;
	JScrollPane personnelJScrollPane = null;

	Vector titleVector = new Vector();
	Vector personnelVector = new Vector();


	public personnel() {
		super("人事管理系统——人事变更界面");
		this.setVisible(true);
		this.setBounds(240, 30, 900, 800);
		this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
		this.setLayout(null);

		per_info.setFont(new Font(null, Font.CENTER_BASELINE, 15));
		per_info.setBounds(400, 20, 100, 40);
		this.add(per_info);

		//表头
		titleVector.add("记录编号");
		titleVector.add("员工号");
		titleVector.add("变更代码");
		titleVector.add("详细记录");

		personnelTable = new JTable(personnelVector, titleVector);
		personnelTable.setRowHeight(32);

personnelTable.getColumnModel().getColumn(3).setPreferredWidth(300);// 设 置 列
宽
			personnelJScrollPane = new JScrollPane(personnelTable);
		personnelTable.setPreferredScrollableViewportSize(new      Dimension(800,
400));

		//分别设置水平和垂直滚动条自动出现
		personnelJScrollPane.setHorizontalScrollBarPolicy(
				JScrollPane.HORIZONTAL_SCROLLBAR_AS_NEEDED);
		personnelJScrollPane.setVerticalScrollBarPolicy(
				JScrollPane.VERTICAL_SCROLLBAR_AS_NEEDED);

		personnelJScrollPane.setBounds(40, 70, 800, 400);
```

```
        this.add(personnelJScrollPane);//设置表格位置并加入框架

        through_id.setBounds(60, 520, 100, 30);
        this.add(through_id);
        input_id.setBounds(180, 520, 200, 30);
        this.add(input_id);

        through_Staff_id.setBounds(60, 570, 100, 30);
        this.add(through_Staff_id);
        input_Staff_id.setBounds(180, 570, 200, 30);
        this.add(input_Staff_id);

        through_change_code.setBounds(60, 620, 120, 30);
        this.add(through_change_code);
        input_change_code.setBounds(180, 620, 200, 30);
        this.add(input_change_code);

        query.setBounds(130, 680, 100, 30);
        this.add(query);

        queryall.setBounds(270, 680, 120, 30);
        this.add(queryall);

        back.setBounds(600, 700, 110, 25);
        this.add(back);

        //添加监听
        query.addActionListener(this);
        queryall.addActionListener(this);
        back.addActionListener(this);
    }

    public void actionPerformed(ActionEvent e) {

        String id = input_id.getText().trim();
        String Staff_id = input_Staff_id.getText().trim();
        String per_change = input_change_code.getText().trim();

        //如果点击的是查询
        if (e.getSource() == query) {

            if (id.isEmpty() && Staff_id.isEmpty() && per_change.isEmpty()) {
                JOptionPane.showMessageDialog(null, "请输入查询条件！", "处
理结果", JOptionPane.WARNING_MESSAGE);
```

```
			} else if (!id.isEmpty() && Staff_id.isEmpty() && per_change.isEmpty())
{
				//通过编号查询

				try {
					// 建立查询条件
					String sql = "select * from personnel where id= " + id + ";";
					System.out.println("query_sql = " + sql);
					DbProcess dbProcess = new DbProcess();
					dbProcess.connect();
					ResultSet rs = dbProcess.executeQuery(sql);

					// 将查询获得的记录数据，转换成适合生成 JTable 的数
据形式
					personnelVector.clear();

					int rowCount = 0;
					while (rs.next()) {
						Vector v = new Vector();
						v.add(Integer.valueOf(rs.getInt("id")));
						v.add(rs.getString("person"));
						v.add(Integer.valueOf(rs.getInt("per_change")));
						v.add(rs.getString("per_description"));
						personnelVector.add(v);
						rowCount++;
					}

					personnelTable.updateUI();

					JOptionPane.showMessageDialog(null, "查询到 " +
rowCount + " 条记录", "处理结果", JOptionPane.INFORMATION_MESSAGE);
					dbProcess.disconnect();
				} catch (SQLException sqle) {
					System.out.println("sqle = " + sqle);
					JOptionPane.showMessageDialog(null,
						"数据操作错误", "错误",
JOptionPane.ERROR_MESSAGE);
				}

			} else if (id.isEmpty() && !Staff_id.isEmpty() && per_change.isEmpty())
{
				//通过员工号查询
				try {
					// 建立查询条件
					String sql = "select * from personnel where person= '" +
Staff_id + "';";
					System.out.println("query_sql = " + sql);
					DbProcess dbProcess = new DbProcess();
					dbProcess.connect();
					ResultSet rs = dbProcess.executeQuery(sql);

