实 验 报 告

院（系）：信息学院 课程名称：JAVA WEB开发及应用 日期：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 班级 | B1531 | 学号 | 01 | 实验室 |  |
| 专业 | 软件技术 | 姓名 | 付炳博 | 计算机号 |  |
| 实验  名称 | 图书馆管理系统 | | | 成绩评定 |  |
| 所用  软件 | JDK, Apache Tomcat , Eclipse,MySQL | | | 教师签名 |  |
| 实  验  目  的  或  要  求 | 实现了一个小型的图书馆管理系统。  美观友好的操作界面，能够保证系统的易用性。 | | | | |
| 实  验  步  骤  、  心  得  体  会 | 图书馆管理系统主要包括主要功能，有系统设置模块，读者管理模块，图书管理模块，图书借还模块，系统查询和口令更改。   1. 设计数据库db\_library，共12个表     1-1表tb\_bookinfo，用来存放图书信息。    1-3表tb\_manager,用来存放管理员登录时所用的信息。    1-4表tb\_purview,用来存放图书管理员的访问权限。     1. 文件结构      1. 主要代码如下。   3.1.先创建合适的包名，新建BaseDao用于连接数据库。  **package cn.jju.library.dao;**  import java.sql.Connection;  import java.sql.DriverManager;  import java.sql.ResultSet;  import java.sql.SQLException;  import java.sql.Statement;  **public class ConnDB** {  public Connection conn = null;  public Statement stmt = null;  public ResultSet rs = null;  //private PreparedStatement pstmt = null;  public ConnDB() {  try {  Class.forName("com.mysql.jdbc.Driver");  conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/db\_library",  "root", "root");  } catch (ClassNotFoundException e) {  e.printStackTrace();  } catch (SQLException e) {  e.printStackTrace();  }  }  public Connection getConnection() {  return conn;  }  /\*  \* 功能：执行查询语句  \*/  public ResultSet executeQuery(String sql) {  try {  conn = getConnection();  stmt = conn.createStatement(ResultSet.TYPE\_SCROLL\_INSENSITIVE,  ResultSet.CONCUR\_READ\_ONLY);  rs = stmt.executeQuery(sql);  } catch (SQLException ex) {  System.err.println(ex.getMessage());  }  return rs;  }  /\*  \* 功能:执行更新操作  \*/  public int executeUpdate(String sql) {  int result = 0;  try {  conn = getConnection(); //调用getConnection()方法构造Connection对象的一个实例conn  stmt = conn.createStatement(ResultSet.TYPE\_SCROLL\_INSENSITIVE,  ResultSet.CONCUR\_READ\_ONLY);  result = stmt.executeUpdate(sql); //执行更新操作  } catch (SQLException ex) {  result = 0;  }  return result;  }  /\*  \* 功能:关闭数据库的连接  \*/  public void close() {  try {  if (rs != null) {  rs.close();  }  if (stmt != null) {  stmt.close();  }  if (conn != null) {  conn.close();  }  } catch (Exception e) {  e.printStackTrace(System.err);  }  }  }   1. 根据数据的设计把实体类的写下来。     主要实体类的代码如下。  4.1  **package cn.jju.library.entity;**  **public class BookForm** {  private String author;  private String barcode;  private String bookName;  private String bookcaseName;  private int bookcaseid;  private int days;  private int del;  private Integer id;  private String inTime;  private String isbn;  private String operator;  private int page;  private Float price;  private String publishing;  private String translator;  private int typeId;  private String typeName;  private String img;  public String getAuthor() {  return author;  }  public void setAuthor(String author) {  this.author = author;  }  public void setTypeName(String typeName) {  this.typeName = typeName;  }  public void setTypeId(int typeId) {  this.typeId = typeId;  }  public void setTranslator(String translator) {  this.translator = translator;  }  public void setPublishing(String publishing) {  this.publishing = publishing;  }  public void setPrice(Float price) {  this.price = price;  }  public void setPage(int page) {  this.page = page;  }  public void setOperator(String operator) {  this.operator = operator;  }  public void setIsbn(String isbn) {  this.isbn = isbn;  }  public void setInTime(String inTime) {  this.inTime = inTime;  }  public void setId(Integer id) {  this.id = id;  }  public void setDel(int del) {  this.del = del;  }  public void setDays(int days) {  this.days = days;  }  public void setBookcaseid(int bookcaseid) {  this.bookcaseid = bookcaseid;  }  public void setBookcaseName(String bookcaseName) {  this.bookcaseName = bookcaseName;  }  public void