

软件工程课程实践报告

项 目 名 称： 超市零售管理信息系统

班 级： 2022级计算机科学与技术三班

小 组 成 员： 侯栩涛 薛琦嘉 朱鹏超

任 课 教 师： 王博

2024年11月

项目基本信息

|  |  |
| --- | --- |
| 项目名称 |  |
| 项目简介 | 该超市零售管理信息系统旨在为超市商品和会员管理提供一个综合的零售管理平台，协助超市存储和管理进货、销售、库存以及会员信息。通过这一系统，超市能够更加高效地处理各种业务操作，确保数据的准确性和实时性，从而显著提高整体管理效率。系统涵盖了从商品采购到销售出库的全流程管理，提供详细的库存跟踪和销售分析功能，使管理者能够实时了解库存状况和销售趋势。该系统不仅简化了日常运营流程，还提供了强大的数据分析和报表功能，帮助管理者做出更明智的决策。 |
| 小组成员 | 侯栩涛 薛琦嘉 朱鹏超 |
| 任务分工 | 薛琦嘉：代码编写，程序开发  侯栩涛：代码编写，文档撰写  朱鹏超：文档撰写，数据库设计 |

课程设计评定成绩记录

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 指导教师意见 | 系统完成情况：优 良 中 差 | | | |
| 报告完成情况：优 良 中 差 | | | |
| 答辩评定成绩 | 团队整体成绩： | | | |
| 成员成绩 |  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
| 综 合 成 绩 |  | | | |

1. **引言**

**（一）背景**

当今时代，零售行业正历经深刻变革，电商冲击、消费者需求多元，让实体超市面临前所未有的挑战。传统人工记录、分散管理模式弊端尽显，进货易混乱、库存难精准把控，销售数据滞后，会员服务也流于形式，严重制约超市发展。

与此同时，数字化技术蓬勃兴起，为行业革新带来曙光。在此背景下，我们全力打造超市零售管理信息系统，旨在集成数据、优化流程，让超市重掌运营主动权，以高效管理、精准决策，从容应对市场风云变幻。

**（二）**研究目的

本研究旨在通过软件工程的方法论，设计并实现一套高效、智能的超市零售管理信息系统。该系统将集成商品管理、库存管理、销售管理、顾客关系管理等功能模块，实现业务流程的自动化处理与数据的高效利用。研究的核心目的在于提高超市的运营效率，为管理层提供精准的数据支持，从而优化决策过程，同时提升顾客满意度，促进超市的长期可持续发展。

**（三）**研究意义

该系统不仅能够有效解决超市零售管理中存在的实际问题，提升运营效率，还能为零售行业的信息化转型提供有力支撑。此外，通过软件工程原理的应用，本研究还将为同类系统的开发提供可借鉴的经验和方法，推动软件工程理论与实践的进一步发展。

**二、可行性研究**

**（一）技术可行性分析**

**1.1开发技术成熟：**当前软件开发领域，Java、Python 等编程语言功能强大、生态完备，搭配主流框架如 Spring Boot、Django，能高效搭建稳定的系统架构，满足超市复杂业务逻辑需求。数据库方面，MySQL、sqlsever提供可靠存储方案，事务处理确保数据完整性，应对高并发读写游刃有余。

**1.2 硬件适配广泛：**普通商用服务器、PC 终端，结合常见网络设备，便能组建系统运行环境。如今网络带宽充足，无论是门店内局域网数据传输，还是云端备份、远程访问，都能保障数据实时交互，流畅运行系统各模块，技术门槛与适配难度均较低。

**（二）经济可行性分析**

**2.1开发成本可控：**自主开发团队组建，招聘有经验程序员、测试员，人力成本平摊到项目周期合理即可。

**2.2运营效益可观：**系统上线减少人工记账、盘点时间超 50%，人力精简直接降本；精准库存管控避免积压缺货，销售分析助力营销，预估提升销售额 15% - 30%，短期投入换来长期丰厚盈利，投资回报率诱人。

**（三）操作可行性分析**

**3.1. 用户接受度高：**超市员工经简单培训，能快速上手便捷系统，摆脱繁琐手工流程；消费者受益于快速结算、会员精准服务，购物体验升级，双方对系统接纳意愿强。

**3.2. 法规政策契合：**零售数据管理契合隐私保护、消费者权益法规，系统设计严守规范，加密存储、授权访问个人信息，全程合法合规运营，无政策阻碍，利于长远推广应用。

综合上述分析，该超市零售管理信息系统从技术、经济、操作维度均具备可行性，值得立项推进。

1. 需求分析

**（一）用户需求**

**1.1 超市管理人员：**期望精准掌握库存情况，实时知晓商品进货、销售及库存数量，依此合理安排补货，降低缺货、积压风险；能迅速获取销售报表，剖析各品类销售趋势，为营销决策提供依据。

**1.2.会员顾客：**能随时查询积分余额、等级权益；接收个性化促销信息推送；消费记录清晰可查，兑换积分礼品流程简便。

**（二）系统需求**

**2.1功能需求**

**商品管理**：商品信息录入、修改与删除，含名称、规格、进价、售价、供应商等；库存盘点、预警设置，库存低于阈值自动提示补货；商品分类、标签管理利于筛选查找。

**会员管理**：会员注册、信息维护；积分规则设定、自动累计与扣除；会员金额；

**2.2数据需求**

**商品数据：**名称、规格、进价、售价、库存数量、进货日期、供应商等，数据实时更新，确保库存精准。

**会员数据：**姓名、性别、联系方式、会员卡号、金额等，用于会员识别与权益匹配。

**2.3其他需求**

**性能需求：**系统响应时间不超 3 秒，确保操作流畅；支持至少 50 个收银终端、上千会员同时访问，高并发场景稳定运行。

**安全需求：**用户权限分级，收银员、管理员权限各异；数据加密存储与传输，防信息泄露；定期备份，遇故障可快速恢复数据。

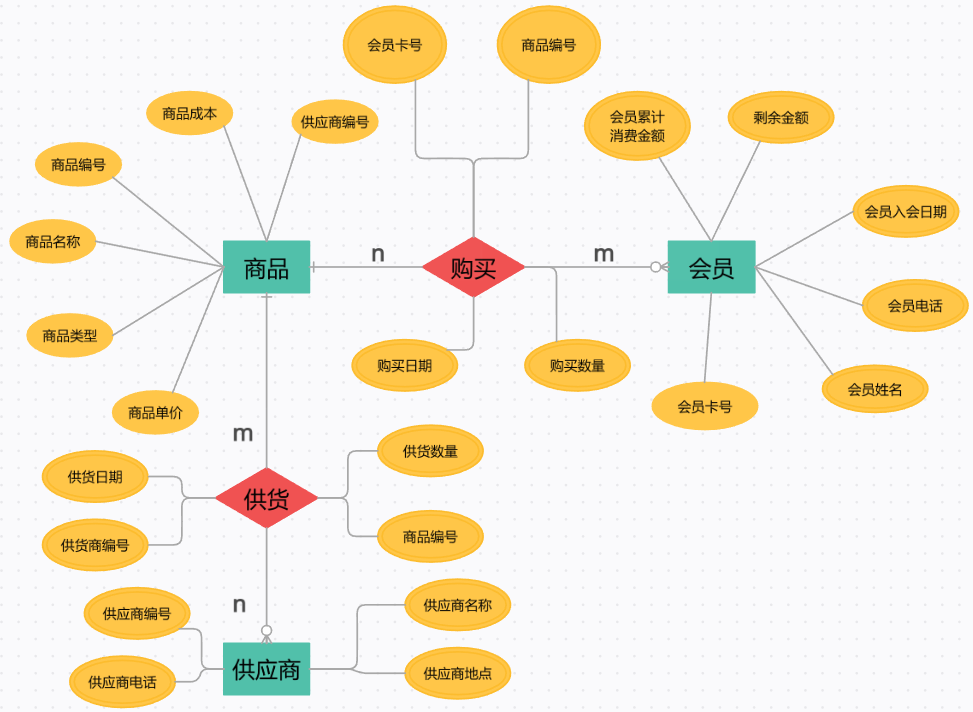
**界面需求：**界面简洁友好，色彩搭配舒适；操作流程引导清晰，适配电脑、平板、收银机等终端屏幕；提供操作手册与在线帮助。

