东北大学秦皇岛分校

Java 程序设计实践设计报告

零售公司销售信息管理系统

学　　院	数学与统计学院
专　　业	数据科学与大数据技术
班级序号	200219
学　　号	202015116
姓　　名	梁祺若
指导教师	王子健，陆杰
开始日期	2022 年 12 月 28 日
结束日期	2023 年 1 月 6 日

教师评阅意见书

一、态度 1．工作态度（□认真、□较好、□一般、□较差） 2．出勤情况（□无缺勤、□缺勤不超过1/3、□缺勤超过1/3） 3．上交时间（□按时、□迟交1天、□迟交1天以上）
二、格式规范 4．文字部分（□符合规范、□较符合规范、□一般、□不符合规范） 5．图表部分（□符合规范、□较符合规范、□一般、□不符合规范） 6．数学公式（□符合规范、□较符合规范、□一般、□不符合规范） 7．参考文献（□符合规范、□较符合规范、□一般、□不符合规范）
三、报告内容 8．任务量和可行性（□合理、□较为合理、□一般、□不合理） 9．报告结构（□合理、□较合理、□一般、□不合理） 10．文字叙述（□清晰流畅、□较为清晰、□一般、□不清晰） 11．图表准确性（□准确、□较准确、□一般、□不准确）
四、综合能力 12．综合运用知识能力（□很强、□较强、□一般、□较弱） 13．实践与动手能力（□很强、□较强、□一般、□较弱） 14．创新意识（□很强、□较强、□一般、□较弱）
综合评价：□优秀、□良好、□中等、□及格、□不及格

评阅教师签字： 日期：

1 引言

随着互联网技术的高速发展，截至 2022 年 6 月，我国网民规模已经达到 10.51 亿，较 2021 年 12 月新增网民 1919 万，互联网普及率达 74.4%。大量的数据正在不断地产生，那么如何安全有效地存储、检索、管理他们呢？于是对数据地有效存储、高效访问、方便共享和安全控制等问题成为了信息时代一个非常重要地问题。

使用数据库可以高效且条理分明地存储数据，它使人们能够更加迅速和方便地管理数据。Java 的语法实际上是 C++语法的精华部分。 在 JAVA 编写程序的时候,可以先连接 MySQL 数据库,可以帮助我们更好的调用程序以及数据。

对于零售公司使用销售管理系统，能够支撑各种市场环境下公司业务的平稳增长、公司盈利、竞争目标的达成和市场营销目标的达成。

2 任务实现过程

2.1 任务简介

2.1.1 任务简介

为实现零售公司销售信息管理系统，建立框架，对成本信息、销售信息、库存信息、定价信息、退货信息、供货商信息和员工信息七项指标，完成增删查的功能，形成基础的信息管理系统。

2.1.2 目标成果

本次任务旨在形成管理系统，需要建立的模块有：

1）成本信息

该模块包括进货价、进货号、商品编号和数量。为零售公司计算成本信息时登记提供便利。成本表设计如下：

名	类型	长度	小数点	允许空值 (	
buyingPrice	int	11	0	☐	
entryNo	varchar	20	0	☐	1
itemNo	varchar	20	0	☐	
quantity	int	11	0	☐	

图：成本表 stock 的设计

2）销售信息

该模块包括销售号、商品编号、销售数量、销售总金额、销售日期和员工号。为零

售公司记录销售信息时登记提供便利。销售表设计如下：

名	类型	长度	小数点	允许空值 (	
transactionNo	varchar	20	0	☐	1
itemNo	varchar	20	0	☐	
saleQuantity	int	11	0	☐	
saleSumPrice	int	11	0	☐	
saleDate	varchar	20	0	☐	

图：销售表 salefact 的设计

3）库存信息

该模块包括商品编号、进货日期、进货价、进货数量和供应商编号。为零售公司记录进货信息时登记提供便利。库存表设计如下：

名	类型	长度	小数点	允许空值 (	
itemNo	varchar	20	0	☐	1
purchaseData	varchar	20	0	☐	
buyingPrice	int	11	0	☐	
itemQuantity	int	11	0	☐	
supplierNo	varchar	20	0	☐	

图：库存表 inventory 的设计

4）定价信息

该模块包括商品编号、商品名称和售价。为零售公司记录定价信息时登记提供便利。定价表设计如下：

名	类型	长度	小数点	允许空值 (	
itemNo	varchar	20	0	☐	1
itemName	varchar	20	0	☐	
salePrice	int	11	0	☐	

