**数据库上机报告**

**一.实验环境**

**操作系统: Windows 10**

**软件：MySQL Community Server, Navicat Premium 15, IntelliJ IDEA Community**

**二.实验内容**

**1.课后习题**

对课后题编写存储过程，实现按供应商号查询该供应商的供应信息；结合教材中SPJ数据库，修改关系S，增加等级属性列(level)，编写触发器，当更新SPJ表中的QTY列时，取该供应商的供应量总和除以100作为其对应等级。

**2.场地预约管理系统**

某实验中心共有二十多间实验室，每个实验室有编号，名称，最大容纳人数，用途(可以提供的服务说明)，全景照片 ，基本设备等(包括，网络，电子屏幕，电脑，摄像头，音频输入输出设备)。

每间实验室开放预约时间从早上8点到晚上22点。

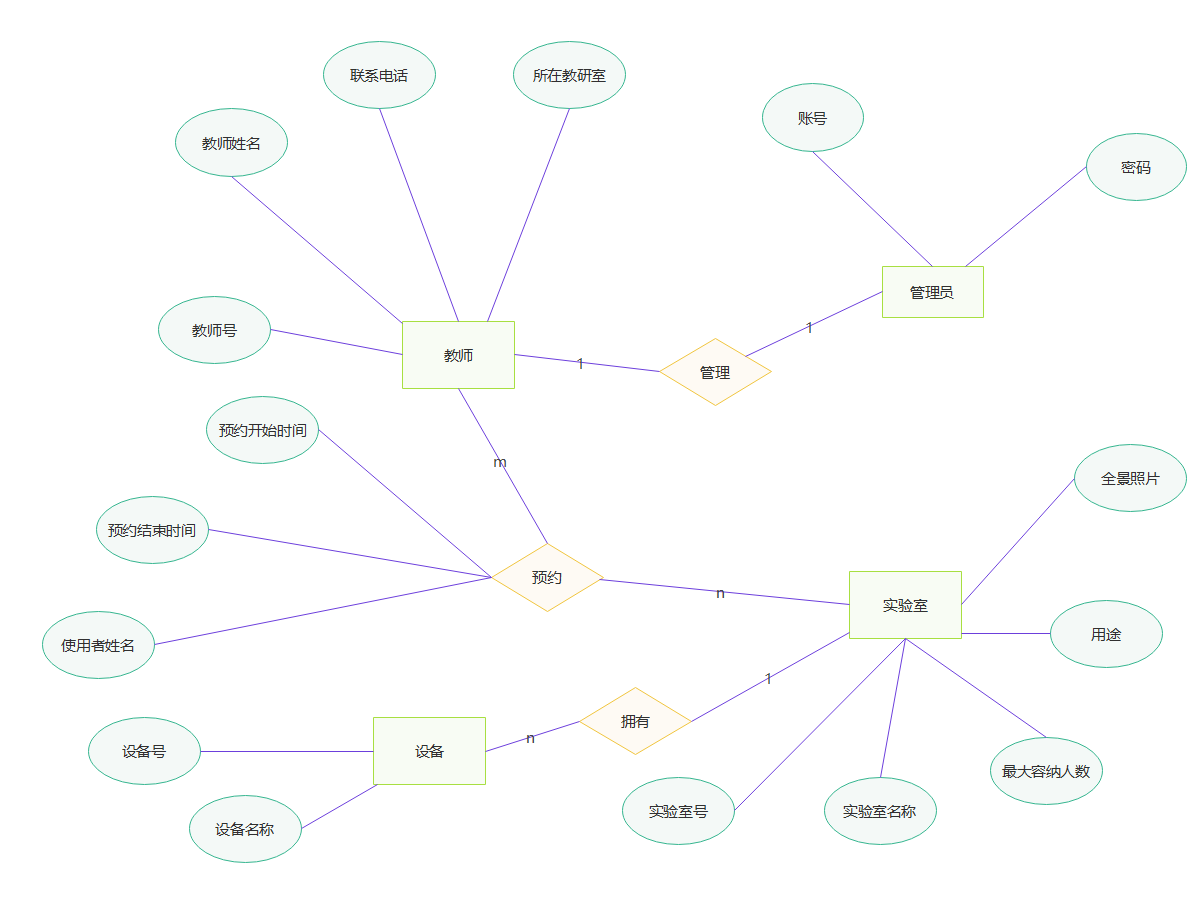
实验室只能被在职教师预约。在职教师具有(编号，姓名，联系电话，所属教研室)等信息。

每间实验室可以被多名教师预约，预约时教师需要指定预约实验室号，预约起止时间，使用者姓名(或组织单位)。同一时间，一个实验室只能被一名教师预约。

每个教师可以预约多间实验室。

**三.实验原理**

**ER图**



**关系模式**

教师(**教师号**,教师姓名,联系电话,所在教研室)

管理员(**账号**,密码,教师号)

实验室(**实验室号**,实验室名称,最大容纳人数,用途,全景照片)

设备(**设备号**,设备名称,实验室号）

预约(**教师号,实验室号,预约开始时间**,预约结束时间,使用者姓名)

**主码**

教师：教师号

管理员：账号

实验室：实验室号

设备：设备号

预约：(教师号,实验室号,预约开始时间)

**外码**

管理员：教师号

设备：实验室号

预约：教师号，实验室号

**相关函数依赖**

**教师：**教师号->教师姓名，教师号->联系电话，教师号->所在教研室

**管理员：**账号->密码，账号->教师号,教师号->账号

**实验室：**实验室号->实验室名称，实验室号->最大容纳人数，实验室号->用途，实验室号->全景照片,实验室名称->用途

**设备：**设备号->设备名称，设备号->实验室号

**预约：**(教师号,实验室号,预约开始时间)->预约结束时间，(教师号,实验室号,预约开始时间)->使用者姓名

**关系模式优化**

**教师：**BCNF

**管理员：**存在非码的函数依赖的决定因素：教师号->账号，为3NF

**实验室：**存在非码的函数依赖的决定因素：实验室名称->用途，为3NF

**设备：**BCNF

**预约：**BCNF

**四.具体步骤及相应源代码**

**课后习题**

**1.编写存储过程：**

CREATE PROCEDURE TEST(IN S\_NO CHAR(5))

BEGIN

SELECT \* FROM S WHERE SNO = S\_NO*;*

END*;*

**2.编写触发器**

CREATE TRIGGER T

AFTER UPDATE ON SPJ

FOR EACH ROW

BEGIN

UPDATE S SET LEVEL=(

SELECT SUM(QTY)/100 FROM SPJ

WHERE S.SNO**=**NEW.SNO AND S.SNO**=**SPJ.SNO

GROUP BY SPJ.SNO)*;*

END*;*

**综合性作业**

根据实验要求，我具体设计了5个表格，分别是管理表，用来存储用户账号和密码信息；教师表，用来存储教师相关信息；实验室表，用来存储实验室相关信息；设备表，用来存储设备相关信息；预约表，用来存储教师的预约记录。

由于该系统的目标用户并非都熟练使用数据库软件，所以要为该系统创建一套良好的人机交互界面，便于用户进行操作，为此我选择使用MySQL进行表的创建，接下来的一系列操作将使用Java来实现。

首先用Java 中的JDBC API 来连接数据库，注册驱动，连接成功后针对该系统的每一项功能执行相应的SQL语句。接下来使用JAVA SWING创造图形界面，实现简易前端，并使用IDEA中的JFORMDESIGNER 插件辅助界面的设计。

以上便是本系统开发过程的简要概述。

**1.信息录入**

//创建教师表

CREATE TABLE TEACHER(

TNO CHAR(10) PRIMARY KEY NOT NULL,

TNAME CHAR(20),

PHONE CHAR(20),

DEPT CHAR(20)

)*;*

//创建管理员表

CREATE TABLE ADMIN(

NUMBER CHAR(20) PRIMARY KEY NOT NULL,

PWD CHAR(20) NOT NULL,

TNO CHAR(10) NOT NULL UNIQUE,

FOREIGN KEY(TNO) REFERENCES LAB(TNO)