四、概要设计

**（一）系统概述**

该系统是一个基于Python语言开发，使用Tkinter库构建图形用户界面，并通过pymssql库与SQL Server数据库（示例中配置为本地服务）进行交互的超市管理应用程序，主要涵盖了商品管理、会员管理以及用户登录注册等功能，旨在方便超市管理人员对商品和会员信息进行操作与维护。

系统的概念结构设计如下图：



**（二）功能模块设计**

**2.1、 登录注册模块**

**登录功能：**提供用户名和密码输入框，用户输入后点击“登录”按钮，系统连接数据库中adminUsers表进行验证，若匹配成功则关闭登录界面进入主管理界面mainui，否则弹出警告提示用户名或密码错误。

**注册功能：**在注册界面，用户输入用户名和密码（有格式长度限制），点击“注册”按钮，系统将信息插入adminUsers表，成功后提示注册成功并返回登录界面，若输入不符合要求则给出警告提示重新输入。

**2.2、商品管理模块**

**查看商品信息功能：**通过查询数据库Goods表以及关联Vendor表获取商品相关信息（编号、名称、类别、单价、成本、供应商等），以树形表格（Treeview）形式展示在界面上，支持按不同条件（如商品编号、名称、类别等）筛选查询商品信息，查询结果同样展示在表格内。

**添加商品信息功能：**弹出添加商品信息的表单界面，用户输入商品编号、名称、类别、单价、成本、供应商等详细信息后，点击“添加”按钮将数据插入Goods表完成添加操作。

**修改商品信息功能：**在商品列表中选中某一商品后，可进入修改界面，修改对应商品的各项信息，点击“修改”按钮，根据选中商品的编号更新数据库中对应商品记录。

**删除商品信息功能：**在商品列表选中商品后，点击“删除”按钮，依据选中商品的编号从Goods表中删除该商品记录。

**2.3、 会员管理模块**

**查看会员信息功能：**查询数据库Menber表，将会员卡号、姓名、电话、注册日期、充值金额等信息展示在树形表格中方便查看。

**添加会员信息功能：**弹出会员注册表单界面，用户输入会员卡号、姓名、电话、注册日期、会员密码、充值金额等信息后，点击“添加”按钮向Menber表插入新会员记录。

**删除会员信息功能：**从会员列表中选中要删除的会员，点击“删除”按钮，按照选中会员的卡号从Menber表中删除对应会员记录。

**（三）模块间关系**

登录注册模块作为系统入口，成功登录后进入主界面，主界面提供商品管理模块和会员管理模块的入口按钮，各管理模块间相对独立，通过与数据库的交互实现对不同数据表（adminUsers、Goods、Menber等）的增删改查操作，以完成对应的业务功能，各模块操作结果可实时反馈在相应的界面展示组件上（如树形表格、提示消息框等）。

**（四）数据存储设计**

系统使用SQL Server数据库，创建了至少三张表：

**4.1、 adminUsers表：**用于存储系统管理员用户的登录信息，包含字段如username（用户名）、userpassword（密码）等，用于登录验证。

|  |  |  |  |
| --- | --- | --- | --- |
| 列名 | 含义 | 数据类型 | 说明 |
| username | 用户名 | Varchar(10) | 主键 |
| userpassword | 密码 | Varchar(20) |  |

**4.2、Goods表：**存储商品相关信息，有Gnum（商品编号）、Gname（商品名称）、Gtype（商品类别）、Gprice（商品单价）、Gbid（商品成本）、Vnum（供应商编号）等字段，支持商品管理模块对商品数据的操作。

|  |  |  |  |
| --- | --- | --- | --- |
| 列名 | 含义 | 数据类型 | 说明 |
| Gnum | 商品编号 | varchar(10) | 主键 |
| Gname | 商品名称 | varchar(10) |  |
| Gtype | 商品类型 | varchar(10) |  |
| Gprice | 商品单价 | money (2) | Gprice>=0 |
| Gbid | 商品成本 | money | Gbid>=0 |
| Vnum | 供应商编号 | Varchar(10) | 是表Vendor的外键 |

**4.3、Menber表：**存放会员信息，涵盖Mnum（会员卡号）、Mname（会员姓名）、Mphone（会员电话）、Mdate（注册日期）、Mcip（会员密码）、Mbalance（充值金额）等字段，为会员管理模块提供数据支持。

|  |  |  |  |
| --- | --- | --- | --- |
| 列名 | 含义 | 数据类型 | 说明 |
| Mnum | 会员卡号 | Varchar(10) | 主键 |
| Mname | 会员姓名 | Varchar(10) |  |
| Mphone | 会员电话 | Varchar(20) |  |
| Mdate | 会员入会日期 | data |  |
| Mbalance | 剩余金额 | money | Mbalance>=0 |
| Mcip | 会员密码 | Varchar |  |

**（五）界面交互设计**

整个系统基于Tkinter构建图形用户界面，采用多窗口、多按钮以及表格展示等交互形式：

**5.1、登录注册界面：**有清晰的用户名、密码输入框，登录、注册、返回等按钮布局合理，方便用户操作以及切换界面。

**5.2、主管理界面：**划分不同功能区域，通过点击不同功能按钮（如查看、添加、修改、删除商品/会员信息按钮）弹出对应操作的子界面，各子界面按操作逻辑合理布局表单元素、操作按钮等，操作完成后可返回主界面或者进行其他相关操作，同时利用树形表格展示查询结果，方便用户直观查看和选择操作对象。

**（六）系统运行流程**

**6.1、** 系统启动先进入登录界面，用户可选择登录（输入正确用户名和密码进入主管理界面）或注册（进入注册界面填写信息注册新账号后返回登录界面）。

**6.2、**成功登录后进入主管理界面，在此界面可按需点击商品管理或会员管理相关按钮，进入对应功能模块执行具体的增删改查操作，操作过程中会与数据库交互，操作完成后根据界面提示或返回主界面继续其他操作，或留在当前功能模块进行进一步操作。

**五、详细设计**

**（一）**登录注册模块详细设计

**1.1、登录功能**

界面设计：使用 Tk() 创建登录窗口 loginwin，设置窗口大小为 300x200，不可改变大小。在窗口内依次放置 llabel1（“用户名:”标签）、luser（用户名输入框）、llabel2（“密码:”标签）、lpassword（密码输入框，设置 show="\*"隐藏输入内容）、loginbtn（“登录”按钮）和 resgisterbt（“注册”按钮），通过 place() 方法精确控制各元素位置，布局简洁清晰，方便用户操作。

业务逻辑：点击“登录”按钮触发 login 函数，函数内首先通过 pymssql.connect() 建立与数据库的连接，并创建游标 cursor。执行 select \* from adminUsers 查询语句获取 adminUsers表中的所有用户记录，通过循环遍历查询结果cnt，将输入的用户名 luser.get() 和密码 lpassword.get()与数据库中每条记录的对应字段进行比对。若匹配成功，则关闭数据库连接，销毁登录窗口 loginwin，调用 mainui 函数进入主管理界面；若遍历完所有记录仍未匹配成功，则关闭数据库连接，并通过 messagebox.showwarning() 弹出警告框提示用户名或密码错误，请重新输入。