setBookName(String bookName) {  this.bookName = bookName;  }  public void setBarcode(String barcode) {  this.barcode = barcode;  }  public String getBarcode() {  return barcode;  }  public String getBookName() {  return bookName;  }  public String getBookcaseName() {  return bookcaseName;  }  public int getBookcaseid() {  return bookcaseid;  }  public int getDays() {  return days;  }  public int getDel() {  return del;  }  public Integer getId() {  return id;  }  public String getInTime() {  return inTime;  }  public String getIsbn() {  return isbn;  }  public String getOperator() {  return operator;  }  public int getPage() {  return page;  }  public Float getPrice() {  return price;  }  public String getPublishing() {  return publishing;  }  public String getTranslator() {  return translator;  }  public int getTypeId() {  return typeId;  }  public String getTypeName() {  return typeName;  }  public String getImg() {  return img;  }  public void setImg(String img) {  this.img = img;  }    }  4.2  **package cn.jju.library.entity;**  **public class ManagerForm** {  private Integer id=new Integer(-1); //管理员ID号  private String name=""; //管理员名称  private String pwd=""; //管理员密码  private int sysset=0; //系统设置权限  private int readerset=0; //读者管理权限  private int bookset=0; //图书管理权限  private int borrowback=0; //图书借还权限  private int sysquery=0; //系统查询权限  public Integer getId() {  return id;  }  public void setId(Integer id) {  this.id = id;  }  public void setPwd(String pwd) {  this.pwd = pwd;  }  public void setName(String name) {  this.name = name;  }  public String getName() {  return name;  }  public String getPwd() {  return pwd;  }  public void setSysset(int sysset){  this.sysset=sysset;  }  public int getSysset(){  return sysset;  }  public int getReaderset(){  return readerset;  }  public void setReaderset(int readerset){  this.readerset=readerset;  }  public void setBookset(int bookset){  this.bookset=bookset;  }  public int getBookset(){  return bookset;  }  public void setBorrowback(int borrowback){  this.borrowback=borrowback;  }  public int getBorrowback(){  return borrowback;  }  public void setSysquery(int sysquery){  this.sysquery=sysquery;  }  public int getSysquery(){  return sysquery;  }  }  5.系统登录设计  5.1系统登录是图书管理系统的的入口。在该界面中，系统管理员可通过输入正确的管理员名称和密码登录到系统。  登录界面如下。    5.1.2关键代码。  首先是页面布局。    在实现系统登录时，主要是在Servlet中获取提交的登录信息，并验证输入管理员信息是否合法，如果合法，则将页面重定向的系统到系统界面。在Servlet中编写方法，用于获取提交的登录信息，以及调用DAO的方法验证登录信息，并根据验证结果做出相应的处理  **package cn.jju.library.servlet;**  **// 管理员身份验证**  public void managerLogin(HttpServletRequest request,  HttpServletResponse response) throws ServletException, IOException {  ManagerForm managerForm = new ManagerForm();//实例化managerForm类  managerForm.setName(request.getParameter("name"));//获取管理员名称并设置name属性  managerForm.setPwd(request.getParameter("pwd"));//获取管理员密码并设置pwd属性  int ret = managerDaoImpl.checkManager(managerForm);//调用ManagerDAO类的checkManager()方法  if (ret == 1) {  /\*\*\*\*\*\*\*\*\*\*将登录到系统的管理员名称保存到session中\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  HttpSession session=request.getSession();  session.setAttribute("manager",managerForm.getName());  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  request.getRequestDispatcher("main.jsp").forward(request, response);//转到系统主界面  } else {  request.setAttribute("error", "您输入的管理员名称或密码错误！");  request.getRequestDispatcher("error.jsp")  .forward(request, response);//转到错误提示页  }  }  //编写dao方法，用于验证管理员身份，返回值1时表示验证成功。  **package cn.jju.library.daoImpl;**  **public class ManagerDaoImpl implements ManagerDao{**  // 管理员身份验证  public int checkManager(ManagerForm managerForm) {  int flag = 0; // 标记变量，值为0时表示不成功，值为1时表示成功  String sql = "SELECT \* FROM tb\_manager where name='"  + ChStr.filterStr(managerForm.getName()) + "'"; // 连接SQL语句，并过滤管理员名称中的危险字符  ResultSet rs = conn.executeQuery(sql);  try {  if (rs.next()) {  String pwd = ChStr.filterStr(managerForm.getPwd()); // 获取输入的密码并过滤输入字符串中的危险字符  if (pwd.equals(rs.getString(3))) {  flag = 1; // 表示验证成功  } else {  flag = 0; // 表示验证不成功  }  } else {  flag = 0; // 表示验证不成功  }  } catch (SQLException ex) {  flag = 0; // 表示验证不成功  } finally {  //conn.close(); // 关闭数据库连接  }  return flag;  }  在实现验证登录的过程中，从网站安全的角度考虑，在页面上加入验证用户是否是登录的代码。  <%  String manager = (String) session.getAttribute("manager");  //验证用户是否登录  if (manager == null || "".equals(manager)) {  response.sendRedirect("index.jsp");  }  %>  6.主页面设计。  6.2首页    6.2.1界面设计    6.3关键代码。  在主界面时，需要显示图书借阅排行榜，所以要编写Dao方法，实现从数据库中统计出借阅排行数据，并保存到Collection集合中。  **package cn.jju.library.daoImpl;**  import java.sql.ResultSet;  import java.sql.SQLException;  import java.text.SimpleDateFormat;  import java.util.ArrayList;  import java.util.Calendar;  import java.util.Collection;  import cn.jju.library.dao.BorrowDao;  import cn.jju.library.dao.ConnDB;  import cn.jju.library.entity.BookForm;  import cn.jju.library.entity.BorrowForm;  import cn.jju.library.entity.ReaderForm;  **public class BorrowDaoImpl implements BorrowDao{**  **...**  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*图书借阅排行\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  public Collection<BorrowForm> bookBorrowSort() {  String sql = "select \* from (SELECT bookid,count(bookid) as degree FROM" +  " tb\_borrow group by bookid) as borr join (select b.\*,c.name as bookcaseName" +  ",p.pubname,t.typename from tb\_bookinfo b left join tb\_bookcase" +  " c on b.bookcase=c.id join tb\_publishing p on b.ISBN=p.ISBN join " +  "tb\_booktype t on b.typeid=t.id where b.del=0)" +  " as book on borr.bookid=book.id order by borr.degree desc limit 10 ";  Collection<BorrowForm> coll = new ArrayList<>(); //创建并实例化Collection对象  BorrowForm form = null; //声明BorrowForm对象  ResultSet rs = conn.executeQuery(sql); //执行查询语句  try {  while (rs.next()) {  form = new BorrowForm(); //实例化BorrowForm对象  form.setBookId(rs.getInt(1)); //获取图书ID  form.setDegree(rs.getInt(2)); //获取借阅次数  form.setBookBarcode(rs.getString(3)); //获取图书条形码  form.setBookName(rs.getString(4)); //获取图书名称  form.setAuthor(rs.getString(6)); //获取作者  form.setPrice(Float.valueOf(rs.getString(9))); //获取定价  form.setBookcaseName(rs.getString(16)); //获取书架名称  form.setPubName(rs.getString(17)); //获取出版社  form.setBookType(rs.getString(18)); //获取图书类型  coll.add(form); //保存到Collection集合中  }  } catch (SQLException ex) {  System.out.println(ex.getMessage()); //输出异常信息  }  //conn.close(); //关闭数据库连接  return coll;  }   1. 图书借阅设计   在实现图书借阅时，因为限定了每位读者可借阅的图书的次数，且该数量由数据表tb\_readerType中的可借数量number决定。  7.1关键代码  **package cn.jju.library.servlet;**  import java.io.IOException;  import javax.servlet.ServletException;  import javax.servlet.annotation.WebServlet;  import javax.servlet.http.HttpServlet;  import javax.servlet.http.HttpServletRequest;  import javax.servlet.http.HttpServletResponse;  import cn.jju.library.daoImpl.BookDaoImpl;  import cn.jju.library.daoImpl.BorrowDaoImpl;  import cn.jju.library.daoImpl.ReaderDaoImpl;  import cn.jju.library.entity.BookForm;  import cn.jju.library.entity.ReaderForm;  /\*\*  \* Servlet implementation class BorrowServlet  \*/  **@WebServlet("/borrow")**  **public class BorrowServlet extends HttpServlet {**  // 在构造方法中实例化Borrow类中应用的持久层类的对象  private BorrowDaoImpl borrowdaoimpl = null;  private ReaderDaoImpl readerdaoimpl = null;  private BookDaoImpl bookdaoimpl = null;  private ReaderForm readerForm = new ReaderForm();  public BorrowServlet() {  this.borrowdaoimpl = new BorrowDaoImpl();  this.readerdaoimpl = new ReaderDaoImpl();  this.bookdaoimpl = new BookDaoImpl();  }  protected void doGet(HttpServletRequest request, HttpServletResponse response)  throws ServletException, IOException {  // TODO Auto-generated