

毕业设计 (论文)

题目: *****商店智能管理系统

学 院： 计算机科学学院

专业班级: 计算机科学与技术 06 级 2 班

指导教师: _____ 职称: _____

学生姓名: _____

学 号: 40605010201

摘 要

当前,我国商业正处在一个高速发展的时期,也是我国商业由传统商业向现代化商业转变的重要时期,正在经历由经验决策到信息决策转变的时期。如何及时准确地获得各种信息,了解市场的发展趋势,来预测、指导经营活动;如何及时准确地掌握商店内部动态和信息,已成为商店最关心的问题。

本系统基于成熟的 C/S 模式,采用 C#作为编程语言,Microsoft Visual Studio 2008 作为开发工具,以 SQL Server 2008 作为后台的数据库,利用 ADO.NET 数据库访问技术实现对数据库的各种管理操作。系统分为两部分:第一部分针对员工使用,包括的功能如:查询商品信息、查询供应商信息、商品销售等;第二部分面向店主使用,包括的功能如:对商品信息、供应商信息、会员和员工信息的增删查改,计算利润等。

为了避免用户打开很多窗口导致桌面混乱,设计过程中使用 MdiParent 方法将所有本系统的子窗体放置于主窗体之内,无论用户怎么拖动,窗体都不会离开主窗体。设计过程中,首先建立了系统的应用原型,然后在此基础上进行需求迭代,详细设计时不断修改和完善,经过测试阶段反复调试和验证,最终形成达到设计要求的可行系统。

关键词: C#, ADO.NET, SQL Server, C/S 模式, 信息化管理

ABSTRACT

At present, our business is in a period of rapid development, but also our business from traditional to modern commercial business important period of change, is going through the experience of making changes to the information decision-making period. How to timely and accurate access to information, understand the market trends, to predict and guide business activities; how timely and accurate grasp the internal dynamics and information store has become shop most concerned about.

The system is based on mature C / S mode, using C # as the programming language, Microsoft Visual Studio 2008 as a development tool to SQL Server 2008 database as a background, using ADO.NET database access technology on a variety of database management operations. System is divided into two parts: the first part for employees to use, including features such as: check product information, inquiries, supplier information, product sales; the second part of the owner for the use, including features such as: product information, supplier information , members and staff additions and deletions of information search change, calculating profits.

In order to avoid the user to open many windows cause desktop clutter, the design methods used during the MdiParent all sub-form of the system placed within the main form, no matter how the user drag the form will not leave the main form. Design process, the first application of the system to establish a prototype, and then based on the needs in this iteration, detailed design, constantly amended and improved, 经过 test phase repeated debugging and verification, form a viable system to meet the design requirements.

KEY WORDS: C#, ADO.NET, SQL Server, C/S mode, Information management

目 录

第1章 引言.....	1
1.1 问题的提出.....	1
1.2 研究现状.....	1
1.3 研究内容.....	2
1.4 研究方法和技术路线.....	2
第2章 系统开发平台和语言介绍.....	3
2.1 Microsoft Visual Studio 2008 的简介.....	3
2.1.1 快速的应用程序开发.....	3
2.1.2 高效的团队协作.....	3
2.1.3 突破性的用户体验.....	4
2.2 Microsoft SQL Server 2008 简介.....	4
2.3 C#语言简介.....	5
2.4 ADO.NET 简介.....	5
2.4.1 SqlConnection 类.....	6
2.4.2 Command 对象.....	6
2.4.3 SqlDataReader 类.....	6
2.4.4 DataSet 对象.....	6
2.4.5 SqlDataAdapter 类.....	7
2.4.6 DataTable 类.....	7
第3章 系统分析与总体设计.....	8
3.1 需求分析.....	8
3.2 系统模块设计.....	8
3.2.1 系统功能模块分析.....	8
3.2.2 系统流程分析.....	9
3.3 数据库设计.....	10
3.3.1 数据库逻辑结构设计.....	10
3.3.2 系统 E-R 图设计.....	12
第4章 系统详细设计.....	14
4.1 底层数据操作模块设计.....	14
4.2 系统登录界面与主界面设计.....	16
4.2.1 登录界面设计.....	16

4.2.2 系统主界面设计.....	17
4.3 商品管理模块设计.....	19
4.3.1 商品信息查询模块.....	19
4.3.2 商品信息维护模块.....	20
4.4 供应商管理模块设计.....	25
4.4.1 供应商信息查询模块.....	25
4.4.2 供应商信息维护模块.....	26
4.5 员工管理模块设计.....	31
4.5.1 员工信息查询模块.....	31
4.5.2 员工信息维护模块.....	32
4.6 会员管理模块设计.....	37
4.6.1 会员信息查询模块.....	37
4.6.2 会员信息维护模块.....	38
4.7 销售管理模块设计.....	43
4.7.1 出售商品模块.....	43
4.7.2 利润计算模块.....	44
4.8 系统管理模块.....	45
4.8.1 权限管理模块.....	45
4.8.2 注销登录模块.....	49
4.8.3 退出系统模块.....	49
第5章 系统测试与性能分析.....	50
第6章 总结.....	51
致 谢.....	52
参考文献.....	53

第1章 引言

1.1 问题的提出

随着人类社会的不断发展,计算机越来越发挥着重要的作用。它提高了人们的工作效率,方便了人们的生活,为人们增添了不少乐趣,甚至于改变了人们传统的生活方式。计算机带来的改变是巨大的,人们可以通过简单的输入几个字符,就能轻松地得到我们想要的东西。

该商店目前存在的问题是:信息化不足,计算机仅仅用于娱乐和查找资料,大量的日常工作皆是手工处理,因此工作效率低,仓库管理也非常不合理,不能及时根据最近的销售量及时调整进货量以及进货品种。

通过该系统,西安昌盛商店可以实现:1、减轻管理者和工作人员的工作量 2、提高商店管理的规范性 3、及时查看商店的盈亏情况 4、合理安排进货时间、数量以及品种。

1.2 研究现状

目前市场上针对大中型企业所做的 ERP 系统层出不穷,但是针对类似西安昌盛商店这种小的商店,却无人问津。孰不知随着改革开放以来,我国经济的飞速发展,这类小商店如雨后春笋般出现在大街小巷。这样子的小商店没有独立的部门,更不会需要如 ERP 企业资源计划系统这样子庞大而又昂贵的软件。但是随着计算机在我国的普及,越来越多的店主系统将手头上的诸多所示交给店里面的计算机来完成,以帮助自己更高效的工作。此时,一款必须能够适应此类店铺的软件应运而生。

此软件前身即为库存管理系统。随着全国零售商业的高速发展,其仓库管理变得日益复杂,日常所需处理的数据量渐渐庞大,商业运转的中间环节也越来越多,原先主要靠人工管理和简单计算机管理的方法,显然已无法适应企业规模的不断扩大。鉴于这种情况,库存管理系统应运而生。现在的大型库存管理系统可实现诸如出入库操作,移库操作,库存物品损毁、修复记录,盘点计划自动生成,设置盘点状态,盘点结果确认,库存订货处理等一系列复杂的功能。未来的发展将进一步的智能化,更大程度的替代人工管理。

该系统面向小型商店而产生,加入了此类商店需要的各种其他的功能。使软件更加贴切此类商店的需求。简化了大型库存管理系统中高级的功能,使软件更加易用。同时也降低了软件开发的成本,使此类小型店铺更易于接受该软件。

随着该系统的发展，未来此软件必将普及于大街小巷的小商店。使他们的工作更加高效，使这些小商店的老板们能够将更多的时间用在如何经营店铺，而不是浪费在人工计算利润等繁琐的事情上。

1.3 研究内容

在了解该商店业务流程的基础上，结合以前所学的编程知识和软件工程知识，使用当今最为流行的编程语言——C#，将需求转化成为功能模块再加以实现。

论文各章简介：

第1章：包含选择论题的背景和原因，以及此论题的研究现状。

第2章：系统开发平台的基本知识。

第3章：系统分析与总体设计

第4章：系统详细设计

第5章：系统测试与性能分析

第6章：总结本次开发运行结果

1.4 研究方法和技术路线

研究方法：首先抱着认真和严谨的态度去研究。具体方法如下：

- 多用时间去看 C# 方面的书籍，给自己打好良好的理论知识。
- 因题目中最重要的就是编程部分，所以在软件设计时，应使软件具有较强的可维护性和更新性。
- 题目中还涉及到较多的数据库知识，所以也要多多研究数据库方面的知识。

技术路线：使用强大的 Visual Studio 2008 和 SQL Server 2008 作为编程的工具和数据库管理工具。采用成熟的 C/S 模式完成软件部分的系统设计。平台选用最先进的 .NET Framework 3.5。

第2章 系统开发平台和语言介绍

2.1 Microsoft Visual Studio 2008 的简介

Microsoft Visual Studio 2008 使开发人员能够快速创建高质量、用户体验丰富而又紧密联系的应用程序，充分展示了 Microsoft 开发智能客户端应用程序的构想。借助 Visual Studio 2008，采集和分析信息将变得更为简单便捷，业务决策也会因此变得更为有效。任何规模的组织都可以使用 Visual Studio 2008 快速创建能够利用 Windows Vista 和 2007 Visual Studio system 的更安全、更易于管理并且更可靠的应用程序。

