实验七 综合实验

实验日期： 2017 年 5 月 5 日 班级： 软件1501

学号（后四位）： 0314 姓名： 马亚卿 成绩：

1. 实验目的

1 了解JDBC的作用，掌握通过JDBC访问数据库的方法。

2 能够实现对数据库中数据的添加、删除、修改和查询。

1. 实验内容（以学生信息管理为例，可以自拟XXX信息管理系统，实现类似的功能即可）

**注意**：在写报告时，不需要写数据库连接的四个参数，也不需要写Class.forName(......)，只要写上“Connection con=....”就行。其它的不能省略。

1.建立一个类，利用数据库来存储多个Student，写完一个方法在main中写一段测试代码，运行以保证目前所做工作的正确性。有以下方法：

a add(Student stu)：可以向其中增加新的学生，并保存在数据库中。

测试add方法是否正确：用add方法向数据库增加一个新的学生，然后在数据库的图形管理界面中查询，确认是否增加。

b dispAll()：可以显示所有的学生信息。

c findById(long id)：可以按照学号来查找，并显示符合条件的学生信息，查无该人的话显示错误信息。

d findByName(String name)：可以按照姓名查找学生，找到后显示其信息，查无此人显示错误信息。

e delById(long id)：可以按照id删除学生的信息，然后显示找到该人。若查无此人，显示相应的错误信息。

2 （选做）以图形界面的方式实现上述系统，并实现相应的功能。菜单如下：

1 显示所有学生信息 2 按学号查找 3 按姓名查找

4 按学号删除 5按成绩排序 6 退出

图书信息管理系统

部分关于数据库操作的代码：

* 1. 检查登录信息的代码：

|  |
| --- |
| package Login;  import java.sql.\*;  import javax.swing.JOptionPane;  public class LoginCheck {  String driver = "org.hsqldb.jdbcDriver";  String url = "jdbc:hsqldb:hsql://localhost";  String user = "sa";  String pass = "";  private Connection con = null;  public LoginCheck() throws ClassNotFoundException, SQLException{  init();  }  public void init() throws ClassNotFoundException, SQLException{  if(con!=null)  return;  Class.forName(driver);  con = DriverManager.getConnection(url,user,pass);  }  public boolean CheckStudent(String id,String pass) throws SQLException{  String sql = "select \* from student where id=\'" + id + "\'";  Statement stat = con.createStatement();  ResultSet rs = stat.executeQuery(sql);    if(rs.next()){  String passwd = rs.getString("pass");  close();  return passwd.equals(pass);  }  else{  JOptionPane.showMessageDialog(null, "用户不存在", "错误信息",JOptionPane.WARNING\_MESSAGE);  close();  return false;  }    }  public boolean CheckTeacher(String id,String pass) throws SQLException{  String sql = "select \* from teacher where id=\'" + id + "\'";  Statement stat = con.createStatement();  ResultSet rs = stat.executeQuery(sql);  if(rs.next()){  String passwd = rs.getString("pass");  close();  return passwd.equals(pass);  }  else{  JOptionPane.showMessageDialog(null, "用户不存在", "错误信息",JOptionPane.WARNING\_MESSAGE);  close();  return false;  }    }  public void close() throws SQLException{  if(con!=null)  con.close();  }  } |