method stub  String action = request.getParameter("action");  if (action == null || "".equals(action)) {  request.setAttribute("error", "您的操作有误！");  request.getRequestDispatcher("error.jsp").forward(request, response);  } else if ("bookBorrowSort".equals(action)) {  bookBorrowSort(request, response);  } else if ("bookborrow".equals(action)) {  bookborrow(request, response); // 图书借阅  } else if ("bookrenew".equals(action)) {  bookrenew(request, response); // 图书续借  } else if ("bookback".equals(action)) {  bookback(request, response); // 图书归还  } else if ("Bremind".equals(action)) {  bremind(request, response); // 借阅到期提醒  } else if ("borrowQuery".equals(action)) {  borrowQuery(request, response); // 借阅信息查询  }  }  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 图书借阅排行 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  private void bookBorrowSort(HttpServletRequest request, HttpServletResponse response)  throws ServletException, IOException {  request.setAttribute("bookBorrowSort", borrowdaoimpl.bookBorrowSort());  request.getRequestDispatcher("bookBorrowSort.jsp").forward(request, response);  }  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 图书借阅查询 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  private void borrowQuery(HttpServletRequest request, HttpServletResponse response)  throws ServletException, IOException {  String str = null;  String flag[] = request.getParameterValues("flag");  if (flag != null) {  String aa = flag[0];  if ("a".equals(aa)) {  if (request.getParameter("f") != null) {  str = request.getParameter("f") + " like '%" + request.getParameter("key") + "%'";  }  }  if ("b".equals(aa)) {  String sdate = request.getParameter("sdate");  String edate = request.getParameter("edate");  if (sdate != null && edate != null) {  str = "borrowTime between '" + sdate + "' and '" + edate + "'";  }  System.out.println("日期" + str);  }  // 同时选择日期和条件进行查询  if (flag.length == 2) {  if (request.getParameter("f") != null) {  str = request.getParameter("f") + " like '%" + request.getParameter("key") + "%'";  }  System.out.println("日期和条件");  String sdate = request.getParameter("sdate");  String edate = request.getParameter("edate");  String str1 = null;  if (sdate != null && edate != null) {  str1 = "borrowTime between '" + sdate + "' and '" + edate + "'";  }  str = str + " and borr." + str1;  System.out.println("条件和日期：" + str);  }  }  request.setAttribute("borrowQuery", borrowdaoimpl.borrowQuery(str));  System.out.print("条件查询图书借阅信息时的str:" + str);  request.getRequestDispatcher("borrowQuery.jsp").forward(request, response);  }  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 到期提醒 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  private void bremind(HttpServletRequest request, HttpServletResponse response)  throws ServletException, IOException {  request.setAttribute("Bremind", borrowdaoimpl.bremind());  request.getRequestDispatcher("bremind.jsp").forward(request, response);  }  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 图书借阅 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  private void bookborrow(HttpServletRequest request, HttpServletResponse response)  throws ServletException, IOException {  // 查询读者信息  readerForm.setBarcode(request.getParameter("barcode")); // 获取读者条形码  ReaderForm reader = (ReaderForm) readerdaoimpl.queryM(readerForm); // 根据读者条形码获取读者信息  request.setAttribute("readerinfo", reader); // 保存读者信息到request中  // 查询读者的借阅信息  request.setAttribute("borrowinfo", borrowdaoimpl.borrowinfo(request.getParameter("barcode")));  // 完成借阅  String f = request.getParameter("f"); // 获取查询条件  String key = request.getParameter("inputkey"); // 获取输入的关键字  if (key != null && !key.equals("")) { // 判断是否有符合条件的图书  String operator = request.getParameter("operator"); //  BookForm bookForm = bookdaoimpl.queryB(f, key); // 