Project3实验报告

---固定资产管理系统

1441904116蒋宇 162210702227王彬162210702231严晨旭

一．实验目的

1. 让学生更好的理解Oracle数据库系统的设计，同时也有利于学生对编程综合能力的提升。
2. 了解并熟悉Java swing和jwt中的一些特性，有利于学生更好的掌握Java图形编程的一些能力
3. 综合性的开发更加有利于学生综合能力的提升，加强学生的综合素养

二．实验内容

利用Java swing技术和Oracle数据库综合开发一个固定资产管理系统

三．实验原理

1. Oracle数据库

首先对于固定资产的管理最为重要的便是数据，而数据一般放在数据库中，因此需要创建相应的表并且对表中的数据实现相应的增删该查工作

1. 数据库中数据的类型

Oracle 数据库的核心是表，表中的列使用到的常见数据类型如下：

类型 含义

CHAR(length) 存储固定长度的字符串。参数 length 指定了长度，如果存储的字符串长

度小于 length，用空格填充。默认长度是 1，最长不超过 2000 字节。

VARCHAR2(length) 存储可变长度的字符串。length 指定了该字符串的最大长度。默认长度

是 1，最长不超过 4000 字符。

NUMBER(p，s) 既可以存储浮点数，也可以存储整数，p 表示数字的最大位数（如果是

小数包括整数部分和小数部分和小数点，p 默认是 38 为），s 是指小数位

数。

DATE 存储日期和时间，存储纪元、4 位年、月、日、时、分、秒，存储时间

从公元前 4712 年 1 月 1 日到公元后 4712 年 12 月 31 日。

TIMESTAMP 不但存储日期的年月日，时分秒，以及秒后 6 位，同时包含时区。

CLOB 存储大的文本，比如存储非结构化的 XML 文档

BLOB 存储二进制对象，如图形、视频、声音等。

表 1 Oracle 的部分数据类型

对应 NUMBER 类型的示例：

格式 输入的数字 实际的存储

NUMBER 1234.567 1234.567

NUMBER（6，2） 123.4567 123.46

NUMBER（4，2） 12345.67 输入的数字超过了所指定的精度，数据库不能存储

表 2 Number 示例

1. 数据库创建

Oracle 创建表同 SQL Server 一样，使用 CREATE TABLE 命令来完成。创建约束则使用如下

命令：

语法格式：ALTER TABLE 命令

ALTER TABLE 表名 ADD CONSTRAINT 约束名 约束内容。

不论创建表还是约束，与 SQL Server 基本相同，注意：在 Oracle 中 default 是一个值，

而 SQL Server 中 default 是一个约束，因此 Oracle 的 default 设置可以在建表的时候创建

1. 数据的增删改查

数据操纵语言（DML）用于对数据库的表中数据进行添加、修改、删除和 SELECT…For

UPDATE(后面专门学习该查询)操作。对比一期学习过的 SQL Server 操作，接下来一一介绍在

Oracle 中的操作。

**简单查询**

数据查询是用 SELECT 命令从数据库的表中提取信息。SELECT 语句的语法是：

语法结构：简单查询

SELECT \*|列名|表达式 FROM 表名 WHERE 条件 ORDER BY 列名

语法解析：

1. \*表示表中的所有列。

2. 列名可以选择若干个表中的列名，各个列表中间用逗号分隔。

3. 表达式可以是列名、函数、常数等组成的表达式。

4. WHERE 子句是查询的条件。

5. ORDER BY 要求在查询的结果中排序，默认是升序。

**数据插入**

用 INSERT 命令完成对数据的插入。

语法结构：根据结果集创建表

INSERT INTO 表名(列名 1，列名 2……) VALUES (值 1，值 2……)