Visual Studio 2008 在三个方面为开发人员提供了关键改进：

- 快速的应用程序开发
- 高效的团队协作
- 突破性的用户体验

Visual Studio 2008 提供了高级开发工具、调试功能、数据库功能和创新功能，帮助在各种平台上快速创建当前最先进的应用程序。Visual Studio 2008 包括各种增强功能，例如可视化设计器（使用 .NET Framework 3.5 加速开发）、对 Web 开发工具的大量改进，以及能够加速开发和处理所有类型数据的语言增强功能。Visual Studio 2008 为开发人员提供了所有相关的工具和框架支持，帮助创建引人注目的、令人印象深刻并支持 AJAX 的 Web 应用程序。开发人员能够利用这些丰富的客户端和服务端框架轻松构建以客户为中心的 Web 应用程序，这些应用程序可以集成任何后端数据提供程序、在任何当前浏览器内运行并完全访问 ASP.NET 应用程序服务和 Microsoft 平台。

2.1.1 快速的应用程序开发

为了帮助开发人员迅速创建先进的软件，Visual Studio 2008 提供了改进的语言和数据功能，例如语言集成的查询 (LINQ)，各个编程人员可以利用这些功能更轻松地构建解决方案以分析和处理信息。Visual Studio 2008 还使开发人员能够从同一开发环境内创建面向多个 .NET Framework 版本的应用程序。开发人员能够构建面向 .NET Framework 2.0、3.0 或 3.5 的应用程序，意味他们可以在同一环境中支持各种各样的项目。

2.1.2 高效的团队协作

Visual Studio 2008 提供了帮助开发团队改进协作的扩展的和改进的服务

项目,包括帮助将数据库专业人员和图形设计人员加入到开发流程的工具。使用 Microsoft .NET Framework 3.0, .NET Framework 提供了用于解决常见编程任务的构建基块(预制的软件),从而能够快速构造具有出色的最终用户体验的紧密联系的应用程序。在 .NET Framework 模型业务流程上有效构建的紧密联系的应用程序有利于在异类环境中实现系统集成。Visual Studio 和 .NET Framework 的结合使用减少了对公用管道代码的需要,从而缩短了开发时间并使开发人员能够集中精力解决业务问题。.NET Framework 3.5 是在 .NET Framework 3.0 的基础上构建的更高版本。得到增强的功能领域包括基类库、Windows workflow foundation、Windows Communication Foundation、Windows Presentation Foundation 和 Windows CardSpace。

2.1.3 突破性的用户体验

Visual Studio 2008 为开发人员提供了在最新平台上加速创建紧密联系的应用程序的新工具,这些平台包括 Web、Windows Vista、Visual Studio 2007、SQL Server 2008 和 Windows Server 2008。对于 Web, ASP.NET AJAX 及其他新技术使开发人员能够迅速创建更高效、交互式更强和更个性化的新一代 Web 体验。

2.2 Microsoft SQL Server 2008 简介

SQL Server 是一个关系数据库管理系统。它最初是由 Microsoft Sybase 和 Ashton-Tate 三家公司共同开发的,于 1988 年推出了第一个 OS/2 版本。在 Windows NT 推出后,Microsoft 与 Sybase 在 SQL Server 的开发上就分道扬镳了,Microsoft 将 SQL Server 移植到 Windows NT 系统上,专注于开发推广 SQL Server 的 Windows NT 版本。Sybase 则较专注于 SQL Server 在 UNIX 操作系统上的应用。

SQL Server 2008 是一个重大的产品版本,它推出了许多新的特性和关键的改进,使得它成为至今为止的最强大和最全面的 SQL Server 版本。

这个平台有以下特点:

- 可信的——使得公司可以以很高的安全性、可靠性和可扩展性来运行他们最关键任务的应用程序。
- 高效的——使得公司可以降低开发和管理他们的数据基础设施的时间和成本。
- 智能的——提供了一个全面的平台,可以在你的用户需要的时候给他发送观察和信息。

2.3 C#语言简介

C#是一种安全的、稳定的、简单的、优雅的,由C和C++衍生出来的面向对象的编程语言。它在继承C和C++强大功能的同时去掉了一些它们的复杂特性(例如没有宏和模版,不允许多重继承)。C#综合了VB简单的可视化操作和C++的高运行效率,以其强大的操作能力、优雅的语法风格、创新的语言特性和便捷的面向组件编程的支持成为.NET开发的首选语言。并且C#成为ECMA与ISO标准规范。C#看似基于C++写成,但又融入其它语言如Pascal、Java、VB等。

微软C#语言定义主要是从C和C++继承而来的,而且语言中的许多元素也反映了这一点。C#在设计者从C++继承的可选项方面比Java要广泛一些(比如说struts),它还增加了自己新的特点(比方说源代码版本定义),但它还太不成熟,不可能挤垮Java。C#还需要进化成一种开发者能够接受和采用的语言,而微软当前为它的这种新语言大造声势也是值得注意的。目前大家的反应是:“这是对Java的反击。”

2.4 ADO.NET 简介

ADO.NET的名称起源于ADO(ActiveX Data Objects),这是一个广泛的类组,用于在以往的Microsoft技术中访问数据。之所以使用ADO.NET名称,是因为Microsoft希望表明,这是在.NET编程环境中优先使用的数据库访问接口。它提供了平台互用性和可伸缩的数据访问。ADO.NET增强了对非连接编程模式的支持,并支持RICH XML。由于传送的数据都是XML格式的,因此任何能够读取XML格式的应用程序都可以进行数据处理。事实上,接受数据的组件不一定要是ADO.NET组件,它可以是基于一个Microsoft Visual Studio的解决方案,也可以是任何运行在其它平台上的任何应用程序。

ADO.NET是一组用于和数据源进行交互的面向对象类库。通常情况下,数据源是数据库,但它同样也能够是文本文件、Excel表格或者XML文件。ADO.NET允许和不同类型的数据源以及数据库进行交互。然而并没有与此相关的一系列类来完成这样的工作。因为不同的数据源采用不同的协议,所以对于不同的数据源必须采用相应的协议。一些老式的数据源使用ODBC协议,许多新的数据源使用OleDb协议,并且现在还不断出现更多的数据源,这些数据源都可以通过.NET的ADO.NET类库来进行连接。

ADO.NET 提供与数据源进行交互的公共方法，但是对于不同的数据源采用一组不同的类库。这些类库称为 Data Providers，并且通常是以与之交互的协议和数据源的类型来命名的。表 1 列处了一些常见的 data providers，以及它们所使用的 API 前缀，和允许进行交互的数据源类型。

2.4.1 SqlConnection 类

和数据库交互，你必须连接它。连接帮助指明数据库服务器、数据库名字、用户名、密码，和连接数据库所需要的其它参数。connection 对象会被 command 对象使用，这样就能够知道是在哪个数据库上面执行命令。

与数据库交互的过程意味着你必须指明想要发生的操作。这是依靠 command 对象执行的。你使用 command 对象来发送 SQL 语句给数据库。command 对象使用 connection 对象来指出与哪个数据库进行连接。你能够单独使用 command 对象来直接执行命令，或者将一个 command 对象的引用传递给 SqlDataAdapter，它保存了一组能够操作下面描述的一组数据的命令。

2.4.2 Command 对象

成功于数据建立连接后，就可以用 COMMAND 对象来执行查询，修改，插入，删除等命令；COMMAND 对象常用的方法有 ExecuteReader 方法，ExcrteScalar() 方法 ExecuteNonQuery() 方法；插入数据可用 ExecuteNonQuery() 方法来执行插入命令。

2.4.3 SqlDataReader 类

许多数据操作要求你只是读取一串数据。data reader 对象允许你获得从 command 对象的 SELECT 语句得到的结果。考虑性能的因素，从 data reader 返回的数据都是快速的且只是“向前”的数据流。这意味着你只能按照一定的顺序从数据流中取出数据。这对于速度来说是有好处的，但是如果你需要操作数据，更好的办法是使用 DataSet。

2.4.4 DataSet 对象

DataSet 对象是数据在内存中的表示形式。它包括多个 DataTable 对象，而 DataTable 包含列和行，就象一个普通的数据库中的表。你甚至能够定义表之间的关系来创建主从关系 (parent-child relationships)。DataSet 是在特定的场景下使用——帮助管理内存中的数据并支持对数据的断开操作的。DataSet 是被所有 Data Providers 使用的对象，因此它并不像 Data Provider 一样需要特别的前缀。

2.4.5 SqlDataAdapter 类

某些时候你使用的数据主要是只读的，并且你很少需要将其改变至底层的数据源。同样一些情况要求在内存中缓存数据，以此来减少并不改变的数据被数据库调用的次数。Data adapter 通过断开模型来帮助你方便的完成对以上情况的处理。当在一单批次的对数据库的读写操作的持续的改变返回至数据库的时候，Data adapter 填充 (fill) DataSet 对象。data adapter 包含对连接对象以及当对数据库进行读取或者写入的时候自动的打开或者关闭连接的引用。另外，data adapter 包含对数据的 SELECT, INSERT, UPDATE 和 DELETE 操作的 command 对象引用。你将为 DataSet 中的每一个 table 都定义 data adapter，它将为你照顾所有与数据库的连接。所有你将做的工作是告诉 data adapter 什么时候装载或者写入到数据库。

