|  |  |  |
| --- | --- | --- |
| 南农大  **数据库课程设计报告**  XH2 | | |
|  | 题 目: |  |
|  | 组 长: | 学号: |
|  | 组 员 | 学号: |
|  |  | 学号: |
|  | 学 院: | 人工智能学院 |
|  | 专 业: |  |
|  | 班 级: |  |
|  | 指导教师: | 黄芬 职称: 副教授 |
| 年 月 日 | | |

目录

**1开发技术、开发工具及技术路线**4

1.1开发技术及开发工具5

**1.1.1开发技术**5

1**.1.2开发工具**5

1.2技术路线6

**2功能分析与设计**7

2.1需求分析7

2**.1.1用户调研**7

2**.1.2初步设计**7

2.2系统设计7

2**.2.1系统流程设计**7

2**.2.2系统功能分析**8

2.3数据库概念结构设计8

2.4数据库逻辑结构设计9

2**.4.1 E-R图向关系模型的转换**9

2**.4.2利用数据依赖的理论对关系模式进行分析**9

2**.4.3利用外码对关系模式进行分析**9

2.5数据库物理结构设计10

2**.5.1确定存储结构**10

2**.5.2评价物理结构**10

2.6模块划分10

**3实现**11

3.1前端界面实现11

3**.1.1用户界面设计**11

3**.1.2经理界面设计**12

3**.1.3管理员界面设计以及其他界面**13

3.2后端代码实现13

3**.2.1 C#与SQL Server的连接和交互**14

3**.2.2部分功能代码**14

3**.2.3 SQL Server建立数据库**17

3**.2.4 SQL Server存储过程的设置**22

**4系统测试与应用**22

4.1系统测试运行环境22

4.2数据库测试与运行22

4**.2.1用户注册**23

4**.2.2用户查看菜品信息**23

4**.2.3经理新增进货信息**23

4**.2.4查询一段时间内销售总额**24

4**.2.5删除存储表中的存货信息**24

**5总结与展望**25

5.1总结25

5.2不足25

5.3展望26

**参考文献**26

食堂食材的进销存管理系统

**1 开发技术、开发工具及技术路线**

1.1 开发技术及开发工具

1.1.1 开发技术

本平台是基于C#开发的，后台数据库采用Microsoft SQL server。

（1）C#

C#是一个现代的、通用的、面向对象的编程语言，它是由微软（Microsoft）开发的，由Ecma和ISO核准认可的。它是由Anders Hejlsberg和他的团队在.Net框架开发期间开发的。

C#专为公共语言基础结构（CLI）设计的，CLI由可执行代码和运行时环境成，允许在不同的计算机平台和体系结构上使用各种高级语言。它是微软公司结合C/C++和Java等语言的特点设计的一种新的程序设计语言，它基于.NET Framework通用平台，C#程序开发人员可以直接使用.NET Framework中完整且丰富的类库设计出跨平台的软件系统。C#具有简单易学、使用方便、采用可视化设计方法开发复杂软件系统的特点。

（2）SQL server

SQL Server是Microsoft公司推出的关系型数据库管理系统，具有使用方便可伸缩性好与相关软件集成度高等优点，可跨越从运行Microsoft Windows 98的微型计算机到Microsoft Windows 2012的大型多处理器的服务器等多种平台使用。

Microsoft Server是一个全面的数据库平台，使用集成的商业智能（BI）工具提供了企业级的数据管理。Microsoft SQL Server数据库引擎为关系型数据和结构化数据提供了更安全可靠的存储功能，使您可以构建和管理用于业务的高可用和高性能的数据应用程序。

1.2.2 开发工具

本系统在Windows 10操作系统上开发，采用Microsoft Visual Studio 2019集成开发环境进行开发，数据库管理系统采用Microsoft SQL Server Management Studio 19。

（1）Microsoft Visual Studio

Microsoft Visual Studio，简称VS。Visual Studio 集成开发环境是一种创新启动板，可用于编辑、调试并生成代码，然后发布应用。集成开发环境 (IDE) 是一个功能丰富的程序，可用于软件开发的许多方面。除了大多数 IDE 提供的标准编辑器和调试器之外，Visual Studio 还包括编译器、代码完成工具、图形设计器和许多其他功能，以简化软件开发过程。

（2）Microsoft SQL Server Management Studio

SQL Server Management Studio 是一个集成环境，用于访问、配置、管理和开发SQL Server的所有组件。SQL Server Management Studio组合了大量图形工具和丰富的脚本编辑器，使各种技术水平的开发人员和管理员都能访问 SQL Server。  
 SQL Server Management Studio将早期版本的SQL Server中所包含的企业管理器、查询分析器和Analysis Manager功能整合到单一的环境中。此外，SQL Server Management Studio还可以和SQL Server的所有组件协同工作，例如Reporting Services Integration Services和SQL Server Compact 3.5 SP1。开发人员可以获得熟悉的体验，而数据库管理员可获得功能齐全的单一实用工具，其中包含易于使用的图形工具和丰富的脚本撰写功能。

1.2 技术路线

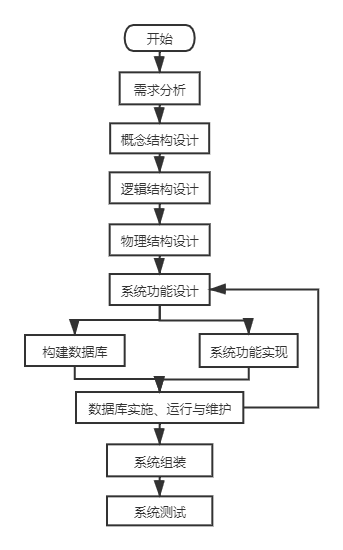
****本系统着重分析当前校园食堂食材的进销存情况，详细设计了针对食堂经理、管理人员以及用户的功能，完成诸如用户可以查看菜品及其价格、剩余量和反馈意见等。食堂经理通过收集用户的反馈来有选择性的联系供应商进行货物的采购以及增加或者删除菜品以进一步满足消费者的需求从而获得更大的利润。食堂的管理人员则可以审查经理和用户的操作行为，避免不法经理和用户进入该管理系统进行不法操作。利用软件工程原理和结构化程序设计思想，结合关系型数据库平台、C#面向对象的程序设计、.NET平台等技术，设计实现了食堂食材的进销存管理系统。  
本平台的开发技术路线如图1-1所示：