					// 将查询获得的记录数据，转换成适合生成 JTable 的数
据形式
					personnelVector.clear();

					int rowCount = 0;
					while (rs.next()) {
						Vector v = new Vector();
						v.add(Integer.valueOf(rs.getInt("id")));
						v.add(rs.getString("person"));
						v.add(Integer.valueOf(rs.getInt("per_change")));
						v.add(rs.getString("per_description"));
						personnelVector.add(v);
						rowCount++;
					}

					personnelTable.updateUI();

					JOptionPane.showMessageDialog(null, " 查 询 到  " +
rowCount + " 条记录", "处理结果", JOptionPane.INFORMATION_MESSAGE);
					dbProcess.disconnect();
				} catch (SQLException sqle) {
					System.out.println("sqle = " + sqle);
					JOptionPane.showMessageDialog(null,
						" 数 据 操 作 错 误 ", " 错 误 ",
JOptionPane.ERROR_MESSAGE);
				}
			} else if (id.isEmpty() && Staff_id.isEmpty() && !per_change.isEmpty())
{
				//通过变更代码
				try {
					// 建立查询条件
					String sql = "select * from personnel where per_change= '" +
per_change + "';";
					System.out.println("query_sql = " + sql);
					DbProcess dbProcess = new DbProcess();
					dbProcess.connect();
					ResultSet rs = dbProcess.executeQuery(sql);
```

```
                    // 将查询获得的记录数据，转换成适合生成 JTable 的数
据形式
                    personnelVector.clear();
                    int rowCount = 0;
                    while (rs.next()) {
                        Vector v = new Vector();
                        v.add(Integer.valueOf(rs.getInt("id")));
                        v.add(rs.getString("person"));
                        v.add(Integer.valueOf(rs.getInt("per_change")));
                        v.add(rs.getString("per_description"));
                        personnelVector.add(v);
                        rowCount++;
                    }

                    personnelTable.updateUI();
                    JOptionPane.showMessageDialog(null, "查询到 " +
rowCount + " 条记录", "处理结果", JOptionPane.INFORMATION_MESSAGE);

                    dbProcess.disconnect();
                } catch (SQLException sqle) {
                    System.out.println("sqle = " + sqle);
                    JOptionPane.showMessageDialog(null,
                            "数据操作错误", "错误",
JOptionPane.ERROR_MESSAGE);
                }
            } else if (id.isEmpty() && !Staff_id.isEmpty() && !per_change.isEmpty())
{
                //通过员工号和变更代码查询
                try {
                    // 建立查询条件
                    String sql = "select * from personnel where person= '" +
Staff_id + "' and per_change='" + per_change + "';";
                    System.out.println("query_sql = " + sql);
                    DbProcess dbProcess = new DbProcess();
                    dbProcess.connect();
                    ResultSet rs = dbProcess.executeQuery(sql);

                    // 将查询获得的记录数据，转换成适合生成 JTable 的数
据形式
                    personnelVector.clear();

                    int rowCount = 0;
                    while (rs.next()) {
                        Vector v = new Vector();
                        v.add(Integer.valueOf(rs.getInt("id")));
                        v.add(rs.getString("person"));
                        v.add(Integer.valueOf(rs.getInt("per_change")));
                        v.add(rs.getString("per_description"));
                        personnelVector.add(v);
                        rowCount++;
                    }

                    personnelTable.updateUI();

                    JOptionPane.showMessageDialog(null, "查询到 " +
rowCount + " 条记录", "处理结果", JOptionPane.INFORMATION_MESSAGE);
                    dbProcess.disconnect();
                } catch (SQLException sqle) {
                    System.out.println("sqle = " + sqle);
                    JOptionPane.showMessageDialog(null,
                            "数据操作错误", "错误",
JOptionPane.ERROR_MESSAGE);
                }
            } else if (!id.isEmpty() && (!Staff_id.isEmpty()
|| !per_change.isEmpty())) {
                JOptionPane.showMessageDialog(null, "由于编号唯一，请先清
空您输入的编号！", "处理结果", JOptionPane.ERROR_MESSAGE);
            }
        } else if (e.getSource() == queryall) {
            queryAll_personnel();
            JOptionPane.showMessageDialog(null, "查询成功", "处理结果",
JOptionPane.INFORMATION_MESSAGE);
        } else if (e.getSource() == back) {
            this.dispose();
            new administrator_mian().queryAllProcess();
        }