语法解析：

1. 列名可以省略。当省略列名时，默认是表中的所有列名，列名顺序为表定义中列的

先后顺序。

2. 值的数量和顺序要与列名的数量和顺序一致。值的类型与列名的类型一致。

**更新数据**

Oracle 在表中更新数据的语法是：

语法结构：UPDATE 操作

UPDATE 表名 SET 列名 1=值，列名 2=值…… WHERE 条件

**删除数据**

Oracle 在表中删除数据的语法是：

语法结构：DELETE 操作

DELETE FROM 表名 WHERE 条件

1. Java swing技术

Swing 是一个为Java设计的GUI工具包。

Swing是JAVA基础类的一部分。

Swing包括了图形用户界面（GUI）器件如：文本框，按钮，分隔窗格和表。

Swing提供许多比AWT更好的屏幕显示元素。它们用纯Java写成，所以同Java本身一样可以跨平台运行，这一点不像AWT。它们是JFC的一部分。它们支持可更换的面板和主题（各种操作系统默认的特有主题），然而不是真的使用原生平台提供的设备，而是仅仅在表面上模仿它们。这意味着你可以在任意平台上使用JAVA支持的任意面板。轻量级组件的缺点则是执行速度较慢，优点就是可以在所有平台上采用统一的行为。

1. Java - odbc桥

开放数据库连接（Open Database Connectivity，ODBC）是为解决异构数据库间的数据共享而产生的，现已成为WOSA(The Windows Open System Architecture ),Windows开放系统体系结构)的主要部分和基于Windows环境的一种数据库访问接口标准ODBC 为异构数据库访问提供统一接口，允许应用程序以SQL 为数据存取标准，存取不同DBMS管理的数据；使应用程序直接操纵DB中的数据，免除随DB的改变而改变。用ODBC 可以访问各类计算机上的DB文件，甚至访问如Excel 表和ASCI I数据文件这类非数据库对象。

四．部分代码展示

一．后台封装代码

一，DBOperation封装好的后台数据库操作代码

package fams.db;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import javax.swing.tree.DefaultTreeCellEditor.EditorContainer;

import org.omg.CosNaming.NamingContextExtPackage.StringNameHelper;

import fams.frame.Register;

public class DBOperation {

Connection conn=null;

PreparedStatement ps=null;

ResultSet rs=null;

String driver="oracle.jdbc.driver.OracleDriver";

String url="jdbc:oracle:thin:@localhost:1521:orcl";

String user="GONGSI";

String password="123456";

String sql1="select \* from users where username=? and password=?";

String sql2="insert into users values(?,?,?,?)";

String sql3="update users set authority='SYS123' where job\_number=?";

public Connection getConn()

{

try {

Class.forName(driver);

System.out.println("驱动加载成功");

} catch (ClassNotFoundException e) {

// TODO Auto-generated catch block

System.out.println("驱动加载失败");

e.printStackTrace();

}

try {

conn=DriverManager.getConnection(url,user,password);

System.out.println("连接成功");

} catch (SQLException e) {

// TODO Auto-generated catch block

System.out.println("连接失败");

e.printStackTrace();

}

return conn;

}

//封装第2个方法：关闭方法

public void getClose()

{

try {

if(rs!=null)

{

rs.close();

System.out.println("成功关闭rs");

}

if(ps!=null)

{

ps.close();

System.out.println("成功关闭ps");

}

if(conn!=null)

{

conn.close();

System.out.println("成功关闭conn");

}

} catch (SQLException e) {

// TODO Auto-generated catch block

System.out.println("程序异常，资源未关闭");

e.printStackTrace();

}

}

public ResultSet selectDate(String us,String psw)

{

//连接

getConn();

//查询

try {

ps=conn.prepareStatement(sql1);

ps.setObject(1,us);

ps.setObject(2,psw);

rs=ps.executeQuery();

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

return rs;

//关闭

}

public void insertUserInfo(int job\_number,String username,String password)

{

// String sql6="insert into info values(?,?,?)";

conn=getConn();

System.out.print("11111");

try {

ps=conn.prepareStatement(sql2);

ps.setObject(1,job\_number);

ps.setObject(2,username);

ps.setObject(3,password);

ps.setObject(4, null);

ps.executeUpdate();

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

//密钥通过

public void insertAuthority()

{

//连接

getConn();

//查询

Register RIS=new Register();

RIS.dispose(); //退出界面

try {

ps=conn.prepareStatement(sql3);

ps.setObject(1,RIS.number);

ps.executeUpdate();

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

//关闭

getClose();

}

//增加界面

public void insertData(String number,String name,String price,String brand,String quantity,String color){

//连接

conn=getConn();

//中间处理

try {

String sql1="insert into fixed\_asset values(?,?,?,?,?,?)";

ps=conn.prepareStatement(sql1);

ps.setString(1, number);

ps.setString(2,name);

ps.setString(3, price);

ps.setString(4, brand);

ps.setString(5, color);

ps.setString(6, quantity);

ps.executeUpdate();