图 1-1 开发路线图

**2 功能分析与设计**

2.1 需求分析

**2.1.1 用户调研**

此食堂食材进销存管理系统针对的用户是食堂管理人员和大学生，满足大学生实时方便地查看当日食堂菜品情况，以便消费；满足管理人员对食材进货，销售，存储管理的需求，以便提高工作效率。

**2.1.2 初步设计**

根据系统用户端需求，将系统功能大致分为如下几块：

(1)用户登录：初始界面为登录界面，同时可以选择注册账户或密码寻回；登录后，会自动跳转至消费用户界面或经理用户界面或管理员用户界面。

(2)消费者用户：跳转至消费者信息界面，可在此界面选择，查看当日菜品销售情况和填写意见反馈表，也可以修改个人信息。

(3)经理用户：跳转至经理信息界面，可在此界面选择，查看或修改（增删改）进货信息，货物存储信息，查询销售额情况，查看消费者的意见反馈情况，也可以在此界面修改个人信息。

(4)管理员用户：跳转至管理员信息界面，可在此界面选择，查看所有用户，对恶意的用户进行账户查封，也可以选择修改个人信息。

2.2 系统设计

**2.2.1 系统流程设计**

根据对需求分析阶段所得到的信息进行分析，将系统流程细化到如图3-1所示：

****

图3-1系统流程图

**2.2.2 系统功能分析**

根据上述系统流程分析，系统功能概括为图3-2：



图3-2系统功能图

2.3 数据库概念结构设计

数据库概念结构设计即通过对用户需求进行综合，归纳与抽象，形成一个独立于具体数据库管理系统的概念模型，能真实、充分地反映现实世界，包括事物和事物之间的联系，能满足用户对数据的处理要求[1]。本课程设计采用E-R图来完成概念结构设计。

根据需求分析，我们建立了如下的E-R模型，如图3-3所示：

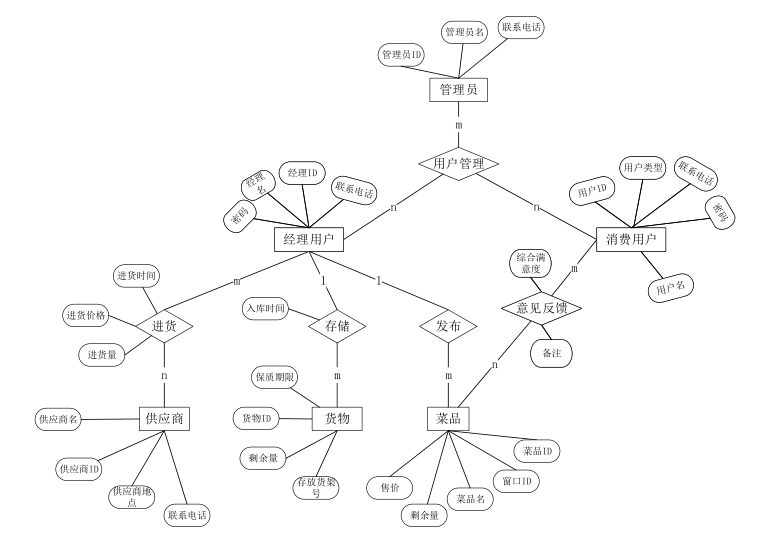


图3-3 E-R模型图

2.4 数据库逻辑结构设计

数据库逻辑结构设计即将概念结构转换为某个数据库管理系统所支持的数据模型，并对其进行优化。本课程设计采用关系数据库，需要实现E-R图到关系模型的转换。

**2.4.1 E-R图向关系模型的转换**

…………

**2.4.2 利用数据依赖的理论对关系模式进行分析**

分析上述关系模式表，……，设计的关系表至少达到3NF。

**2.4.3最终关系模式设计**

（按照5元组描述清楚各关系模式的逻辑结构与设计，以及主键、外健、完整性约束等）

2.5 数据库物理结构设计

数据库在物理设备上的存储结构与存取方法称为数据库的物理结构，它依赖于选定的数据库管理系统。为一个给定的逻辑数据模型选取一个最适合应用要求的物理结构的过程，就是数据库的物理设计[2]。

通常包括两步：（1）确定数据库的物理结构，在关系数据库中主要指存取方法和存储结构；（2）对物理结构进行评价，重点是时间和空间效率。

**2.5.1确定存储结构**

本平台采用的数据库平台是SQL Server2019，数据库以文件形式存储在计算机磁盘中，其中主要的文件有主数据文件，次数据文件，事物日志文件：

（1）主数据文件

主数据文件扩展名是.mdf，包含了数据库的初始信息，并记录数据库还拥有哪些文件。每个数据库有且只能有一个主数据文件。

（2）次数据文件

除了主数据文件以外的所有其他数据文件都是次数据库文件，扩展名是.ndf。次数据文件不是数据库必需的文件，可以有多个。

（3）事务日记文件

每个数据库至少拥有一个自己的日记文件（也可以拥有多个日记文件）。日默认扩展名是.ldf,用来记录数据库的事务以及每个数据对数据库所做的修改，用于数据库恢复阶段。

（4）索引设计

……

**2.5.2评价物理结构**

……

2.6 模块划分

按照上述分析将系统最终划分为如下5个模块：

（1）用户注册模块：

用户如果希望使用本平台的功能则必须完成注册，需要设定账号，密码，并且上传一张自己校园卡的照片供管理员查验，信息填写完整后即可提交注册申请并使用本平台；

（2）登录模块：

* 消费用户填写消费用户账号、密码登录平台，进入消费用户界面；
* 经理用户填写经理用户账号、密码登录平台，进入经理用户界面；
* 管理员填写管理员账号、密码来，进入管理员界面；
* 点击“密码寻回”，帮助找回密码