    }

    public void queryAll_personnel() {
        try {
            // 建立查询条件
            String sql = "select * from personnel ;";
            System.out.println("queryAllProcess(). sql = " + sql);
            DbProcess dbProcess = new DbProcess();
            dbProcess.connect();
            ResultSet rs = dbProcess.executeQuery(sql);
            // 将查询获得的记录数据，转换成适合生成 JTable 的数据形式
            personnelVector.clear();
            while (rs.next()) {
                Vector v = new Vector();
                v.add(rs.getString("id"));
                v.add(rs.getString("person"));
                v.add(Integer.valueOf(rs.getInt("per_change")));
                v.add(rs.getString("per_description"));
                personnelVector.add(v);
            }

            personnelTable.updateUI();

            dbProcess.disconnect();
        } catch (SQLException sqle) {
            System.out.println("sqle = " + sqle);
            JOptionPane.showMessageDialog(null,
                    "数据操作错误", "错误", JOptionPane.ERROR_MESSAGE);
        }
    }
```

7.员工身份登陆后个人信息查询与修改密码界面

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.ResultSet;
import java.sql.SQLException;

public class Staff extends JFrame    implements ActionListener {
    JLabel jLID = new JLabel("员工号：");
    JLabel jLPassword = new JLabel("密码：");
    JLabel jLAuthority =new JLabel("权限：");
    JLabel jLName = new JLabel("姓名：");
    JLabel jLSex = new JLabel("性别：");
    JLabel jLBirthday = new JLabel("生日：");
    JLabel jLDepartment = new JLabel("部门：");
    JLabel jLJob = new JLabel("职务：");
    JLabel jLEdu_level = new JLabel("受教育程度：");
    JLabel jLSpecialty = new JLabel("专业技能：");
    JLabel jLAdress = new JLabel("家庭住址：");
    JLabel jLTel = new JLabel("联系电话：");
    JLabel jLEmail = new JLabel("电子邮箱：");
    JLabel jLState = new JLabel("当前状态：");
    JLabel jLRemark = new JLabel("备注：");

    JLabel info = new JLabel("个人信息");//个人信息

    String login_id = new get_ID().id_get();
    JLabel show_ID = new JLabel(login_id);//获得登录成功后的员工 ID

    JLabel show_Password = new JLabel();
    JLabel show_Authority =new JLabel();
    JLabel show_Name = new JLabel();
    JLabel show_Sex = new JLabel();
    JLabel show_Birthday = new JLabel();
    JLabel show_Department = new JLabel();
    JLabel show_Job = new JLabel();
    JLabel show_Edu_level = new JLabel();
    JLabel show_Specialty = new JLabel();
    JLabel show_Adress = new JLabel();
    JLabel show_Tel = new JLabel();
    JLabel show_Email = new JLabel();
    JLabel show_State = new JLabel();
    JLabel show_Remark = new JLabel();//显示该员工的个人信息

    JLabel password1    = new JLabel("设置新密码：");
    JLabel password2 = new JLabel("确认新密码：");
    JPasswordField set_password = new JPasswordField();
    JPasswordField iden_password = new JPasswordField();
    JButton update_password = new JButton("修改密码");
    JButton back = new JButton("返回");

    public Staff(){
        super("人事管理系统——员工个人界面");
        this.setVisible(true);
        this.setBounds(500,50,650,750);
        this.setLayout(null);

        info.setFont(new Font(null,Font.PLAIN,18));//设置字体
        info.setBounds(260,10,120,24);
        this.add(info);

        jLID.setBounds(150,50,60,24);
        this.add(jLID);
```

```
jLPassword.setBounds(150,80,60,24);
this.add(jLPassword);

jLAuthority.setBounds(150,110,60,24);
this.add(jLAuthority);

jLName.setBounds(150,140,60,24);
this.add(jLName);

jLSex.setBounds(150,170,60,24);
this.add(jLSex);


jLBirthday.setBounds(150,200,60,24);
this.add(jLBirthday);

jLDepartment.setBounds(150,230,60,24);
this.add(jLDepartment);

jLJob.setBounds(150,260,60,24);
this.add(jLJob);

jLEdu_level.setBounds(150,290,75,24);
this.add(jLEdu_level);

jLSpecialty.setBounds(150,320,60,24);
this.add(jLSpecialty);

jLAdress.setBounds(150,350,60,24);
this.add(jLAdress);

jLTel.setBounds(150,380,60,24);
this.add(jLTel);

jLEmail.setBounds(150,410,60,24);
this.add(jLEmail);

jLState.setBounds(150,440,60,24);
this.add(jLState);

jLRemark.setBounds(150,470,60,24);
this.add(jLRemark);
```

```
/*****************************************************************
***/

        show_ID.setBounds(265,50,260,24);
        this.add(show_ID);

        show_Password.setBounds(265,80,260,24);
        this.add(show_Password);

        show_Authority.setBounds(265,110,260,24);