**2.2、 注册功能**

界面设计：registerui 函数用于创建注册窗口 registerwin，同样设置窗口大小为 300x200。 在窗口内布置 rlabel1（“用户名:”标签）、ruser（用户名输入框）、rlabel2（“密码:”标签）、rpassword（密码输入框，show="\*"）、resgisterbtn（“注册”按钮）和 backbtn（“返回”按钮），各元素按合适坐标位置放置，方便用户输入信息和操作按钮。

业务逻辑：点击“注册”按钮触发 register 函数，函数首先对输入的用户名和密码长度进行合法性校验（用户名长度需在 0 < len(ruser.get()) < 10，密码长度需在 0 < len(rpassword.get()) < 20 范围内）。

若输入合法，通过 pymssql.connect()连接数据库（此处字符集设为 UTF-8），创建游标后执行插入语句，将新用户信息插入 adminUsers表，提交事务后关闭数据库连接，再通过 messagebox.showinfo() 弹出提示框告知注册成功，请登录，最后销毁注册窗口 registerwin并调用login返回登录界面；若输入不合法，则通过 messagebox.showwarning()弹出警告框提示输入非法请重新输入。

**（二）商品管理模块详细设计**

**2.1、查看商品信息功能**

界面设计：在 selectgoods 函数中创建主框架 frame1，设置其在主窗口 mainwin 内的位置、大小及边框样式等属性，作为商品管理相关操作界面的容器。放置多个界面元素，包括 pselectgbtn（“查看商品信息”按钮，设置为凹陷样式表示当前所在功能）、addgbtn（“添加商品信息”按钮）、addmbtn（“添加会员信息”按钮）、selectmbtn（“查看会员信息”按钮）用于切换不同功能模块；添加 ttk.Combobox（下拉选择框 combobox）用于选择查询筛选条件（有‘查看全部’、‘商品编号’、‘商品名称’等多个选项），配合 lable1（“查询筛选：”标签）、lable2（“筛选条件：”标签）、selentry（输入筛选条件的文本框）以及 btn1（“查询”按钮）实现按条件查询商品功能；还有 btn2（“删除”按钮）、btn3（“修改”按钮）用于执行对应操作，以及最重要的 ttk.Treeview树形表格 tree）用于展示商品信息，通过设置表格的列名、列宽、表头显示文本以及绑定 TreeviewSelect事件（触发 gchangeid函数获取选中商品编号）来实现良好的数据展示和交互效果。

业务逻辑：首先通过 pymssql.connect() 连接数据库，执行查询语句分别获取商品基础信息和供应商相关信息，对获取的商品信息数据cnt1 进行处理（将元组转换为列表方便后续操作），并通过循环比对将商品记录中的供应商编号替换为实际供应商名称。

点击“查询”按钮 btn1 触发 seltgood 函数，根据 combobox 选择的筛选条件以及 selentry 输入的内容构建不同的查询语句，再次连接数据库执行查询语句获取筛选后的商品信息并更新树形表格展示内容。树形表格展示商品信息时，对商品单价和成本字段进行格式化输出（保留两位小数），并且支持排序展示（通过 cnt1.sort(key=takeFirst) 按商品编号排序），方便用户查看。

**2.2、添加商品信息功能**

界面设计： addgoods函数创建 frame1 框架作为添加商品信息的操作界面，在框架内依次放置 selectgbtn（“查看商品信息”按钮）、paddgbtn（“添加商品信息”按钮，设置为凹陷样式表示当前功能）、addmbtn（“添加会员信息”按钮）、selectmbtn（“查看会员信息”按钮）用于界面导航；添加多个 Label（如 idlabel、namelabel 等）用于提示输入信息的字段名，以及对应的 Entry（如 idEntry、nameEntry 等）文本输入框用于输入商品编号、名称、类别、单价、成本、供应商等详细信息，最后放置 addbtn（“添加”按钮）用于触发添加操作。

业务逻辑：点击“添加”按钮 addbtn 触发addg函数，函数内通过 pymssql.connect() 连接数据库，根据输入框获取的用户输入信息构建插入语句，将新商品信息插入 Goods表，提交事务后关闭数据库连接完成添加操作。

**2.3、 修改商品信息功能**

界面设计：modgood函数在用户选中商品（通过判断tree.set(tree.focus())是否有值，若没有则弹出警告提示选择商品并返回查看商品信息界面）后创建修改商品信息的界面 frame1，在框架内放置 namelabel（“商品名称：”标签）、typelabel（“商品类别：”标签）、pricelabel（“商品单价：”标签）、bidlabel（“商品成本：”标签）、vendorlabel（“供 应 商 ：”标签）以及对应的nameEntry、typeEntry、priceEntry、bidEntry、vendorEntry 文本输入框用于显示和修改商品各字段信息，同时添加 btn1（“修改”按钮）用于保存修改操作，btn2（“返回”按钮）用于返回查看商品信息界面。

业务逻辑：点击“修改”按钮btn1触发modify函数，函数内先连接数据库查询 Vendor 表获取供应商相关信息，比对输入的供应商名称获取对应的供应商编号stats，然后根据输入框中的商品信息以及获取的供应商编号构建更新语句，更新数据库中对应商品记录，提交事务后关闭数据库连接，通过 messagebox.showinfo()弹出提示框告知修改成功，最后调用 selectgoods函数刷新商品信息展示界面。

**2.4、 删除商品信息功能**

界面设计：与其他操作共用 selectgoods界面中的“删除”按钮btn2，按钮设置合适的样式及位置，方便用户操作。

业务逻辑：点击“删除”按钮触发delgood函数，函数首先判断是否有选中商品（通过 tree.set(tree.focus())判断），若没有则弹出警告提示选择商品并返回查看商品信息界面；若有选中商品，根据获取的商品编号 id构建删除语句，连接数据库执行该语句从 Goods表中删除对应商品记录，提交事务后关闭数据库连接，最后调用 selectgoods 函数刷新商品信息展示界面，确保展示数据的实时性。

**（三）会员管理模块详细设计**

**3.1、 查看会员信息功能**

界面设计：在selectmenber函数中创建 frame1 框架作为会员管理操作界面容器，在主窗口 mainwin内设置其合适的位置、大小及边框样式等属性。在框架内放置 selectgbtn（“查看商品信息”按钮）、addgbtn（“添加商品信息”按钮）、addmbtn（“添加会员信息”按钮）、pselectmbtn（“查看会员信息”按钮，设置为凹陷样式表示当前功能）以及 btn（“删除”按钮）用于不同功能操作切换；重点是创建 ttk.Treeview（树形表格 tree2）用于展示会员信息，设置表格的列名（“会员卡号”、“会员姓名”、“会员电话”、“注册日期”、“充值金额”）、列宽、表头显示文本以及绑定TreeviewSelect事件（触发 gchangeid2 函数获取选中会员卡号），方便用户查看和选择会员记录。

业务逻辑：通过 pymssql.connect()连接数据库，执行 select \* from Menber查询语句获取会员信息，将查询结果 cnt1 通过循环插入树形表格 tree2 进行展示，展示内容直观清晰，方便用户查看会员详细信息。

**3.2、 添加会员信息功能**

界面设计：addmenber 函数创建 frame1框架作为添加会员信息的操作界面，在框架内放置多个界面元素，包括用于导航的功能按钮（如 selectgbtn、addgbtn、selectmbtn、paddmbtn），还有多个 Label用于提示输入会员卡号、姓名、电话、注册日期、会员密码、充值金额等各字段信息，对应的 Entry（如e1、e2等）文本输入框用于用户输入具体内容，最后放置 B1（“添加”按钮）用于触发添加操作，B2（“返回”按钮）用于返回上一级界面。

业务逻辑：点击“添加”按钮B1触发add\_huiyuan函数，函数内先通过 pymssql.connect()连接数据库，根据输入框获取的用户输入信息构建插入语句将新会员信息插入Menber表，提交事务后关闭数据库连接，通过 tk.messagebox.showinfo()弹出提示框告知数据添加成功，完成添加操作。