（3）消费用户模块：

消费用户模块又分成如下3个子模块：

a. 查看菜品信息子模块：

b. 填写意见反馈子模块：

c. 个人信息管理模块：

（4）经理用户模块：

经理用户模块又分为如下5个子模块：

a.发布菜品信息

b.查看或修改货物存储信息

c.查看或修改进货信息

d.查看消费意见反馈情况

e.个人信息管理模块

（5）管理员模块：

管理员模块又分成如下2个子模块：

a. 消费用户和经理用户管理子模块

b. 个人信息管理模块

**3实现**

3.1 前端界面实现

此次数据库课程设计的前端界面开发主要借助Visual Studio 2019集成开发环境中针对C#窗体应用程序的设计器来实现项目前端界面的设计和开发工作。开发界面如图4-1所示。

利用VS中针对C#窗体应用的设计器，可以较为方便、快捷地实现所需窗体界面的设计。在实际操作中，只需要将所需的控件拖拽至窗体，再设置其属性，如位置、大小等，设计器将自动生成后缀名为.cs的文件，记录当前窗体和控件的属性信息。通过这种交互式的界面设计方法，大大减轻了程序员的工作量，使得界面开发更为轻松、便捷[3]。

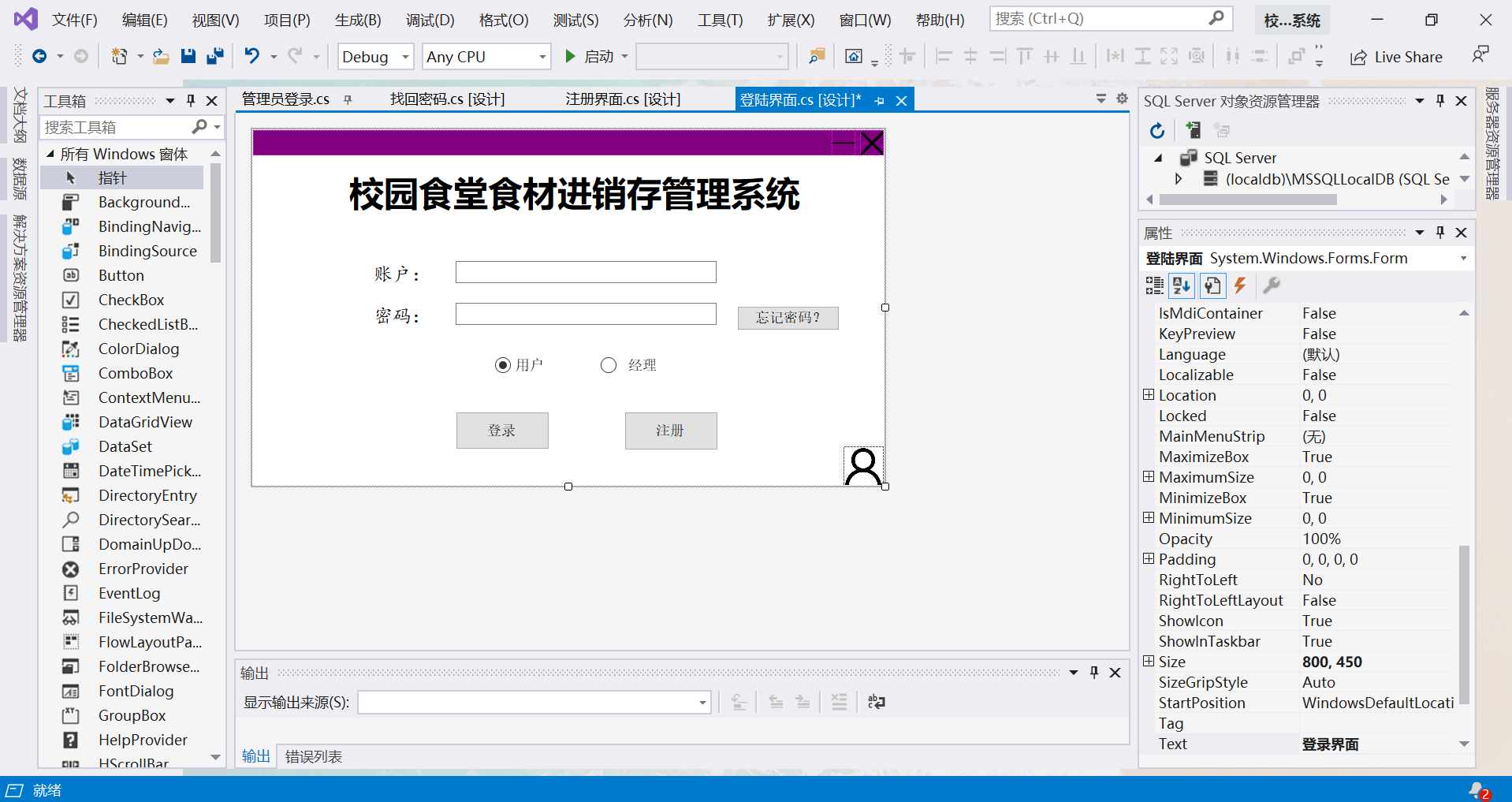


图4-1 VS2019前端界面开发设计器

**3.1.1 用户界面设计**

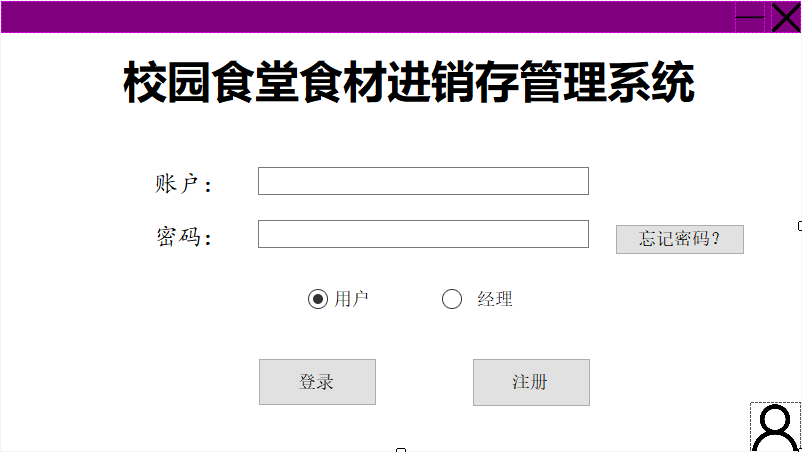
 用户界面如图4-2-1至图4-2-6所示。其中主要展示了登录界面、菜单界面、具体功能界面等主要界面设计。

图4-2-1 登录界面 图4-2-2 用户菜单界面

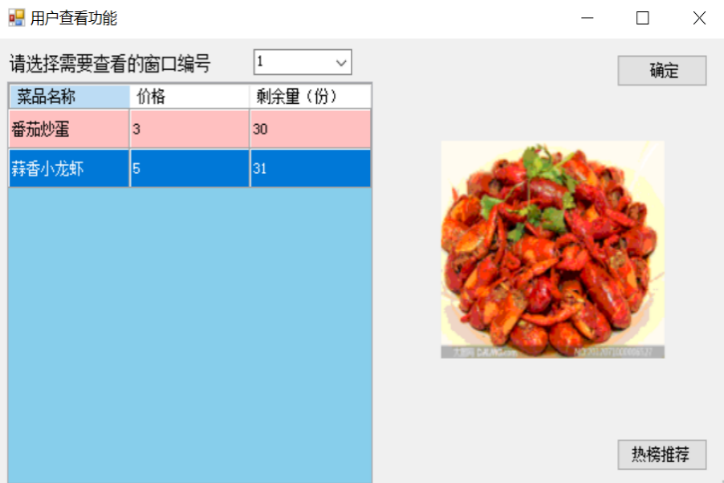


 图4-2-3 查看界面 图4-2-4 热销榜单界面



图4-2-5 意见反馈界面 图4-2-6 个人信息修改界面

**3.1.2 经理界面设计**

经理界面如图4-2-7至图4-2-10所示。由于登录界面与菜单界面与用户的界面类似，这里不再赘述，主要展示经理的具体功能界面设计。



 图4-2-7 进货管理界面 图4-2-8 修改进货信息界面

 图4-2-9 销售管理界面 图4-2-10 存储管理界面