2.4.6 DataTable 类

DataTable 是一个数据网格控件。它可以被应用在 VB 和 ASP 上。它无须代码就可以简单的绑定数据库。它具有微软风格的用户界面。

第3章 系统分析与总体设计

3.1 需求分析

西安昌盛商店长期以来都是采用传统的销售模式，存在着以下问题：

- (1) 手工记录每日的售出商品，晚上老板伏案工作，计算盈亏情况。
- (2) 由于商品非常之多，而且商品上没有标售价，因此会经常出现售货员忘记售价的情况。给营业造成诸多不便。
- (3) 手工记录销售情况，经常由于出现一时忙碌，售货员忘了记录商品的销售情况。
- (4) 供应商方面的管理也存在各种问题，经常会存在找不到供应商名片的情况。
- (5) 会员管理方面，由于没有办理会员卡，所以存在新来的员工不认识老会员的问题。

所以我们的系统应该具备以下功能：

- (1) 对商品按类别分开，使用户分类查询、添加、修改、删除商品信息。
- (2) 按照供应商所供应的货物将供应商也分类管理。
- (3) 录入会员信息，方便员工查询。
- (4) 使用计算机统计商品的销售情况，以及盈利情况。

3.2 系统模块设计

3.2.1 系统功能模块分析

根据需求分析，西安昌盛商店智能管理系统的系统模块图如下图所示。



图 3-1 西安昌盛商店智能管理系统模块图

登录界面模块：登录名采用下拉框形式，有效地避免了用户输入错误，方便用户使用，密码输入框采用加密文字显示，避免用户在输入时泄露密码。

商品管理模块：包含日用类、食品类、烟草类、劳保类这四大类商品信息的查询和维护。根据登录权限的不同，显示或隐藏部分子功能。

供应商管理模块：包含日用类、食品类、烟草类、劳保类这四大类供应商信息的查询和维护。根据登录权限的不同，显示或隐藏部分子功能。

员工管理模块：包含员工信息查询和维护两个子功能。“员工”权限不可见此模块。

会员管理模块：包含会员信息的查询和维护两个子功能。“员工”权限不可见会员信息维护子功能。

销售管理模块：包含商品出售和利润计算两个子功能。“员工”权限不可见利润计算子功能。

系统管理模块：包含注销登录和退出系统两个子功能。

3.2.2 系统流程分析

该系统根据权限的不同，登陆后能使用的功能也不同。权限则分为：admin(店主)和 normal(员工)两种。店主身份可以使用所有的功能，而员工只能使用其中类似查询这样的基本功能。员工登录系统的流程图如图 3-2 所示。

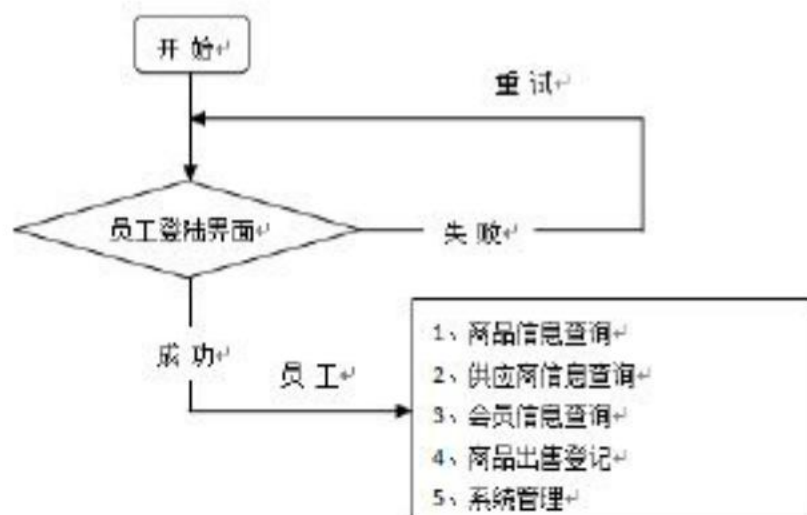


图 3-2 员工登录系统流程图

店主登录系统的流程图如图 3-3 所示。

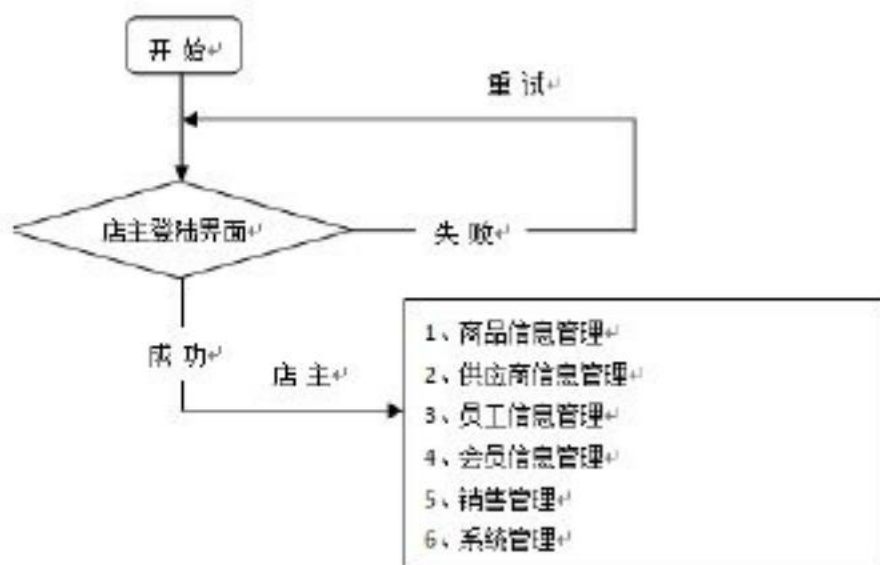


图 3-3 店主登录系统流程图

3.3 数据库设计

本系统的数据库采用的是 SQL Server 2008 数据库管理系统。根据系统的需求分析和功能模块设计，本系统定义的数据库中包含 8 个表：员工信息表 Employee，会员信息表 Member，供应商信息表 Supplier，劳保类商品信息表 LaoBao，日用类商品信息表 RiYong，食品类商品信息表 ShiPin，烟草类商品信息表 YanCao，登录权限表 User_Right。

3.3.1 数据库逻辑结构设计

(1) 员工信息表 Employee：用来保存员工信息，结构如表 3-1 所示。

表 3-1 员工信息表

编号	字段名称	数据结构	说明
1	EID	Int	员工编号
2	ENAME	Varchar(50)	员工姓名
3	EAGE	Int	员工年龄
4	ESex	Varchar(50)	员工性别
5	EHometown	Varchar(50)	员工籍贯

(2) 会员信息表 Member：用来保存会员信息，结构如表 3-2 所示。

表 3-2 会员信息表

编号	字段名称	数据结构	说明
1	MemID	Int	会员编号
2	MemName	Varchar(50)	会员姓名
3	MemAge	Int	会员年龄
4	MemSex	Varchar(50)	会员性别
5	MemHometown	Varchar(50)	会员籍贯

(3) 供应商信息表 Supplier: 用来保存供应商信息, 结构如表 3-3 所示。

表 3-3 供应商信息表

编号	字段名称	数据结构	说明
1	SID	Int	供应商编号
2	SortID	Int	商品种类编号
3	SName	Varchar(50)	供应商名称
4	SAddr	Varchar(50)	供应商地址
5	SPhone	Varchar(50)	供应商电话

(4) 劳保类商品信息表 LaoBao: 用来保存劳保类商品的信息, 结构如表 3-4 所示。

表 3-4 劳保类商品信息表

编号	字段名称	数据结构	说明
1	GoodsID	Int	商品编号
2	GoodsName	Varchar(50)	商品名称
3	GoodsPrice	Float	商品价格
4	MemPrice	Float	会员价格
5	Unit	Varchar(50)	计价单位
6	Stock	int	库存余量

(5) 日用类商品信息表 RiYong: 用来保存日用类商品的信息, 结构如表 3-5 所示。

表 3-5 日用类商品信息表

编号	字段名称	数据结构	说明
1	GoodsID	Int	商品编号
2	GoodsName	Varchar(50)	商品名称
3	GoodsPrice	Float	商品价格
4	MemPrice	Float	会员价格

5	Unit	Varchar(50)	计价单位
6	Stock	int	库存余量

(6) 食品类商品信息表 ShiPin: 用来保存食品类商品的信息, 结构如表 3-6 所示。

表 3-6 食品类商品信息表

编号	字段名称	数据结构	说明
1	GoodsID	Int	商品编号
2	GoodsName	Varchar(50)	商品名称
3	GoodsPrice	Float	商品价格
4	MemPrice	Float	会员价格
5	Unit	Varchar(50)	计价单位
6	Stock	int	库存余量