ON DELETE CASCADE ON UPDATE CASCADE

)*;*

//创建实验室表

CREATE TABLE LAB(

LNO CHAR(10) PRIMARY KEY NOT NULL,

LNAME CHAR(20),

MAXNUM CHAR(20),

PURPOSE CHAR(20),

PHOTO LONGBLOB

)*;*

//创建设备表

 CREATE TABLE DEVICE(

 DNO CHAR(10) PRIMARY KEY NOT NULL,

 DNAME CHAR(20),

 LNO CHAR(10),

 FOREIGN KEY(LNO) REFERENCES LAB(LNO)

 ON DELETE CASCADE ON UPDATE CASCADE

 )*;*

 //创建预约表

 CREATE TABLE BOOK(

 TNO CHAR(10) NOT NULL,

 LNO CHAR(10) NOT NULL,

 BEGINTIME TIMESTAMP NOT NULL,

 ENDTIME TIMESTAMP NOT NULL,

 USERNAME CHAR(20),

 PRIMARY KEY(TNO,LNO,BEGINTIME),

 FOREIGN KEY(TNO) REFERENCES TEACHER(TNO)

 ON DELETE CASCADE ON UPDATE CASCADE,

 FOREIGN KEY(LNO) REFERENCES LAB(LNO)

 ON DELETE CASCADE ON UPDATE CASCADE

 )*;*

INSERT INTO TEACHER VALUES(1,'BILL',1362343332,'CS')*;*

INSERT INTO TEACHER VALUES(2,'JACK',1387777677,'SE')*;*

INSERT INTO TEACHER VALUES(3,'TOM',1382324343,'CN')*;*

INSERT INTO TEACHER VALUES(4,'MARK',1387777677,'PE')*;*

INSERT INTO ADMIN VALUES(12345,12345,1)*;*

INSERT INTO ADMIN VALUES(23456,23456,2)*;*

INSERT INTO ADMIN VALUES(34567,34567,3)*;*

INSERT INTO ADMIN VALUES(45678,45678,4)*;*

INSERT INTO LAB VALUES(11,'COMPUTER\_LAB',70,'PROGRAMMING',LOAD\_FILE("D:\javafile\test\src\computer\_lab.jpg"))*;*

INSERT INTO LAB VALUES(12,'PHYSICS\_LAB',80,'STUDY',LOAD\_FILE("D:\javafile\test\src\physics\_lab.jpg"))*;*

INSERT INTO DEVICE VALUES(21,'COMPUTER',11)*;*

INSERT INTO DEVICE VALUES(22,'BALANCE',12)*;*

INSERT INTO BOOK VALUES(1,11,'2022-12-01 09:00:00','2022-12-01 16:00:00','JHON')*;*

INSERT INTO BOOK VALUES(2,12,'2022-12-03 10:00:00','2022-12-03 20:00:00','MARY')*;*

INSERT INTO BOOK VALUES(2,12,'2022-12-01 10:00:00','2022-12-01 20:00:00','MARY')*;*

INSERT INTO BOOK VALUES(2,11,'2022-12-01 17:00:00','2022-12-01 21:30:00','MARY')*;*

INSERT INTO BOOK VALUES(1,12,'2022-12-04 12:00:00','2022-12-04 19:35:00','TONY')*;*

INSERT INTO BOOK VALUES(4,11,'2022-12-06 13:00:00','2022-12-06 17:35:00','MARK')*;*

INSERT INTO BOOK VALUES(3,11,'2022-12-05 12:00:00','2022-12-05 19:35:00','TOM')*;*

**2.信息查询**

**(1)通过教师编号或教师名查询该教师某一时间段内的预定记录(教**

**师编号，预定起止时间，预定实验室，使用者)**

private void button1MouseClicked(MouseEvent e) {

        // TODO add your code here

        String s = new String(textField1.getText());**//教师名**

        String s1 = new String(textField2.getText());**//预定开始时间**

        String s2 = new String(textField3.getText());**//预定结束时间**

        String s3 = new String(textField4.getText());**//使用者姓名**

        Connection connection = null;

        Statement statement = null;

        ResultSet rs = null;

        try {

            //获取数据连接

            connection = jdbcUtil.openConnection();

            //获取发送sql指令执行sql对象

            statement = connection.createStatement();

            //返回查询结果集用于保存数据库查询内容

            String sql = "SELECT TNO,BEGINTIME,ENDTIME,LNO,USERNAME FROM BOOK WHERE (TNO="+s+" OR TNO=(SELECT TNO FROM TEACHER WHERE TNAME="+s2+")) AND BEGINTIME>="+s2+" AND ENDTIME<="+s3;

            rs = statement.executeQuery(sql);

            //遍历结果集拿到数据

            if(rs!=null)

            System.out.println("教师编号\t\t起始时间\t\t\t\t结束时间\t\t\t\t\t实验室编号\t使用者");

            while (rs.next()) {

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

            }

        } catch (Exception e1) {

            e1.printStackTrace();

        }finally {

            //执行完数据库操作后记得关闭数据库连接资源

            try{

                rs.close();

                statement.close();

                connection.close();

            }catch (SQLException e2){

                e2.printStackTrace();

            }

        }

    }

(2)**通过实验室号及日期查询本实验室当天场地已预定时间段**

String s1 = new String(textField1.getText())*;***//实验号**

String s2 = new String(textField2.getText())*;***//实验日期**

String sql = "SELECT TIME(BEGINTIME),TIME(ENDTIME) FROM BOOK WHERE LNO="+s1+" AND DATE(BEGINTIME)<=>"+s2*;*

**(3)** **通过设备号查询设备所在实验室**

String s1 = new String(textField1.getText())*;*

String sql="SELECT LNO FROM DEVICE WHERE DNO="+s1*;*

**(4)** **按照年月日查询所有实验室的空闲时间段**

我这里用了两个字符串数组来存储实验室的空闲开始时间和空闲结束时间，由于创建表的时候也许并没有08:00:00和22:00:00这两个时间点，但显示给用户的时候可能要用到这两个时间点所以要把他们两个加上，之后排序。如果之前预约表里有这两个时间点(两个数组长度不一样)，那么还要进行去重。这里的空闲开始时间也就是该实验室上个使用者上次预定结束时间，空闲结束时间也就是下一个使用者预定开始时间。

        String s1 = new String(textField1.getText())*;*

        String s2 = null,temp =null*;*

        ArrayList<String> s3 = new ArrayList<>()*;***//存储空闲开始时间**

        ArrayList<String> s4 = new ArrayList<>()*;***//存储空闲结束时间**

        Connection connection = null*;*

        Statement statement = null*;*

        ResultSet rs = null*;*

        try {

            //获取数据连接

            connection = jdbcUtil.openConnection()*;*

            //获取发送sql指令执行sql对象

            statement = connection.createStatement()*;*

            //返回查询结果集用于保存数据库查询内容

            String sql ="SELECT LNO,TIME(ENDTIME) FREE\_BEGIN," +

                    "TIME(BEGINTIME) FREE\_END FROM BOOK WHERE"