2对图书信息的数据库进行操作的代码：（由于有图形界面，故只列出部分）

|  |
| --- |
| package Login;  import java.sql.\*;  public class BookSql {  String driver = "org.hsqldb.jdbcDriver";  String url = "jdbc:hsqldb:hsql://localhost";  String user = "sa";  String pass = "";  private Connection con;  public BookSql() throws ClassNotFoundException, SQLException{  init();  }  public void init() throws ClassNotFoundException, SQLException{  if(con!=null)  return;  Class.forName(driver);  con = DriverManager.getConnection(url,user,pass);  }  public void query(){  try {  Statement stat = con.createStatement();  String sql = "select \* from book";  ResultSet rs = stat.executeQuery(sql);  while(rs.next()){  String name = rs.getString("name");  System.out.println(name);  String press = rs.getString("press");  System.out.println(press);  }  } catch (SQLException e) {  e.printStackTrace();  }  }  public String getName(String id){  Statement stat;  try {  stat = con.createStatement();  String sql = "select \* from book where id = '" + id + "'";  ResultSet rs = stat.executeQuery(sql);  rs.next();  // System.out.println(rs.getString("name"));  return rs.getString("name");  } catch (SQLException e) {    }  return null;    }  public int getBorrowed(String id){  Statement stat;  try {  stat = con.createStatement();  String sql = "select \* from book where id = '" + id + "'";  ResultSet rs = stat.executeQuery(sql);  rs.next();  // System.out.println(rs.getString("name"));  return rs.getInt("borrowed");  } catch (SQLException e) {    }  return 0;  }  public int getBorrow(String id){  Statement stat;  try {  stat = con.createStatement();  String sql = "select \* from book where id = '" + id + "'";  ResultSet rs = stat.executeQuery(sql);  rs.next();  // System.out.println(rs.getString("name"));  return rs.getInt("borrow");  } catch (SQLException e) {    }  return 0;  }  public String[][] search(){  String sql = "select \* from book";  return getString(sql);  }  public String[][] getString(String sql){  String [][]s = null;  try {  Statement stat = con.createStatement(ResultSet.TYPE\_SCROLL\_INSENSITIVE,  ResultSet.CONCUR\_READ\_ONLY);  ResultSet rs = stat.executeQuery(sql);  ResultSetMetaData rsmd = rs.getMetaData();  int col = rsmd.getColumnCount();  rs.last();  int row = rs.getRow();  rs.beforeFirst();    int index = 0;  if(row>0){  s = new String[row][col];  while(rs.next()){  for(int i = 0;i<col;i++){  s[index][i] = rs.getString(i+1);    }  s[index][6] = rs.getInt(7) + "";  s[index][7] = rs.getInt(8) + "";  s[index][8] = rs.getDouble(9) + "";  index++;  }  }    } catch (SQLException e) {    }  return s;  }  public String[][] searchBy(String mode,String name){  String sql = "select \* from book where "+mode+" like '" + name +"'";  return getString(sql);  }  /\*-1 失败 1 成功 0 已存在\*/  public int add(String [] s){  String str = "select \*from book where id = ?";    try {  PreparedStatement ps1 = con.prepareStatement(str);  ps1.setString(1, s[0]);  ResultSet rs = ps1.executeQuery();  ps1.close();  if(rs.next())  return 0;  } catch (SQLException e1) {  return -1;  }  String sql = "insert into book values(?,?,?,?,?,?,?,?,?)";  try {  PreparedStatement ps = con.prepareStatement(sql);  ps.setString(1, s[0]);  ps.setString(2, s[1]);  ps.setString(3, s[2]);  ps.setString(4, s[3]);  ps.setString(5, s[4]);  ps.setString(6, s[5]);  ps.setInt(7, Integer.parseInt(s[6]));  ps.setInt(8, Integer.parseInt(s[7]));  ps.setDouble(9, Double.parseDouble(s[8]));  ps.executeUpdate();  ps.close();  } catch (SQLException e) {  return -1;  }  return 1;  }  public boolean delete(String id) {  String sql = "delete from book where id = ?";  PreparedStatement ps;  try {  ps = con.prepareStatement(sql);  ps.setString(1, id);  ps.executeUpdate();  ps.close();  } catch (SQLException e) {  try {  con.rollback();  } catch (SQLException e1) {    }  return false;  }  return true;  }  public boolean update(String[] s) {  String sql = "update book set name = ? , press = ? , author = ?,type = ?,time = ?,";  sql += "borrow = ?,borrowed = ?,price = ? where id = ?";  try {  PreparedStatement ps = con.prepareStatement(sql);  for(int i = 1;i<6;i++){  ps.setString(i, s[i]);  }  ps.setInt(6, Integer.parseInt(s[6]));  ps.setInt(7, Integer.parseInt(s[7]));  ps.setDouble(8, Double.parseDouble(s[8]));  ps.setString(9, s[0]);  ps.executeUpdate();  } catch (SQLException e) {  // TODO Auto-generated catch block  e.printStackTrace();  }  return false;  }  public String[] getAll() {  String []s = null;  String sql = "select \* from book";  return null;  }  } |