(7) 烟草类商品信息表 YanCao: 用来保存烟草类商品的信息, 结构如表 3-7 所示。

表 3-7 烟草类商品信息表

编号	字段名称	数据结构	说明
1	GoodsID	Int	商品编号
2	GoodsName	Varchar(50)	商品名称
3	GoodsPrice	Float	商品价格
4	MemPrice	Float	会员价格
5	Unit	Varchar(50)	计价单位
6	Stock	int	库存余量

(8) 登录权限表 User_Right: 用来保存用户的登录信息, 结构如表 3-8 所示。

表 3-8 登录权限表

编号	字段名称	数据结构	说明
1	Login ID	Varchar(50)	登录编号
2	LoginName	Varchar(50)	登录名称
3	LoginPwd	Varchar(50)	登录密码
4	LoginRight	Varchar(50)	登录权限

3.3.2 系统 E-R 图设计

由于店员和店主之间存在不同的权限, 所能使用的功能也不同, 所以他们的 E-R 图也不同。

店员只能对系统中的信息进行查看，而不能改动任何信息。所以，店员的 E-R 图如图 3-4 所示。

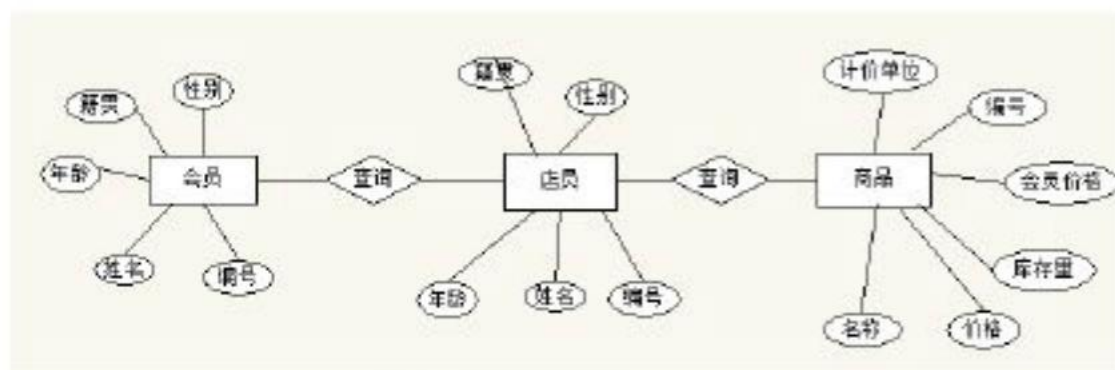


图 3-4 店员 E-R 图

店主具备所有操作的权限，可以对系统中的所有信息进行查看，增加，修改，删除。店主的 E-R 图如图 3-5 所示。

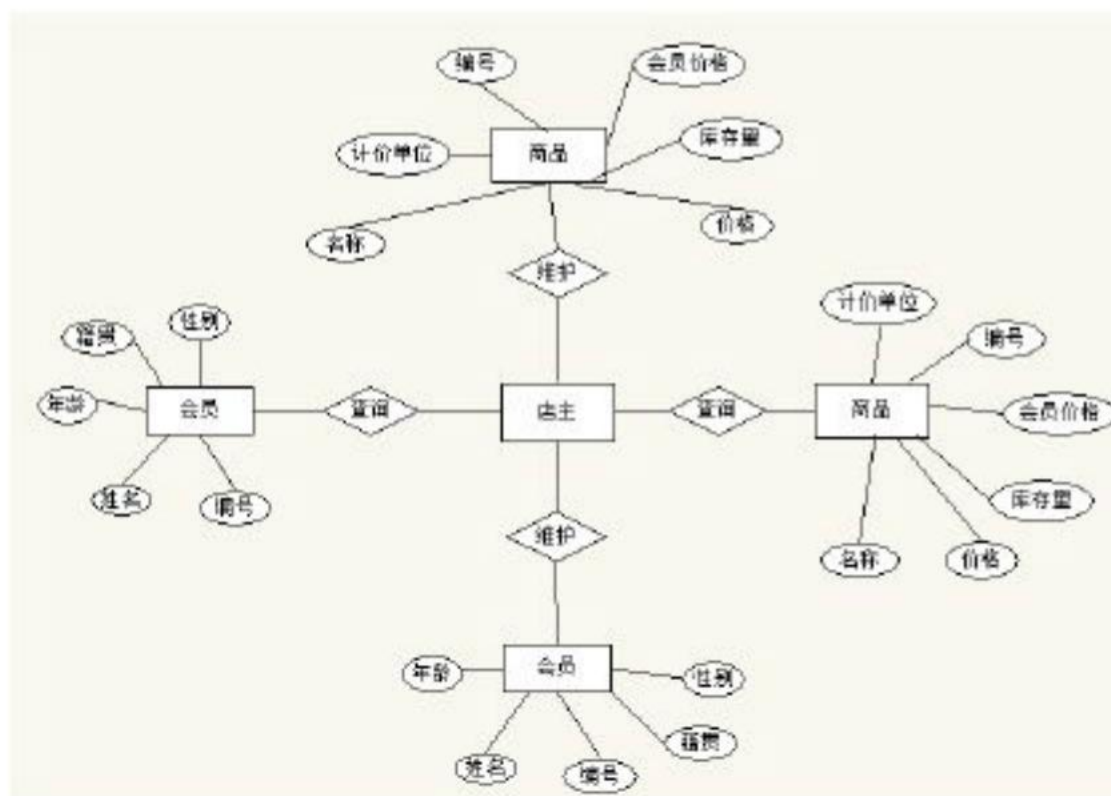


图 3-5 店主 E-R 图

第4章 系统详细设计

4.1 底层数据操作模块设计

此模块是用来给本系统中所有需要用到数据库的程序提供的一类方法。里面定义了各种有关数据库操作的功能，包含：数据库连接函数，获取数据集函数，执行数据操作函数等。代码如下：

```
class SQLHelper
{
    /// <summary>
    /// 创建SqlConnection类，为后面的连接数据库操作提供支持。
    /// </summary>
    /// <returns></returns>
    public SqlConnection getConn()
    {
        string strConn = "server = (local);
Database=XACSSStoreManage;Integrated Security = true";
        SqlConnection myConn = new SqlConnection(strConn);
        return myConn;
    }

    /// <summary>
    /// 获取指定的数据集。作为DATAGRIDVIEW的数据源使用。
    /// </summary>
    /// <param name="mySqlStr"></param>
    /// <param name="myDataTable"></param>
    /// <returns></returns>
    public DataSet getDataSet(string mySqlStr, string myDataTable)
    {
        SqlConnection myConn = this.getConn();
        SqlDataAdapter myAdapter = new SqlDataAdapter(mySqlStr,
```

```
myConn);  
  
    DataSet myDataSet = new DataSet();  
    myAdapter.Fill(myDataSet, myDataTable);  
    return myDataSet;  
}  
  
/// <summary>  
/// 按照指定的SQL语句获取数据库中的相应数据  
/// </summary>  
/// <param name="SqlStr"></param>  
/// <returns></returns>  
public SqlDataReader getReader(string SqlStr)  
{  
    SqlConnection myConn = this.getConn();  
    SqlCommand myComm = new SqlCommand(SqlStr, myConn);  
    myConn.Open();  
    SqlDataReader myReader =  
myComm.ExecuteReader(CommandBehavior.CloseConnection);  
    return myReader;  
}  
  
/// <summary>  
/// 执行相应SQL语句。  
/// </summary>  
/// <param name="SqlStr"></param>  
public void getComm(string SqlStr)  
{  
    SqlConnection myConn = this.getConn();  
    myConn.Open();  
    SqlCommand myComm = new SqlCommand(SqlStr, myConn);  
    myComm.ExecuteNonQuery();  
    myComm.Dispose();  
    myConn.Close();  
}
```

```

        myConn.Dispose();
    }
}

```

4.2 系统登录界面与主界面设计

4.2.1 登录界面设计

本系统登录界面为 loginForm.cs, 登录名的选择采用下拉框的形式, 这样子可以很好的避免了用户输入登录名不合法的问题。密码的输入采用的是加密字符串的形式, 以此来避免用户所输入的密码被人看到。当用户输入密码错误时会有错误提示, 并且自动清空密码输入框, 以便用户重新输入。

显示界面如下所示:



图 4-1 登录界面

实现代码如下:

```

SQLHelper sh = new SQLHelper(); //实例化SQLHelper类
private void loginbutton_Click(object sender, EventArgs e)
{
    SqlDataReader myReader = sh.getReader("select LoginName, LoginRight
from User_Right where LoginID=' " + comboBox1.Text + "' and LoginPwd=' "
+ textBox2.Text + "'");
    myReader.Read();
    if (myReader.HasRows)

```

```
{
    LoginName = myReader["LoginName"].ToString().Trim();
    UserRight = myReader["LoginRight"].ToString().Trim();
    MainForm mForm = new MainForm();
    this.Hide();
    mForm.Show();
}
else
{
    MessageBox.Show("用户名或密码错误!", "提示",
        MessageBoxButtons.OK, MessageBoxIcon.Information);
    textBox2.Text = "";
}
myReader.Close();
}
```