根据查询条件获取图书信息  if (bookForm != null) {  int ret = borrowdaoimpl.insertBorrow(reader, bookdaoimpl.queryB(f, key), operator); // 保存图书借阅信息  if (ret == 1) {  request.setAttribute("bar", request.getParameter("barcode"));//实现当图书借阅成功后，不是跳回借阅首页而是进行下一次的借阅  request.getRequestDispatcher("bookBorrow\_ok.jsp").forward(request, response);  } else {  request.setAttribute("error", "添加借阅信息失败!"); // 保存提示信息到request中  request.getRequestDispatcher("error.jsp").forward(request, response); // 转到错误提示页  }  } else {  request.setAttribute("error", "没有该图书!"); // 保存提示信息到request中  request.getRequestDispatcher("error.jsp").forward(request, response); // 转到错误提示页  }  } else {  request.getRequestDispatcher("bookBorrow.jsp").forward(request, response);  }  }  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 图书继借 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  private void bookrenew(HttpServletRequest request, HttpServletResponse response)  throws ServletException, IOException {  // 查询读者信息  readerForm.setBarcode(request.getParameter("barcode")); // 获取读者条形码  ReaderForm reader = (ReaderForm) readerdaoimpl.queryM(readerForm); // 根据读者条形码查询读者信息  request.setAttribute("readerinfo", reader);  // 查询读者的借阅信息  request.setAttribute("borrowinfo", borrowdaoimpl.borrowinfo(request.getParameter("barcode")));  if (request.getParameter("id") != null) {  int id = Integer.parseInt(request.getParameter("id"));  if (id > 0) { // 执行继借操作  int ret = borrowdaoimpl.renew(id);  if (ret == 0) {  request.setAttribute("error", "图书继借失败!");  request.getRequestDispatcher("error.jsp").forward(request, response);  } else {  request.setAttribute("bar", request.getParameter("barcode"));  request.getRequestDispatcher("bookRenew\_ok.jsp").forward(request, response);  }  }  } else {  request.getRequestDispatcher("bookRenew.jsp").forward(request, response);  }  }  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 图书归还 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  private void bookback(HttpServletRequest request, HttpServletResponse response)  throws ServletException, IOException {  // 查询读者信息  readerForm.setBarcode(request.getParameter("barcode"));  ReaderForm reader = (ReaderForm) readerdaoimpl.queryM(readerForm);  request.setAttribute("readerinfo", reader);  // 查询读者的借阅信息  request.setAttribute("borrowinfo", borrowdaoimpl.borrowinfo(request.getParameter("barcode")));  if (request.getParameter("id") != null) {  int id = Integer.parseInt(request.getParameter("id"));  String operator = request.getParameter("operator");  if (id > 0) { // 执行归还操作  int ret = borrowdaoimpl.back(id, operator);  if (ret == 0) {  request.setAttribute("error", "图书归还失败!");  request.getRequestDispatcher("error.jsp").forward(request, response);  } else {  request.setAttribute("bar", request.getParameter("barcode"));  request.getRequestDispatcher("bookBack\_ok.jsp").forward(request, response);  }  }  } else {  request.getRequestDispatcher("bookBack.jsp").forward(request, response);  }  }  protected void doPost(HttpServletRequest request, HttpServletResponse response)  throws ServletException, IOException {  // TODO Auto-generated method stub  doGet(request, response);  }  }  在实现图书借阅的方法中，还需要调用ReaderDao类中的queryM()、BorrowDao类的borrowinfo()、insertBorrow（）方法，BookDao类的queryB()方法。  **public class ReaderDaoImpl implements ReaderDao {**  // 用于修改的查询  **public ReaderForm queryM(ReaderForm readerForm) {**  ReaderForm readerForm1 = null;  String sql = "";  if (readerForm.getId() != null) { // 根据读者ID查询读者信息  sql = "select r.\*,t.name as typename,t.number from tb\_reader r left join tb\_readerType t on r.typeid=t.id where r.id="  + readerForm.getId() + "";  } else if (readerForm.getBarcode() != null) {// 根据读者条形码查询读者信息  sql = "select r.