                    +" DATE(BEGINTIME)<=>"+s1+" ORDER BY LNO,BEGINTIME,ENDTIME"*;*

            rs = statement.executeQuery(sql)*;*

            //遍历结果集拿到数据

            if(rs.next()){

                System.out.println("实验室\t空闲开始时间\t空闲结束时间")*;*

                do {

                    s2**=**rs.getString(1)*;*

                    if(!s2.equals(temp)){

                        Collections.sort(s3)*;*

                        Collections.sort(s4)*;*

                        if(s3.size()>s4.size()){//去重

                            s3.remove(0)*;***//08:00:00在最前面**

                        }else if(s4.size()>s3.size()){

                            s4.remove(s4.size()-1)*;***//22:00:00**

                        }

                        for(int i**=**0*;i<s3.size();i++){*

                            System.out.println(temp+"\t\t"+s3.get(i)+"\t"+s4.get(i))*;*

                        }

                        s3.clear()*;*

                        s4.clear()*;*

                        s3.add("08:00:00")*;*

                        s4.add("22:00:00")*;*

                        temp**=**s2*;*

                    }

                    s3.add(rs.getString(2))*;*

                    s4.add(rs.getString(3))*;*

                } while (rs.next())*;*

**3.信息修改**

**(1)** **修改设备所在实验室相关信息**

ArrayList<String> s = new ArrayList<>()*;*

String s1 = new String(textField1.getText())*;***//设备号**

String s2 = new String(textField3.getText())*;***//实验室名称**

String s3 = new String(textField4.getText())*;***//实验室最大容纳人数**

s.add("UPDATE LAB SET LNAME = "+s2+" WHERE LNO = (SELECT LNO FROM DEVICE WHERE DNO = "+s1+")")*;*

s.add("UPDATE LAB SET MAXNUM = "+s3+" WHERE LNO = (SELECT LNO FROM DEVICE WHERE DNO = "+s1+")")*;*

**(2)** **修改已录入的实验室相关信息**

ArrayList<String> s = new ArrayList<>()*;*

String s1 = new String(textField1.getText())*;***//实验室号**

String s2 = new String(textField2.getText())*;***//实验室名称**

String s3 = new String(textField3.getText())*;***//实验室最大容纳人数**

String s4 = new String(textField4.getText())*;***//实验室用途**

s.add("UPDATE LAB SET LNAME = "+s2+" WHERE LNO ="+s1)*;*

s.add("UPDATE LAB SET MAXNUM = "+s3+" WHERE LNO ="+s1)*;*

s.add("UPDATE LAB SET PURPOSE = "+s4+" WHERE LNO ="+s1)*;*

**(3)** **修改已录入的教师相关信息**

ArrayList<String> s = new ArrayList<>()*;*

String s1 = new String(textField1.getText())*;***//教师号**

String s2 = new String(textField2.getText())*;***//教师姓名**

String s3 = new String(textField3.getText())*;***//教师电话**

String s4 = new String(textField4.getText())*;***//所在教研室**

s.add("UPDATE TEACHER SET TNAME = "+s2+" WHERE TNO ="+s1)*;*

s.add("UPDATE TEACHER SET PHONE = "+s3+" WHERE TNO ="+s1)*;*

s.add("UPDATE TEACHER SET DEPT = "+s3+" WHERE TNO ="+s1)*;*

**4.信息删除**

**(1)** **按照设备号删除设备信息**

String s = new String(textField1.getText())*;***//设备号**

String sql = "DELETE FROM DEVICE WHERE DNO ="+s*;*

**(2)** **按照教师编号删除教师相关信息**

String s = new String(textField1.getText())*;***//教师号**

String sql = "DELETE FROM TEACHER WHERE TNO ="+s*;*

**(3)** **按照实验室号删除实验室相关信息**

String s = new String(textField1.getText())*;***//实验室号**

String sql = "DELETE FROM LAB WHERE LNO ="+s*;*

**(4)** **按日期删除某教师某实验室的预约记录**

String s1 = new String(textField1.getText())*;***//日期**

String s2 = new String(textField2.getText())*;***//教师号**

String s3 = new String(textField3.getText())*;***//实验室号**

String sql = "DELETE FROM BOOK WHERE TNO = "+s2+" AND LNO = "

+s3+" AND DATE(BEGINTIME)<=>"+s1*;*

**(5)** **按照年份删除实验室预约记录**

String s = new String(textField1.getText())*;***//年份**

String sql = "DELETE FROM BOOK WHERE YEAR(BEGINTIME) <=>"+s*;*

**5.信息统计**

**统计各实验室被预约的总时长(按年或按月)**

我们把总时长先转换成分钟，再按照分钟和月份年份的转换关系来计算，这里默认每个月30天。如果直接使用TIMESTAMPDIFF函数进行月份和年份计算的话，那么不足一个月的情况结果就为0了。

//按照年份

String sql="SELECT LNO,SUM(TIMESTAMPDIFF(MINUTE,BEGINTIME,ENDTIME))/43200/12 YEAR " +

"FROM BOOK GROUP BY LNO"*;*

//按照月份

String sql="SELECT LNO,SUM(TIMESTAMPDIFF(MINUTE,BEGINTIME,ENDTIME))/43200 MONTH " +

"FROM BOOK GROUP BY LNO"*;*

**6.登录**

**设计登录窗口，验证教师登录账号和密码**

这里将用户输入的账号和密码与数据库内存储的进行比对，如果却有此账号密码，则可以登录，否则显示账号密码输入不正确。并且登录之后才可以进行账号密码的修改。同时使用SWING中的密码框，在用户输入的时候不显示真实密码，以保障用户安全。

private void button1MouseClicked(MouseEvent e) {

        // TODO add your code here

        Connection connection = null*;*

        Statement statement = null*;*

        ResultSet resultSet = null*;*

        String s1 = new String(textField1.getText())*;***//账号**

        String s2 = new String(passwordField1.getPassword())*;***//密码**

        boolean flag = false*;*

        try {

            connection = jdbcUtil.openConnection()*;*

            String sql = "SELECT NUMBER,PWD FROM ADMIN"*;*

            statement = connection.createStatement()*;*

            resultSet = statement.executeQuery(sql)*;*

            while (resultSet.next()){

                if(s1.equals(resultSet.getString(1))&&s2.equals(resultSet.getString(2))){

                    flag = true*;*

                    break*;*

                }

            }

        } catch (Exception e1) {

            e1.printStackTrace()*;*

        }finally{

            try {

                if (resultSet != null){

                    resultSet.close()*;*

                }

                if (statement != null) {

                    statement.close()*;*

                }

                if (connection != null) {

                    connection.close()*;*

                }

            } catch (SQLException e2) {

                e2.printStackTrace()*;*

            }

        }

        if(flag){

            JOptionPane.showMessageDialog(null,"登录成功")*;*

            //this.setVisible(false)*;*

            new menu().setVisible(true)*;*

        }else{

            JOptionPane.showMessageDialog(null,"账号或密码不正确")*;*

        }

    }

**7.密码维护**

**对教师账户密码的修改**

用户先输入教师号，原账号和密码，如果存在的话方可进行账号和密码的修改。账号密码可以只修改其中之一。

private void button1MouseClicked(MouseEvent e) {

        // TODO add your code here

        Connection connection = null;

        Statement statement = null;

        String s1 = new String(textField1.getText());

        String s2 = new String(passwordField1.getPassword());

        String s5 = new String(textField3.getText());

        boolean flag = false;

        ResultSet resultSet = null;

        try {

                connection = jdbcUtil.openConnection();

                String sql = "SELECT NUMBER,PWD,TNO FROM ADMIN";

                statement = connection.createStatement();

                resultSet = statement.executeQuery(sql);

                while (resultSet.next()){

                    if(s1.equals(resultSet.getString(1))&&s2.equals(resultSet.getString(2))

                            &&s5.equals(resultSet.getString(3))){

                        flag = true;

                        break;

                    }

                }

            } catch (Exception e1) {

                e1.printStackTrace();

            }finally{

                try {

                    if (resultSet != null){

                        resultSet.close();

                    }

                    if (statement != null) {

                        statement.close();

                    }

                    if (connection != null) {

                        connection.close();

                    }

                } catch (SQLException e2) {

                    e2.printStackTrace();

                }

            }

        String s3 = new String(textField2.getText());

        String s4 = new String(passwordField2.getPassword());

        ArrayList<String> s = new ArrayList<>();

        s.add("UPDATE ADMIN SET NUMBER ="+s3+" WHERE NUMBER ="+s1);

        s.add("UPDATE ADMIN SET PWD ="+s4+" WHERE PWD ="+s2);

        try {

            //获取数据连接

            connection = jdbcUtil.openConnection();

            //获取发送sql指令执行sql对象

            statement = connection.createStatement();

            if(flag==false) {

                JOptionPane.showMessageDialog(null, "教师号或原账号或密码不正确请重新输入");

                this.setVisible(false);

            }else{

                statement.addBatch(s.get(0));

                statement.addBatch(s.get(1));

                int[] success = statement.executeBatch();  //批量执行所有sql返回一个更新计数的数组

                JOptionPane.showMessageDialog(null, "修改成功");

                this.setVisible(false);

            }

            //执行成功返回更新计数的数组

        } catch (Exception e1) {

            e1.printStackTrace();

        } finally {

            //执行完数据库操作后记得关闭数据库连接资源

            try {

                statement.close();

                connection.close();

            } catch (SQLException e2) {

                e2.printStackTrace();

