

实 验 报 告

**课程名称： 数据库应用原理**

**学 年： 2018-2019 第 1 学期**

**专业班级： 2016软件工程1班**

**姓 名： 党士麟**

**学 号： 201624133128**

**指导教师： 王敏琴**

**开始时间： 20181012**

**结束时间： 20181125**

**实验成绩：**

**目录**

**[1. 概述](#_Toc15774_WPSOffice_Level1)** **[2](#_Toc15774_WPSOffice_Level1)**

**[2. 代码结构](#_Toc5695_WPSOffice_Level1)** **[2](#_Toc5695_WPSOffice_Level1)**

[M&V部分](#_Toc15774_WPSOffice_Level2) [2](#_Toc15774_WPSOffice_Level2)

[C部分](#_Toc5695_WPSOffice_Level2) [3](#_Toc5695_WPSOffice_Level2)

**[3. ER图](#_Toc28860_WPSOffice_Level1)** **[3](#_Toc28860_WPSOffice_Level1)**

[提示：具体操作详细见 本文档所在文件夹的 《图书商城使用说明书》中](#_Toc28860_WPSOffice_Level2) [5](#_Toc28860_WPSOffice_Level2)

**[4. 详细代码for User](#_Toc5975_WPSOffice_Level1)** **[6](#_Toc5975_WPSOffice_Level1)**

[（1） 链接数据库](#_Toc5975_WPSOffice_Level2) [6](#_Toc5975_WPSOffice_Level2)

[（2） 有关字体的过滤器](#_Toc6078_WPSOffice_Level2) [7](#_Toc6078_WPSOffice_Level2)

[（3） 商场主页](#_Toc13972_WPSOffice_Level2) [8](#_Toc13972_WPSOffice_Level2)

[（4） 购书](#_Toc17381_WPSOffice_Level2) [9](#_Toc17381_WPSOffice_Level2)

[（5） 购物车](#_Toc16553_WPSOffice_Level2) [10](#_Toc16553_WPSOffice_Level2)

[（6） 订单](#_Toc30118_WPSOffice_Level2) [11](#_Toc30118_WPSOffice_Level2)

[记录到数据库中并且更新书本的库存](#_Toc18563_WPSOffice_Level2) [13](#_Toc18563_WPSOffice_Level2)

**[5. 详细代码for Manger](#_Toc6078_WPSOffice_Level1)** **[13](#_Toc6078_WPSOffice_Level1)**

[（1） 用户管理页面](#_Toc24414_WPSOffice_Level2) [13](#_Toc24414_WPSOffice_Level2)

[（2） 书本管理](#_Toc32166_WPSOffice_Level2) [17](#_Toc32166_WPSOffice_Level2)

**[6. 数据库](#_Toc13972_WPSOffice_Level1)** **[21](#_Toc13972_WPSOffice_Level1)**

[数据库详细在本文档所在文件夹的 bookshop.sql文件中](#_Toc1717_WPSOffice_Level2) [21](#_Toc1717_WPSOffice_Level2)

[Book](#_Toc23172_WPSOffice_Level2) [21](#_Toc23172_WPSOffice_Level2)

[messager](#_Toc533_WPSOffice_Level2) [21](#_Toc533_WPSOffice_Level2)

[Useroder （购物清单）](#_Toc4760_WPSOffice_Level2) [21](#_Toc4760_WPSOffice_Level2)

**[7. 补充说明(登陆)](#_Toc17381_WPSOffice_Level1)** **[22](#_Toc17381_WPSOffice_Level1)**

**[8. 心得总结](#_Toc16553_WPSOffice_Level1)** **[24](#_Toc16553_WPSOffice_Level1)**

**实验名称:简易购书商城**

1. **概述**

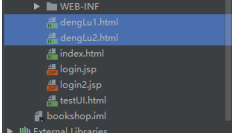
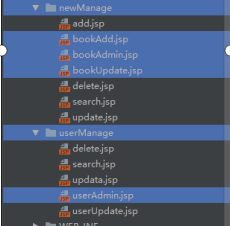
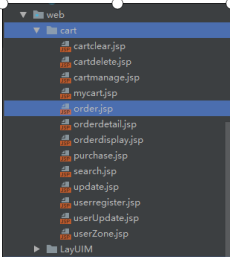
使用工具：IntelliJ IDEA 2018.2.4,navicat Premium 12，processon（UML作图工具）；