**3.1.3 管理员界面设计以及其他界面**

管理员的主要功能是对用户和经理进行人事管理，包括查看信息和回收权限。并不是这次课程设计的主体部分，界面相对较少。因此，对管理员界面和一些其他界面做简要展示。

 图4-2-11 管理员菜单界面 图4-2-12 管理权限界面

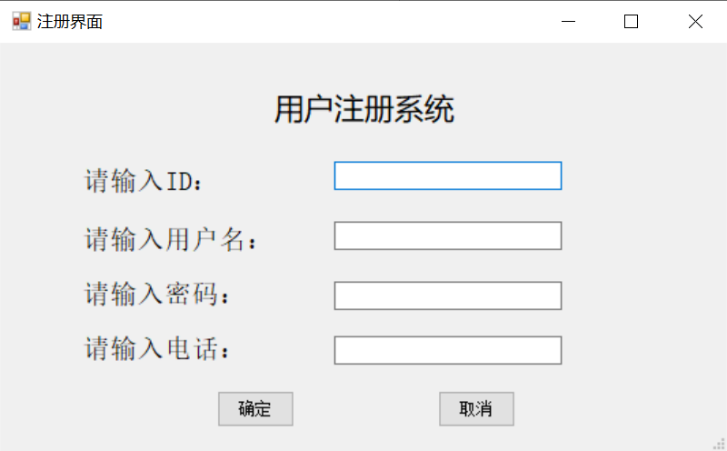


图4-2-13 设置密码界面 图4-2-14 注册界面

3.2 后端代码实现

此次数据库课程设计的代码量较多且具有冗余性，为此仅选取核心和关键部分的代码进行说明和解释。

**3.2.1 C#与SQL Server的连接和交互**

在此次课程设计中，为了实现C#与SQL Server的交互功能，新建了一个类以封装针对数据库的各种操作。下面针对该类方法中的使用到的对象做一些简单介绍[4]：

（1）SqlConnection对象：如果要连接SQL Server数据库，必须使用System.Data.SqlClient命名空间下的SqlConnection类，所以首先要通过using System.Data.SqlClient命令引用命名空间。连接数据库之后，通过调用SqlConnection对象的Open方法打开数据库。当完成对数据库的操作后，要关闭与数据库的连接，以释放占用的资源。调用SqlConnection对象的Close方法关闭与数据库的连接即可。

（2）SqlCommand对象：该对象用于向SQL Server数据库发送SQL语句，位于System.Data.SqlClient命名空间。下面介绍该对象的两个方法：

ExecuteNonQuery方法：用于执行SQL语句，并返回受影响的行数。在使用SqlCommand对象向数据库发送增、删、改命令时，通常用ExecuteNonQuery方法执行发送的SQL语句。

ExecuteReader方法：用于执行SQL语句，并生成一个包含数据的SqlDataReader对象的实例。

（3）SqlDataReader对象：该对象是数据读取器对象，提供只读向前的游标。通过调用该对象的Read方法读取数据，使SqlDataReader前进到下一条记录。