**3.3、删除会员信息功能**

界面设计：与查看会员信息共用 selectmenber界面中的“删除”按钮 btn，按钮设置合理的样式及位置，便于操作。

业务逻辑：点击“删除”按钮触发 delmenber 函数，函数首先判断是否有选中会员（通过 tree2.set(tree2.focus()) 判断），若没有则弹出警告提示选择会员并重新执行查看会员信息操作；若有选中会员，根据获取的会员卡号 id2 构建删除语句，连接数据库执行该语句从Menber表中删除对应会员记录，提交事务后关闭数据库连接，最后调用 selectmenber函数刷新会员信息展示界面，保证数据展示的及时性和准确性。

**（四）数据库设计详细说明**

**4.1、 adminUsers表：**包含 username（VARCHAR 类型，用于存储用户名，长度根据实际需求设定，设置为主键，保证用户名唯一性）、userpassword（VARCHAR 类型，存储用户密码，长度满足密码安全要求）两个主要字段，用于系统用户登录验证相关操作。

数据操作说明：登录功能中通过查询该表验证用户输入的用户名和密码是否匹配；注册功能则向该表插入新用户的用户名和密码信息，插入前需对输入的合法性进行校验，确保数据的完整性和正确性。

**4.2、 Goods表**：有 Gnum（INT 类型，商品编号，作为主键，一般设置为自增，用于唯一标识商品）、Gname（VARCHAR 类型，商品名称，长度根据实际商品名称长度设定）、Gtype（VARCHAR 类型，商品类别）、Gprice（DECIMAL 类型，商品单价，设置合适的精度，如保留两位小数）、Gbid（DECIMAL 类型，商品成本，同样保留合适精度）、Vnum（INT`类型，可能关联供应商相关编号，与 Vendor表存在关联关系）等字段，用于存储商品全方位的基础信息，支持商品管理模块中各类对商品数据的增删改查操作。

数据操作说明：查看商品信息时，通过多表关联查询（与 Vendor 表关联获取供应商名称）获取商品详细信息展示给用户；添加商品信息功能根据用户输入构建插入语句向该表插入新商品记录；修改商品信息依据用户修改后的内容更新对应商品记录；删除商品信息则根据商品编号删除相应记录，每次操作后确保数据库中商品数据的一致性和准确性。

**4.3、 Menber表：**包含 Mnum（INT 类型，会员卡号，作为主键，可唯一标识会员）、Mname（VARCHAR类型，会员姓名）、Mphone（VARCHAR 类型，会员电话）、Mdate（DATE 类型，注册日期）、Mcip（VARCHAR 类型）、Mbalance（INT 类型，充值金额）等字段，用于全面记录会员相关信息，满足会员管理模块的业务需求。

数据操作说明：查看会员信息时查询该表获取所有会员记录展示在界面；添加会员信息功能按照用户输入向表中插入新会员数据；删除会员信息则依据会员卡号删除对应的会员记录，操作过程中同样注重数据的完整性和实时性维护，确保会员信息管理的准确有序。

**（五）系统整体流程与交互详细说明**

**5.1、启动与登录注册流程：**

系统启动运行 loginui 函数展示登录界面，用户在此界面可选择输入用户名和密码点击“登录”按钮尝试登录系统，若选择“注册”按钮则进入注册界面 registerui 进行新用户注册操作，注册成功后返回登录界面继续登录。成功登录后通过 mainui`函数进入主管理界面开展后续商品管理和会员管理相关操作。

**5.2、 主管理界面交互流程**：

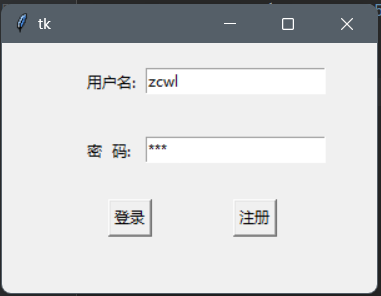
在主管理界面mainwin中，通过各个功能按钮（“查看商品信息”、“添加商品信息”、“查看会员信息”、“添加会员信息”等）清晰划分不同功能模块入口，点击相应按钮弹出对应的操作子界面，用户在子

**六、系统实现**

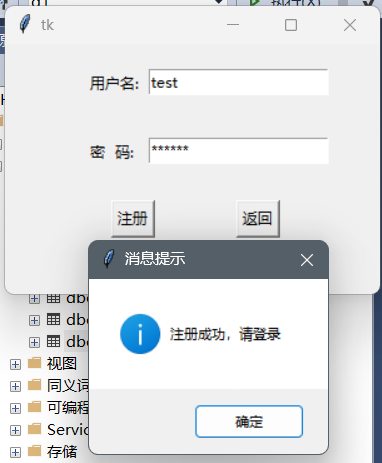
**（一）系统界面设计并测试系统功能**

1.超简洁的登录界面：输入账号和密码，点击回车键或点击“登录”按钮，系统会将此账号和密码与数据库数据进行比对，如果存在，则会进入主界面，否则会提示“账号或密码错误！”。

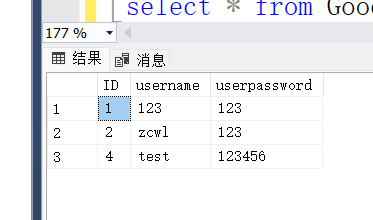
备注：登录名：zcwl 密码：123



同时可以进行注册账号，注册之后会在数据库中保存。



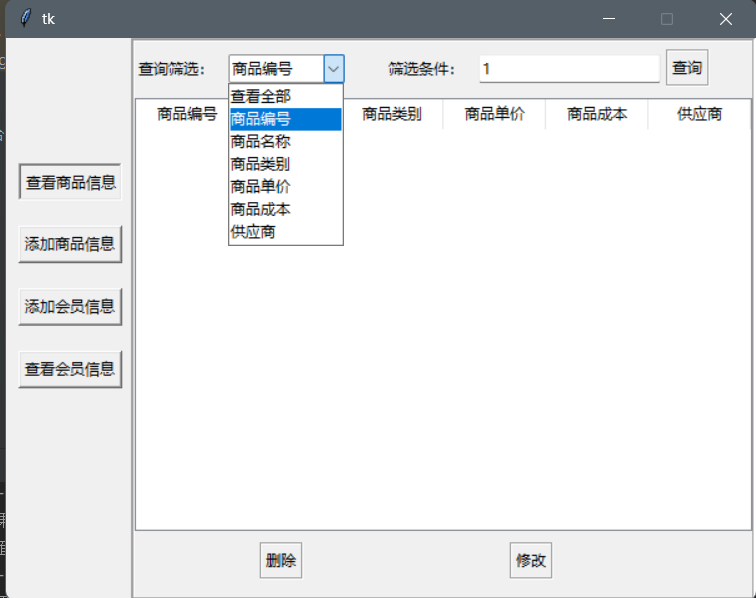
对比数据库，注册账号已保存。



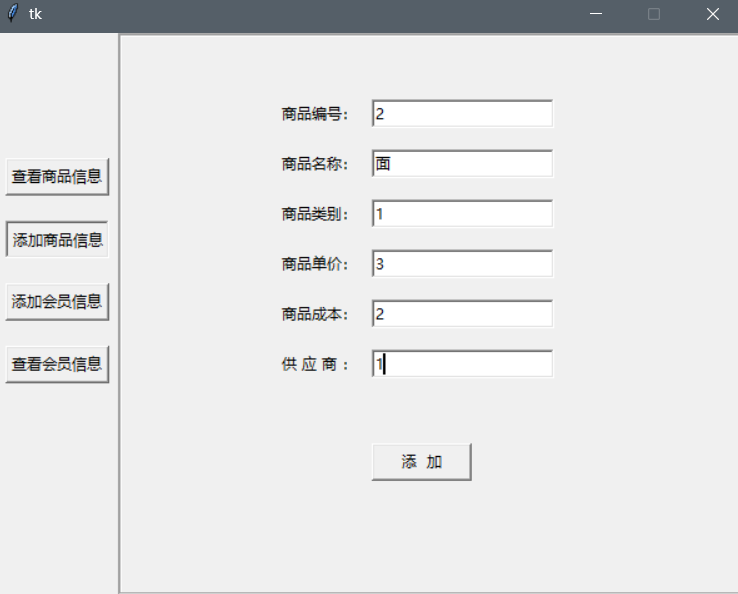
2.主界面：为便于超市管理的工作，界面设置简单明了，同时兼具查看商品信息，添加商品信息，添加会员信息，查看会员信息等功能。