开发环境：win5/win7；

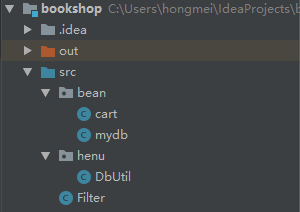
编写目的：实现简单的购书商场；

1. **代码结构**

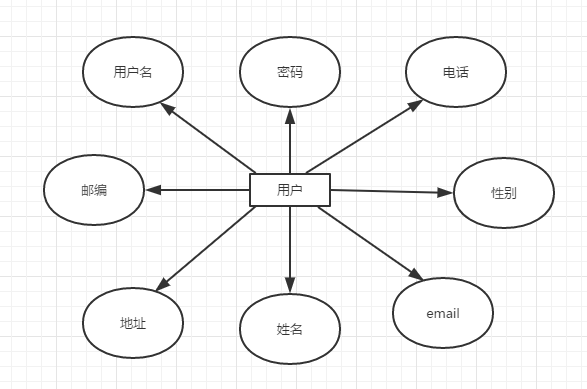
M&V部分



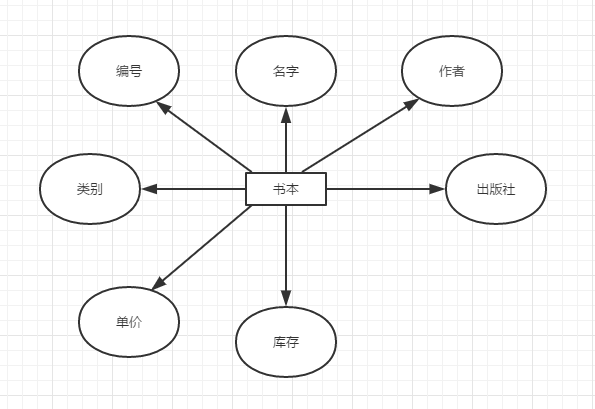
C部分



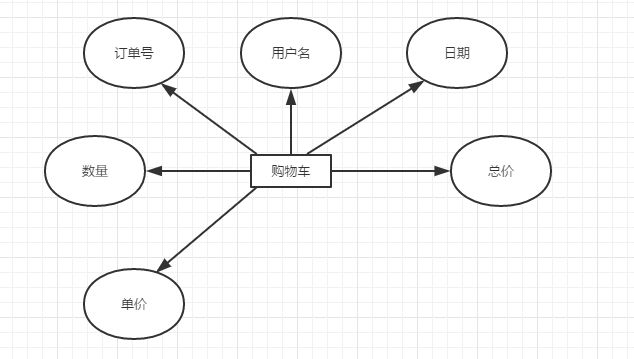
1. **ER图**



用户登陆时首先是用到 **用户名** 和 **密码**；因为为购物网站，**电话**，**姓名**，**地址**，**邮编**必不可少；**email**用于账号的找回，**性别**为基本信息



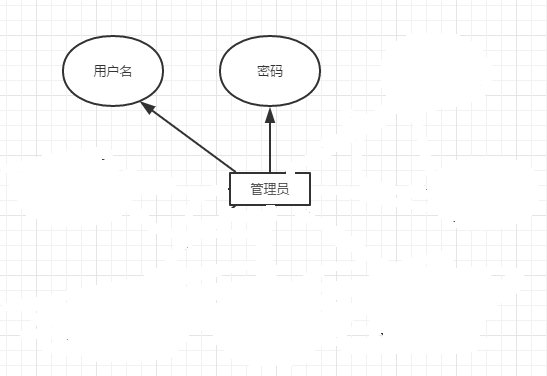
书本为本网站的商品：其主要包含了**名字**，**作者**，**出版社**，**编号**（即ISBN），**类别**，**单价**和**库存**