图：定价表 comminfo 的设计

5）退货信息

该模块包括价格、退货号、销售号、商品编号、供应商编号和退货。为零售公司记录退货信息时登记提供便利。退货表设计如下：

名	类型	长度	小数点	允许空值 (	
price	int	11	0	☐	
returnNo	varchar	20	0	☐	1
transactionNo	varchar	20	0	☐	
itemNo	varchar	20	0	☐	
supplierNo	varchar	20	0	☐	

图：退货表 returnit 的设计

6）供货商信息

该模块包括供应商名称、供应商编号、供应商地址和联系电话。为零售公司记录供应商信息时登记提供便利。供应商表设计如下：

名	类型	长度	小数点	允许空值 (	
supplierName	varchar	20	0	☐	
supplierNo	varchar	20	0	☐	1
address	varchar	20	0	☐	
teleNum	varchar	20	0	☐	

图：供应商表 supplier 的设计

7）员工信息

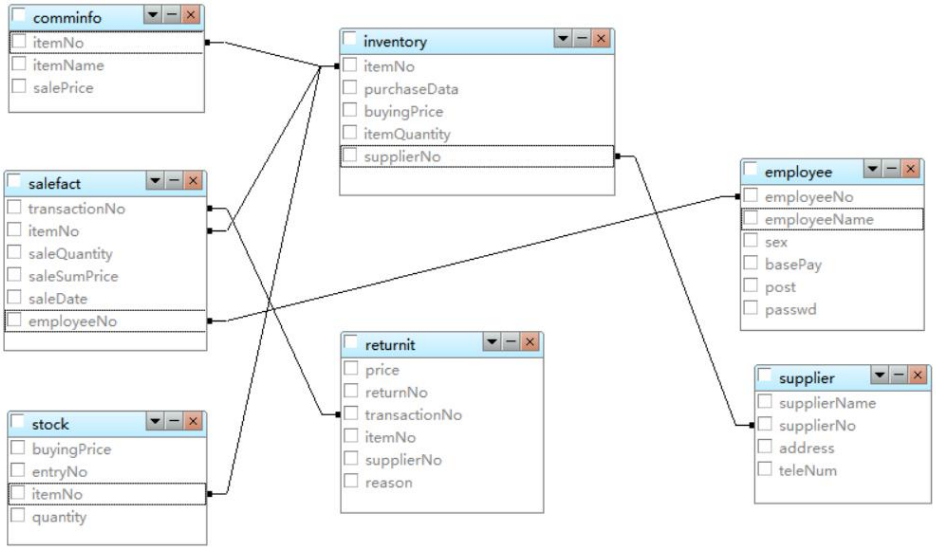
该模块包括员工号、员工姓名、性别、基本工资、职务和密码。为零售公司记录员工信息时登记提供便利，同时为登陆系统提供注册表信息。员工表设计如下：

名	类型	长度	小数点	允许空值 (	
employeeNo	varchar	20	0	☐	1
employeeName	varchar	20	0	☐	
sex	varchar	20	0	☐	
basePay	int	11	0	☐	
post	varchar	20	0	☐	

图：员工表 employee 的设计

2.1.2 框架说明

本次任务设计的数据表视图如下所示：



图：框架设计

首先登录 MySQL，开启 MySQL 服务并建立数据库。接着把 MySQL 链接到 Java EE 中处理，为每个数据表都增加增删查功能，用 JSP 动态页面实现可系统的视化和操作窗口，用 Java 代码写 Serverlet 文件实现后台的增删查功能。