} catch (SQLException e) {

e.printStackTrace();

}

//关闭

getClose();

}

public void DeleteValue(int index,String name) {

if(conn == null) {

conn = getConn();

}

try {

String sql = "DELETE FROM FIXED\_ASSET WHERE NUMBERS = ? AND NAME = ?";

ps = conn.prepareStatement(sql);

ps.setInt(1, index);

ps.setString(2, name);

ps.executeUpdate();

} catch (SQLException e) {

e.printStackTrace();

}

getClose();

}

public static void main(String[] args) {

// TODO Auto-generated method stub

DBOperation dbo = new DBOperation();

dbo.getConn();

dbo.getClose();

}

}

二.MyRegExp正则表达式的后台代码

package fams.db;

import java.util.regex.Matcher;

import java.util.regex.Pattern;

import javax.swing.text.AttributeSet;

import javax.swing.text.BadLocationException;

import javax.swing.text.PlainDocument;

public class MyRegExp extends PlainDocument {

private Pattern pattern;

private Matcher m;

public MyRegExp(String pat)

{

super();

this.pattern=Pattern.compile(pat);

}

@Override

public void insertString

(int offset, String str, AttributeSet attr)

throws BadLocationException {

if (str == null){

return;

}

String tmp=getText(0, offset).concat(str);

m=pattern.matcher(tmp);

if(m.matches())

super.insertString(offset, str, attr);

}

}

三．SelectDao查询功能封装代码

package fams.db;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

//用ps对象完成查询

public class SelectDao {

Connection conn=null;

PreparedStatement ps=null;

ResultSet rs=null;

String sql1="select \* from good";

String sql2="select \* from good where ID=?";

String sql3="select \* from good where PRICE=?";

DBOperation me=new DBOperation();

public void findAll() throws Exception {

conn=me.getConn();

ps=conn.prepareStatement(sql1);

rs=ps.executeQuery();

while(rs.next()) {

System.out.println(rs.getInt(1)+"\t"+rs.getString(2)+"\t"+rs.getFloat(3)+"\t"+rs.getString(4)+"\t"+rs.getString(5)+"\t"+rs.getString(6));

}

me.getClose();

}

public void findById() throws Exception {

conn=me.getConn();

ps=conn.prepareStatement(sql2);

ps.setInt(1,2);

rs=ps.executeQuery();

while(rs.next()) {

System.out.println(rs.getInt(1)+"\t"+rs.getString(2)+"\t"+rs.getFloat(3)+"\t"+rs.getString(4)+"\t"+rs.getString(5)+"\t"+rs.getString(6));

}

me.getClose();

}

public void findByName() throws Exception {

conn=me.getConn();

ps=conn.prepareStatement(sql2);

ps.setInt(1,10);

rs=ps.executeQuery();

while(rs.next()) {

System.out.println(rs.getInt(1)+"\t"+rs.getString(2)+"\t"+rs.getFloat(3)+"\t"+rs.getString(4)+"\t"+rs.getString(5)+"\t"+rs.getString(6));

}

me.getClose();

}

public void findByPrice() {

}

public void findByColor() {

}

}

三．前端代码

package fams.frame;

import java.awt.BorderLayout;

import java.awt.Color;

import java.awt.Container;

import java.awt.Font;

import java.awt.FontMetrics;

import java.awt.Graphics;

import java.awt.GridLayout;

import java.awt.HeadlessException;

import java.awt.Image;

import java.awt.Rectangle;

import java.awt.Shape;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.awt.image.ImageObserver;

import java.sql.Connection;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.text.AttributedCharacterIterator;

import javax.swing.ImageIcon;

import javax.swing.JButton;

import javax.swing.JFrame;

import javax.swing.JLabel;

import javax.swing.JOptionPane;

import javax.swing.JPanel;

import javax.swing.JPasswordField;

import javax.swing.JTextField;

import javax.swing.UIManager;

import fams.db.DBOperation;

import fams.db.MyRegExp;

//登录

public class Login extends JFrame implements ActionListener {

/\*\*

\* 自动生成的序列化版本号

\*/

//private static final long serialVersionUID = 3620904701704635361L;