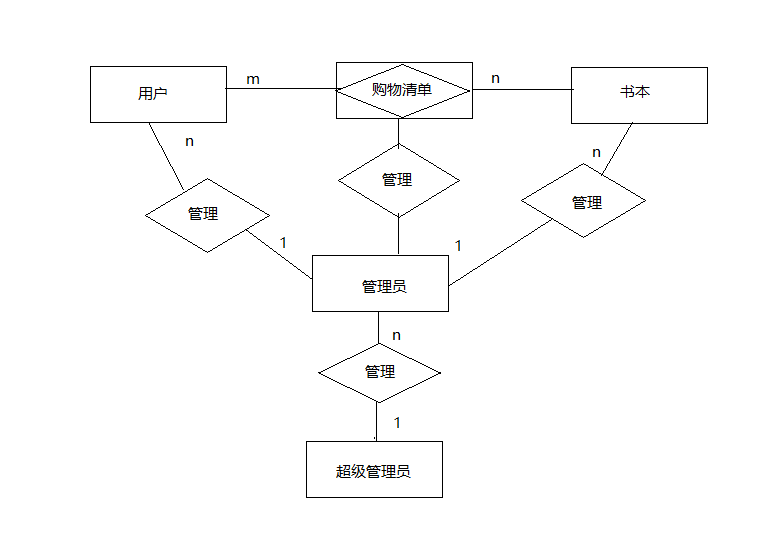
购物清单

***购物车收货地址默认为用户注册时填写的地址，用户可以通过修改个人信息修改送货地址；***

购物清单是用户购买时产生的：有**订单号**，**用户名**，**日期**，**数量**，**单价**和**总价**



管理员:管理员**用户名**，**密码**由超级管理员分配，负责管理书本的下架（删除），修改信息库存，上架（添加）的管理；



***提示：具体操作详细见 本文档所在文件夹的 《图书商城使用说明书》中***

1. **详细代码for User**
2. **链接数据库**

public class DbUtil {  
 private static final String *URL*="jdbc:mysql://localhost:3306/bookshop";  
 private static final String *USER*="root";  
 private static final String *PASSWORD*="dang0913";  
  
 protected static Statement *s*=null;  
 protected static ResultSet *rs*=null;  
 protected static Connection *conn*=null;  
  
 public static Connection getConnection(){  
 try{  
 Class.*forName*("com.mysql.jdbc.Driver");  
 *conn*= DriverManager.*getConnection*(*URL*,*USER*,*PASSWORD*);  
 }catch (Exception e){  
 e.printStackTrace();  
 }  
 return *conn*;  
 }  
 public static int executeUpdate(String sql){  
 int result=0;  
 try {  
 *s*=*getConnection*().createStatement();  
 result= *s*.executeUpdate(sql);  
 }catch (SQLException e){  
 e.printStackTrace();  
 }  
 return result;  
 }  
 public static ResultSet executeQuery(String sql){  
 try{  
 *s*=*getConnection*().createStatement();  
 *rs*=*s*.executeQuery(sql);  
 }catch(SQLException e){  
 e.printStackTrace();  
 }  
 return *rs*;  
 }  
 public static PreparedStatement executePreparedStatement(String sql){  
 PreparedStatement ps=null;  
 try{  
 ps=*getConnection*().prepareStatement(sql);  
 }catch (Exception e){  
 e.printStackTrace();  
 }  
 return ps;  
 }  
 //回滚  
 public static void rollback(){  
 try{  
 *getConnection*().rollback();  
 }catch (SQLException e){  
 e.printStackTrace();  
 }  
 }  
 public static void close(){  
 try{  
 if(*rs*!=null)  
 *rs*.close();  
 if(*s*!=null)  
 *s*.close();  
 if(*conn*!=null)  
 *conn*.close();  
 }catch (SQLException e){  
 e.printStackTrace();  
 }  
 }  
}

1. **有关字体的过滤器**

public void doFilter(ServletRequest req, ServletResponse resp, FilterChain chain) throws ServletException, IOException {  
 req.setCharacterEncoding("UTF-8");  
 resp.setCharacterEncoding("UTF-8");  
 chain.doFilter(req, resp);  
}