\*,t.name as typename,t.number from tb\_reader r left join tb\_readerType t on r.typeid=t.id where r.barcode="  + readerForm.getBarcode() + "";  }  ResultSet rs = conn.executeQuery(sql); // 执行查询语句  String birthday = "";  try {  while (rs.next()) {  readerForm1 = new ReaderForm();  readerForm1.setId(Integer.valueOf(rs.getString(1))); // 获取读者ID  readerForm1.setName(rs.getString(2)); // 获取读者姓名  readerForm1.setSex(rs.getString(3)); // 获取读者性别  readerForm1.setBarcode(rs.getString(4)); // 获取读者条形码  readerForm1.setVocation(rs.getString(5)); // 获取职业  birthday = rs.getString(6); // 获取生日  readerForm1.setBirthday(birthday == null ? "" : birthday);  readerForm1.setPaperType(rs.getString(7)); // 获取证件类型  readerForm1.setPaperNO(rs.getString(8)); // 获取证件号码  readerForm1.setTel(rs.getString(9));// 获取联系电话  readerForm1.setEmail(rs.getString(10));// 获取E-mail地址  readerForm1.setCreateDate(rs.getString(11));// 获取创建日期  readerForm1.setOperator(rs.getString(12)); // 获取操作员  readerForm1.setRemark(rs.getString(13));// 获取备注  readerForm1.setTypeid(rs.getInt(14));// 获取读者类型ID  readerForm1.setTypename(rs.getString(15)); // 获取读者类型名称  readerForm1.setNumber(rs.getInt(16)); // 获取可借数量  }  } catch (SQLException ex) {  }  // conn.close();//关闭数据库连接  return readerForm1;  }  **package cn.jju.library.daoImpl;**  import java.sql.ResultSet;  import java.sql.SQLException;  import java.text.SimpleDateFormat;  import java.util.ArrayList;  import java.util.Calendar;  import java.util.Collection;  import cn.jju.library.dao.BorrowDao;  import cn.jju.library.dao.ConnDB;  import cn.jju.library.entity.BookForm;  import cn.jju.library.entity.BorrowForm;  import cn.jju.library.entity.ReaderForm;  **public class BorrowDaoImpl implements BorrowDao{**  ConnDB conn = new ConnDB();  public int insert() {  String sql = "INSERT INTO tb\_borrow (bookid) vlaues(1) ";  int ret = conn.executeUpdate(sql);  return ret;  }  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*图书借阅\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  **public int insertBorrow(ReaderForm readerForm, BookForm bookForm,**  **String operator) {**  String sql1 = "select t.days from tb\_bookinfo b left join tb\_booktype t on"  + " b.typeid=t.id where b.id=" + bookForm.getId() + ""; // 获取可借天数的SQL语句  ResultSet rs = conn.executeQuery(sql1); // 执行SQL语句  int days = 0;  try {  if (rs.next()) {  days = rs.getInt(1); // 获取可借天数  }  } catch (SQLException ex) {  }  // 计算归还时间  Calendar calendar = Calendar.getInstance(); // 获取系统日期  SimpleDateFormat format = new SimpleDateFormat("yyyy-MM-dd");  java.sql.Date date = java.sql.Date.valueOf(format.format(calendar  .getTime()));// 借书日期  calendar.add(calendar.DAY\_OF\_YEAR, days);// 加上可借天数  java.sql.Date backTime = java.sql.Date.valueOf(format.format(calendar  .getTime())); // 归还日期  String sql = "Insert into tb\_borrow (readerid,bookid,borrowTime,backTime,operator) values("  + readerForm.getId()  + ","  + bookForm.getId()  + ",'"  + date  + "','" + backTime + "','" + operator + "')";  System.out.println("添加图书借阅信息的SQL：" + sql);  int falg = conn.executeUpdate(sql); // 执行更新语句  //conn.close();// 关闭数据库连接  return falg;  }  **//查询图书借阅信息**  **public Collection<BorrowForm> borrowinfo(String str){**  String sql="select borr.\*,book.bookname,book.price,pub.pubname," +  "bs.name bookcasename,r.barcode from (select \* from tb\_borrow " +  "where ifback=0) as borr left join tb\_bookinfo book on borr.bookid" +  "=book.id join tb\_publishing pub on book.isbn=pub.isbn join" +  " tb\_bookcase bs on book.bookcase=bs.id join tb\_reader r on" +  " borr.readerid=r.id where r.barcode='"+str+"'";  ResultSet rs=conn.executeQuery(sql);//执行查询语句  Collection<BorrowForm> coll=new ArrayList<>();  BorrowForm form=null;  try {  while (rs.next()) {  form = new BorrowForm();  form.setId(Integer.valueOf(rs.getInt(1)));//获取ID号  form.setBorrowTime(rs.getString(4));//获取借阅时间  form.setBackTime(rs.getString(5));//获取归还时间  form.setBookName(rs.getString(8));//获取图书名称  form.setPrice(Float.valueOf(rs.getFloat(9)));//获取定价  form.setPubName(rs.getString(10));//获取出版社  form.setBookcaseName(rs.getString(11)); //获取书价名称  coll.add(form);//添加借阅信息到Collection集合中  }  } catch (SQLException ex) {  System.out.println("借阅信息："+ex.getMessage());//输出异常信息  }  //conn.close();//关闭数据库连接  return coll;  }  **public class BookDaoImpl implements BookDao {**  **public BookForm queryB(String f, String key) {//借阅书籍的查询**  BookForm bookForm=null;  String sql="select b.