        this.add(show_Authority);

        show_Name.setBounds(265,140,260,24);
        this.add(show_Name);

        show_Sex.setBounds(265,170,260,24);
        this.add(show_Sex);

        show_Birthday.setBounds(265,200,260,24);
        this.add(show_Birthday);

        show_Department.setBounds(265,230,260,24);
        this.add(show_Department);

        show_Job.setBounds(265,260,260,24);
        this.add(show_Job);

        show_Edu_level.setBounds(265,290,260,24);
        this.add(show_Edu_level);


        show_Specialty.setBounds(265,320,260,24);
        this.add(show_Specialty);

        show_Adress.setBounds(265,350,260,24);
        this.add(show_Adress);

        show_Tel.setBounds(265,380,260,24);
        this.add(show_Tel);

        show_Email.setBounds(265,410,260,24);
        this.add(show_Email);

        show_State.setBounds(265,440,260,24);
        this.add(show_State);

        show_Remark.setBounds(265,470,260,24);
        this.add(show_Remark);


        password1.setBounds(140,530,75,24);
        this.add(password1);
        set_password.setBounds(230,530,240,24);
        this.add(set_password);

        password2.setBounds(140,560,75,24);
        this.add(password2);
        iden_password.setBounds(230,560,240,24);
        this.add(iden_password);

        update_password.setBounds(270,600,110,35);
        this.add(update_password);

        back.setBounds(470,650,100,30);
        this.add(back);

        update_password.addActionListener(this);
        back.addActionListener(this);

        this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }

    public void showAll()    throws SQLException{
        DbProcess dbProcess = new DbProcess();
        String sql = "select * from person where id='"+login_id+"';";
        System.out.println("queryStaff(). sql = " + sql);

        dbProcess.connect();
        ResultSet rs = dbProcess.executeQuery(sql);

        while (rs.next()){
            show_Password.setText(rs.getString("password"));
            show_Authority.setText(rs.getString("authority"));
            show_Name.setText(rs.getString("name"));
            show_Sex.setText(rs.getString("sex"));
            show_Birthday.setText(rs.getString("birthday"));
            show_Department.setText(rs.getString("department"));
            show_Job.setText(rs.getString("job"));
				show_Edu_level.setText(rs.getString("edu_level"));
				show_Specialty.setText(rs.getString("specialty"));
				show_Adress.setText(rs.getString("address"));
				show_Tel.setText(rs.getString("tel"));
				show_Email.setText(rs.getString("email"));
				show_State.setText(rs.getString("state"));
				show_Remark.setText(rs.getString("remark"));

			}

		}//设置显示该员工的信息


		//实现方法
		public void actionPerformed(ActionEvent e)
		{
			//点击修改密码则更改员工密码
			if(e.getSource() == update_password){
				char[] Pwd1 = set_password.getPassword();
				String pwd1 = "";
				for(int i = 0 ; i < Pwd1.length ; i++)
				{
					pwd1 = pwd1 + Pwd1[i];
				}//获取输入的密码
				char[] Pwd2 = iden_password.getPassword();
				String pwd2 = "";
				for(int i = 0 ; i < Pwd2.length ; i++)
				{
					pwd2 = pwd2 + Pwd2[i];
				}//获取输入的密码

					if(pwd1.equals("")|| pwd2.equals(""))
					{
						JOptionPane.showMessageDialog(null,"不能设置密码为空！
","处理结果",JOptionPane.ERROR_MESSAGE);
					}
					else if(!pwd1.equals(pwd2))
					{
						JOptionPane.showMessageDialog(null,"两个密码不匹配！
","处理结果",JOptionPane.ERROR_MESSAGE);
						set_password.setText("");
						iden_password.setText("");
					}
					else{
```

```
                    DbProcess dbProcess = new DbProcess();
                    String sql = "update person set password='"+pwd1+"' where
id='"+login_id+"';";
                    System.out.println("updatePassword_sql = " + sql);

                    dbProcess.executeUpdate(sql);
                    JOptionPane.showMessageDialog(null,"密码修改成功","处
理结果",JOptionPane.INFORMATION_MESSAGE);
                    try {
                        showAll();
                        set_password.setText("");
                        iden_password.setText("");
                    } catch (SQLException throwables) {
                        throwables.printStackTrace();
                    }
                }
        }else if(e.getSource() == back)
        {
            this.dispose();
            new login();
        }
    }
}
```