3.主界面支持对商品和会员信息进行增加、删除、修改、查询功能，其中查询功能可以按商品编号、商品名称、商品类别或商品单价等信息查询，均支持模糊查询。

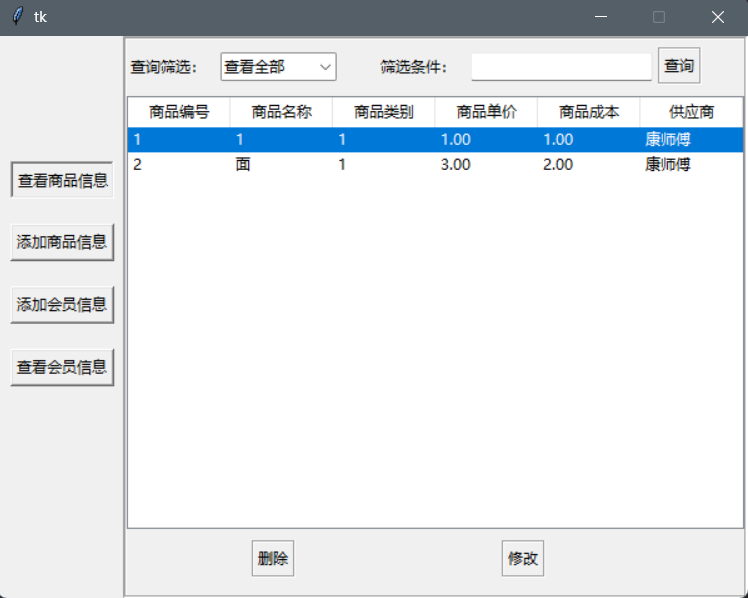


4.商品信息添加功能展示，供应商中1表示“康师傅”，商品编号不能与之前的商品编号重复。



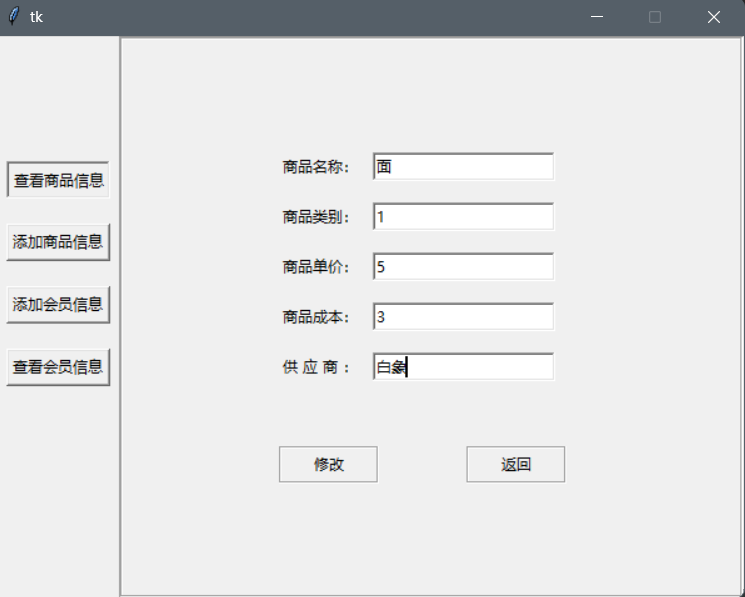


5. 商品信息删除功能展示，选中所要删除的那一栏信息，点击删除即可。





6.商品信息修改功能展示，选中要修改的那一栏，点击修改，然后填写除了商品编号的其他信息。





7.商品信息查询功能展示，可以查看全部商品信息，也可以按照某一项信息进行查询。

按供应商查询



按商品类别查询

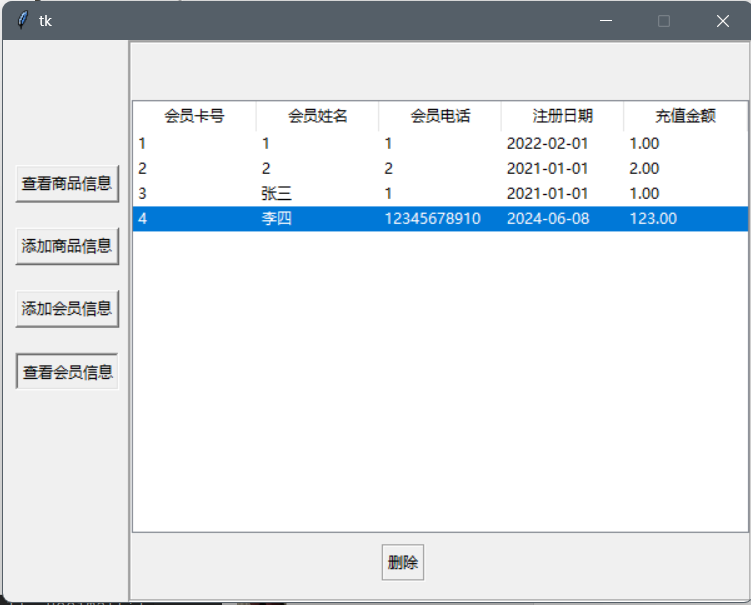


8.会员信息添加功能展示，注册日期要使用标准格式“xxxx-xx-xx”，卡号不能与之前注册的卡号信息相同。





9.会员信息删除功能展示，选中所要删除的那一栏信息，然后点击删除。





**界面设计Python代码：**

import pymssql  
import tkinter as tk  
from tkinter import messagebox  
from tkinter import ttk  
  
  
# 主函数  
def mainui():  
 global mainwin  
 mainwin = tk.Tk()  
 mainwin.geometry("600x450")  
 mainwin.resizable(0, 0)  
 selectgoods()  
 mainwin.mainloop()  
  
  
def gchangeid2(e):  
 global id2  
 itm = tree2.set(tree2.focus())  
 id2 = itm['会员卡号']  
 print(id2)  
  
  
def delmenber():  
 if not tree2.set(tree2.focus()):  
 msg = messagebox.showwarning(title="消息警告", message="请选择一位会员")  
 print(msg)  
 selectmenber()  
 return  
 sql = "delete from Menber where Mnum='%s'" % id2  
 con = pymssql.connect(host="127.0.0.1", port="1433", database="d1", user="sa", password="123", charset="cp936")  
 cursor = con.cursor()  
 cursor.execute(sql)  
 con.commit()  
 con.close()  
 selectmenber()  
  