运行结果如图：

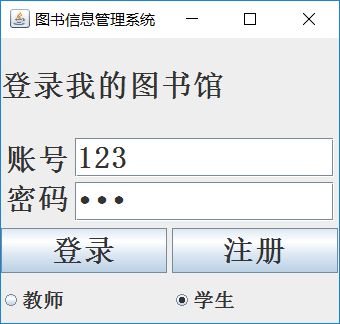
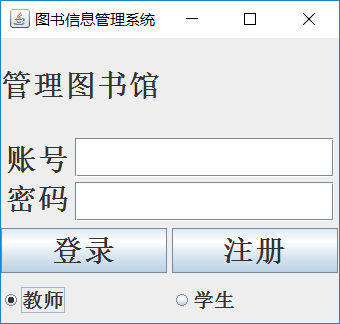
 

图1 学生登录界面 图2 教师登录界面

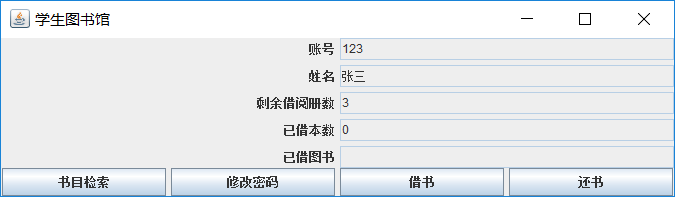


图3 学生菜单

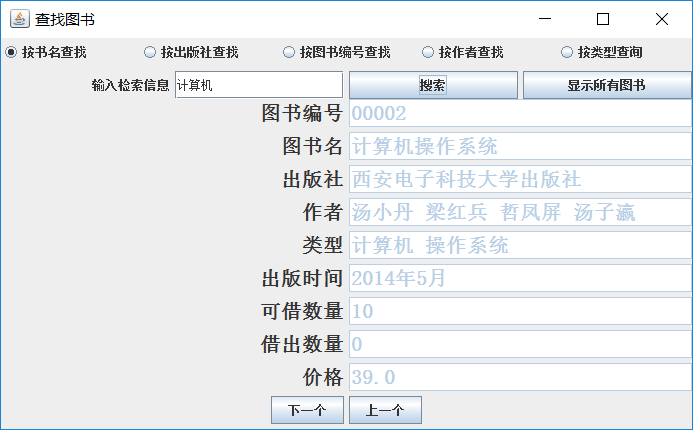


图4 图书查找界面

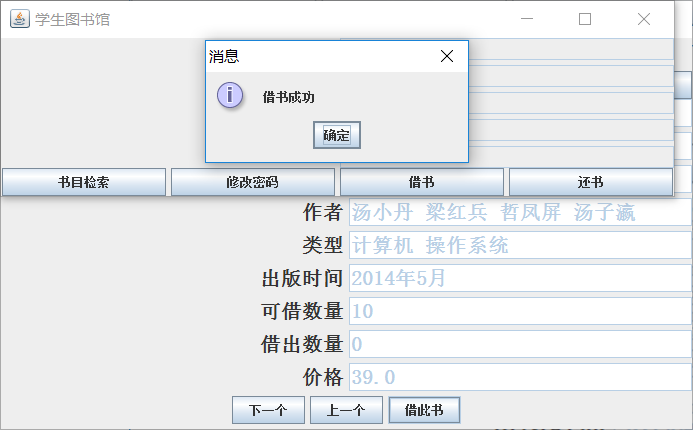
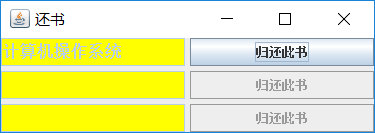