2.2 任务处理过程

2.2.1 MySQL：创建数据库并输入初始数据

1. 登录 MySQL

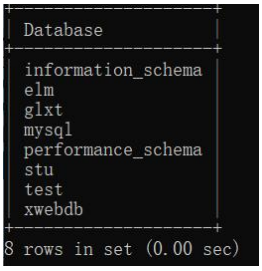
进入指令提示符界面，启动并登录 MySQL 服务。具体操作如下图所示：

图：登录 MySQL

2. 查看原有数据库

输入指令查看原有数据库。具体命令如下：

```
show databases;
```

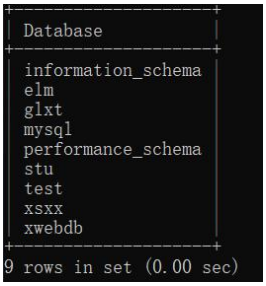


图：原有数据库

3. 新建数据库并查询

输入指令新建数据库，用来保存本次任务的所有数据表，并查询添加情况。具体命令如下：

```
create database xxsx;
```



图：新建后的数据库信息

4. 选择工作库

输入指令选择我们建立的工作库，并查看当前库的表格情况。当前数据库为空。具

体命令如下：

```
use xxsx;
```

2.2.2 Navicat：创建数据表并输入初始数据

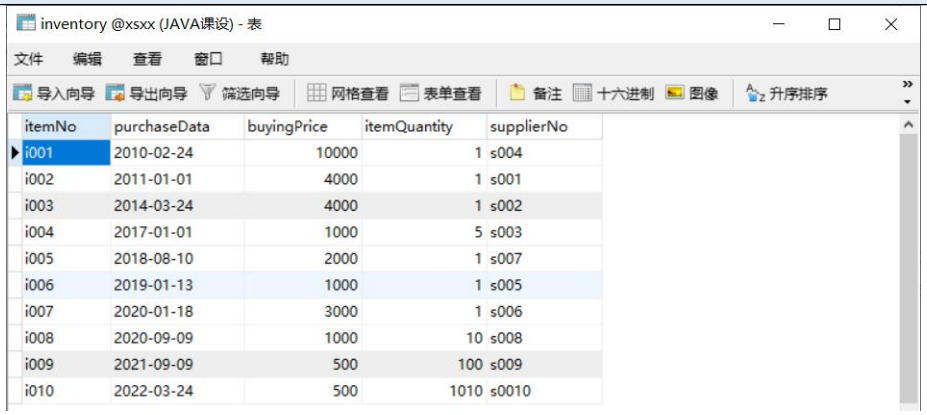
1. 新建表格并插入数据（以 inventory 表的建立为例）

在 xxsx 数据库内新建查询，建立表格，代码如下：

```
create table inventory(
    item varchar(20) PRIMARY KEY,
    purchaseData varchar(20),
    buyingPrice int,
    itemQuantity int,
    supperlierNo varchar(20)
)charset=utf8;
```

随后插入数据，代码如下：

```
INSERT INTO inventory VALUES ('i001', '2010-02-24', 10000, 1, 's004');
INSERT INTO inventory VALUES ('i002', '2011-01-01', 4000, 1, 's001');
INSERT INTO inventory VALUES ('i003', '2014-03-24', 4000, 1, 's002');
INSERT INTO inventory VALUES ('i004', '2017-01-01', 1000, 5, 's003');
INSERT INTO inventory VALUES ('i005', '2018-08-10', 2000, 1, 's007');
INSERT INTO inventory VALUES ('i006', '2019-01-13', 1000, 1, 's005');
INSERT INTO inventory VALUES ('i007', '2020-01-18', 3000, 1, 's006');
INSERT INTO inventory VALUES ('i008', '2020-09-09', 1000, 10, 's008');
INSERT INTO inventory VALUES ('i009', '2021-09-09', 500, 100, 's009');
INSERT INTO inventory VALUES ('i010', '2022-03-24', 500, 1010, 's010');
```



itemNo	purchaseData	buyingPrice	itemQuantity	supplierNo
i001	2010-02-24	10000	1	s004
i002	2011-01-01	4000	1	s001
i003	2014-03-24	4000	1	s002
i004	2017-01-01	1000	5	s003
i005	2018-08-10	2000	1	s007
i006	2019-01-13	1000	1	s005
i007	2020-01-18	3000	1	s006
i008	2020-09-09	1000	10	s008
i009	2021-09-09	500	100	s009
i010	2022-03-24	500	1010	s0010