  
def selectmenber():  
 global tree2  
 frame1 = tk.Frame(mainwin, bd=3, width=500, height=450, relief="groove")  
 selectgbtn = tk.Button(mainwin, text="查看商品信息", command=selectgoods)  
 addgbtn = tk.Button(mainwin, text="添加商品信息", command=addgoods)  
 addmbtn = tk.Button(mainwin, text="添加会员信息", command=addmenber)  
 pselectmbtn = tk.Button(mainwin, text="查看会员信息", relief="sunken")  
 btn = tk.Button(frame1, text='删除', command=delmenber, relief="groove")  
 sql1 = "select \* from Menber"  
 con = pymssql.connect(host="127.0.0.1", port="1433", database="d1", user="sa", password="123", charset="cp936")  
 cursor = con.cursor()  
 cursor.execute(sql1)  
 cnt1 = cursor.fetchall()  
 print(cnt1)  
 tree2 = ttk.Treeview(frame1, height=16)  
 tree2["selectmode"] = 'browse'  
 tree2["show"] = 'headings'  
 tree2["columns"] = ("会员卡号", "会员姓名", "会员电话", "注册日期", "充值金额")  
 tree2.column("会员卡号", width=99)  
 tree2.column("会员姓名", width=98)  
 tree2.column("会员电话", width=98)  
 tree2.column("注册日期", width=98)  
 tree2.column("充值金额", width=99)  
 tree2.heading("会员卡号", text="会员卡号")  
 tree2.heading("会员姓名", text="会员姓名")  
 tree2.heading("会员电话", text="会员电话")  
 tree2.heading("注册日期", text="注册日期")  
 tree2.heading("充值金额", text="充值金额")  
 tree2.bind("<<TreeviewSelect>>", gchangeid2)  
 index = 0  
 for item in cnt1:  
 tree2.insert("", index, values=(item[0], item[1], item[2], item[3], item[5]))  
 index = index + 1  
 tree2.place(x=0, y=45)  
 selectgbtn.place(x=10, y=100)  
 addgbtn.place(x=10, y=150)  
 addmbtn.place(x=10, y=200)  
 pselectmbtn.place(x=10, y=250)  
 btn.place(x=200, y=400)  
 frame1.place(x=100, y=0)  
  
  
def add\_huiyuan():  
 # 连接数据库  
 connect = pymssql.connect(server="localhost:1433", database='d1', user='sa', password='123', charset='utf8')  
 cursor = connect.cursor()  
 if cursor:  
 print("连接成功!")  
 # 创建光标  
 cursor = connect.cursor()  
 # 编写SQL语句  
 sql = "insert into Menber(Mnum,Mname,Mphone,Mdate,Mcip,Mbalance) values('%s','%s','%s','%s','%s','%s')" % (  
 e1.get(),  
 e2.get(),  
 e3.get(),  
 e4.get(),  
 e5.get(),  
 e6.get())  
 # 执行SQL语句，并且输出完成提示信息，否则回滚  
 cursor.execute(sql)  
 connect.commit()  
 tk.messagebox.showinfo("提示", "数据添加成功！")  
 # 关闭数据库连接，防止泄露  
 connect.close()  
  
  
# 增加会员选项  
def addmenber():  
 global e1, e2, e3, e4, e5, e6  
 selectgbtn = tk.Button(mainwin, text="查看商品信息", command=selectgoods)  
 addgbtn = tk.Button(mainwin, text="添加商品信息", command=addgoods)  
 selectmbtn = tk.Button(mainwin, text="添加会员信息", relief="sunken")  
 paddmbtn = tk.Button(mainwin, text="查看会员信息", command=selectmenber)  
 frame1 = tk.Frame(mainwin, bd=3, width=500, height=450, relief="groove")  
 l0 = tk.Label(frame1, text="会员注册系统", font=('Arial', 22), width=10, height=1)  
 l0.place(x=150, y=25)  
 l1 = tk.Label(frame1, text="会员卡号", font=('Arial', 12), width=8, height=1)  
 l1.place(x=50, y=100)  
 l2 = tk.Label(frame1, text="会员姓名", font=('Arial', 12), width=8, height=1)  
 l2.place(x=50, y=150)  
 l3 = tk.Label(frame1, text="会员电话", font=('Arial', 12), width=8, height=1)  
 l3.place(x=50, y=200)  
 l4 = tk.Label(frame1, text="注册日期", font=('Arial', 12), width=8, height=1)  
 l4.place(x=50, y=250)  
 l5 = tk.Label(frame1, text="会员密码", font=('Arial', 12), width=8, height=1)  
 l5.place(x=50, y=300)  
 l6 = tk.Label(frame1, text="充值金额", font=('Arial', 12), width=8, height=1)  
 l6.place(x=50, y=350)  
 e1 = tk.Entry(frame1, show=None, width=40)  
 e1.place(x=150, y=100)  
 e2 = tk.Entry(frame1, show=None, width=40)  
 e2.place(x=150, y=150)  
 e3 = tk.Entry(frame1, show=None, width=40)  
 e3.place(x=150, y=200)  
 e4 = tk.Entry(frame1, show=None, width=40)  
 e4.place(x=150, y=250)  
 e5 = tk.Entry(frame1, show=None, width=40)  
 e5.place(x=150, y=300)  
 e6 = tk.Entry(frame1, show=None, width=40)  
 e6.place(x=150, y=350)  
 # B1是添加按钮  
 B1 = tk.Button(frame1, text="添加", font=('Arial', 12), width=8, height=1, command=add\_huiyuan)  
 B1.place(x=100, y=400)  
 # B2是返回按钮  
 B2 = tk.Button(frame1, text="返回", font=('Arial', 12), width=8, height=1)  
 B2.place(x=270, y=400)  
 # 窗口刷新  
 frame1.place(x=100, y=0)  
 selectgbtn.place(x=10, y=100)  
 addgbtn.place(x=10, y=150)  
 selectmbtn.place(x=10, y=200)  
 paddmbtn.place(x=10, y=250)  
  
  
# 添加商品操作  
def addg():  
 con = pymssql.connect(host="127.0.0.1", port="1433", database="d1", user="sa", password="123")  
 cursor = con.cursor()  
 sql = "insert into Goods (Gnum,Gname,Gtype,Gprice,Gbid,Vnum) values (%s,'%s','%s',%s,%s,'%s')" % \  
 (idEntry.get(), nameEntry.get(), typeEntry.get(), priceEntry.get(), bidEntry.get(), vendorEntry.get())  
 cursor.execute(sql)  
 con.commit()  
 con.close()  
  
  
# 增加商品选项  
def addgoods():  
 global idEntry  
 global nameEntry  
 global typeEntry  
 global priceEntry  
 global bidEntry  
 global vendorEntry  
 selectgbtn = tk.Button(mainwin, text="查看商品信息", command=selectgoods)  
 paddgbtn = tk.Button(mainwin, text="添加商品信息", relief="sunken", )  
 addmbtn = tk.Button(mainwin, text="添加会员信息", command=addmenber)  
 selectmbtn = tk.Button(mainwin, text="查看会员信息", command=selectmenber)  
 frame1 = tk.Frame(mainwin, bd=3, width=500, height=450, relief="groove")  
 idlabel = tk.Label(frame1, text="商品编号：")  
 namelabel = tk.Label(frame1, text="商品名称：")  
 typelabel = tk.Label(frame1, text="商品类别：")  
 pricelabel = tk.Label(frame1, text="商品单价：")  
 bidlabel = tk.Label(frame1, text="商品成本：")  
 vendorlabel = tk.Label(frame1, text="供 应 商 ：")  
 idEntry = tk.Entry(frame1, bd=2)  
 nameEntry = tk.Entry(frame1, bd=2)  
 typeEntry = tk.Entry(frame1, bd=2)  
 priceEntry = tk.Entry(frame1, bd=2)  
 bidEntry = tk.Entry(frame1, bd=2)  
 vendorEntry = tk.Entry(frame1, bd=2)  
 addbtn = tk.Button(frame1, text="添 加", width="10", command=addg)  
 selectgbtn.place(x=10, y=100)  
 paddgbtn.place(x=10, y=150)  
 addmbtn.place(x=10, y=200)  
 selectmbtn.place(x=10, y=250)  
 idlabel.place(x=125, y=50)  
 namelabel.place(x=125, y=90)  
 typelabel.place(x=125, y=130)  
 pricelabel.place(x=125, y=170)  
 bidlabel.place(x=125, y=210)  
 vendorlabel.place(x=125, y=250)  
 idEntry.place(x=200, y=50)  
 nameEntry.place(x=200, y=90)  
 typeEntry.place(x=200, y=130)  
 priceEntry.place(x=200, y=170)  
 bidEntry.place(x=200, y=210)  
 vendorEntry.place(x=200, y=250)  
 addbtn.place(x=200, y=325)  
 frame1.place(x=100, y=0)  
  