**3.2.2 部分功能代码**

为了更好地展示此次数据库课程设计的具体实现，节选了一些基本功能的代码，来说明C#是如何与SQL server进行交互的。

（1）登录功能

using System;

using System.Windows.Forms;

…

private void button1\_Click(object sender, EventArgs e)

{

if (textBox1.Text != "" && textBox2.Text != "")//如果输入不为空

{

Login();

}

else

{

MessageBox.Show("输入有空项，请重新输入。","提示信息");

}

}

public void Login()

{

//用户

if (radioButtonuser.Checked == true)

{

Dao dao = new Dao();

string sql = $"select \*from t\_user where Uid='{textBox1.Text}' and Upsw='{textBox2.Text}'";

IDataReader dc = dao.read(sql);

if (dc.Read())

{

//存储用户的ID和姓名

Data.UID = dc["Uid"].ToString();

Data.UName = dc["Uname"].ToString();

Data.Uphone = dc["Uphone"].ToString();

Data.Upsw = dc["Upsw"].ToString();

MessageBox.Show("登陆成功","提示信息");

用户界面 user = new 用户界面();

this.Hide();

user.ShowDialog();

this.Show();

}

else

{

MessageBox.Show("很抱歉，您还不是用户，登陆失败。请先注册！","提示信息");

}

dao.DaoClose();

}

if (radioButtonadmin.Checked == true)

{//管理员

Dao dao = new Dao();

string sql = $"select \*from t\_admin where Aid='{textBox1.Text}' and Apsw='{textBox2.Text}'";

IDataReader dc = dao.read(sql);

if (dc.Read())

{

Data.AID = dc["Aid"].ToString();

Data.ANAME = dc["Aname"].ToString();

Data.Aphone = dc["Aphone"].ToString();

Data.Apsw = dc["Apsw"].ToString();

MessageBox.Show("登陆成功","提示信息");

管理员界面 a = new 管理员界面();

this.Hide();

a.ShowDialog();

this.Show();

}

else

{

MessageBox.Show("很抱歉，您还不是管理员，登陆失败。", "提示信息");

}

dao.DaoClose();

}

}

(2) 删除功能

private void button2\_Click(object sender, EventArgs e)

{

try

{

string id = dataGridView1.SelectedRows[0].Cells[0].Value.ToString();

DialogResult dr = MessageBox.Show("确认删除吗？", "信息提示", MessageBoxButtons.OKCancel, MessageBoxIcon.Question);//中间内容为第一个参数

if (dr == DialogResult.OK)//如果点击了"确定"

{

string sql = $"delete from t\_stock where JHid='{id}'";

Dao dao = new Dao();

if (dao.Execute(sql) > 0)//返回了行数

{

MessageBox.Show("删除成功");

Table();//相当于刷新，重新把数据库表中的内容传到dataGridView1

}

else

{

MessageBox.Show("删除失败" + sql);

}

dao.DaoClose();

}

}

catch

{

MessageBox.Show("请先在表格中选中要删除的进货记录", "信息提示", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

public void Table()

{

dataGridView1.Rows.Clear();//清空旧数据

Dao dao = new Dao();

string sql = $"select \*from t\_stock";

IDataReader dc = dao.read(sql);

while (dc.Read())

{

dataGridView1.Rows.Add(dc[0].ToString(),dc[1].ToString(), dc[2].ToString(), dc[3].ToString(), dc[4].ToString(),dc[5].ToString(),dc[6].ToString(),dc[7].ToString());

}//将数据库中的一行记录添加到dataGridView1的一行当中

dc.Close();

dao.DaoClose();

}

**3.2.3 SQL Server建立数据库**

建立以下8个基本表（表4-1至表4-8）

表4-1 Purchase（进货表）

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 序号 | 字段 | 别名 | 数据类型 | 数据长度 | 码 | 是否可空 |
| 1 | purchaseid | 货物ID | int |  | PK | 否 |
| 2 | purchasename | 货物名 | char | 20 |  | 否 |
| 3 | suppliersid | 供应商ID | int |  | FK | 否 |
| 4 | purchasemoney | 进货价格 | smallmoney |  |  | 是 |
| 5 | purchasenumber | 货物数量 | int |  |  | 是 |
| 6 | purchasetime | 进货时间 | smalldatetime |  |  | 是 |

表4-2 Sales（销售表）

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 序号 | 字段 | 别名 | 数据类型 | 数据长度 | 码 | 是否可空 |
| 1 | windowid | 窗口ID | int |  | PK | 否 |
| 2 | foodid | 菜品ID | int |  | PK | 否 |
| 3 | foodname | 菜品名 | char | 20 |  | 否 |
| 4 | salesmoney | 售价 | smallmoney |  |  | 是 |
| 5 | salessave | 剩余量 | int |  |  | 是 |

表4-3 Storage（存储货物表）

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 序号 | 字段 | 别名 | 数据类型 | 数据长度 | 码 | 是否可空 |
| 1 | purchaseid | 货物ID | int |  | PK  FK | 否 |
| 2 | shelves | 存放货架号 | int |  |  | 是 |
| 3 | storagesave | 库存量 | int |  |  | 是 |
| 4 | intime | 入库时间 | smalldatetime |  |  | 是 |
| 5 | outtime | 保质期限 | smalldatetime |  |  | 是 |

表4-4 Managers（经理用户表）

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 序号 | 字段 | 别名 | 数据类型 | 数据长度 | 码 | 是否可空 |
| 1 | managersid | 经理ID | int |  | PK | 否 |
| 2 | managersname | 经理名 | char | 20 |  | 否 |
| 3 | mpassword | 密码 | char | 20 |  | 否 |
| 4 | mphone | 电话号码 | char | 15 |  | 是 |