//step1:定义变量与对象

JPanel jp1,jp2,jp3;

JLabel jl1,jl2,jl3;

JButton jbt1,jbt2;

JTextField jtf;

JPasswordField jpf;

Container con = new Container();

//Step2:初始化()

public Login () {

//重写构造方法，完成初始化、实例化

jp1=new JPanel();

jp2=new JPanel();

jp3=new JPanel();

Font font1=new Font("宋体",Font.BOLD,32);

Font font2=new Font("宋体",Font.BOLD,20);

jl1=new JLabel("FAMS - Login");

jl2=new JLabel(" 用户： ",JTextField.CENTER);

jl3=new JLabel(" 密码： ",JTextField.CENTER);

jl1.setFont(font1);

jl2.setFont(font2);

jl3.setFont(font2);

//实例化按钮

jbt1=new JButton("登录");

jbt2=new JButton("注册");

//给按钮添加监听

jbt1.addActionListener(this);

jbt2.addActionListener(this);

jtf=new JTextField(20);

jpf=new JPasswordField(20);

//String sql="select \* from info where username=? and password=?";

//明确组件与面板的归属

//面板1包含：标签1

jp1.add(jl1);

//面板2包含：标签2、3+文本框+密码框

//设计jp2的布局(网格布2行2列)

//jp2.setLayout(new GridLayout(2,2));

jp2.add(jl2);

jp2.add(jtf);

jp2.add(jl3);

jp2.add(jpf);

//面板3包含：按钮1、2

jp3.add(jbt1);

jp3.add(jbt2);

//将面板加入到窗体中

ImageIcon img = new ImageIcon("Images/0.jpg");//这是背景图片

JLabel imgLabel = new JLabel(img);//将背景图放在标签里。

this.getLayeredPane().add(imgLabel, new Integer(Integer.MIN\_VALUE));//注意这里是关键，将背景标签添加到jframe的LayeredPane面板里。

imgLabel.setBounds(0,0,img.getIconWidth(), img.getIconHeight());//设置背景标签的位置

//窗体包含：面板1、2、3 并添加布局(边界布局)

this.add(jp1,BorderLayout.NORTH);

this.add(jp2,BorderLayout.CENTER);

this.add(jp3,BorderLayout.SOUTH);

this.setTitle("FAMS V1.0");

this.setSize(420,360);

this.setLocationRelativeTo(null);//窗体居中

this.setResizable(false);//禁止放大

this.setVisible(true);//可见

String outLookAndFeel ="com.jtattoo.plaf.aluminium.AluminiumLookAndFeel";

try {

UIManager.setLookAndFeel(outLookAndFeel);

} catch (Exception e) {

e.printStackTrace();

}

}

//Step3:调用

public static void main(String[] args) {

// TODO Auto-generated method stub

new Login();

}

@Override

public void actionPerformed(ActionEvent e)

{

ResultSet rs=null;

//方法重写赋予新的功能

//响应用户前台动作

if(e.getSource()==jbt1)

{

//登录

//JOptionPane.showMessageDialog(this, "登录啊");

//获取前台用户信息

String usernames=this.jtf.getText();

String passwords=this.jpf.getText();

//根据用户查询信息表中有无此人

DBOperation dbo=new DBOperation();

rs=dbo.selectDate(usernames, passwords);//查询用户

//登录结果

try {

if(rs.next())

{

JOptionPane.showMessageDialog(this, "信息正确。登录成功");

//登陆成功跳转到系统的主界面

if(rs.getObject(4)!=null)

{

this.dispose();

ManagerMainFrame mmf=new ManagerMainFrame();

}

else {

this.dispose();

UserMainFrame umf=new UserMainFrame();

}

}

else {

JOptionPane.showMessageDialog(this, "信息错误，请重新登录");

//登陆失败跳转到失败界面或不做处理

}

} catch (HeadlessException e1) {

// TODO Auto-generated catch block

e1.printStackTrace();

} catch (SQLException e1) {

// TODO Auto-generated catch block

e1.printStackTrace();

}

}

else if(e.getSource()==jbt2) {

//JOptionPane.showMessageDialog(this, "注册啊");

Register register=new Register();