图5 借书操作

1. (b)

图6 还书操作

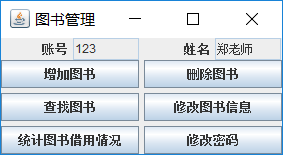
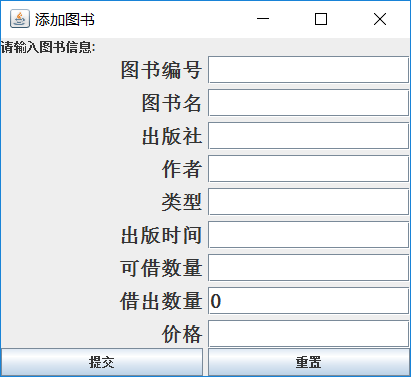
 

图7教师菜单 图8添加图书

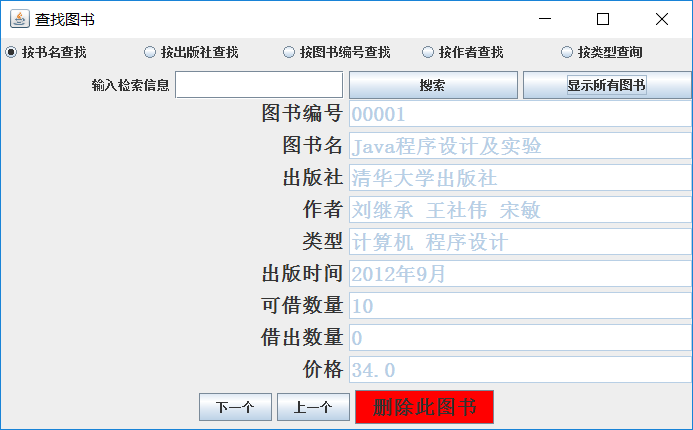


图9删除图书操作



图10 统计图书借出情况

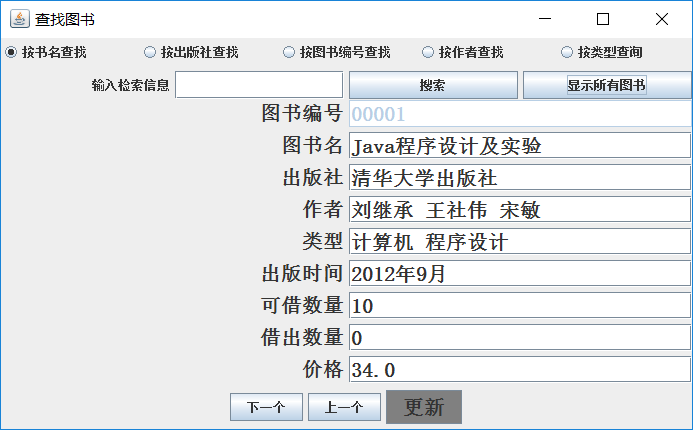


图11修改图书信息

1. 在本次实验中的问题和解决方法

问题1：总是打不开数据库。

解决办法：需要将数据库的服务器一直打开才能访问数据库。

问题2：在进行借书操作时，如果一直点击借书，图书信息出错。

解决办法：在每次借书之后都调用all.doClick()方法更新一下。

四．心得体会

这次试验让我初步了解了JDBC的作用，掌握了通过JDBC访问数据库的方法。能够实现对数据库中数据的添加、删除、修改和查询。感觉到了数据库操作的简洁性（以前是经过文件对数据的操作）。

这次试验还让我对图形用户界面的使用有了更加深入的认识。在写试验时，对图形用户界面的设计就占据了实验的大部分时间，但是效果却很不理想。过程繁琐而且容易出错。设计出来的界面没有条理，可读性差，在我修改时需要花费大量时间查找组件，没有可维护性。

今后要多多联系数据库，熟悉sql语句的使用。还要在图形用户界面上下功夫，争取写出有条理，可读性好的代码。