  
# 获取选择的编号  
def gchangeid(e):  
 global id  
 itm = tree.set(tree.focus())  
 id = itm['商品编号']  
 print(id)  
  
  
# 修改操作  
def modify():  
 con = pymssql.connect(host="127.0.0.1", port="1433", database="d1", user="sa", password="123", charset="cp936")  
 cursor = con.cursor()  
 cursor.execute("select \* from Vendor")  
 temp = cursor.fetchall()  
 stats = ''  
 for item in temp:  
 if item[1] == vendorEntry.get():  
 stats = item[0]  
 sql = "update Goods set Gname = '%s',Gtype = '%s', Gprice = %s,Gbid = %s, Vnum = %s where Gnum = %s " % \  
 (nameEntry.get(), typeEntry.get(), priceEntry.get(), bidEntry.get(), stats, id)  
 cursor = con.cursor()  
 cursor.execute(sql)  
 con.commit()  
 con.close()  
 msg = messagebox.showinfo(title="消息提示", message="修改成功")  
 print(msg)  
 selectgoods()  
  
  
# 修改商品界面  
def modgood():  
 if not tree.set(tree.focus()):  
 msg = messagebox.showwarning(title="消息警告", message="请选择一项商品")  
 print(msg)  
 selectgoods()  
 return  
 global nameEntry  
 global typeEntry  
 global priceEntry  
 global bidEntry  
 global vendorEntry  
 print(tree.set(tree.focus()))  
 frame1 = tk.Frame(mainwin, bd=3, width=500, height=450, relief="groove")  
 namelabel = tk.Label(frame1, text="商品名称：")  
 typelabel = tk.Label(frame1, text="商品类别：")  
 pricelabel = tk.Label(frame1, text="商品单价：")  
 bidlabel = tk.Label(frame1, text="商品成本：")  
 vendorlabel = tk.Label(frame1, text="供 应 商 ：")  
 nameEntry = tk.Entry(frame1, bd=2)  
 typeEntry = tk.Entry(frame1, bd=2)  
 priceEntry = tk.Entry(frame1, bd=2)  
 bidEntry = tk.Entry(frame1, bd=2)  
 vendorEntry = tk.Entry(frame1, bd=2)  
 btn1 = tk.Button(frame1, text="修改", width="10", relief='groove', command=modify)  
 btn2 = tk.Button(frame1, text="返回", width="10", relief='groove', command=selectgoods)  
 namelabel.place(x=125, y=90)  
 typelabel.place(x=125, y=130)  
 pricelabel.place(x=125, y=170)  
 bidlabel.place(x=125, y=210)  
 vendorlabel.place(x=125, y=250)  
 nameEntry.place(x=200, y=90)  
 typeEntry.place(x=200, y=130)  
 priceEntry.place(x=200, y=170)  
 bidEntry.place(x=200, y=210)  
 vendorEntry.place(x=200, y=250)  
 btn1.place(x=125, y=325)  
 btn2.place(x=275, y=325)  
 frame1.place(x=100, y=0)  
  
  
# 删除商品  
def delgood():  
 if not tree.set(tree.focus()):  
 msg = messagebox.showwarning(title="消息警告", message="请选择一项商品")  
 print(msg)  
 selectgoods()  
 return  
 sql = "delete from Goods where Gnum='%s'" % id  
 con = pymssql.connect(host="127.0.0.1", port="1433", database="d1", user="sa", password="123", charset="cp936")  
 cursor = con.cursor()  
 cursor.execute(sql)  
 con.commit()  
 con.close()  
 selectgoods()  
  
  
# 按条件查询  
def seltgood():  
 stats = selentry.get()  
 if combobox.get() == '商品编号': # '查看全部', '商品编号', '商品名称', '商品类别', '商品单价', '商品成本', '供应商'  
 sql1 = "select \* from Goods where Gnum = %s" % stats  
 sql2 = "select \* from Vendor"  
 else:  
 if combobox.get() == '商品名称':  
 sql1 = "select \* from Goods where Gname = %s" % stats  
 sql2 = "select \* from Vendor"  
 else:  
 if combobox.get() == '商品类别':  
 sql1 = "select \* from Goods where Gtype = %s" % stats  
 sql2 = "select \* from Vendor"  
 else:  
 if combobox.get() == '商品单价':  
 sql1 = "select \* from Goods where Gprice = %s" % stats  
 sql2 = "select \* from Vendor"  
 else:  
 if combobox.get() == '商品成本':  
 sql1 = "select \* from Goods where Gbid = %s" % stats  
 sql2 = "select \* from Vendor"  
 else:  
 if combobox.get() == '供应商':  
 con = pymssql.connect(host="127.0.0.1", port="1433", database="d1", user="sa",  
 password="123", charset="cp936")  
 cursor = con.cursor()  
 cursor.execute("select \* from Vendor")  
 temp = cursor.fetchall()  
 for item in temp:  
 if item[1] == stats:  
 stats = item[0]  
 con.close()  
 sql1 = "select \* from Goods where Vnum = %s" % stats  
 sql2 = "select \* from Vendor"  
 else:  
 sql1 = "select \* from Goods"  
 sql2 = "select \* from Vendor"  
 con = pymssql.connect(host="127.0.0.1", port="1433", database="d1", user="sa", password="123", charset="cp936")  
 cursor = con.cursor()  
 cursor.execute(sql1)  
 cnt1 = cursor.fetchall()  
 cursor.execute(sql2)  
 cnt2 = cursor.fetchall()  
 con.close()  
 cnt1 = list(cnt1)  
 index = 0  
 for item in cnt1:  
 cnt1[index] = list(item)  
 index = index + 1  
 for item1 in cnt1:  
 for item2 in cnt2:  
 if item1[5] == item2[0]:  
 item1[5] = item2[1]  
 break  
 print("商品查询结果：")  
 print(cnt1)  
 print("--------------")  
  
 for child in tree.get\_children():  
 tree.delete(child)  
 index = 0  
 for item in cnt1:  
 tree.insert("", index, values=(item[0], item[1],  
 item[2], format(item[3], '.2f'), format(item[4], '.2f'), item[5]))  
 index = index + 1  
  
  
def takeFirst(elem):  
 return int(elem[0])  
  