}

}

}

四．数据类型

package fams.frame;

import java.awt.BorderLayout;

import java.awt.Color;

import java.awt.Container;

import java.awt.Font;

import java.awt.FontMetrics;

import java.awt.Graphics;

import java.awt.GridLayout;

import java.awt.HeadlessException;

import java.awt.Image;

import java.awt.Rectangle;

import java.awt.Shape;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.awt.image.ImageObserver;

import java.sql.Connection;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.text.AttributedCharacterIterator;

import javax.swing.ImageIcon;

import javax.swing.JButton;

import javax.swing.JFrame;

import javax.swing.JLabel;

import javax.swing.JOptionPane;

import javax.swing.JPanel;

import javax.swing.JPasswordField;

import javax.swing.JTextField;

import javax.swing.UIManager;

import fams.db.DBOperation;

import fams.db.MyRegExp;

//登录

public class Login extends JFrame implements ActionListener {

/\*\*

\* 自动生成的序列化版本号

\*/

//private static final long serialVersionUID = 3620904701704635361L;

//step1:定义变量与对象

JPanel jp1,jp2,jp3;

JLabel jl1,jl2,jl3;

JButton jbt1,jbt2;

JTextField jtf;

JPasswordField jpf;

Container con = new Container();

//Step2:初始化()

public Login () {

//重写构造方法，完成初始化、实例化

jp1=new JPanel();

jp2=new JPanel();

jp3=new JPanel();

Font font1=new Font("宋体",Font.BOLD,32);

Font font2=new Font("宋体",Font.BOLD,20);

jl1=new JLabel("FAMS - Login");

jl2=new JLabel(" 用户： ",JTextField.CENTER);

jl3=new JLabel(" 密码： ",JTextField.CENTER);

jl1.setFont(font1);

jl2.setFont(font2);

jl3.setFont(font2);

//实例化按钮

jbt1=new JButton("登录");

jbt2=new JButton("注册");

//给按钮添加监听

jbt1.addActionListener(this);

jbt2.addActionListener(this);

jtf=new JTextField(20);

jpf=new JPasswordField(20);

//String sql="select \* from info where username=? and password=?";

//明确组件与面板的归属

//面板1包含：标签1

jp1.add(jl1);

//面板2包含：标签2、3+文本框+密码框

//设计jp2的布局(网格布2行2列)

//jp2.setLayout(new GridLayout(2,2));

jp2.add(jl2);

jp2.add(jtf);

jp2.add(jl3);

jp2.add(jpf);

//面板3包含：按钮1、2

jp3.add(jbt1);

jp3.add(jbt2);

//将面板加入到窗体中

ImageIcon img = new ImageIcon("Images/0.jpg");//这是背景图片

JLabel imgLabel = new JLabel(img);//将背景图放在标签里。

this.getLayeredPane().add(imgLabel, new Integer(Integer.MIN\_VALUE));//注意这里是关键，将背景标签添加到jframe的LayeredPane面板里。

imgLabel.setBounds(0,0,img.getIconWidth(), img.getIconHeight());//设置背景标签的位置

//窗体包含：面板1、2、3 并添加布局(边界布局)

this.add(jp1,BorderLayout.NORTH);

this.add(jp2,BorderLayout.CENTER);

this.add(jp3,BorderLayout.SOUTH);

this.setTitle("FAMS V1.0");

this.setSize(420,360);

this.setLocationRelativeTo(null);//窗体居中

this.setResizable(false);//禁止放大

this.setVisible(true);//可见

String outLookAndFeel ="com.jtattoo.plaf.aluminium.AluminiumLookAndFeel";

try {

UIManager.setLookAndFeel(outLookAndFeel);

} catch (Exception e) {

e.printStackTrace();

}

}

//Step3:调用

public static void main(String[] args) {

// TODO Auto-generated method stub

new Login();

}

@Override

public void actionPerformed(ActionEvent e)

{

ResultSet rs=null;

//方法重写赋予新的功能

//响应用户前台动作

if(e.getSource()==jbt1)

{

//登录

//JOptionPane.showMessageDialog(this, "登录啊");

//获取前台用户信息

String usernames=this.jtf.getText();

String passwords=this.jpf.getText();

//根据用户查询信息表中有无此人

DBOperation dbo=new DBOperation();

rs=dbo.selectDate(usernames, passwords);//查询用户

//登录结果

try {

if(rs.next())