图：添加数据

2.2.3 Eclipse：进行具体功能实现

1. 新建项目，并为每一个表格生成类

javabean 可以通过提供符合一致性设计模式的公共方法将内部域暴露成员属性，set 和 get 方法获取，随后封装成类，命名为表名方便使用。以 inventory 表为例，代码如下：

```
package javabean;

public class Inventory {
    private String itemNo;
    private String purchaseData;
    private int buyingPrice;
    private int itemQuantity;
    private String supplierNo;
    public String getItemNo() {
        return itemNo;
    }
    public void setItemNo(String itemNo) {
        this.itemNo = itemNo;
    }
    public String getPurchaseData() {
        return purchaseData;
    }
    public void setPurchaseData(String purchaseData) {
        this.purchaseData = purchaseData;
    }
    public int getBuyingPrice() {
        return buyingPrice;
    }
    public void setBuyingPrice(int buyingPrice) {
        this.buyingPrice = buyingPrice;
    }
    public int getItemQuantity() {
        return itemQuantity;
    }
    public void setItemQuantity(int itemQuantity) {
        this.itemQuantity = itemQuantity;
    }
    public String getSupplierNo() {
        return supplierNo;
    }
    public void setSupplierNo(String supplierNo) {
        this.supplierNo = supplierNo;
    }
}
```

2. 实现查找功能

在实现查找之前，首先在 Java Resources 文件夹下 src 内建立相对应的 package，用来存放对应表格的操作。随后在 package 内新建 servlet 文件，命名为 InvList.java，用来实现查询功能。doGet 方法内代码如下：

```
List <Inventory> invs = new ArrayList<Inventory>();
Connection con = null;
try{
    Class.forName("com.mysql.jdbc.Driver");
```

```
    con =
DriverManager.getConnection("jdbc:mysql://127.0.0.1:3306/xsxx?characterEnco
ding=utf8","root","");
    PreparedStatement pst = con.prepareStatement("select * from inventory");
    ResultSet rs = pst.executeQuery();
    while(rs.next()){
        Inventory inv = new Inventory();
        inv.setItemNo(rs.getString("itemNo"));
        inv.setPurchaseData(rs.getString("purchaseData"));
        inv.setBuyingPrice(rs.getInt("buyingPrice"));
        inv.setItemQuantity(rs.getInt("itemQuantity"));
        inv.setSupplierNo(rs.getString("supplierNo"));
        invs.add(inv);
    }
    request.setAttribute("invs", invs);
    request.getRequestDispatcher("inventory.jsp").forward(request,response);
}catch(Exception e){
    e.printStackTrace();
}
finally{
    try{
        if(null != con)con.close();
    }catch(SQLException e){
        e.printStackTrace();
    }
}
```

然后新建 JSP 文件，来实现功能可视化动态 Web 界面，命名为 invIns.jsp。在编写时考虑到后期需要增加添加和删除功能，因此在这里提前写好超链接。代码如下：

```
<%@ page language="java" contentType="text/html; charset=utf-8"
    pageEncoding="utf-8" import="java.util.*,javabean.*"%>
<!DOCTYPE>
<html>
<head>
<meta charset="utf-8">
<title>查询库存信息</title>
</head>
<body>
    <%
    List <Inventory> invs = (List <Inventory>)request.getAttribute("invs");
    %>
    <table border="1">
        <tr>
            <th>商品编号</th>
            <th>进货日期</th>
            <th>进货价</th>
            <th>进货数量</th>
            <th>供应商编号</th>
        </tr>
    <%
        for(Inventory inv : invs){
    %>
        <tr>
            <td><%=inv.getItemNo()%></td>
            <td><%=inv.getPurchaseData()%></td>
            <td><%=inv.getBuyingPrice()%></td>
            <td><%=inv.getItemQuantity()%></td>
```

```
            <td><%=inv.getSupplierNo()%></td>
            <td><a href="InvDel?itemNo=<%=inv.getItemNo() %>">删除</a></td>
        </tr>
    <%}%>
    </table>
    <a href="ToInvIns">添加库存信息</a>
</body>
</html>
```

3. 实现添加功能

要实现添加功能，首先我们需要一个添加界面。为了到达添加界面并传输添加页面中所添加的内容，我们需要一个 Servlet 文件实现，命名为 ToInvIns.java。doGet 方法内代码如下：

```
request.setCharacterEncoding("utf-8");
response.setContentType("test/html;charset=utf-8");
request.getRequestDispatcher("invIns.jsp").forward(request, response);
```

随后，我们设计引导用户输入新库存信息的 JSP 动态网络页面，命名为 invIns.jsp。

```
<%@ page language="java" contentType="text/html; charset=utf-8"
    pageEncoding="utf-8"%>
<!DOCTYPE>
<html>
<head>
<meta charset="utf-8">
<title>添加库存信息</title>
</head>
<body>
    <form action="InvIns" method="post">
        商品编号：<input type="text" name="itemNo"><br />
        进货日期：<input type="text" name="purchaseData"><br />
        进货价：<input type="text" name="buyingPrice"><br />
        进货数量：<input type="text" name="itemQuantity"><br />
        供应商编号：<input type="text" name="supplierNo"><br />
        <input type="submit" value="添加库存信息">
        <input type="reset">
    </form>
</body>
</html>
```