  
# 查询修改删除商品选项  
def selectgoods():  
 global frame1  
 global tree  
 global combobox  
 global selentry  
 pselectgbtn = tk.Button(mainwin, text="查看商品信息", relief="sunken")  
 addgbtn = tk.Button(mainwin, text="添加商品信息", command=addgoods)  
 addmbtn = tk.Button(mainwin, text="添加会员信息", command=addmenber)  
 selectmbtn = tk.Button(mainwin, text="查看会员信息", command=selectmenber)  
 frame1 = tk.Frame(mainwin, bd=3, width=500, height=450, relief="groove")  
 combobox = ttk.Combobox(frame1,  
 values=['查看全部', '商品编号', '商品名称', '商品类别', '商品单价', '商品成本', '供应商'],  
 width=10)  
 lable1 = tk.Label(frame1, text='查询筛选：')  
 lable2 = tk.Label(frame1, text='筛选条件：')  
 btn1 = tk.Button(frame1, text='查询', command=seltgood, relief='groove')  
 selentry = ttk.Entry(frame1)  
 btn2 = tk.Button(frame1, text='删除', command=delgood, relief="groove")  
 btn3 = tk.Button(frame1, text='修改', command=modgood, relief="groove")  
 combobox.current(0)  
 sql1 = "select \* from Goods"  
 sql2 = "select \* from Vendor"  
 con = pymssql.connect(host="127.0.0.1", port="1433", database="d1", user="sa", password="123", charset="cp936")  
 cursor = con.cursor()  
 cursor.execute(sql1)  
 cnt1 = cursor.fetchall()  
 cursor.execute(sql2)  
 cnt2 = cursor.fetchall()  
 cnt1 = list(cnt1)  
 index = 0  
 for item in cnt1:  
 cnt1[index] = list(item)  
 index = index + 1  
 con.close()  
 for item1 in cnt1:  
 for item2 in cnt2:  
 if item1[5] == item2[0]:  
 item1[5] = item2[1]  
 break  
 print("商品查询结果：")  
 print(cnt1)  
 print("--------------")  
 # 表结构建立  
 tree = ttk.Treeview(frame1, height=16)  
 tree["selectmode"] = 'browse'  
 tree["show"] = 'headings'  
 tree["columns"] = ("商品编号", "商品名称", "商品类别", "商品单价", "商品成本", "供应商")  
 tree.column("商品编号", width=82)  
 tree.column("商品名称", width=82)  
 tree.column("商品类别", width=82)  
 tree.column("商品单价", width=82)  
 tree.column("商品成本", width=82)  
 tree.column("供应商", width=82)  
 tree.heading("商品编号", text="商品编号")  
 tree.heading("商品名称", text="商品名称")  
 tree.heading("商品类别", text="商品类别")  
 tree.heading("商品单价", text="商品单价")  
 tree.heading("商品成本", text="商品成本")  
 tree.heading("供应商", text="供应商")  
 tree.bind("<<TreeviewSelect>>", gchangeid)  
 index = 0  
 cnt1.sort(key=takeFirst)  
 for item in cnt1:  
 tree.insert("", index, values=(item[0], item[1],  
 item[2], format(item[3], '.2f'), format(item[4], '.2f'), item[5]))  
 index = index + 1  
 pselectgbtn.place(x=10, y=100)  
 addgbtn.place(x=10, y=150)  
 addmbtn.place(x=10, y=200)  
 selectmbtn.place(x=10, y=250)  
 frame1.place(x=100, y=0)  
 tree.place(x=0, y=45)  
 combobox.place(x=75, y=10)  
 lable1.place(x=0, y=10)  
 lable2.place(x=200, y=10)  
 selentry.place(x=275, y=10)  
 btn1.place(x=425, y=6)  
 btn2.place(x=100, y=400)  
 btn3.place(x=300, y=400)  
  
  
# 登录操作  
def login():  
 con = pymssql.connect(host="127.0.0.1", port="1433", database="d1", user="sa", password="123", charset="cp936")  
 cursor = con.cursor()  
 cursor.execute("select \* from adminUsers")  
 cnt = cursor.fetchone()  
 print(luser.get() + lpassword.get())  
 while cnt:  
 print(cnt)  
 if cnt[1] == luser.get() and cnt[2] == lpassword.get():  
 con.close()  
 loginwin.destroy()  
 mainui()  
 return  
 cnt = cursor.fetchone()  
 con.close()  
 msg = messagebox.showwarning(title="消息警告", message="用户名或密码错误，请重新输入")  
 print(msg)  
  
  
# 注册界面的返回选项  
def resreturnlogin():  
 registerwin.destroy()  
 loginui()  
  
  
# 注册操作  
def register():  
 if 0 < len(ruser.get()) < 10 and 0 < len(rpassword.get()) < 20:  
 con = pymssql.connect(host="127.0.0.1", port="1433", database="d1", user="sa", password="123", charset="UTF-8")  
 cursor = con.cursor()  
 print(ruser.get())  
 sql = "insert into adminUsers (username,userpassword) values('%s','%s')" % (ruser.get(), rpassword.get())  
 cursor.execute(sql)  
 con.commit()  
 con.close()  
 msg = messagebox.showinfo(title="消息提示", message="注册成功，请登录")  
 print(msg)  
 registerwin.destroy()  
 loginui()  
 else:  
 msg = messagebox.showwarning(title="消息警告", message="输入非法请重新输入")  
 print(msg)  
  
  
# 注册界面  
def registerui():  
 global registerwin  
 global ruser  
 global rpassword  
 loginwin.destroy()  
 registerwin = tk.Tk()  
  
 registerwin.geometry("300x200")  
 rlabel1 = tk.Label(registerwin, text="用户名:")  
 rlabel2 = tk.Label(registerwin, text="密 码:")  
 ruser = tk.Entry(registerwin)  
 rpassword = tk.Entry(registerwin, show="\*")  
 resgisterbtn = tk.Button(registerwin, text="注册", command=register)  
 backbtn = tk.Button(registerwin, text="返回", command=resreturnlogin)  
  
 rlabel1.place(x=65, y=20)  
 rlabel2.place(x=65, y=75)  
 ruser.place(x=115, y=20)  
 rpassword.place(x=115, y=75)  
 resgisterbtn.place(x=85, y=125)  
 backbtn.place(x=185, y=125)  
 loginwin.mainloop()  
  
  
# 登陆界面  
def loginui():  
 global loginwin  
 global luser  
 global lpassword  
 loginwin = tk.Tk()  
 loginwin.geometry("300x200")  
 llabel1 = tk.Label(loginwin, text="用户名:")  
 llabel2 = tk.Label(loginwin, text="密 码:")  
 luser = tk.Entry(loginwin)  
 lpassword = tk.Entry(loginwin, show="\*")  
 loginbtn = tk.Button(loginwin, text="登录", command=login)  
 resgisterbtn = tk.Button(loginwin, text="注册", command=registerui)  
  
 llabel1.place(x=65, y=20)  
 llabel2.place(x=65, y=75)  
 luser.place(x=115, y=20)  
 lpassword.place(x=115, y=75)  
 loginbtn.place(x=85, y=125)  
 resgisterbtn.place(x=185, y=125)  
 loginwin.mainloop()  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 mainui()

**（二）数据库设计主要代码**

use d1

/\*--------------------------------

--------------------------------\*/

/\*用户登录数据库\*/

drop table adminUsers

select \* from adminUsers

create table adminUsers (

ID int primary key identity(1,1),

username varchar(10) not null,

userpassword varchar(20) not null

);

SET IDENTITY\_INSERT adminUsers off;

insert into adminUsers

(username,userpassword)

values('zcwl','123')

/\*--------------------------------

--------------------------------\*/

/\*商品\*/

create table Goods(

Gnum varchar(10) primary key,/\*商品编号\*/

Gname varchar(10) not null,/\*商品名称\*/

Gtype varchar(10) not null,/\*商品类型\*/

Gprice money check(Gprice>=0),/\*商品单价\*/

Gbid money check(Gbid>=0),/\*商品成本\*/

Vnum varchar(10) not null,/\*供应商编号\*/

foreign key(Vnum) references Vendor(Vnum)

)

insert into Goods

(Gnum,Gname,Gtype,Gprice,Gbid,Vnum)

values ('1','1','1',1,1,'1')

select \* from Goods

/\*--------------------------------

--------------------------------\*/

/\*供应商\*/

insert into Vendor values (

'1','康师傅','12345678901','安阳市'

),(

'2','白象','12345678901','北京市'

)

select \* from Vendor

create table Vendor(

Vnum varchar(10) primary key,/\*编号\*/

Vname varchar(10) not null,/\*供应商名称\*/

Vphone varchar(20) not null,/\*供应商电话\*/

Vpalce varchar(10) not null/\*供应商地点\*/

)

/\*--------------------------------

--------------------------------

会员\*/

create table Menber(

Mnum varchar(10) primary key,/\*会员卡号\*/

Mname varchar(10) not null,/\*会员姓名\*/

Mphone varchar(20) not null,/\*会员电话\*/

Mdate data,/\*会员入会日期\*/

Mcip varchar),/\*会员密码\*/

Mbalance money check(Mbalance>=0),/\*剩余金额\*/

)

drop table Menber

/\*--------------------------------

--------------------------------\*/

七、总结