            }

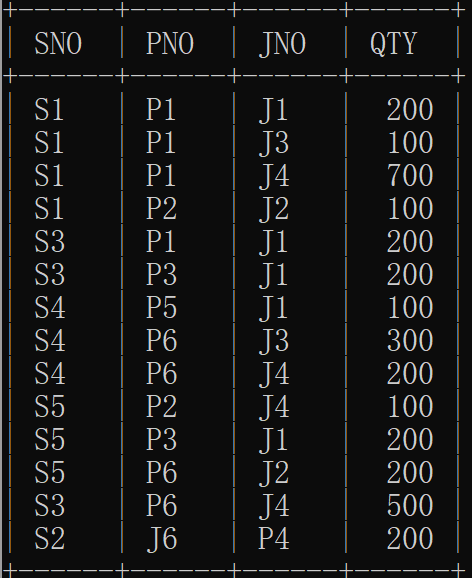
        }

    }

**五.实验结果**

**课后习题**

**SPJ表**

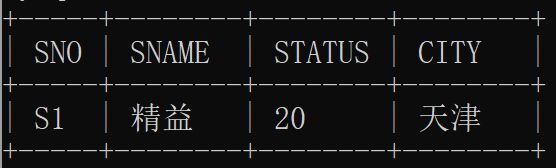


**S表**

****

示例：

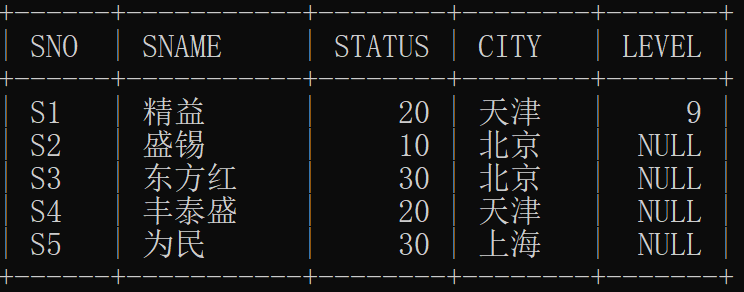
CALL TEST('S1');



将S1,P1,J1 对应的供应总量更新为0时，S1对应的供应总量为900，所以level值为9;

UPDATE SPJ SET QTY=0 WHERE SNO='S1' AND PNO='P1' AND JNO='J1';

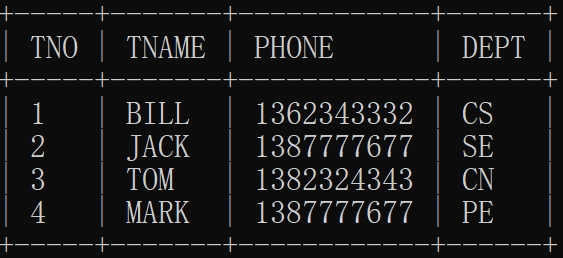
SELECT \* FROM S;