{

JOptionPane.showMessageDialog(this, "信息正确。登录成功");

//登陆成功跳转到系统的主界面

if(rs.getObject(4)!=null)

{

this.dispose();

ManagerMainFrame mmf=new ManagerMainFrame();

}

else {

this.dispose();

UserMainFrame umf=new UserMainFrame();

}}

else {

JOptionPane.showMessageDialog(this, "信息错误，请重新登录");

//登陆失败跳转到失败界面或不做处理

}

} catch (HeadlessException e1) {

// TODO Auto-generated catch block

e1.printStackTrace();

} catch (SQLException e1) {

// TODO Auto-generated catch block

e1.printStackTrace();

}

}

else if(e.getSource()==jbt2) {

//JOptionPane.showMessageDialog(this, "注册啊");

Register register=new Register();

}

}}

五．用户界面框架

package fams.frame;

import java.awt.BorderLayout;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import javax.swing.JButton;

import javax.swing.JComboBox;

import javax.swing.JFrame;

import javax.swing.JLabel;

import javax.swing.JPanel;

import javax.swing.JScrollPane;

import javax.swing.JTable;

import javax.swing.JTextField;

import fams.frame.MyModel;

public class UserMainFrame extends JFrame implements ActionListener{

Connection conn=null;

PreparedStatement ps=null;

ResultSet rs=null;

//1.定义-13个

//2 JLabel+1 JTextField+5 JButton+3 JPanel

JPanel jp1,jp2,jp3;

JLabel jl1,jl2;

JButton jbt1,jbt2,jbt3,jbt4;

JTextField jtf;

JComboBox jcb;//组合下拉列表框1

//表相关：3个

MyModel mm;

JTable jt;

JScrollPane jsp;//滚动面板，放置表

public UserMainFrame(){

mm=new MyModel();

jt=new JTable(mm);

jsp=new JScrollPane(jt);

jcb=new JComboBox<>();//实例化下拉列表

jcb.addItem("NUMBERS");

jcb.addItem("NAME");

jcb.addItem("PRICE");

jcb.addItem("BRAND");

jcb.addItem("QUANTITY NUMBER");

jcb.addItem("COLOR");

//实例化面板3个

jp1=new JPanel();

jp2=new JPanel();

jp3=new JPanel();

//标签实例化2个

jl1=new JLabel("请输入物品名");

jl2=new JLabel("请选择操作分类");

//按钮实例化3个

jbt1=new JButton("查询");

jbt2=new JButton("增加");

jbt3=new JButton("退出");

jbt4=new JButton("刷新");

jbt1.addActionListener(this);

jbt2.addActionListener(this);

jbt3.addActionListener(this);

jbt4.addActionListener(this);

//文本框实例化1个

jtf=new JTextField(20);

jp1.add(jl1);

jp1.add(jcb);

jp1.add(jtf);

jp1.add(jbt1);

jp2.add(jsp);

jp3.add(jl2);

jp3.add(jbt2);

jp3.add(jbt4);

jp3.add(jbt3);

this.add(jp1,BorderLayout.NORTH);

this.add(jsp,BorderLayout.CENTER);

this.add(jp3,BorderLayout.SOUTH);

this.setTitle("SGMS V1.0");

this.setBounds(30,30,600,600);

this.setLocationRelativeTo(null);

this.setResizable(false);

this.setVisible(true);

}

public static void main(String[] args) {

new UserMainFrame();

}

@Override

public void actionPerformed(ActionEvent arg0) {

// TODO 自动生成的方法存根

if(arg0.getSource()==jbt1) {

String selectN=(String) jcb.getSelectedItem();

String names=jtf.getText();

if(selectN.equals("NUMBERS")) {

if(names.equals("")) {

try {

mm=new MyModel();

} catch (Exception e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

jt.setModel(mm);

}

else {String sql="select \* from fixed\_asset where numbers='"+names+"'";

try {

mm=new MyModel(sql);

} catch (Exception e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

jt.setModel(mm);}

}else if(selectN.equals("NAME")) {

if(names.equals("")) {

try {

mm=new MyModel();

} catch (Exception e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

jt.setModel(mm);

}

else {String sql="select \* from fixed\_asset where name='"+names+"'";

try {

mm=new MyModel(sql);