最后，新建一个 Servlet 文件，编写 Java 代码实现添加功能，命名为 InvIns.java。doGet 方法内代码如下：

```
request.setCharacterEncoding("utf-8");
response.setContentType("test/html;charset=utf-8");
String itemNo = request.getParameter("itemNo");
String purchaseData = request.getParameter("purchaseData");
int buyingPrice = Integer.parseInt(request.getParameter("buyingPrice"));
int itemQuantity = Integer.parseInt(request.getParameter("itemQuantity"));
String supplierNo = request.getParameter("supplierNo");
Connection con = null;
try{
    Class.forName("com.mysql.jdbc.Driver");
    con =
```

```
DriverManager.getConnection("jdbc:mysql://127.0.0.1:3306/xsxx?characterEncoding=utf8","root","");
    PreparedStatement pst = con.prepareStatement("insert into inventory values (? , ? , ? , ? , ?)");
    pst.setString(1, itemNo);
    pst.setString(2, purchaseData);
    pst.setInt(3, buyingPrice);
    pst.setInt(4, itemQuantity);
    pst.setString(5, supplierNo);
    pst.execute();
}catch(Exception e){
    e.printStackTrace();
}
finally{
    try{
        if(null != con)con.close();
    }catch(SQLException e){
        e.printStackTrace();
    }
}
request.getRequestDispatcher("InvList").forward(request, response);
```

4. 实现删除功能

新建一个 Servlet 文件，编写 Java 代码实现删除功能，命名为 InvDel.java。doGet 方法内代码如下：

```
request.setCharacterEncoding("utf-8");
response.setContentType("test/html;charset=utf-8");
String itemNo = request.getParameter("itemNo");
Connection con = null;
try{
    Class.forName("com.mysql.jdbc.Driver");
    con =
DriverManager.getConnection("jdbc:mysql://127.0.0.1:3306/xsxx?characterEncoding=utf8","root","");
    PreparedStatement pst = con.prepareStatement("delete from inventory where itemNo=?");
    pst.setString(1, itemNo);
    pst.execute();
}catch(Exception e){
    e.printStackTrace();
}
finally{
    try{
        if(null != con)con.close();
    }catch(SQLException e){
        e.printStackTrace();
    }
}
request.getRequestDispatcher("InvList").forward(request, response);
```

5. 添加主界面

作为管理系统，我们需要将我们设计好的七个表格组合在同一界面，实现资源整合。新建 JSP 动态页面，命名为 main.jsp，作为我们访问零售公司销售信息管理系统的主界面。代码如下：

```
<%@ page language="java" contentType="text/html; charset=utf-8"
    pageEncoding="utf-8"%>
<!DOCTYPE>
<html>
<head>
<meta charset="utf-8">
<title>导航页面</title>
</head>
<frameset rows="10%,90%">
    <frame src="title.jsp" />
    <frameset cols="10%,90%">
        <frame src="navicate.jsp" />
        <frame src="InvList" name="view" />
    </frameset>
</frameset>
</html>
```

新建 JSP 动态界面保存我们创建的数据表名，并设计超链接进行跳转，命名为 navicate.jsp。代码如下：

```
<%@ page language="java" contentType="text/html; charset=utf-8"
    pageEncoding="utf-8"%>
<!DOCTYPE>
<html>
<head>
<meta charset="utf-8">
<title>Insert title here</title>
</head>
<body>
    <a href="StoList" target="view">成本信息</a><br />
    <a href="SaleList" target="view">销售统计</a><br />
    <a href="InvList" target="view">库存信息</a><br />
    <a href="ComList" target="view">定价信息</a><br />
    <a href="ReList" target="view">退货信息</a><br />
    <a href="SupList" target="view">供货商信息</a><br />
    <a href="EmpList" target="view">员工信息</a><br />
</body>
</html>
```