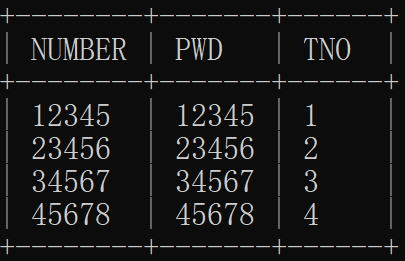
**综合性作业**

首先展示每一个表的初始状态，由于实验表中的图片用MySQL展示并不直观并且16进制字符流篇幅过长，所以选择用Navicat进行展示。

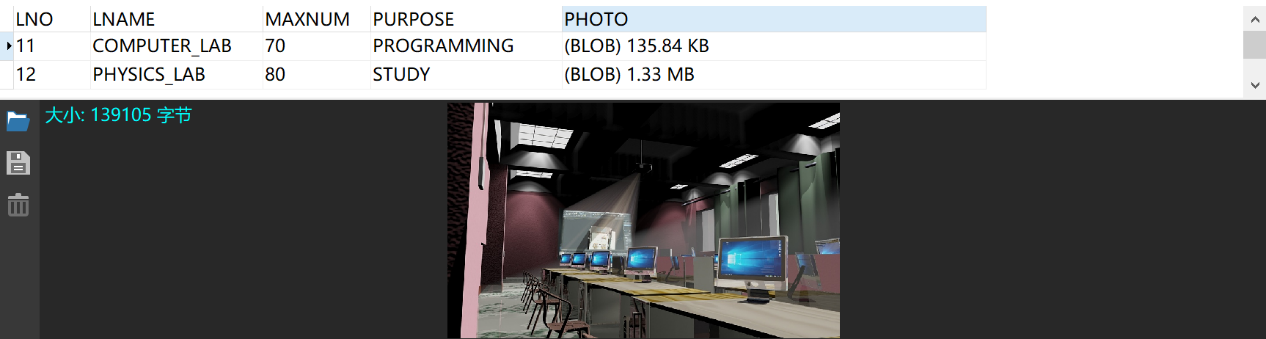
**教师表**

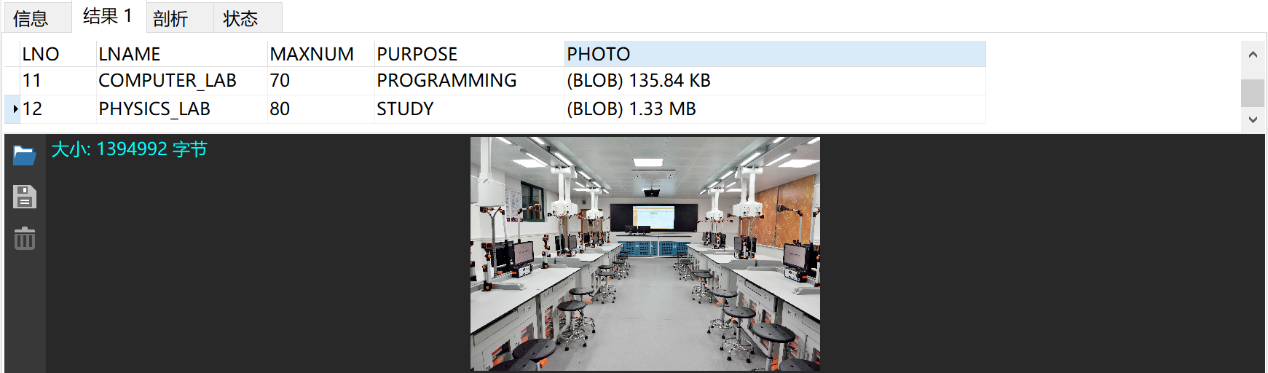


**管理员表**

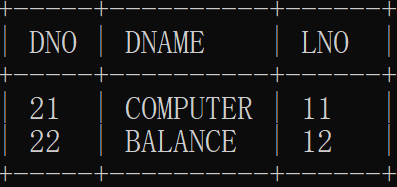


**实验室表(上图为电脑机房，下图为物理实验室)**

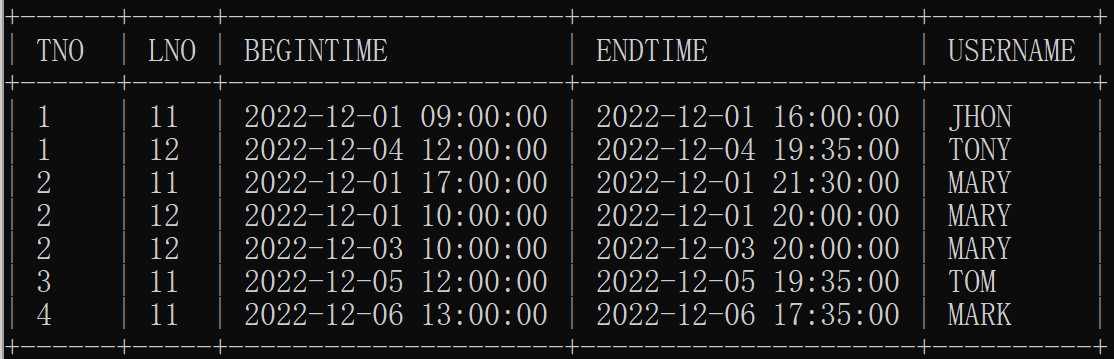




**设备表**



**预约表**



**图形化界面展示**

**设计登录窗口，验证教师登录账号和密码**

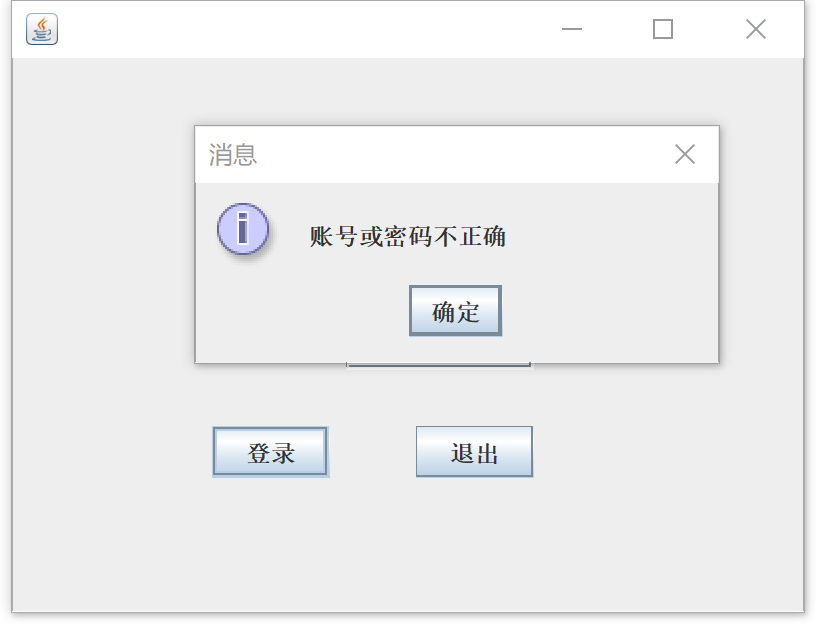
运行idea中的login.java函数弹出登录界面。



首先输入错误账号和密码



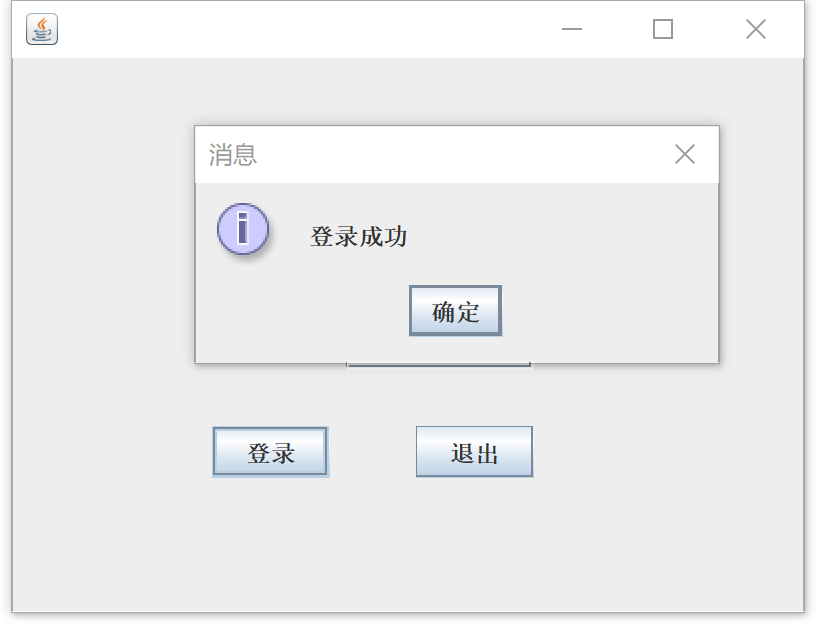
这时点击登录会显示错误信息



接下来输出正确信息



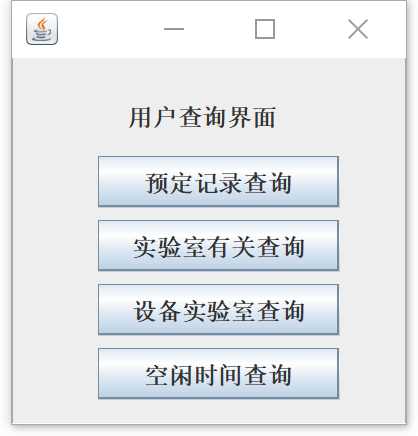