4.2.2 系统主界面设计

本系统的主界面为 mainForm.cs, 在主界面的下方采用状态栏显示当前登录用户的登录权限以及用户名。系统根据登录权限的不同显示或隐藏部分功能模块。

(1) 店主登录界面:

“admin”权限可以使用本系统的所有功能，包括：商品管理，供应商管理，员工管理，会员管理，销售管理，系统管理。其界面如下所示：



图 4-2 店主登录系统主界面

(2) 店员登录界面：

“normal”权限的用户登录系统后，系统只给其提供部分功能，如：商品管理、供应商管理、会员管理这三大模块下的查询子功能，销售管理模块下的出售商品子功能，以及系统管理。

显示界面如下：



图 4-3 店员登录系统主界面

实现隐藏功能的代码如下：

```
private void mainForm_Load(object sender, EventArgs e)
{
    //当登录的用户权限为“normal”时，隐藏部分内容。
    if (loginForm.UserRight == "normal")
    {
        供应商信息维护ToolStripMenuItem.Visible = false;
        员工管理ToolStripMenuItem.Visible = false;
        会员基本信息维护ToolStripMenuItem.Visible = false;
        信息修改ToolStripMenuItem.Visible = false;
        信息修改ToolStripMenuItem1.Visible = false;
        信息修改ToolStripMenuItem2.Visible = false;
        信息修改ToolStripMenuItem3.Visible = false;
    }

    toolStripStatusLabel1.Text = "当前用户为：" + loginForm.LoginName +
    "， 权限为：" + loginForm.UserRight;
}
```

4.3 商品管理模块设计

由于该商店销售四大类商品，为了方便用户管理，此模块将四大类商品分别进行管理，但其原理都是一样的，所以只以一类商品的管理作为示范。

4.3.1 商品信息查询模块

该模块用于查询商品的各种信息，如商品的价格，会员价格，库存余量，计价单位等。显示界面如下所示：



图 4-4 商品信息查询界面

实现代码如下:

```

/// <summary>
/// 加载此窗体时, 加载数据到dataGridView1.
/// </summary>
/// <param name="sender"></param>
/// <param name="e"></param>
private void YanCaoGoodsForm1_Load (object sender, EventArgs e)
{
    string SqlStr = "select * from YanCao";
    DataSet dataSet = SQLHelper.getDataSet(SqlStr, "YanCao");
    dataGridView1.DataSource = dataSet.Tables[0];
}

```

4.3.2 商品信息维护模块

该模块具有对商品信息进行增加、删除、修改的功能。用户选中要修改的行, 商品的各种信息则对应的出现在上方的输入框里, 再单击修改按钮, 就可以在上方的输入框里对商品的信息进行修改, 完成后点击确定按钮, 则保存数据; 或者单击删除按钮, 则可删除这条数据。单击添加按钮, 上方原本只读状态的输入框变为可写, 完成后点击确定按钮, 则保存数据。

点击添加时, 显示界面如下:



图 4-5 添加商品信息界面

点击修改时，显示界面如下：



图 4-6 修改商品信息界面

实现代码如下所示：

```

/// <summary>
/// 点击增加按钮时，将所有输入框置为可写，并且清空所有输入框。将确定按钮设置为可用。
/// </summary>
/// <param name="sender"></param>

```

```
/// <param name="e"></param>
private void btnAdd_Click(object sender, EventArgs e)
{
    textBox1.ReadOnly = false;
    textBox2.ReadOnly = false;
    textBox3.ReadOnly = false;
    textBox4.ReadOnly = false;
    textBox5.ReadOnly = false;

    textBox1.Text = "";
    textBox2.Text = "";
    textBox3.Text = "";
    textBox4.Text = "";
    textBox5.Text = "";

    judge = 1;
    Button4.Enabled = true;
}
/// <summary>
/// 当点击修改按钮时，将输入框置为可写。并将确定按钮置为可用。
/// </summary>
/// <param name="sender"></param>
/// <param name="e"></param>
private void btnModify_Click(object sender, EventArgs e)
{
    textBox1.ReadOnly = false;
    textBox2.ReadOnly = false;
    textBox3.ReadOnly = false;
    textBox4.ReadOnly = false;
    textBox5.ReadOnly = false;

    judge = 2;
    Button4.Enabled = true;
}
```

```

/// <summary>
/// 按照前面标志的类型，执行相应SQL语句。保存数据，并刷新页面。
/// </summary>
/// <param name="sender"></param>
/// <param name="e"></param>
private void btnSave_Click(object sender, EventArgs e)
{
    string comm;
    if (judge == 1)
    {
        comm = "insert into YanCao values(' " + textBox1.Text +
            ", ' " + Convert.ToInt32(textBox2.Text) + ", ' " + " " +
            Convert.ToInt32(textBox3.Text) + ", ' " + textBox4.Text +
            Convert.ToInt32(textBox5.Text) + ", ' " + ")";
        SQLHelper.getComm(comm);
    }
    if (judge == 2)
    {
        comm = "update YanCao set GoodsName=' " + textBox1.Text
            + ", GoodsPrice=' " + Convert.ToInt32(textBox2.Text) + ", MemPrice="
            + Convert.ToInt32(textBox3.Text) + ", Unit=' " + textBox4.Text + ", Stock=' "
            + Convert.ToInt32(textBox5.Text) + " ";
        SQLHelper.getComm(comm);
    }

    YanCaoGoodsForm2_Load(sender, e);
    judge = 0;
}

/// <summary>
/// 删除选择的数据。
/// </summary>
/// <param name="sender"></param>

```



```

/// <param name="e"></param>
private void btnDelete_Click(object sender, EventArgs e)
{
    try
    {
        if (MessageBox.Show("确定要删除该商品信息吗?", "提示",
            MessageBoxButtons.OKCancel, MessageBoxIcon.Question) ==
            DialogResult.OK)
        {
            baseofForms.getComm("delete from YanCao where
            GoodsID=' " + Convert.ToString(dataGridView1[0,
            dataGridView1.CurrentCell.RowIndex].Value).Trim() + "'");
            YanCaoGoodsForm2_Load (sender, e);
            MessageBox.Show("删除数据成功!", "提示",
            MessageBoxButtons.OK, MessageBoxIcon.Information);
        }
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message, "提示",
            MessageBoxButtons.OKCancel, MessageBoxIcon.Information);
    }
}

/// <summary>
/// 将dataGridView1里选择的数据显示到相应输入框中。
/// </summary>
/// <param name="sender"></param>
/// <param name="e"></param>
private void dataGridView1_CellClick(object sender,
DataGridViewCellEventArgs e)
{
    textBox1.Text = Convert.ToString(dataGridView1[0,
    dataGridView1.CurrentCell.RowIndex].Value).Trim();
}

```

```

        textBox2.Text = Convert.ToString(dataGridView1[1,
dataGridView1.CurrentCell.RowIndex].Value).Trim();
        textBox3.Text = Convert.ToString(dataGridView1[2,
dataGridView1.CurrentCell.RowIndex].Value).Trim();
        textBox4.Text = Convert.ToString(dataGridView1[3,
dataGridView1.CurrentCell.RowIndex].Value).Trim();
        textBox5.Text = Convert.ToString(dataGridView1[4,
dataGridView1.CurrentCell.RowIndex].Value).Trim();
    }

    /// <summary>
    /// 当选择变化时，还原输入框的只读状态。
    /// </summary>
    /// <param name="sender"></param>
    /// <param name="e"></param>
    private void dataGridView1_CellStateChanged(object sender,
DataGridViewCellStateChangedEventArgs e)
    {
        textBox1.ReadOnly = true;
        textBox2.ReadOnly = true;
        textBox3.ReadOnly = true;
        textBox4.ReadOnly = true;
        textBox5.ReadOnly = true;
    }
}

```

4.4 供应商管理模块设计

4.4.1 供应商信息查询模块

该模块用于查询供应商的各种信息，如供应商的名称、地址、电话等。程序界面如下：



图 4-7 查询供应商信息

实现代码如下：

```

/// <summary>
    /// 加载此窗体时，加载数据到dataGridView1。
    /// </summary>
    /// <param name="sender"></param>
    /// <param name="e"></param>
    private void SerchS_Load (object sender, EventArgs e)
    {
        string SqlStr = "select * from Supplier";
        DataSet dataSet = SQLHelper.getDataSet(SqlStr, "
Supplier");
        dataGridView1.DataSource = dataSet.Tables[0];
    }

```

4.4.2 供应商信息维护模块

该模块具有对供应商信息进行增加、删除、修改的功能。用户选中要修改的行，供应商的各种信息则对应的出现在上方的输入框里，再单击修改按钮，就可以在上方的输入框里对供应商的信息进行修改，完成后点击确定按钮，则保存数据；或者单击删除按钮，则可删除这条数据。单击添加按钮，上方原本不可输入的输入框变得可以输入数据，完成后点击确定按钮，则保存数据。