1. **商场主页**



**<%** String bookid=request.getParameter("id");  
 String num=request.getParameter("amount");  
  
 String price="";  
  
 if((Vector)session.getAttribute("cartList")==null){  
**%>**<p>目前你的购物车是空的</p>  
**<%**}else{  
 Vector list=(Vector)session.getAttribute("cartList");  
 int n=list.size();  
**%>**

**进来主页先到购物车界面，判断是否为空，若空则显示目前你的购物车是空的的提示**

**点击购书进入购书界面**

1. **购书**



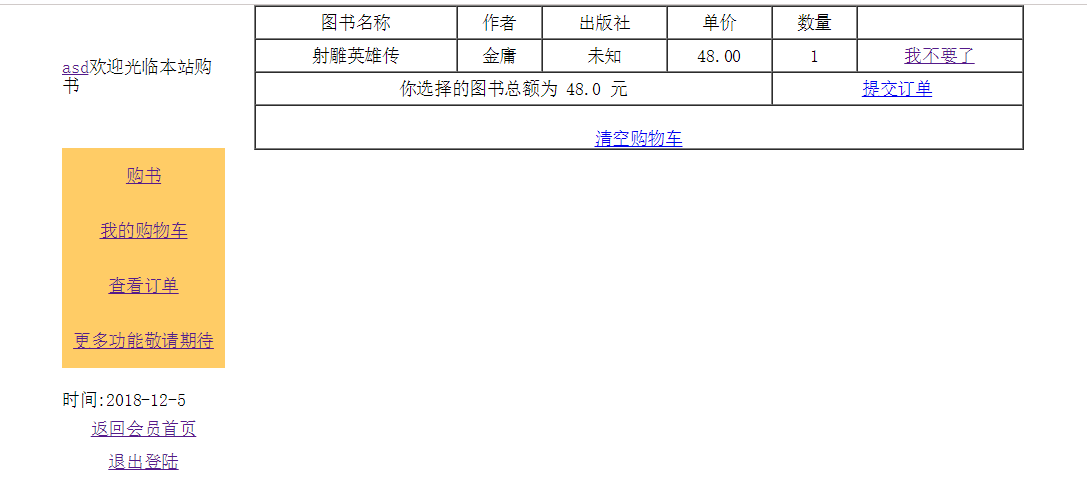
**<%** ResultSet rs=null;  
 db.*connection*();  
 String query="select \* from book";  
 rs=db.*query*(query);  
 while(rs.next()){  
 String s1=rs.getString(1);//书ID  
 String s2=rs.getString(2);//图书名  
 String s3=rs.getString(3);//作者  
 String s4=rs.getString(4);//出版社  
 String s5=rs.getString(5);//类别  
 String s6=rs.getString(6);//单价  
 String s7=rs.getString(7);//库存  
**%>**<tr>  
 <td height="30" align="center" valign="middle">**<%=**s1**%>**</td>  
 <td height="30" align="center" valign="middle">**<%=**s2**%>**</td>  
 <td height="30" align="center" valign="middle">**<%=**s3**%>**</td>  
 <td height="30" align="center" valign="middle">**<%=**s4**%>**</td>  
 <td height="30" align="center" valign="middle">**<%=**s5**%>**</td>  
 <td height="30" align="center" valign="middle">**<%=**s6**%>**</td>

**获取数据库所有图书的信息显示在页面上**

<td height="30" align="center" valign="middle"><span class="STYLE18"></span>  
 <form id="form1" name="form1" method="post" action="purchase.jsp?id=**<%=**s1**%>**">  
 <label><input type="text" name="amount" id="textfield" size="5"></label>  
 <input type="submit" name="button2" value="购买"/><span class="STYLE18"></span>  
 </form>  
</td>

**点击购买后，会吧购买的数量以及对应书本的单价发送到购物车的购物单中**

1. **购物车**



**<%** float money=0;  
 for(int i=0;i<n;i++){  
 cart c=(cart)list.elementAt(i);  
 String sql="select id,name,author,publisher,price from book where id='"+c.getBookId()+"'";  
 //db.connection();  
 ResultSet rs=db.*query*(sql);  
 rs.next();  
 String s1=rs.getString("id");  
 String s2=rs.getString("name");  
 String s3=rs.getString("author");  
 String s4=rs.getString("publisher");  
 String s5=rs.getString("price");  
 price=s5;  
 money+=Float.*parseFloat*(price)\*c.getNum();  
**%>**