点击登录显示登录成功

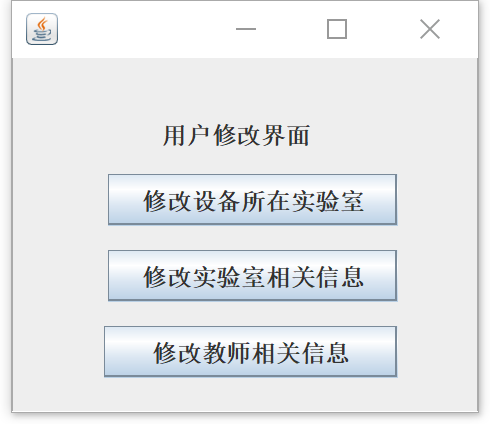


点击确定之后会显示主菜单

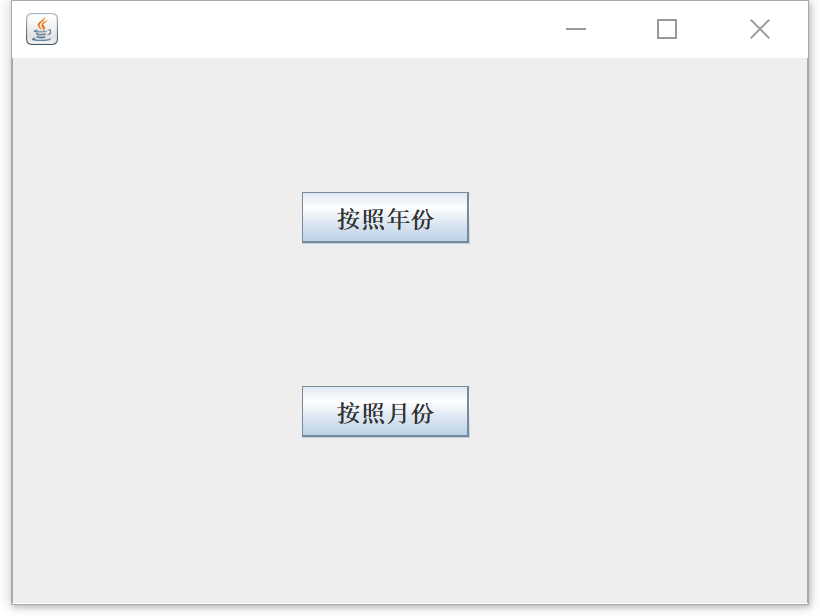


分别点击这5个按键会显示如下各图











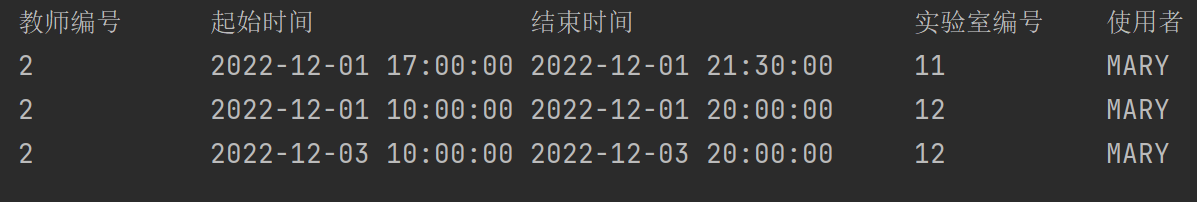
接下来逐一验证每个模块的功能

**查询模块**

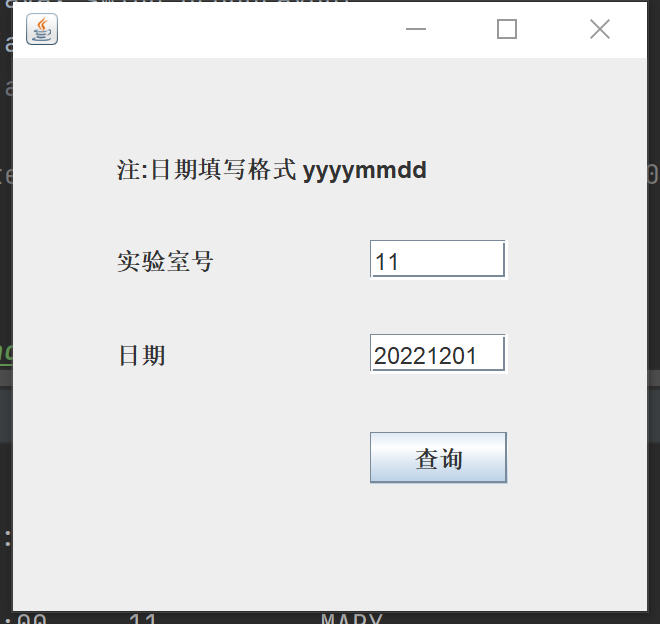
**(1)通过教师编号或教师名查询该教师某一时间段内的预定记录(教**

**师编号，预定起止时间，预定实验室，使用者)**



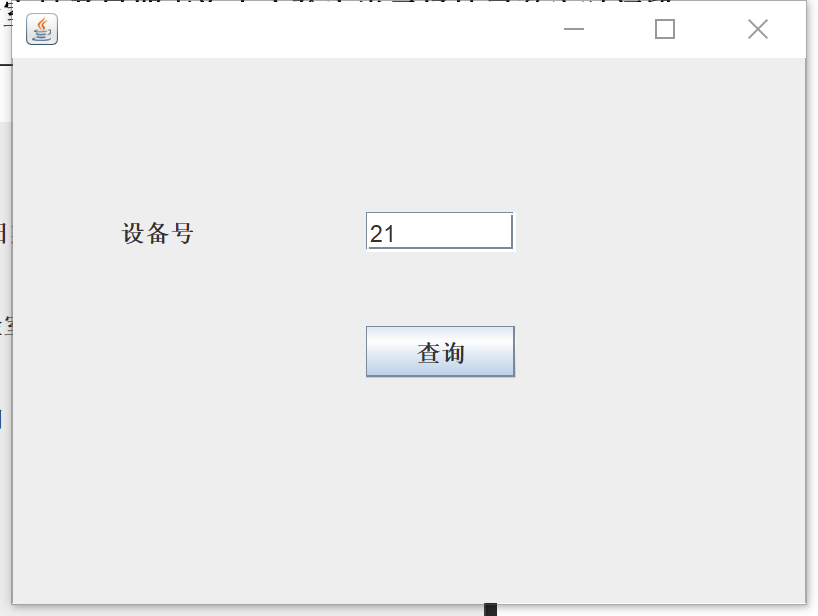


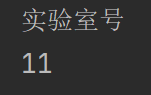
(2)**通过实验室号及日期查询本实验室当天场地已预定时间段**



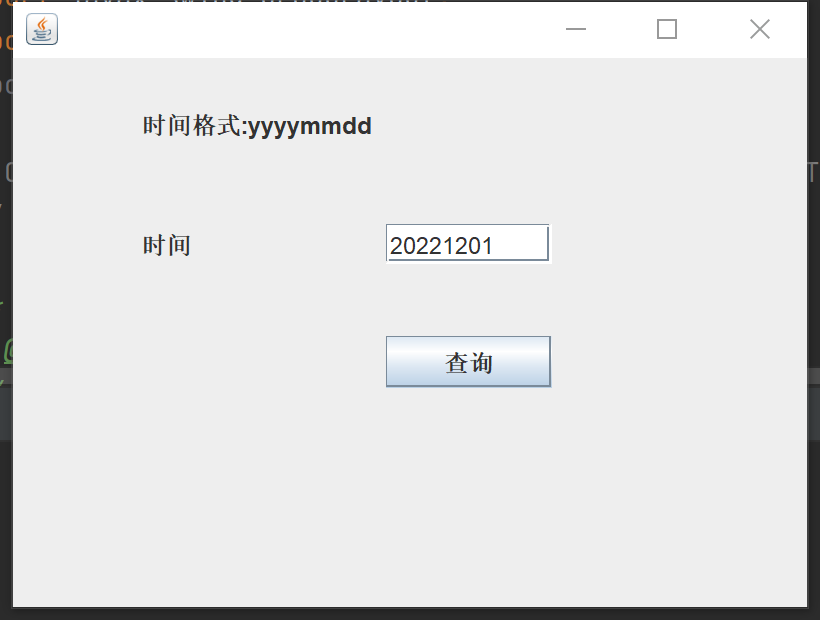


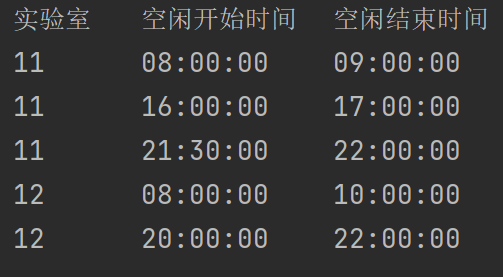
**(3)** **通过设备号查询设备所在实验室**





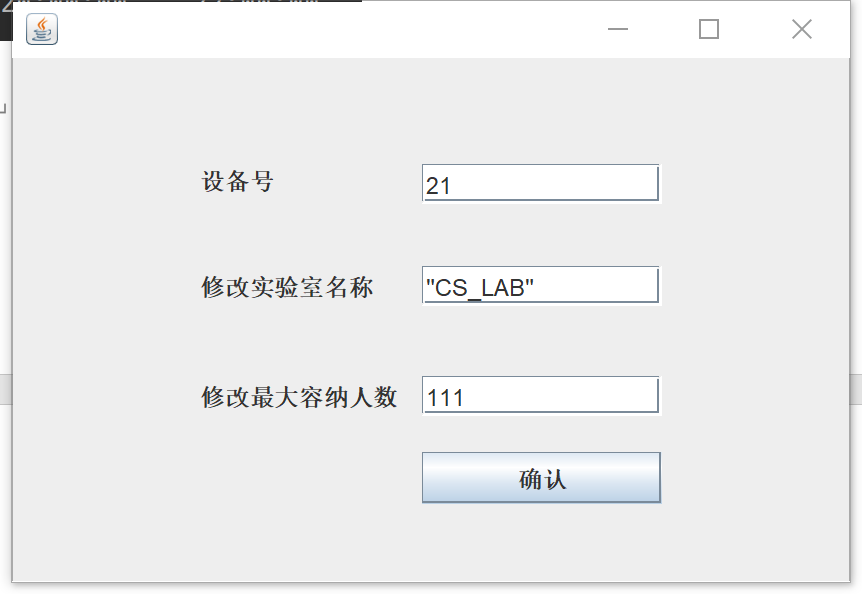