} catch (Exception e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

jt.setModel(mm);}

}else if(selectN.equals("PRICE")) {

if(names.equals("")) {

try {

mm=new MyModel();

} catch (Exception e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

jt.setModel(mm);

}

else {String sql="select \* from fixed\_asset where price='"+names+"'";

try {

mm=new MyModel(sql);

} catch (Exception e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

jt.setModel(mm);}

}else if(selectN.equals("BRAND")) {

if(names.equals("")) {

try {

mm=new MyModel();

} catch (Exception e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

jt.setModel(mm);

}

else {String sql="select \* from fixed\_asset where brand='"+names+"'";

try {

mm=new MyModel(sql);

} catch (Exception e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

jt.setModel(mm);}

}else if(selectN.equals("COLOR")) {

if(names.equals("")) {

try {

mm=new MyModel();

} catch (Exception e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

jt.setModel(mm);

}

else {String sql="select \* from fixed\_asset where color='"+names+"'";

try {

mm=new MyModel(sql);

} catch (Exception e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

jt.setModel(mm);}

}else if(selectN.equals("QUANTITY")) {

if(names.equals("")) {

try {

mm=new MyModel();

} catch (Exception e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

jt.setModel(mm);

}

else {String sql="select \* from fixed\_asset where quantity='"+names+"'";

try {

mm=new MyModel(sql);

} catch (Exception e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

jt.setModel(mm);}

}

}

else if(arg0.getSource()==jbt2)

{

new InsertData();

}

else if(arg0.getSource()==jbt3)

{

this.dispose();

}

else if(arg0.getSource()==jbt4)

{

this.dispose();

new UserMainFrame();

}

}

}

六．注册代码

package fams.frame;

import java.awt.BorderLayout;

import java.awt.Font;

import java.awt.GridLayout;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import javax.swing.JButton;

import javax.swing.JFrame;

import javax.swing.JLabel;

import javax.swing.JOptionPane;

import javax.swing.JPanel;

import javax.swing.JPasswordField;

import javax.swing.JTextField;

import fams.db.DBOperation;

import fams.db.MyRegExp;

//

import oracle.jdbc.Const;

public class Register extends JFrame implements ActionListener {

// 1定义相关变量

public static int number;

// 3个标签+3个文本框+2个按钮+3个面板

JPanel jp1, jp2,jp3;

JLabel jl1, jl2, jl3,jl4,jl5,jl6,jl7; // 标签

JTextField jtf1, jtf2;

JPasswordField jpf;

JButton jbt1, jbt2, jbt3;

Connection conn = null;

PreparedStatement ps = null;

ResultSet rs = null;

String sql1 = "select \* from users where job\_number=?";

// String sql2="insert into users values(?,?,?)";

// 2.初始化（构造方法）

public Register() {

// 2.1实例化new

jp1 = new JPanel();

jp2 = new JPanel();

jp3=new JPanel();

Font font1=new Font("宋体",Font.BOLD,32);

Font font2=new Font("宋体",Font.BOLD,20);

Font font3=new Font("宋体",Font.ITALIC,12);

jl1 = new JLabel(" 编 号：", JTextField.CENTER);

jl2 = new JLabel(" 用户名：", JTextField.CENTER);

jl3 = new JLabel(" 密 码：", JTextField.CENTER);

jl4=new JLabel("FAMS - Register", JTextField.CENTER);

jl5=new JLabel("1-9位数字 ", JTextField.CENTER);

jl6=new JLabel("1-10位数字或字母", JTextField.CENTER);

jl7=new JLabel("1-10位数字或字母", JTextField.CENTER);

jl4.setFont(font1);

jl1.setFont(font2);

jl2.setFont(font2);

jl3.setFont(font2);

jl5.setFont(font3);

jl6.setFont(font3);

jl7.setFont(font3);

jtf1 = new JTextField();

jtf2 = new JTextField();

jpf = new JPasswordField();

// jtf1.setDocument(new MyRegExp("^[0-9]\*$"));

jbt1 = new JButton("用户注册");

jbt2 = new JButton("管理员注册");

jbt3 = new JButton("退出");

jtf1 = new JTextField(20);

jtf1.setDocument(new MyRegExp("^[0-9]{0,9}$"));

jtf2 = new JTextField(20);

jtf2.setDocument(new MyRegExp("^[a-zA-Z0-9]{0,10}$"));

jpf = new JPasswordField(20);

jpf.setDocument(new MyRegExp("^[a-zA-Z0-9]{0,10}$"));

jbt1.addActionListener(this);

jbt2.addActionListener(this);

jbt3.addActionListener(this);