6. 添加登录系统

为了管理系统的安全性，我们为此系统添加登录界面。登录信息关联到 employee 表中，方便录入。新建 JSP 动态页面作为登录界面，命名为 login.jsp。代码如下：

```
<%@ page language="java" contentType="text/html; charset=utf-8"
    pageEncoding="utf-8"%>
<!DOCTYPE>
<html>
<head>
<meta charset="utf-8">
<title>登录</title>
</head>
<body>
    <form action="Login" method="post">
        员工姓名：<input type="text" name="employeeName"/><br />
        密码：<input type="password" name="passwd"/><br />
```

```
        <input type="submit" value="登录"/>
        <input type="reset" />
    </form>
</body>
</html>
```

新建 package，命名为 login，在 package 内新建 Servlet 文件，读取 employee 表中的姓名和密码作为登录识别，命名为 Login.java。doGet 方法内代码如下：

```
request.setCharacterEncoding("utf-8");
response.setContentType("test/html;charset=utf-8");
String employeeName = request.getParameter("employeeName");
String passwd = request.getParameter("passwd");

Connection con = null;
try{
    Class.forName("com.mysql.jdbc.Driver");
    con =
DriverManager.getConnection("jdbc:mysql://127.0.0.1:3306/xsxx?characterEnco
ding=utf8","root","");
    PreparedStatement pst = con.prepareStatement("select * from employee where
employeeName=? and passwd=?");
    pst.setString(1, employeeName);
    pst.setString(2, passwd);
    ResultSet rs = pst.executeQuery();
    if(rs.next()){
        request.getSession().setAttribute("emps", employeeName);
        response.sendRedirect("main.jsp");
    }else{
        response.sendRedirect("login.jsp");
    }

}catch(Exception e){
    e.printStackTrace();
}
finally{
    try{
        if(null != con)con.close();
    }catch(SQLException e){
        e.printStackTrace();
    }
}
```

为安全性考虑，我们为了防止非员工越过登录界面访问主界面，需要再为系统增加过滤器。只允许访客访问登录界面，登录成功后才可以进入主界面做读写操作。新建 package，命名为 filter。在 package 内新建 Filter 文件，命名为 Filter.java。doFilter 方法内代码如下：

```
HttpServletRequest req = (HttpServletRequest)request;
String uri = req.getRequestURI();
if(uri.equals("/XSXXXT/login.jsp")|| uri.equals("/XSXXXT/Login")){
    chain.doFilter(request, response);
}else{
    Object obj = req.getSession().getAttribute("emps");
    if(null != obj){
        chain.doFilter(request, response);
```

```
    }
    else{
        HttpServletResponse resp = (HttpServletResponse)response;
        resp.sendRedirect("login.jsp");
    }
}
```

3 系统运行结果及使用说明

运行代码后我们的零售公司学生信息管理系统正式运行。此时，我们只需在浏览器输入 http://localhost:8080/XSXXXT/login.jsp 进入登录界面，如下图所示：

图：登录界面

在员工姓名框和密码框内输入正确的员工姓名及密码，进入零售公司销售信息管理系统。进入后主界面为导航页面，其顶部为系统名，左侧为导航栏，点击超链接可以转移到不同的信息表中，右侧为对应点击的超链接的表格，此为查询功能的实现。导航页面如下图所示：

图：导航页面

单击每行信息右侧的“删除”按钮即可完成删除操作。以 stock 成本数据表最后一行数据为例，单击右侧“删除”即可删除本行信息。

图：删除后的成本数据表

单击表格下方的“添加成本信息”按钮，即可跳转至添加界面。如下图所示：

图：添加成本信息页面

输入成本信息后，单击“添加成本信息”按钮，即可将信息添加至表格。如下图所示：

图：输入新成本信息

随后回到导航页面即可看到新添加的数据。如下图所示：

图：新成本信息表格

如果想要对数据进行修改，可用先删除后添加的方式修改。

结　　论

本次对成本信息、销售信息、库存信息、定价信息、退货信息、供货商信息和员工信息七项指标，完成增删查的功能，形成基础的信息管理系统。设计数据库时将各个表中的关键信息用外键相连，比如库存信息表中的 itemNo 属性就与定价信息表、销售信息表、成本信息表和退货信息表中的 itemNo 属性相连，库存信息表中的 supplierNo 属性就与供货商信息表中的 supplierNo 属性相连等等。

本次任务还可以在添加修改按钮，将修改操作以新页面的形式，让员工填写修改内容。还可以添加操作成功与失败提示，对于登录信息不匹配，或者添加非法格式信息时进行提醒。

参考文献