**(4)** **按照年月日查询所有实验室的空闲时间段**

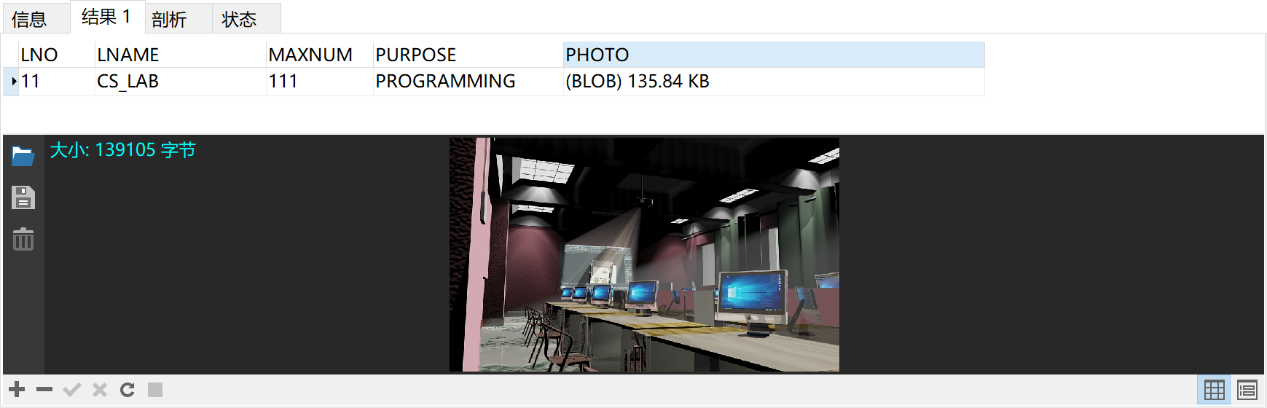




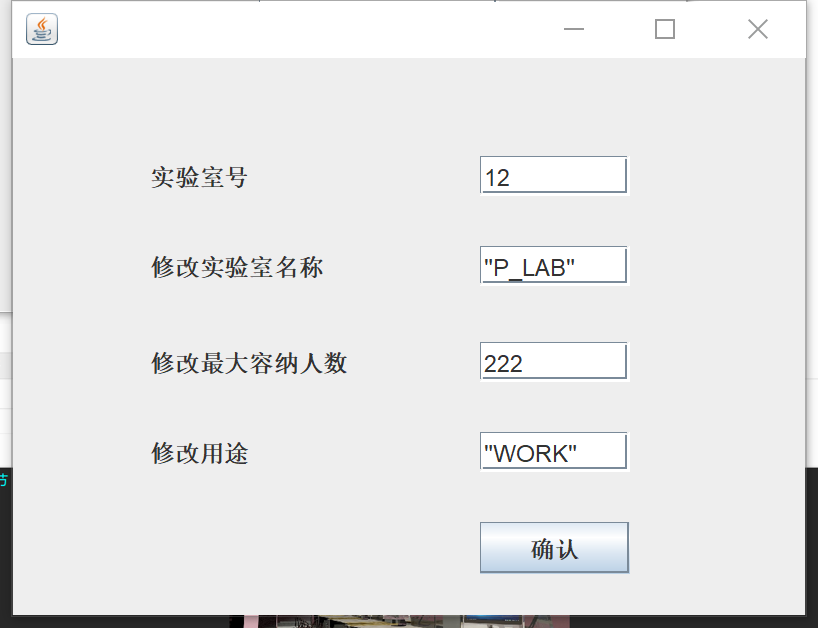
**修改模块**

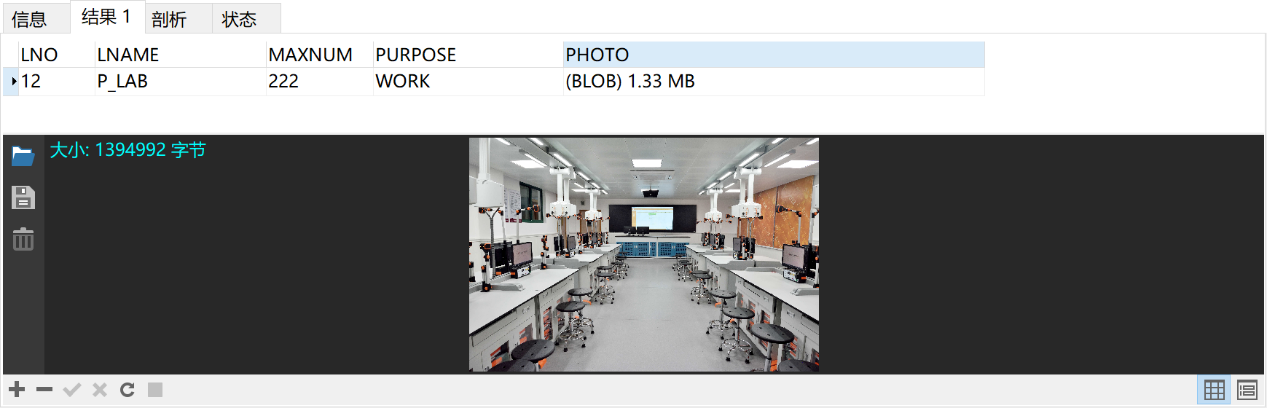
**(1)** **修改设备所在实验室相关信息**

****

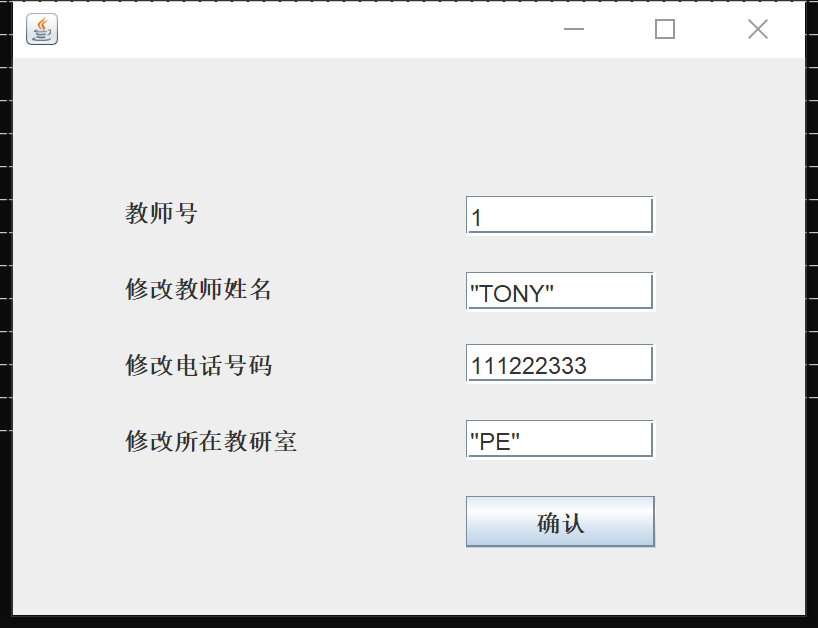
****

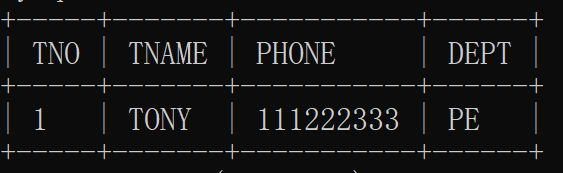
**(2) 修改已录入的实验室相关信息**

****

****

**(3)** **修改已录入的教师相关信息**

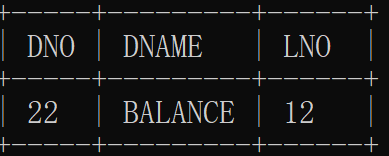
****

****

**删除模块**

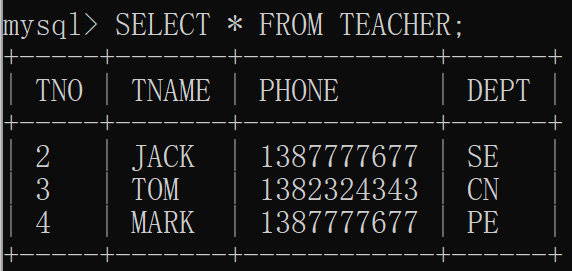
**(1)** **按照设备号删除设备信息**

****

****

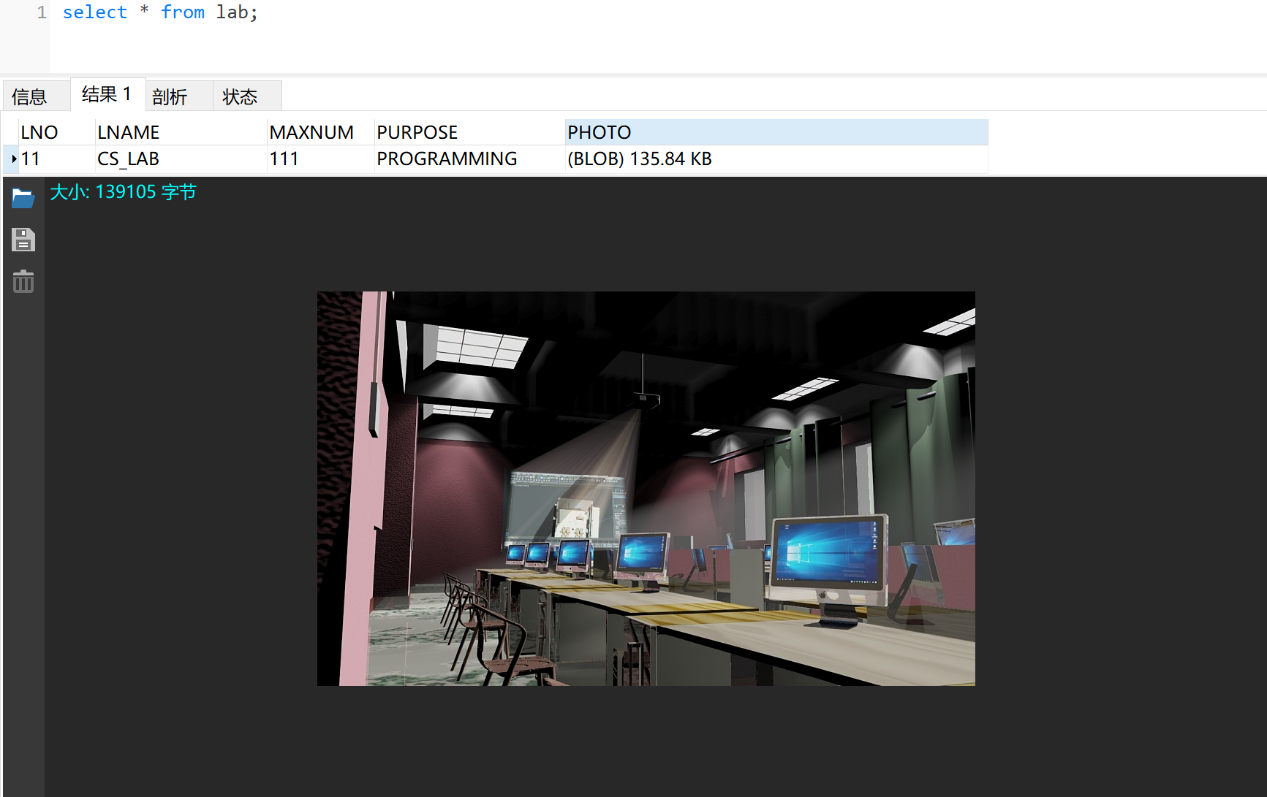
**(2)** **按照教师编号删除教师相关信息**

****

****

**(3) 按照实验室号删除实验室相关信息**

****

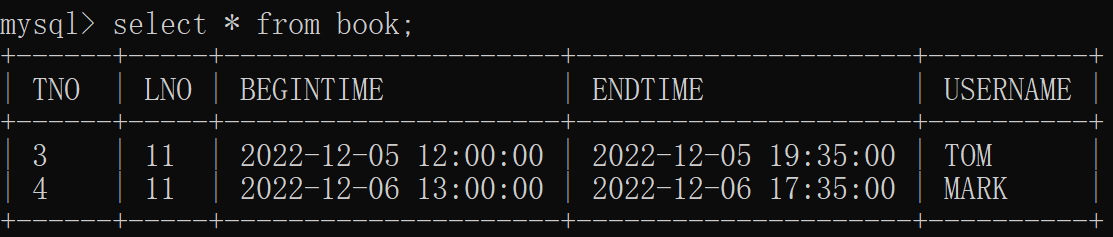