点击添加时，显示界面如下：



图 4-8 添加供应商信息界面

点击修改时，显示界面如下：



图 4-9 修改供应商信息界面

实现代码如下所示：

```
/// <summary>
```

```
/// 点击增加按钮时，将所有输入框置为可写，并且清空所有输入框。将确定按钮设置为可用。
```



```
/// </summary>
/// <param name="sender"></param>
/// <param name="e"></param>
private void btnAdd_Click(object sender, EventArgs e)
{
    textBox1.ReadOnly = false;
    textBox2.ReadOnly = false;
    textBox3.ReadOnly = false;
    textBox4.ReadOnly = false;
    textBox5.ReadOnly = false;

    textBox1.Text = "";
    textBox2.Text = "";
    textBox3.Text = "";
    textBox4.Text = "";
    textBox5.Text = "";

    judge = 1;
    Button4.Enabled = true;
}
/// <summary>
/// 当点击修改按钮时，将输入框置为可写。并将确定按钮置为可用。
/// </summary>
/// <param name="sender"></param>
/// <param name="e"></param>
private void btnModify_Click(object sender, EventArgs e)
{
    textBox1.ReadOnly = false;
    textBox2.ReadOnly = false;
    textBox3.ReadOnly = false;
    textBox4.ReadOnly = false;
    textBox5.ReadOnly = false;

    judge = 2;
```



```

        Button4.Enabled = true;
    }

    /// <summary>
    /// 按照前面标志的类型, 执行相应SQL语句。保存数据, 并刷新页面。
    /// </summary>
    /// <param name="sender"></param>
    /// <param name="e"></param>
    private void btnSave_Click(object sender, EventArgs e)
    {
        string comm;
        if (judge == 1)
        {
            comm = "insert into Supplier values(' " + textBox1.Text
            + "', ' " + textBox2.Text + "', ' " + textBox3.Text + "', ' " +
            textBox4.Text + textBox5.Text + "', ' " + ")";
            SQLHelper.getComm(comm);
        }
        if (judge == 2)
        {
            comm = "update Supplier set SortID=' " + textBox1.Text +
            "', SSort=' " + extBox2.Text + "', SName=' " + textBox3.Text + "', SAddr=' " +
            textBox4.Text + "', SPhone=' " + textBox5.Text + "'";
            SQLHelper.getComm(comm);
        }

        SetS_Load(sender, e);
        judge = 0;
    }

    /// <summary>
    /// 删除选择的数据。
    /// </summary>
    /// <param name="sender"></param>

```

```

    /// <param name="e"></param>
    private void btnDelete_Click(object sender, EventArgs e)
    {
        try
        {
            if (MessageBox.Show("确定要删除该供应商信息吗?", "提示",
                MessageBoxButtons.OKCancel, MessageBoxIcon.Question) ==
                DialogResult.OK)
            {
                baseofForms.getComm("delete from Supplier where
                SID='" + Convert.ToString(dataGridView1[0,
                dataGridView1.CurrentCell.RowIndex].Value).Trim() + "'");
                SetS_Load (sender, e);
                MessageBox.Show("删除数据成功!", "提示",
                MessageBoxButtons.OK, MessageBoxIcon.Information);
            }
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.Message, "提示",
                MessageBoxButtons.OKCancel, MessageBoxIcon.Information);
        }
    }

    /// <summary>
    /// 将dataGridView1里选择的数据显示到相应输入框中。
    /// </summary>
    /// <param name="sender"></param>
    /// <param name="e"></param>
    private void dataGridView1_CellClick(object sender,
    DataGridViewCellEventArgs e)
    {
        textBox1.Text = Convert.ToString(dataGridView1[0,
        dataGridView1.CurrentCell.RowIndex].Value).Trim();
    }

```

```

        textBox2.Text = Convert.ToString(dataGridView1[1,
dataGridView1.CurrentCell.RowIndex].Value).Trim();
        textBox3.Text = Convert.ToString(dataGridView1[2,
dataGridView1.CurrentCell.RowIndex].Value).Trim();
        textBox4.Text = Convert.ToString(dataGridView1[3,
dataGridView1.CurrentCell.RowIndex].Value).Trim();
        textBox5.Text = Convert.ToString(dataGridView1[4,
dataGridView1.CurrentCell.RowIndex].Value).Trim();
    }

    /// <summary>
    /// 当选择变化时，还原输入框的只读状态。
    /// </summary>
    /// <param name="sender"></param>
    /// <param name="e"></param>
    private void dataGridView1_CellStateChanged(object sender,
DataGridViewCellStateChangedEventArgs e)
    {
        textBox1.ReadOnly = true;
        textBox2.ReadOnly = true;
        textBox3.ReadOnly = true;
        textBox4.ReadOnly = true;
        textBox5.ReadOnly = true;
    }
}

```

4.5 员工管理模块设计

4.5.1 员工信息查询模块

该模块用于查询员工的各种信息，如员工的姓名、性别、籍贯、年龄等。显示界面如下：



图 4-10 员工信息查询界面

实现代码如下：

```

/// <summary>
/// 加载此窗体时，加载数据到dataGridView1。
/// </summary>
/// <param name="sender"></param>
/// <param name="e"></param>
private void SerchS_Load (object sender, EventArgs e)
{
    string SqlStr = "select * from Employee";
    DataSet dataSet = SQLHelper.getDataSet(SqlStr, "
Employee");
    dataGridView1.DataSource = dataSet.Tables[0];
}

```

4.5.2 员工信息维护模块

该模块具有对员工信息进行增加、删除、修改的功能。用户选中要修改的行，员工的各种信息则对应的出现在上方的输入框里，再单击修改按钮，就可以在上方的输入框里对员工的信息进行修改，完成后点击确定按钮，则保存数据；或者单击删除按钮，则可删除这条数据。单击添加按钮，上方原本只读状态的输入框变为可写状态，完成后点击确定按钮，则保存数据。

点击添加时，显示界面如下：



图 4-11 添加员工信息界面

点击修改时，显示界面如下：



图 4-12 修改员工信息界面

实现代码如下所示：

```
/// <summary>
```

```
/// 点击增加按钮时，将所有输入框置为可写，并且清空所有输入框。
```

将确定按钮设置为可用。


```
/// </summary>
/// <param name="sender"></param>
/// <param name="e"></param>
private void btnAdd_Click(object sender, EventArgs e)
{
    textBox1.ReadOnly = false;
    textBox2.ReadOnly = false;
    textBox3.ReadOnly = false;
    textBox4.ReadOnly = false;

    textBox1.Text = "";
    textBox2.Text = "";
    textBox3.Text = "";
    textBox4.Text = "";

    judge = 1;
    Button4.Enabled = true;
}
/// <summary>
/// 当点击修改按钮时，将输入框置为可写。并将确定按钮置为可用。
/// </summary>
/// <param name="sender"></param>
/// <param name="e"></param>
private void btnModify_Click(object sender, EventArgs e)
{
    textBox1.ReadOnly = false;
    textBox2.ReadOnly = false;
    textBox3.ReadOnly = false;
    textBox4.ReadOnly = false;

    judge = 2;
    Button4.Enabled = true;
}
```

```

    /// <summary>
    /// 按照前面标志的类型, 执行相应SQL语句。保存数据, 并刷新页面。
    /// </summary>
    /// <param name="sender"></param>
    /// <param name="e"></param>
    private void btnSave_Click(object sender, EventArgs e)
    {
        string comm;
        if (judge == 1)
        {
            comm = "insert into Employee values(' " + textBox1.Text
            + " ', ' " + textBox2.Text + " ', ' " + " + " ', ' " + textBox3.Text + " ', ' " +
            textBox4.Text + " ', ' " + " ) ";
            SQLHelper.getComm(comm);
        }
        if (judge == 2)
        {
            comm = "update Employee set EName=' " + textBox1.Text +
            " ', ESex=' " + extBox2.Text + " ', EAge=' " + textBox3.Text + " ', EHometown=' "
            + textBox4.Text + " ' ";
            SQLHelper.getComm(comm);
        }

        YuanGForm2_Load(sender, e);
        judge = 0;
    }

    /// <summary>
    /// 删除选择的数据。
    /// </summary>
    /// <param name="sender"></param>
    /// <param name="e"></param>
    private void btnDelete_Click(object sender, EventArgs e)
    {

```

```

        try
        {
            if (MessageBox.Show("确定要删除该员工信息吗?", "提示",
                MessageBoxButtons.OKCancel, MessageBoxIcon.Question) ==
                DialogResult.OK)
            {
                baseofForms.getComm("delete from Employee where
                EID=' " + Convert.ToString(dataGridView1[0,
                dataGridView1.CurrentCell.RowIndex].Value).Trim() + "'");
                SetS_Load(sender, e);
                MessageBox.Show("删除数据成功!", "提示",
                MessageBoxButtons.OK, MessageBoxIcon.Information);
            }
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.Message, "提示",
                MessageBoxButtons.OKCancel, MessageBoxIcon.Information);
        }
    }

    /// <summary>
    /// 将dataGridView1里选择的数据显示到相应输入框中。
    /// </summary>
    /// <param name="sender"></param>
    /// <param name="e"></param>
    private void dataGridView1_CellClick(object sender,
    DataGridViewCellEventArgs e)
    {
        textBox1.Text = Convert.ToString(dataGridView1[0,
        dataGridView1.CurrentCell.RowIndex].Value).Trim();
        textBox2.Text = Convert.ToString(dataGridView1[1,
        dataGridView1.CurrentCell.RowIndex].Value).Trim();
        textBox3.Text = Convert.ToString(dataGridView1[2,