**购物车根据书本id列出相关信息以及上个页面获取的单价和数量的信息并算出总价**

1. **订单**



**<%** String id=request.getParameter("id");  
 db.*connection*();  
 String sql="select \* from orderlist where id='"+id+"'";  
 ResultSet rs=db.*query*(sql);  
 rs.next();  
 String s1=rs.getString("id");//订单号id  
 System.*out*.println(s1);  
 String s2=rs.getString(2);//订单人 用户username  
 String s5=rs.getString(4);//金额 总价money  
 Vector list=(Vector)session.getAttribute("cartList");  
 int n=list.size();  
 for(int i=0;i<n;i++){  
 cart cart=(cart)list.elementAt(i);  
 String bId=cart.getBookId();  
 sql="select name,price from book where id='"+bId+"'";  
 db.*connection*();  
 rs=db.*query*(sql);  
 rs.next();  
 String s3=rs.getString(1);//书本name  
 String s4=rs.getString(2);//书本单价price  
**%>**

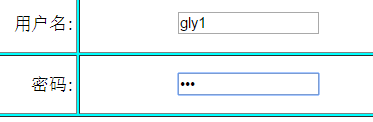
**提交订单后列出详细信息**

**<%** int key=0;  
 String id=" ";  
 String date=" ";  
 String money=request.getParameter("money");  
 String user=request.getParameter("username");  
 String price=request.getParameter("price");  
 String num=request.getParameter("amount");  
 out.println(num);  
 out.println(price);  
 GregorianCalendar d=new GregorianCalendar();  
 String year=Integer.*toString*(d.get(Calendar.*YEAR*));  
 String month=Integer.*toString*(d.get(Calendar.*MONTH*)+1);  
 String day=Integer.*toString*(d.get(Calendar.*DAY\_OF\_MONTH*));  
 date=year+month+day;  
 String query="select \* from userorder";  
 ResultSet rs=mydb.*executeQuery*(query);  
  
 while(rs.next()){  
 key=rs.getRow();  
 }  
 key++;  
 id=Integer.*toString*(key);  
 Vector list=(Vector)session.getAttribute("cartList");  
 int n=list.size();  
 //out.println(n);  
 for(int i=0;i<n;i++){  
 cart cart=(cart)list.elementAt(i);  
 String sql="select name,price from book where id='"+cart.getBookId()+"'";  
 rs=db.*query*(sql);  
 rs.next();  
 String name=rs.getString("name");  
 int sum=cart.getNum();  
  
 //库存重新计算数量  
 int stock=0;  
 rs=db.*query*("select stock from book where id='"+cart.getBookId()+"'");  
 while(rs.next())  
 stock=Integer.*parseInt*(rs.getString(1));  
 stock=stock-cart.getNum();  
 String sql3="update book set stock="+stock+" where id="+cart.getBookId();  
 db.*update*(sql3);  
 }  
 //订单表  
 String sql="insert into userorder values("+id+","+user+","+date+","+money+","+price+","+num+")";  
 int rsn1=db.*update*(sql);  
 response.sendRedirect("orderdisplay.jsp");  
**%>**