//jp1.setLayout(new GridLayout(3, 2));

jp1.add(jl1);

jp1.add(jtf1);// 文本框

jp1.add(jl5);

jp1.add(jl2);

jp1.add(jtf2);// 文本框

jp1.add(jl6);

jp1.add(jl3);

jp1.add(jpf);// 密码框

jp1.add(jl7);

jp2.add(jbt1);

jp2.add(jbt2);

jp2.add(jbt3);

jp3.add(jl4);

this.add(jp3, BorderLayout.NORTH);

this.add(jp1, BorderLayout.CENTER);

this.add(jp2, BorderLayout.SOUTH);

// 2.2设计注册窗体及设置属性

this.setTitle("欢迎注册FAMS系统");

this.setBounds(30, 30, 475, 360);

this.setLocationRelativeTo(null);

this.setVisible(true);

this.setResizable(false);

}

// 3.调用（main方法调用构造方法）（省略）

public static void main(String[] args) {

Register reg = new Register();

}

@Override

public void actionPerformed(ActionEvent e) {

if (e.getSource() == jbt1) {

try {

int job\_number = Integer.parseInt(this.jtf1.getText());

String usernames = this.jtf2.getText();

String passwords = this.jpf.getText();

DBOperation dbo = new DBOperation();

conn = dbo.getConn();

try {

ps = conn.prepareStatement(sql1);

ps.setObject(1, job\_number);

rs = ps.executeQuery();

if (rs.next()) {

JOptionPane.showMessageDialog(this, "编号重复，请重新输入");

} else {

JOptionPane.showMessageDialog(this, "编号不重复注册成功");

dbo.insertUserInfo(job\_number, usernames, passwords);

}

} catch (SQLException e1) {

// TODO Auto-generated catch block

e1.printStackTrace();

}

dbo.getClose();

} catch (NumberFormatException e2) {

//System.out.println("请正确输入。。。。。。。。");

e2.printStackTrace();

}

this.dispose(); // 退出界面

} else if (e.getSource() == jbt3) {

this.dispose(); // 退出系统

// this.dispose();//销毁当前窗体

} else if (e.getSource() == jbt2) {

int job\_number = Integer.parseInt(this.jtf1.getText());

String usernames = this.jtf2.getText();

String passwords = this.jpf.getText();

DBOperation dbo = new DBOperation();

conn = dbo.getConn();

try {

ps = conn.prepareStatement(sql1);

ps.setObject(1, job\_number);

rs = ps.executeQuery();

if (rs.next()) {

JOptionPane.showMessageDialog(this, "编号重复，请重新输入");

} else {

// JOptionPane.showMessageDialog(this, "编号不重复注册成功");

dbo.insertUserInfo(job\_number, usernames, passwords);

}

} catch (SQLException e1) {

// TODO Auto-generated catch block

e1.printStackTrace();

}

dbo.getClose();

number = job\_number;

this.dispose(); // 退出界面

AuthenticateFrame af = new AuthenticateFrame();

}

}

}

五．实验心得

通过此次实验，我们充分的仍认知到数据库设计在整个系统开发中的重要作用数据库设计在整个软件开发的起到的举足轻重的作用，尤其是我说的设计原则的第一点，数据库与需求是相辅相成的，我经常把软件开发比作汽车制造。汽车制造会经过图纸设计，模型制作，样车制造，小批量试生产，最后是批量生产等步骤。整个过程环环相扣，后一过程是建立在前一过程正确的前提基础之上的。如果在图纸设计阶段发现了一个纰漏，我们可以重新进行图纸设计，如果到了样车制造阶段发现这个错误，那么我们就要把从图纸设计到样车制造的阶段重来，越到后面发现设计上的问题，所付出的代价越大，修改的难度也越大。

数据库设计难度其实要比单纯的技术实现的难很多，他充分体现了一个人的全局设计能力和掌控能力，所以在今后的项目中大家一定要着重培养这方面的能力。