表4-5 Consumers（消费用户表）

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 序号 | 字段 | 别名 | 数据类型 | 数据长度 | 码 | 是否可空 |
| 1 | consumersid | 用户ID | int |  | PK | 否 |
| 2 | consumersname | 用户名 | char | 20 |  | 否 |
| 3 | cpassword | 密码 | char | 20 |  | 否 |
| 4 | cphone | 电话号码 | char | 15 |  | 是 |
| 5 | consumerstype | 用户类型 | smallint |  |  | 是 |

表4-6 Administrators（管理员表）

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 序号 | 字段 | 别名 | 数据类型 | 数据长度 | 码 | 是否可空 |
| 1 | administratrorsid | 管理员ID | int |  | PK | 否 |
| 2 | administratorname | 管理员名 | char | 20 |  | 否 |
| 3 | aphone | 电话号码 | char | 15 |  | 是 |

表4-7 Suppliers（供应商表）

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 序号 | 字段 | 别名 | 数据类型 | 数据长度 | 码 | 是否可空 |
| 1 | suppliersid | 供应商ID | int |  | PK | 否 |
| 2 | suppliersname | 供应商名 | char | 20 |  | 否 |
| 3 | suppliersplace | 供应商地点 | char | 100 |  | 是 |
| 4 | sphone | 电话号码 | char | 15 |  | 是 |

表4-8 Feedback（意见反馈表）

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 序号 | 字段 | 别名 | 数据类型 | 数据长度 | 码 | 是否可空 |
| 1 | consumersid | 用户ID | int |  | PK  FK | 否 |
| 2 | windowid | 窗口ID | int |  | PK  FK | 否 |
| 3 | foodid | 菜品ID | int |  | PK  FK | 否 |
| 4 | satisfy | 综合满意度 | int |  |  | 是 |
| 5 | marks | 备注 | char | 200 |  | 是 |

数据库关系图如图4-9所示：

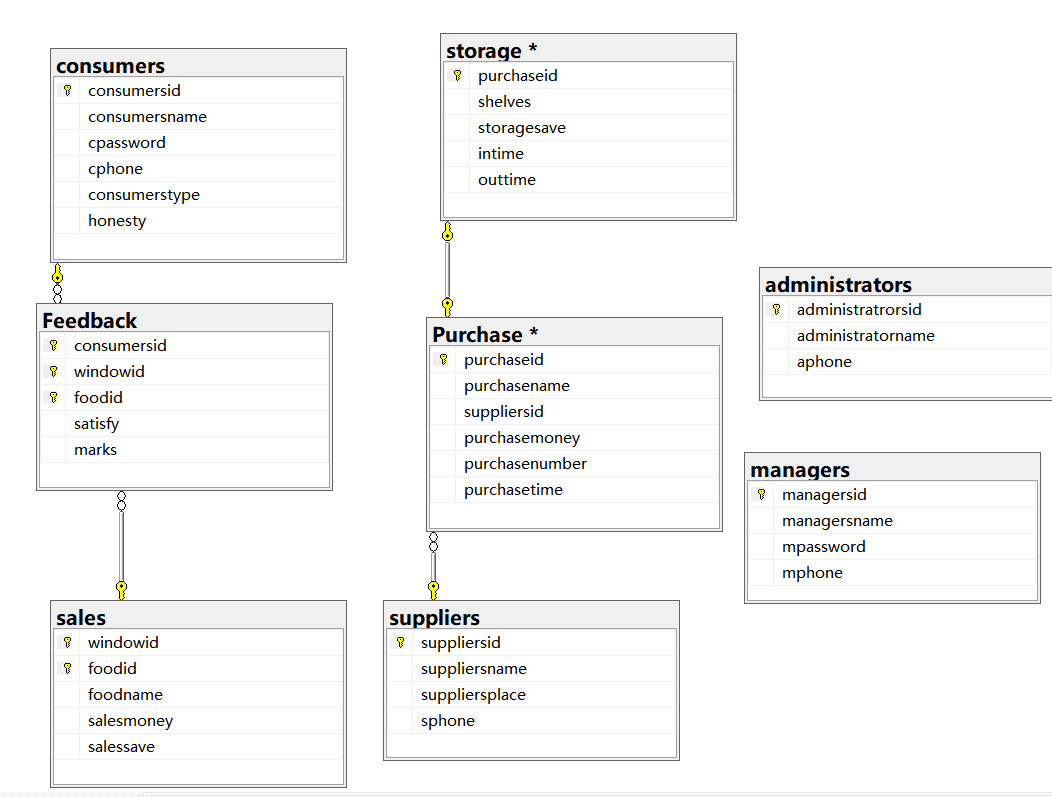


图4-9 数据库关系图

根据数据库逻辑结构设计，编写SQL语句建立数据库。

具体实现代码如下：

/\*建立进货表\*/

drop table if exists Purchase

create table Purchase

(

purchaseid int not null,

purchasename char(20) not null,

suppliersid int not null,

purchasemoney smallmoney,

purchasenumber int,

purchasetime smalldatetime,

Constraint purchase1 primary key(purchaseid),

Constraint purchase2 Foreign key(suppliersid) references suppliers(suppliersid)

)

/\*建立销售表\*/

drop table if exists sales

create table sales

(

windowid int not null,

foodid int not null,

foodname char(20) not null,

salesmoney smallmoney,

salessave int,

Constraint sales1 primary key(windowid,foodid)

)

/\*建立存储货物表\*/

drop table if exists storage

create table storage

(

purchaseid int not null,

shelves int ,

storagesave int,

intime smalldatetime,

outtime smalldatetime,

Constraint storage1 primary key(purchaseid)，

Constraint storage2 foreign key(purchaseid)references Purchase(purchaseid)

)

/\*建立经理用户表\*/

drop table if exists managers

create table managers

(

managersid int not null,

managersname char(20) not null,

mpassword char(20) ,

mphone char(20),

Constraint managers1 primary key(managersid)