**记录到数据库中并且更新书本的库存**

1. **详细代码for Manger**

管理员账户由超级管理员生成 *在表messager*



1. **用户管理页面**



**<%** //获取查询方式和关键字  
 String type = request.getParameter("column");  
 String keyword = request.getParameter("value");  
//默认查询所有的用户信息  
 String sqlSearch = "SELECT \* FROM customer";  
//如果查询关键字不为空，则重定义SQL语句  
 if(keyword != null) {  
 sqlSearch = "SELECT \* FROM customer WHERE " + type + " LIKE '%" + keyword + "%'";  
 }  
 ResultSet rs = null;  
 rs = DbUtil.*executeQuery*(sqlSearch);  
 StringBuffer sb = new StringBuffer();  
 try {  
 while(rs.next()) {  
 String userId = "";  
 userId = rs.getString("id");  
 sb.append("<tr><td>");  
 sb.append(userId);  
 sb.append("</td><td>");  
 sb.append(rs.getString("password"));  
 sb.append("</td><td>");  
 sb.append(rs.getString("name"));  
 sb.append("</td><td>");  
 sb.append(rs.getString("sex"));  
 sb.append("</td><td>");  
 sb.append(rs.getString("address"));  
 sb.append("</td><td>");  
 sb.append(rs.getString("code"));  
 sb.append("</td><td>");  
 sb.append(rs.getString("tel"));  
 sb.append("</td><td>");  
 sb.append(rs.getString("email"));  
 sb.append("</td><td>");  
 //查询字符串，传递一个名为bookId的参数，该参数值为id即用户id  
 sb.append("&nbsp;");sb.append("&nbsp;");  
 sb.append("<a href='delete.jsp?id=").append(userId).append("'>删除</a>");  
 sb.append("&nbsp;");  
 sb.append("<a href='userUpdate.jsp?id=").append(userId).append("'>修改</a>");  
 sb.append("</td></tr>");  
 }  
 DbUtil.*close*();  
 } catch(SQLException e) {  
 e.printStackTrace();  
 }  
 session.setAttribute("search", sb);  
 out.print(sb);  
 response.sendRedirect("userAdmin.jsp");  
**%>**

**从数据库中获取用户信息；有删除用户（恶意用户）修改用户信息（若不合理），默认用显示全部用户，可以用关键字查询用户**



//编写SQL语句  
 String sql = "UPDATE customer SET password=?,name=?,sex=?," +  
 "address=?, code=?,tel=?,email=? WHERE id=?";  
 int result = 0;  
 //为动态SQL的参数赋值  
 try{  
 PreparedStatement ps = DbUtil.*executePreparedStatement*(sql);  
// ps.setString(1, id);  
 ps.setString(1, password);  
 ps.setString(2, name);  
 ps.setString(3, sex);  
 ps.setString(4, address);  
 ps.setString(5, code);  
 ps.setString(6, tel);  
 ps.setString(7, email);  
 ps.setString(8, id);  
 //执行SQL语句  
 result =ps.executeUpdate();  
 ps.close();  
 }catch(SQLException e)  
 {  
 e.printStackTrace();  
 }  
 //如果执行成功，查询数据库  
 if(result >0 )  
 {  
 //查询所有的用户信息  
 String sqlSearch = "SELECT \* FROM customer";  
 ResultSet rs = null;  
 rs = DbUtil.*executeQuery*(sqlSearch);  
 StringBuffer sb = new StringBuffer();  
 try{  
 //编历查询结果，拼接为StringBuffer对象。  
 while(rs.next())  
 {  
 sb.append("<tr><td>");  
 sb.append(rs.getString("id"));  
 sb.append("</td><td>");  
 sb.append(rs.getString("password"));  
 sb.append("</td><td>");  
 sb.append(rs.getString("name"));  
 sb.append("</td><td>");  
 sb.append(rs.getString("sex"));  
 sb.append("</td><td>");  
 sb.append(rs.getString("address"));  
 sb.append("</td><td>");  
 sb.append(rs.getString("code"));  
 sb.append("</td><td>");  
 sb.append(rs.getString("tel"));  
 sb.append("</td><td>");  
 sb.append(rs.getString("email"));  
 sb.append("</td><td>");  
 sb.append("&nbsp;");sb.append("&nbsp;");  
 sb.append("<a href='delete.jsp?id=").append(id).append("'>删除</a>");  
 sb.append("&nbsp;");  
 sb.append("<a href='bookUpdate.jsp?id=").append(id).append("'>修改</a>");  
 sb.append("</td></tr>");  
 }  
 DbUtil.*close*();  
 }catch(SQLException e)  
 {  
 e.printStackTrace();  
 }  
 //将查询结果存入session名为search的属性中  
 session.setAttribute("search", sb);  
 //跳转至bookAdmin.jsp页面  
// out.print("更新成功");  
 response.sendRedirect("userAdmin.jsp");  
 }