```

```

dataGridView1.CurrentRow.RowIndex].Value).Trim();
        textBox4.Text = Convert.ToString(dataGridView1[3,
dataGridView1.CurrentRow.RowIndex].Value).Trim();
    }
    /// <summary>
    /// 当选择变化时，还原输入框的只读状态。
    /// </summary>
    /// <param name="sender"></param>
    /// <param name="e"></param>
    private void dataGridView1_CellStateChanged(object sender,
DataGridViewCellStateChangedEventArgs e)
    {
        textBox1.ReadOnly = true;
        textBox2.ReadOnly = true;
        textBox3.ReadOnly = true;
        textBox4.ReadOnly = true;
    }

```

4.6 会员管理模块设计

4.6.1 会员信息查询模块

该模块用于查询会员的各种信息，如会员的姓名、性别、籍贯、年龄等。显示界面如下：



图 4-13 查询会员信息界面

实现代码如下：

```

/// <summary>
    /// 加载此窗体时，加载数据到dataGridView1。
    /// </summary>
    /// <param name="sender"></param>
    /// <param name="e"></param>
    private void HuiYForm1_Load (object sender, EventArgs e)
    {
        string SqlStr = "select * from Member";
        DataSet dataSet = SQLHelper.getDataSet(SqlStr, "Member");
        dataGridView1.DataSource = dataSet.Tables[0];
    }

```

4.6.2 会员信息维护模块

该模块具有对会员信息进行增加、删除、修改的功能。用户选中要修改的行，会员的各种信息则对应的出现在上方的输入框里，再单击修改按钮，就可以在上方的输入框里对会员的信息进行修改，完成后点击确定按钮，则保存数据；或者单击删除按钮，则可删除这条数据。单击添加按钮，上方原本不可输入的输入框变得可以输入数据，完成后点击确定按钮，则保存数据。

点击添加时，显示界面如下：



图 4-14 添加会员信息界面

点击修改时，显示界面如下：



图 4-15 修改会员信息界面

实现代码如下所示：

```

/// <summary>
    /// 点击增加按钮时，将所有输入框置为可写，并且清空所有输入框。
    将确定按钮设置为可用。
/// </summary>
/// <param name="sender"></param>
/// <param name="e"></param>

```

```

private void btnAdd_Click(object sender, EventArgs e)
{
    textBox1.ReadOnly = false;
    textBox2.ReadOnly = false;
    textBox3.ReadOnly = false;
    textBox4.ReadOnly = false;

    textBox1.Text = "";
    textBox2.Text = "";
    textBox3.Text = "";
    textBox4.Text = "";

    judge = 1;
    Button4.Enabled = true;
}

/// <summary>
/// 当点击修改按钮时，将输入框置为可写。并将确定按钮置为可用。
/// </summary>
/// <param name="sender"></param>
/// <param name="e"></param>
private void btnModify_Click(object sender, EventArgs e)
{
    textBox1.ReadOnly = false;
    textBox2.ReadOnly = false;
    textBox3.ReadOnly = false;
    textBox4.ReadOnly = false;

    judge = 2;
    Button4.Enabled = true;
}

/// <summary>
/// 按照前面标志的类型，执行相应SQL语句。保存数据，并刷新页面。
/// </summary>

```

```

    /// <param name="sender"></param>
    /// <param name="e"></param>
    private void btnSave_Click(object sender, EventArgs e)
    {
        string comm;
        if (judge == 1)
        {
            comm = "insert into Employee values(' " + textBox1.Text
+ " ', ' " + textBox2.Text + " ', ' " + " ' " + textBox3.Text + " ', ' " +
textBox4.Text + " ', ' " + " ' ";
            SQLHelper.getComm(comm);
        }
        if (judge == 2)
        {
            comm = "update Employee set EName=' " + textBox1.Text +
" ', ESex=' " + extBox2.Text + " ', EAge=' " + textBox3.Text + " ', EHometown=' "
+ textBox4.Text + " ' ";
            SQLHelper.getComm(comm);
        }

        HuiYForm2_Load(sender, e);
        judge = 0;
    }

    /// <summary>
    /// 删除选择的数据。
    /// </summary>
    /// <param name="sender"></param>
    /// <param name="e"></param>
    private void btnDelete_Click(object sender, EventArgs e)
    {
        try
        {
            if (MessageBox.Show("确定要删除该会员信息吗?", "提示

```

```

", MessageBoxButtons.OKCancel, MessageBoxIcon.Question) ==
DialogResult.OK)
    {
        baseofForms.getComm("delete from Member where
MemID=' " + Convert.ToString(dataGridView1[0,
dataGridView1.CurrentCell.RowIndex].Value).Trim() + "'");
        SetS_Load (sender, e);
        MessageBox.Show("删除数据成功!", "提示",
MessageBoxButtons.OK, MessageBoxIcon.Information);
    }
}
catch (Exception ex)
{
    MessageBox.Show(ex.Message, "提示",
MessageBoxButtons.OKCancel, MessageBoxIcon.Information);
}
}

/// <summary>
/// 将dataGridView1里选择的数据显示到相应输入框中。
/// </summary>
/// <param name="sender"></param>
/// <param name="e"></param>
private void dataGridView1_CellClick(object sender,
DataGridViewCellEventArgs e)
{
    textBox1.Text = Convert.ToString(dataGridView1[0,
dataGridView1.CurrentCell.RowIndex].Value).Trim();
    textBox2.Text = Convert.ToString(dataGridView1[1,
dataGridView1.CurrentCell.RowIndex].Value).Trim();
    textBox3.Text = Convert.ToString(dataGridView1[2,
dataGridView1.CurrentCell.RowIndex].Value).Trim();
    textBox4.Text = Convert.ToString(dataGridView1[3,
dataGridView1.CurrentCell.RowIndex].Value).Trim();
}

```



```

    }
    /// <summary>
    /// 当选择变化时，还原输入框的只读状态。
    /// </summary>
    /// <param name="sender"></param>
    /// <param name="e"></param>
    private void dataGridView1_CellStateChanged(object sender,
DataGridViewCellStateChangedEventArgs e)
    {
        textBox1.ReadOnly = true;
        textBox2.ReadOnly = true;
        textBox3.ReadOnly = true;
        textBox4.ReadOnly = true;
    }

```

4.7 销售管理模块设计

4.7.1 出售商品模块

该模块包含出售商品的功能，每出售一件商品，库存量都会自动减1。并将商品的出售情况记录在一张数据表里，方便日后的利润计算。

```

/// <summary>
/// 加载此窗体时，加载数据到dataGridView1。
/// </summary>
/// <param name="sender"></param>
/// <param name="e"></param>
private void SalesForm1_Load (object sender, EventArgs e)
{
    string SqlStr = "select * from Sale";
    DataSet dataSet = SQLHelper.getDataSet(SqlStr, "Sale");
    dataGridView1.DataSource = dataSet.Tables[0];
}

```

其界面如下：



图 4-16 出售商品界面

4.7.2 利润计算模块

该模块可以让用户通过自己定义日期，来统计商品出售情况以及计算利润。显示界面如下：



图 4-17 利润计算界面

实现代码如下：

```
private void button1_Click(object sender, EventArgs e)
{
    string sqlstr = "select * from Sales where datetime between"
```

```

+ Convert.ToDateTime(dateTimePicker1.Text.ToString())
    + "and" +
Convert.ToDateTime(dateTimePicker2.Text.ToString());
SQLHelper.getcomm(sqlstr);
string lstr = "select sum(GoodsPrice-GoodsInPrice) from
Sales";
textBox1.Text = SQLHelper.getcomm(lstr);
}

```

4.8 系统管理模块

4.8.1 权限管理模块

此模块可以使店主管理下属员工的登录权限，增删查改各种登录信息。点击添加时显示界面如下：



图 4-18 权限管理界面

实现代码如下所示：

```

/// <summary>
    /// 点击增加按钮时，将所有输入框置为可写，并且清空所有输入框。
    将确定按钮设置为可用。
    /// </summary>
    /// <param name="sender"></param>
    /// <param name="e"></param>
    private void btnAdd_Click(object sender, EventArgs e)