)

/\*建立消费用户表\*/

drop table if exists consumers

create table consumers

(

consumersid int not null,

consumersname char(20) not null,

cpassword char(20) not null,

cphone char(15),

consumerstype smallint,

Constraint consumers1 primary key(consumersid)

)

/\*建立管理员表\*/

drop table if exists administrators

create table administrators

(

administratrorsid int not null,

administratorname char(20) not null,

aphone char(15),

Constraint administrators1 primary key(administratrorsid )

)

/\*建立供应商表\*/

drop table if exists suppliers

create table suppliers

(

suppliersid int not null,

suppliersname char(20) not null,

suppliersplace char(100) ,

sphone char(15),

Constraint suppliers1 primary key(supplierid )

）

/\*建立意见反馈表\*/

drop table if exists Feedback

create table Feedback

(

consumersid int not null,

windowid int not null,

foodid int not null,

satisfy int ,

marks char(200),

Constraint feedback1 primary key(consumersid,windowid,foodid),

Constraint feedback2 foreign key(consumersid) references consumers(consumersid),

Constraint feedback3 foreign key(windowid,foodid) references sales(windowid,foodid)

Constraint feedsatisfy check (satisfy between 0 and 100)

)

针对上述SQL语句的的说明如下：

1. 在建表前加入drop table if exists 表名的语句，是方便如果建表之后希望对表进行修改，只需要对create table语句进行修改，执行的时候与drop table语句一起执行，DBMS就会删除原来的表再建一个新的表，就可以完成对表的修改。便于操作。
2. primary key约束语句体现了实体完整性,不为空置，foreign key约束语句体现了参照完整性；要么取空值，要么等于被参照表的主码值；not null,check约束语句体现了用户定义的完整性，not null保证列值唯一，check保证列值满足一个条件表达式。
3. constraint语句用来对完整性约束条件命名，从而可以灵活地增加。删除一个完整性约束条件。

**3.2.4 SQL Server存储过程的设置**

在进货表中，只提供了供应商ID，如果知道了货物ID，需要知道供应商信息，经理还需到供应商表中根据对应的供应商ID查找对应信息，增加了查找的复杂性，由此，我们采用了采取数据库技术中的“存储过程”和“游标”来优化查找过程。当提供了货物ID，可以返回其（供应商名，供应商地点，电话号码)。

具体实现代码如下：

create procedure purchase\_procedure

@purchaseid int,@supplied\_cursor cursor varying output

/\*定义参数列表@purchaseid为用户提供的货物ID号，作为输入变量；

@supplied\_cursor作为游标变量，返回结果集\*/

as

set @supplied\_cursor=cursor for

select

purchaseid,suppliersname,suppliersplace,sphone

from

suppliers,purchase /\*进货表和供应商表通过供应商ID做连接\*/

where

suppliers.suppliersid=purchase.suppliersid

and purchaseid=@purchaseid

open @supplied\_cursor

针对上述SQL语句的的说明如下[6]：

1. 存储过程包括过程首部和过程体。在过程首部，“过程名”是数据库服务器合法的对象标识；参数列表[参数1，参数2···]用名字来标识调用时给出的参数值，必须指定值的数据类型。可以定义输入参数，输出参数或输入/输出参数，默认为输入参数，也可以无参数，每个参数名前规定要有一个“@”符号,输出参数要加output。
2. 过程体是一个<过程化SQL块>，作用是描述该存储过程的操作，包括声明部分和可执行部分。在as之后定义存储过程体。

**4 系统测试与应用**

4.1 系统测试运行环境

测试系统软件环境：操作系统为Windows 10并且已经安装了.NET Framework 4.7.2，同时已安装SQL Server Management Studio v18.6。

测试系统硬件配置：处理器为Intel Core i5-8250U,内存8G.