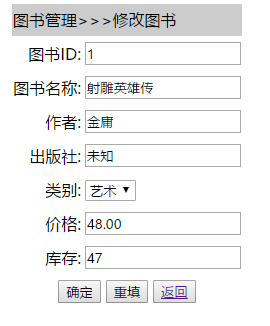
**修改时，获取用户的所有信息，更改时动态赋值，并且更新数据库**

1. **书本管理**



**<%** //获取查询方式和关键字  
 String type = request.getParameter("column");  
 String value = request.getParameter("value");  
//默认查询所有的用户信息  
 String sqlSearch = "SELECT \* FROM book";  
//如果查询关键字不为空，则重定义SQL语句  
 if(value != null) {  
 sqlSearch = "SELECT \* FROM book WHERE " + type + " LIKE '%" + value + "%'";  
 }  
 ResultSet rs = null;  
 rs = DbUtil.*executeQuery*(sqlSearch);  
 StringBuffer sb = new StringBuffer();  
 try {  
 while(rs.next()) {  
 String bookid = "";  
 bookid = rs.getString("id");  
 sb.append("<tr><td>");  
 sb.append(bookid);  
 sb.append("</td><td>");  
 sb.append(rs.getString("name"));  
 sb.append("</td><td>");  
 sb.append(rs.getString("author"));  
 sb.append("</td><td>");  
 sb.append(rs.getString("publisher"));  
 sb.append("</td><td>");  
 sb.append(rs.getString("type"));  
 sb.append("</td><td>");  
 sb.append(rs.getString("price"));  
 sb.append("</td><td>");  
 sb.append(rs.getString("stock"));  
 sb.append("</td><td>");  
 //查询字符串，传递一个名为bookid的参数，该参数值为id即用户名  
 sb.append("<a href='delete.jsp?id=").append(bookid).append("'>删除</a>");  
 sb.append("&nbsp;");  
 sb.append("<a href='bookUpdate.jsp?id=").append(bookid).append("'>修改</a>");  
 sb.append("</td></tr>");  
 }  
 DbUtil.*close*();  
 } catch(SQLException e) {  
 e.printStackTrace();  
 }  
 session.setAttribute("search", sb);  
 out.print(sb);  
 response.sendRedirect("bookAdmin.jsp");  
**%>**

**从数据库中获取书本信息；有删除书本（下架）修改书本信息（若不合理，或者上货后修改库存量），默认用显示全部书本，可以用关键字查询书本。**



//编写SQL语句  
 String sql = "UPDATE book SET name=?,author=?,publisher=?," +  
 "type=?, price=?,stock=? WHERE id=?";  
 int result = 0;  
 //为动态SQL的参数赋值  
 try{  
 PreparedStatement ps = DbUtil.*executePreparedStatement*(sql);  
// ps.setString(1, id);  
 ps.setString(1, name);  
 ps.setString(2, author);  
 ps.setString(3, publisher);  
 ps.setString(4, type);  
 ps.setString(5, price);  
 ps.setString(6, stock);  
 ps.setString(7,id);  
 //执行SQL语句  
 result =ps.executeUpdate();  
 ps.close();  
 }catch(SQLException e)  
 {  
 e.printStackTrace();  
 }  
 //如果执行成功，查询数据库  
 if(result >0 )  
 {  
 //查询所有的用户信息  
 String sqlSearch = "SELECT \* FROM book";  
 ResultSet rs = null;  
 rs = DbUtil.*executeQuery*(sqlSearch);  
 StringBuffer sb = new StringBuffer();  
 try{  
 //编历查询结果，拼接为StringBuffer对象。  
 while(rs.next())  
 {  
 sb.append("<tr><td>");  
 sb.append(rs.getString("id"));  
 sb.append("</td><td>");  
 sb.append(rs.getString("name"));  
 sb.append("</td><td>");  
 sb.append(rs.getString("author"));  
 sb.append("</td><td>");  
 sb.append(rs.getString("publisher"));  
 sb.append("</td><td>");  
 sb.append(rs.getString("type"));  
 sb.append("</td><td>");  
 sb.append(rs.getString("price"));  
 sb.append("</td><td>");  
 sb.append(rs.getString("stock"));  
 sb.append("</td><td>");  
 sb.append("<a href='delete.jsp?id=").append(id).append("'>删除</a>");  
 sb.append("&nbsp;");  
 sb.append("<a href='bookUpdate.jsp?id=").append(id).append("'>修改</a>");  
 sb.append("</td></tr>");  
 }  
 DbUtil.*close*();  
 }catch(SQLException e)  
 {  
 e.printStackTrace();  
 }  
 //将查询结果存入session名为search的属性中  
 session.setAttribute("search", sb);  
 //跳转至bookAdmin.jsp页面  
// out.print("更新成功");  
 response.sendRedirect("bookAdmin.jsp");