\*,c.name as bookcaseName,p.pubname as publishing,t.typename" +  " from tb\_bookinfo b left join tb\_bookcase c on b.bookcase=c.id join" +  " tb\_publishing p on b.ISBN=p.ISBN join tb\_booktype t on" +  " b.typeid=t.id where b."+f+"='"+key+"'"; //查询图书信息的SQL语句  ResultSet rs=conn.executeQuery(sql); //执行查询语句  try {  if (rs.next()) {  bookForm=new BookForm();  bookForm.setBarcode(rs.getString(1)); //获取图书条形码  bookForm.setBookName(rs.getString(2)); //获取图书名称  bookForm.setTypeId(rs.getInt(3));//获取图书类型ID  bookForm.setAuthor(rs.getString(4));//获取作者  bookForm.setTranslator(rs.getString(5)); //获取译者  bookForm.setIsbn(rs.getString(6)); //获取图书的ISBN号  bookForm.setPrice(Float.valueOf(rs.getString(7))); //此处必须进行类型转换  bookForm.setPage(rs.getInt(8));//获取页码  bookForm.setBookcaseid(rs.getInt(9)); //获取书架ID  bookForm.setInTime(rs.getString(10)); //获取入库时间  bookForm.setOperator(rs.getString(11));//获取操作员  bookForm.setDel(rs.getInt(12));//获取是否删除  bookForm.setId(Integer.valueOf(rs.getString(13)));//获取图书ID号  bookForm.setBookcaseName(rs.getString(14));//获取书架名称  bookForm.setPublishing(rs.getString(15));//获取出版社  bookForm.setTypeName(rs.getString(16));//获取类型名称  }  } catch (SQLException ex) {  }  //conn.close();//关闭数据库连接  return bookForm;  }     1. 图书续借设计     在图书续借时，会把读者条形码和借阅者ID号一起传递到图书续借的Servlet控制类中。    主要实现了dao中的实现图书续借的方法bookrenew()和保存图书续借信息的方法renew（）。  **package cn.jju.library.servlet;**  **@WebServlet("/borrow")**  **public class BorrowServlet extends HttpServlet {**  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 图书继借 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  private void bookrenew(HttpServletRequest request, HttpServletResponse response)  throws ServletException, IOException {  // 查询读者信息  readerForm.setBarcode(request.getParameter("barcode")); // 获取读者条形码  ReaderForm reader = (ReaderForm) readerdaoimpl.queryM(readerForm); // 根据读者条形码查询读者信息  request.setAttribute("readerinfo", reader);  // 查询读者的借阅信息  request.setAttribute("borrowinfo", borrowdaoimpl.borrowinfo(request.getParameter("barcode")));  if (request.getParameter("id") != null) {  int id = Integer.parseInt(request.getParameter("id"));  if (id > 0) { // 执行继借操作  int ret = borrowdaoimpl.renew(id);  if (ret == 0) {  request.setAttribute("error", "图书继借失败!");  request.getRequestDispatcher("error.jsp").forward(request, response);  } else {  request.setAttribute("bar", request.getParameter("barcode"));  request.getRequestDispatcher("bookRenew\_ok.jsp").forward(request, response);  }  }  } else {  request.getRequestDispatcher("bookRenew.jsp").forward(request, response);  }  }  **package cn.jju.library.daoImpl;**  **public class BorrowDaoImpl implements BorrowDao{**  **//图书继借**  **public int renew(int id){**  String sql0="SELECT bookid FROM tb\_borrow WHERE id="+id+""; //根据借阅ID查询图书ID的SQL语句  ResultSet rs1=conn.executeQuery(sql0); //执行查询语句  int flag=0;  try {  if (rs1.next()) {  //获取可借天数  String sql1 = "select t.days from tb\_bookinfo b left join" +  " tb\_booktype t on b.typeid=t.id where b.id="  +rs1.getInt(1) + ""; //获取可借天数的SQL语句  ResultSet rs = conn.executeQuery(sql1); //执行查询语句  int days = 0;  try {  if (rs.next()) {  days = rs.getInt(1); //获取可借天数  }  } catch (SQLException ex) {  }  //计算归还时间  Calendar calendar=Calendar.getInstance(); //获取系统日期  SimpleDateFormat format = new SimpleDateFormat("yyyy-MM-dd");//设置日期格式  java.sql.Date date=java.sql.Date.valueOf(  format.format(calendar.getTime()));//借书日期  calendar.add(calendar.DAY\_OF\_YEAR, days); //加上可借天数  java.sql.Date backTime= java.sql.Date.valueOf(format.format(calendar.getTime())); //归还日期  String sql = "UPDATE tb\_borrow SET backtime='" + backTime +  "' where id=" + id + ""; //更新归还时间完成续借  flag = conn.executeUpdate(sql);//执行更新语句  }  } catch (Exception ex1) {}  //conn.close();//关闭数据库连接  return flag;  }   1. 