****

**(4) 按日期删除某教师某实验室的预约记录**

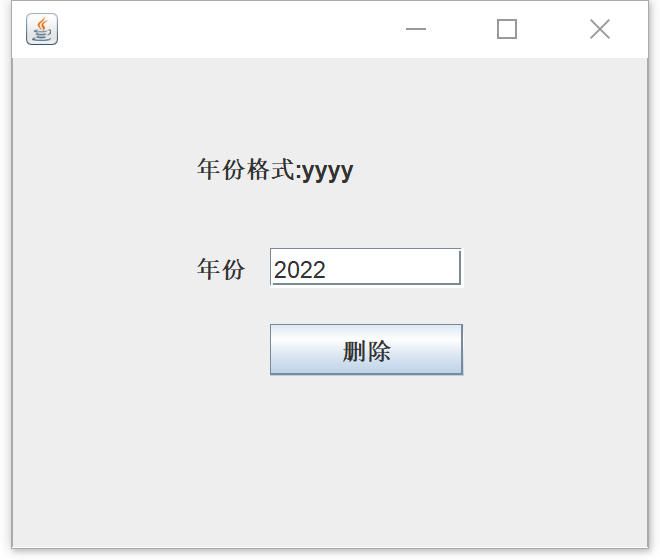
**删除前：(由于之前删除了实验室和教师相关信息，所以相关预定记录也自动删除了)**

****

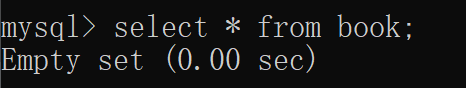
**删除后：**

****

**(5)** **按照年份删除实验室预约记录**

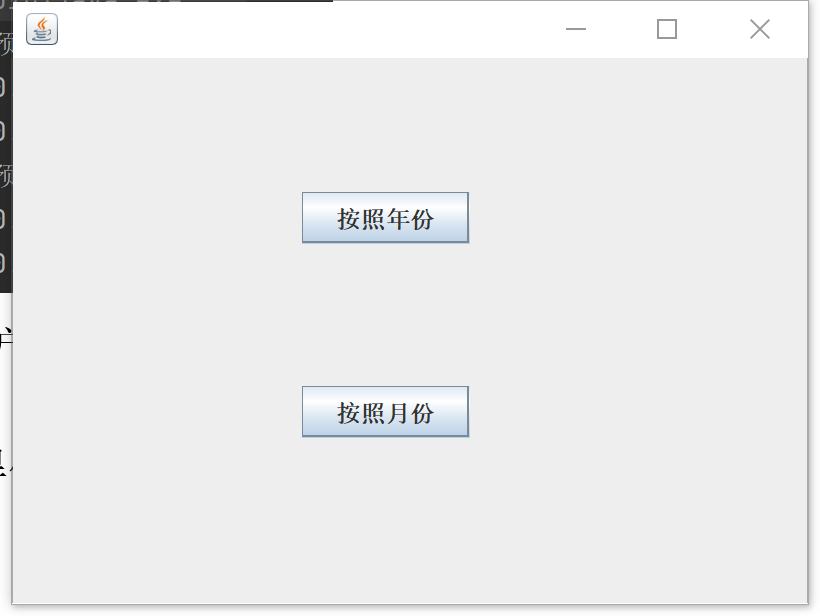
****

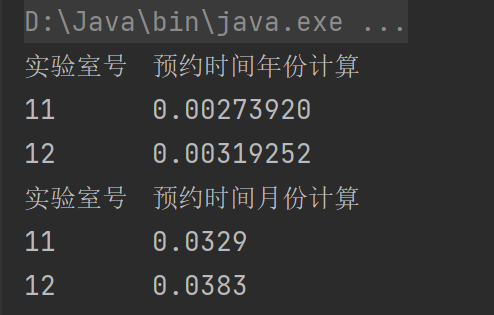
由于预约记录都是2022年的所以删除之后预约表清空：

****

**统计各实验室被预约的总时长(按年或按月)**

**注：由于刚刚把预约表清空了，所以在执行此项操作之前，我把之前每个表删除的内容都恢复了。**

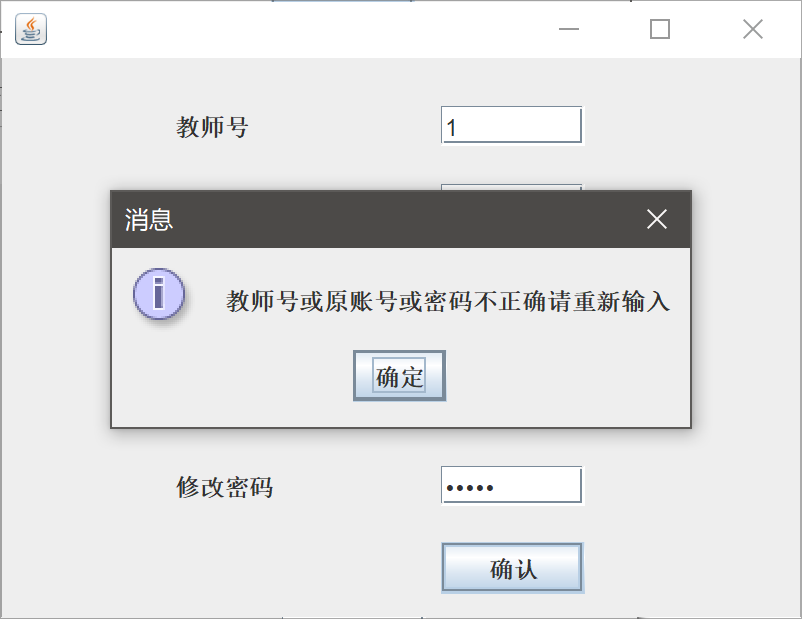
****

****

**对教师账户密码的修改**

首先输入错误账号密码，会有弹窗提示



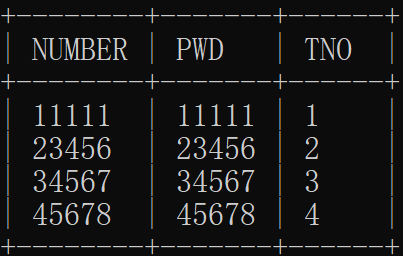


接下来输入正确账号和密码





查询修改结果



尝试登录显示成功