```

```
{
    textBox1.ReadOnly = false;
    textBox2.ReadOnly = false;
    textBox3.ReadOnly = false;

    textBox1.Text = "";
    textBox2.Text = "";
    textBox3.Text = "";

    judge = 1;
    Button4.Enabled = true;
}

/// <summary>
/// 当点击修改按钮时，将输入框置为可写。并将确定按钮置为可用。
/// </summary>
/// <param name="sender"></param>
/// <param name="e"></param>
private void btnModify_Click(object sender, EventArgs e)
{
    textBox1.ReadOnly = false;
    textBox2.ReadOnly = false;
    textBox3.ReadOnly = false;

    judge = 2;
    Button4.Enabled = true;
}

/// <summary>
/// 按照前面标志的类型，执行相应SQL语句。保存数据，并刷新页面。
/// </summary>
/// <param name="sender"></param>
/// <param name="e"></param>
private void btnSave_Click(object sender, EventArgs e)
{
```

```

        string comm;
        if (judge == 1)
        {
            comm = "insert into User_Right values(' " + textBox1.Text
+ " ', ' " + textBox2.Text + " ', ' " + " ' " + textBox3.Text + " ' )";
            SQLHelper.getComm(comm);
        }
        if (judge == 2)
        {
            comm = "update User_Right set LoginName=' " +
textBox1.Text + " ', LoginPwd=' " + extBox2.Text + " ', LoginRight=' " +
textBox3.Text + " ' ";
            SQLHelper.getComm(comm);
        }

        RightForm_Load(sender, e);
        judge = 0;
    }

    /// <summary>
    /// 删除选择的数据。
    /// </summary>
    /// <param name="sender"></param>
    /// <param name="e"></param>
    private void btnDelete_Click(object sender, EventArgs e)
    {
        try
        {
            if (MessageBox.Show("确定要删除该条信息?", "提示",
MessageBoxButtons.OKCancel, MessageBoxIcon.Question) ==
DialogResult.OK)
            {
                baseofForms.getComm("delete from User_Right where
LoginID=' " + Convert.ToString(dataGridView1[0,

```



```

dataGridView1.CurrentRow.RowIndex].Value).Trim() + "");
        SetS_Load (sender, e);
        MessageBox.Show("删除数据成功!", "提示",
        MessageBoxButtons.OK, MessageBoxIcon.Information);
    }
}
catch (Exception ex)
{
    MessageBox.Show(ex.Message, "提示",
    MessageBoxButtons.OKCancel, MessageBoxIcon.Information);
}
}

/// <summary>
/// 将dataGridView1里选择的数据显示到相应输入框中。
/// </summary>
/// <param name="sender"></param>
/// <param name="e"></param>
private void dataGridView1_CellClick(object sender,
DataGridViewCellEventArgs e)
{
    textBox1.Text = Convert.ToString(dataGridView1[0,
dataGridView1.CurrentRow.RowIndex].Value).Trim();
    textBox2.Text = Convert.ToString(dataGridView1[1,
dataGridView1.CurrentRow.RowIndex].Value).Trim();
    textBox3.Text = Convert.ToString(dataGridView1[2,
dataGridView1.CurrentRow.RowIndex].Value).Trim();
}

/// <summary>
/// 当选择变化时, 还原输入框的只读状态。
/// </summary>
/// <param name="sender"></param>
/// <param name="e"></param>
private void dataGridView1_CellStateChanged(object sender,

```



```

DataGridViewCellStateChangedEventArgs e)
{
    textBox1.ReadOnly = true;
    textBox2.ReadOnly = true;
    textBox3.ReadOnly = true;
    textBox4.ReadOnly = true;
}

```

4.8.2 注销登录模块

此模块可以注销当前用户的登录,使页面重新返回登录界面。实现代码如下:

```

private void 注销登录ToolStripMenuItem_Click(object sender, EventArgs
e)
{
    loginForm logForm = new loginForm();
    this.Hide();
    logForm.Show();
}

```

4.8.3 退出系统模块

此模块可以使用户退出系统,作用等同于系统页面右上角的红色叉叉。实现代码如下:

```

private void 退出系统ToolStripMenuItem_Click(object sender, EventArgs
e)
{
    this.Close();
}
private void mainForm_FormClosed(object sender,
FormClosedEventArgs e)
{
    Application.Exit();
}

```

第 5 章 系统测试与性能分析

本系统所采用的测试方法是首先进行各模块的测试,看其模块的功能是否符合需求分析中所要求具有的功能。如果未完善,则进行修改和调整。最后进行总体测试。测试结果各项功能均已经或基本达到设计要求。

本系统所采用的测试环境是:Window7 操作系统 +.NetFramework 3.5 平台。

测试的主要内容包括:登录界面,根据权限不同显示不同的主界面内容,查看和维护商品信息、供应商信息、员工信息、会员信息等,商品出售管理,利润计算等功能。测试结果正常,基本达到设计的要求。

本系统采用一个底层数据操作模块作为公用基础类,所有的数据操作都基于此,很好的实现了软件设计思想中“代码复用”的理念。这样子也避免了在各个功能模块中都写入数据操作的函数,最终导致系统崩溃的潜在问题。与此同时,系统在性能上也得到了质的飞跃,系统运行速度明显提升。

第6章 总结

经过三个多月的学习和工作，顺利完成了“西安昌盛商店智能管理系统”的设计。从开始接到设计题目到系统功能的实现，每走一步对我来说都是新的尝试与挑战，不断地发现问题，解决问题使我的学习工作能力有了很大的提高，这次论文也是我在大学期间独立完成的最大的项目。这次学习和实践提高了我的思维和动手能力，为我今后的学习和发展奠定了扎实的基础。

本系统最终实现了预先定制的所有功能。系统的两部分功能为：第一部分针对员工使用，包括的功能如：查询商品信息、查询供应商信息、商品销售等；第二部分面向店主使用，包括的功能如：对商品信息、供应商信息、会员和员工信息的增删查改，计算利润等。

这次设计的任务是该商店管理的智能化，信息化。在设计的过程中我遇到了很多的问题，但是通过查阅资料和师红宇老师的悉心指导，我解决了很多问题。由于时间的限制，目前所做的系统还有很多不完善的地方，比如界面设计过于简单，不够美观等。

在开发过程中我体会到了做软件行业的严谨性，你稍不注意，就会犯很大的错误，并且对知识的追求也应该严谨，每一步，每一个环节都应该理解清楚，否则会越往下做越做不动了，应该保持很高的求学精神，一步一个脚印，踏踏实实的做完工作的每一个环节，这是软件编程者应具备的基本素质。

如今，该系统已是西安昌盛商店必不可少的管理系统，所以，以后应尽力完善此系统，再多添加些功能，实现各种信息的全面管理。如：将会员管理模块做更细的划分，实现会员赊账的信息化管理。进而提高该商店的管理效率，为商店带来更大的价值。

致 谢

在此衷心感谢*****老师在我的毕业设计过程中,对我的细心指导与帮助。从而使我能够顺利完成课题的设计和论文的撰写。从课题确定之日起,师红宇老师便指导我提前准备好毕业设计所需要提前做的准备工作:了解所用编程语言的资料、打好知识基础、过好理论关、确定编程思想和设计环境关;师红宇老师严格按照软件开发的步骤给我分配任务,而且还利用新的软件开发管理方法来指导与协调我的设计进程,并定期检查我的进度是否完成预定目标,直至设计完成。虽然中间遇到了各种各样的问题,但是在老师的讲解下使我们了解了许多关于软件开发的前沿技术与思想,使我从中学到了很多知识,开拓了视野,丰富了知识面,同时老师严谨的治学作风,认真踏实的工作态度将使我受益终身。所有这些都为我今后的学习和工作打下了坚实的基础。

参考文献

- [1] (美)Paul Sarknas 著, 高猛、王海涛译 Asp.net2.0 电子商务高级编程 (c#2005 版) 清华大学出版社 2007.11
- [2] (美)Christain Nagel、Bill Evjen、Jay Glynn 等著, 李敏波译, 黄静审核 C#高级编程 (第四版) 清华大学出版社 2006.10
- [3] 张洪举, 芦桂章. 循序渐进 SQL Server 2005 入门、进阶与应用实例, 北京: 人民邮电出版社, 2008
- [4] 张青, 郭亚萍. C#编程思想与实践[M]. 北京: 冶金工业出版社, 2005.
- [5] 陈明. 实用软件工程基础[M]. 北京: 清华大学出版社, 2002.
- [6] 黄德志, 汤庆恩. C#程序设计实用教程[M]. 北京: 冶金工业出版社, 2002.
- [7] 王敏. C#线程参考手册[M]. 北京: 清华大学出版社, 2003.
- [8] 葛昊晗, 烫涌涛, 李强. C#大学教程[M]. 北京: 清华大学出版社, 2003.
- [9] 李泽, 陈彬, 唐俊瞿, 张曜. C#函数实用手册[M]. 北京: 冶金工业出版社, 2006.
- [10] 熊盛心, 许志庆, 李钦. Visual C# .NET 语言参考手册[M]. 北京: 清华大学出版社, 2002.
- [11] (美)富尔斯汀(Feuersteinin, S.)主编 龚波译. Oracle PL/SQL 最佳实践, 北京: 机械工业出版社, 2009.02
- [12] 王恒山. 管理信息系统, 北京: 机械工业出版社, 2008
- [13] 黄达. 基于.net 的网站业务逻辑函数设计原则初探[J]. 岳阳职业技术学院报. 2008. (03)
- [14] 陈金开. 基于.net 泛型 web 系统轻量级框架实现 [J]. 福建电脑, 2010, (01):102-103.