4.2 数据库测试与运行

由于此次数据库课程设计中的功能有类似或重复的情况，比如修改个人信息、增添商品信息等，在这里展示主要功能的测试。

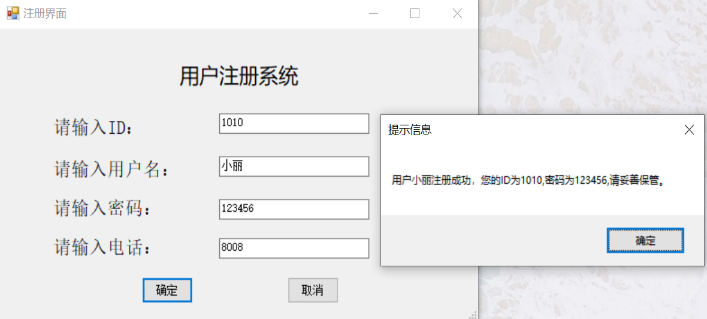
4**.2.1 用户注册**

图5-1 用户注册界面、传入数据库的数据

4**.2.2 用户查看菜品信息**

图5-2 查看数据库中的菜品数据

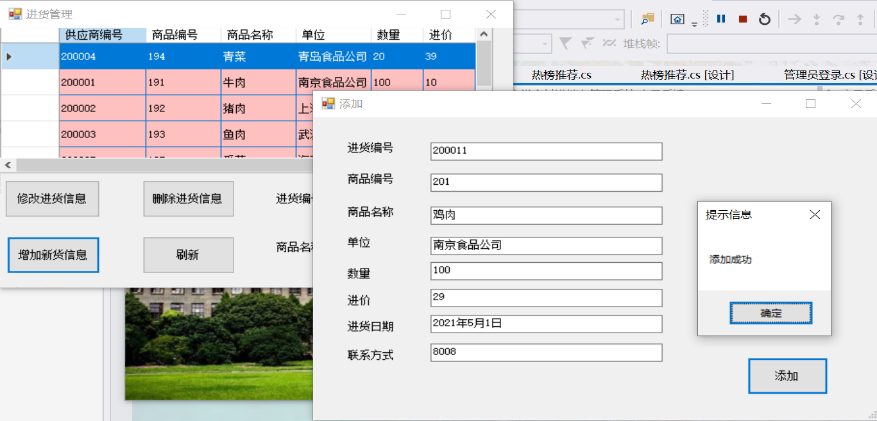
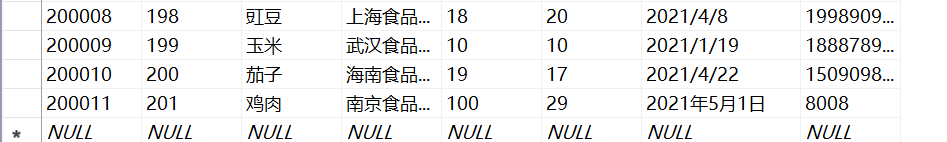
4**.2.3 经理新增进货信息**

图5-3 向数据库中增添进货信息

4**.2.4 查询一段时间内销售总额**

图5-4 根据数据库中的数据得销售额

4**.2.5 删除存储表中的存货信息**

图5-5 删除数据库中的一条记录

**5 总结与展望**

5.1 总结

本系统实现了如下核心功能：

* 用户中心：

（1）用户注册系统：用户按照我们界面的提示通过输入ID、用户名、密码即 可注册。

（2）用户登录系统：用户通过自己的账户和密码登陆该系统。注意：只有在系统内存在的用户才可进行登录操作。用户登录之后可以完成的功能有：

（a）查看：用户可以选择要查看的窗口编号，随后查看所选择窗口的菜品名称、价格、剩余量（份）以及该菜品的样式图例，除此之外用户还可以看到我们的热榜推荐菜品。

（b）反馈：用户在食用菜品之后可以通过该系统填写自己的意见反馈，按照我们界面提示选择自己所选择的窗口号、消费的菜品，然后可以反馈自己对此菜品的满意度、问题描述以及可以将有关该菜品的相关菜品上传反馈给我们，可留下自己的联系方式便于后期我们调查联系，最后提交反馈即可。

（c）个人信息修改：用户可以修改自己的个人信息，同时也可以修改密码

（d）返回：返回登陆时的主界面

* 经理中心：

（1）经理登录系统：经理通过自己的账户和密码登陆该系统。注意：只有在系统内存在的经理才可进行登录操作，且经理的账号不允许注册。经理登录之后可以完成的功能有：

（a）进货管理：经理通过进货编号和商品名称可以查询该货品的单位、数量、进价、进货日期等进货信息，同时也可以增加、修改、删除所选择的进货信息

（b）销售管理：经理通过该界面可以清楚的知道窗口号、窗口名、菜品名称、价格及其销量等信息，除此之外经理可以选择查询某一时间段内某一菜品的销售额同时也可以选择查询某一时间段内的销售总额

（c）存储管理：经理通过该界面能够查询不同区或者是所有区的货物ID、货物名称、货物单元、剩余数量、保质时长等信息，同时经理可以根据需要来添加、修改和删除存货信息

（d）个人信息修改：经理可以修改自己的个人信息，同时也可以修改密码

（e）返回：返回登陆时的主界面

* 管理员中心：

（1）个人信息修改：管理员可以修改自己的个人信息，同时也可以修改密码

（2）管理系统：管理员可以对经理和用户进行管理，根据需求来修改、删除用户和经理信息禁止不法经理和用户对该系统进行不法操作

通过对本课题的研究以及该管理系统的设计与开发，我们对数据库的理论知识有了更进一步的理解，也能更加熟练地使用SQL语句，同时也学习了C#的基本语法并可以使用C#完成基础的软件开发。在整个数据库课设中，本小组成员分工明确，相互配合，积极讨论，团结互助，在整个过程中我们小组成员一起成长一起进步。在我们小组的团结合作下，通过查阅资料、网上搜索等方式共同完成了本次课程设计，最终我们都收获很多。

5.2 不足

目前，本系统的大体核心功能都基本实现，但是整个系统的应用上来说还存在一些不足：

* 用户和经理在注册信息时账号是随意的，没有限制
* 用户端对于食堂窗口菜品等信息仅仅局限于查看和反馈功能，不能查询所需等待时间以避免一到饭点就出现排长龙的现象等等，未能扩充更多的功能；
* 用户端通过该系统不能线上进行点餐结账以及提前预定等功能，以省去一些繁琐复杂的环节
* 在数据库设计方面未考虑日志文件，以防止数据库发生故障后能够及时对数据库进行恢复
* 在数据库、方面未能完全考虑数据库的安全性与完整性问题
* 整个界面可以设计得更加美观

5.3 展望

本管理系统面向食堂经理、管理员、消费者用户等，希望本小组研究的食堂食材的进销存管理系统能够为大家带来一定程度上的便利。

本管理系统是对于食堂食材的进销存的一次尝试，我们仅仅只做出一个小小的雏形。希望以后我们能以本次课程设计为基础，在已有的基础之上优化并扩展新的功能，我们可以设计用户可以线上点餐、结账，食堂经理可以给通过线上接收消息并提供上门送餐服务等等更多的功能，让该管理系统更好的切实所需，进一步满足人们的需求。

**参考文献**

[1] 王珊, 萨师煊. 《数据库系统概论（第五版）》 高等教育出版社.

[2] 马晓梅. 《SQL Server实验指导》 清华大学出版社.

[3] 李春葆,曾平,喻丹丹.《C#程序设计教程（第3版）》 清华大学出版社.

[4] 王小科.《C#开发实战》清华大学出版社.

[5] 王立平，刘祥淼.《SQL Server从入门到精通》清华大学出版社.

[6] 李俊民，王国胜.《SQL Server基础与案例开发详解》清华大学出版社.