**获取书本信息，修改时动态赋值，并且更新数据库，操作与用户的修改类型**

1. **数据库**

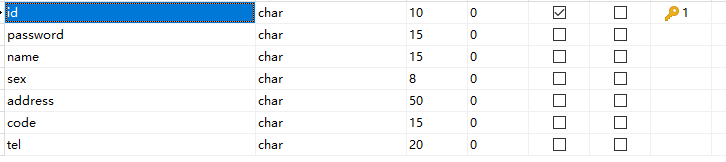


***数据库详细在本文档所在文件夹的 bookshop.sql文件中***

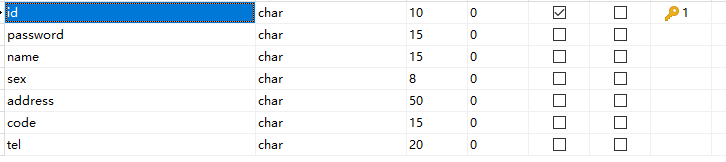
Book



customer



messager



Useroder （购物清单）



1. **补充说明(登陆)**

**用户登陆 login2.jsp**

**<%** String username=request.getParameter("username");  
 String password=request.getParameter("password");  
 String role=request.getParameter("role");  
 ResultSet rs=null;  
  
 if(role.equals("user")){  
// db.connection();  
// String sql="select \* from customer where id="+"\'"+username+"\'"+"and password="+"\'"+password+"\'";  
// rs=db.query(sql);  
// while(rs.next()){  
// session.setAttribute("username", username);  
// response.sendRedirect("cart/mycart.jsp");  
// }

//为方便测试所用的验证  
 if(username.equals("123")&&password.equals("123")){  
 session.setAttribute("username", username);  
 response.sendRedirect("cart/mycart.jsp");  
 }  
 }

//为方便测试所用的验证  
**%>**

**为方便测试用户的账户密码默认为123，若恢复原来功能，则把上面代码的注释去掉，并吧** //为方便测试所用的验证 **之间的代码注释调即可**

**管理员登陆 login.jsp**

**<%** String username=request.getParameter("username");  
 String password=request.getParameter("password");  
 String role=request.getParameter("role");  
 ResultSet rs=null;  
  
 if(role.equals("user")){  
// db.connection();  
// String sql="select \* from messager where id="+"'"+username+"'"+" and password="+"'"+password+"'";  
// rs=db.query(sql);  
// while(rs.next()){  
// session.setAttribute("username", username);  
// /\*response.sendRedirect("newManage/bookAdmin.jsp");\*/  
// response.sendRedirect("userManage/userAdmin.jsp");  
// }

//为方便测试所用的验证

if(username.equals("gly")&&password.equals("123")){  
 session.setAttribute("username", username);  
 response.sendRedirect("userManage/userAdmin.jsp");  
 }  
 }  
 //为方便测试所用的验证  
 **%>**

**为方便测试用户的账户默认为gly密码默认为123，若恢复原来功能，则把上面代码的注释去掉，并吧** //为方便测试所用的验证 **之间的代码注释调即可**

1. **心得总结**

利用本学期学的java web知识以及上学期的数据库原理的知识实现本系统，虽然简陋，功能不强大，但更加熟练了相关操作，并且更加深了对数据库的操作是若何实现的。

数据无处不在，所以我们要更加合理地管理我们需要的数据，以及理清楚相关的管理，对于er图的熟练必不可少；