图书归还设计     归还之后    9.1在实现此页面时，和图书续借有很多相似的地方。  <td width="12%" align="center"><a href="borrow?action=bookback&barcode=<%=barcode%>&id=<%=id%>&operator=<%=manager%>">归还</a>&nbsp;</td>  图书归还时将读者的条形码和借阅ID一起传值到Servlet类中。  9.2关键代码。  **public int back(int id,String operator){**  //根据借阅ID获取读者ID和图书ID  String sql0="SELECT readerid,bookid FROM tb\_borrow WHERE id="+id+"";  ResultSet rs1=conn.executeQuery(sql0); //执行查询语句  int flag=0;  try {  if (rs1.next()) {  Calendar calendar=Calendar.getInstance(); //获取系统日期  SimpleDateFormat format = new SimpleDateFormat("yyyy-MM-dd");  java.sql.Date date=java.sql.Date.valueOf(format.format(calendar.getTime()));//还书日期  int readerid=rs1.getInt(1); //获取读者ID  int bookid=rs1.getInt(2);//获取图书ID  String sql1="INSERT INTO tb\_giveback (readerid,bookid,backTime" +  ",operator) VALUES("+readerid+","+bookid+",'"  +date+"','"+operator+"')"; //保存归还信息  int ret=conn.executeUpdate(sql1); //执行更新语句  if(ret==1){  String sql2 = "UPDATE tb\_borrow SET ifback=1 where id=" + id +  ""; //将借阅信息标记为已归还  flag = conn.executeUpdate(sql2); //执行更新语句  }else{  flag=0;  }  }  } catch (Exception ex1) {  }  //conn.close();//关闭数据库连接  return flag;  }  **package cn.jju.library.servlet;**  import java.io.IOException;  import javax.servlet.ServletException;  import javax.servlet.annotation.WebServlet;  import javax.servlet.http.HttpServlet;  import javax.servlet.http.HttpServletRequest;  import javax.servlet.http.HttpServletResponse;  import cn.jju.library.daoImpl.BookDaoImpl;  import cn.jju.library.daoImpl.BorrowDaoImpl;  import cn.jju.library.daoImpl.ReaderDaoImpl;  import cn.jju.library.entity.BookForm;  import cn.jju.library.entity.ReaderForm;  @WebServlet("/borrow")  **public class BorrowServlet extends HttpServlet {**  **private void bookback(HttpServletRequest request, HttpServletResponse response)**  **throws ServletException, IOException {**  // 查询读者信息  readerForm.setBarcode(request.getParameter("barcode"));  ReaderForm reader = (ReaderForm) readerdaoimpl.queryM(readerForm);  request.setAttribute("readerinfo", reader);  // 查询读者的借阅信息  request.setAttribute("borrowinfo", borrowdaoimpl.borrowinfo(request.getParameter("barcode")));  if (request.getParameter("id") != null) {  int id = Integer.parseInt(request.getParameter("id"));  String operator = request.getParameter("operator");  if (id > 0) { // 执行归还操作  int ret = borrowdaoimpl.back(id, operator);//调用back()方法执行图书归还操作  if (ret == 0) {  request.setAttribute("error", "图书归还失败!");  request.getRequestDispatcher("error.jsp").forward(request, response);  } else {  request.setAttribute("bar", request.getParameter("barcode"));  request.getRequestDispatcher("bookBack\_ok.jsp").forward(request, response);  }  }  } else {  request.getRequestDispatcher("bookBack.jsp").forward(request, response);  }  }   1. 其他页面展示。       友情链接的页面。    11.在IE浏览器中输入: http://localhost:8080/LibrarySystem/main.jsp  实验总结与心得体会  在此次实现期末大作业中，加深了对面向对象程序设计思想的理解，能对网站功能进行分析，并设计合理的类结构，掌握了JSP网站的基本开发流程，同时对JDBC技术在实际中的开发的运用有进一步的理解和学习。Servlet在实际中的开发应用，使用Servlet，Servlet对象主要封装了对HTTP请求的处理。可以处理各种逻辑业务，在安全性扩展性以及性能方面都十分优秀。主要是对web请求的处理功能是十分的强大。同时对于JSP经典设计模式中的Model2的开发流程学习很多。在此次学习中，掌握了网站的开发能力，能够运用合理控制流程编写高效的代码；虽然在此次实验中很多不足的地方，比如在创建工程的时候没有注意到一些不需要的代码删除掉，有点投机取巧的成分，在使用Servlet时采用的是注释配置Servlet，比起3.0版本更简单快捷一点。即使对Java web的学习还不够深入，但是对此产生浓厚的兴趣，对进一步的学习有很大的帮助。在老师的提醒下是学习不在盲目。 | | | | |