商品订单管理系统设计报告

学号：072110112，姓名：冉中益， 联系方式：18651876960

专业：计算机科学与技术

Email：815459626@qq.com

**摘 要:** 本报告旨在设计一个有效的商品订单管理系统，以提高企业的商品、买家及其订单管理水平。根据需求分析，系统需要具备数据存储、插入、删除、更新和检索功能。采用面向对象的方法进行设计，将系统划分为多个模块，并明确了各模块之间的关系和接口。为实现数据存储和检索功能，采用了关系型数据库管理系统，并设计了合理的数据结构和算法。同时，为提高系统的可靠性和安全性，采用了数据备份和加密等技术。最后，通过性能测试，验证了系统的可行性和效果。使用Python进行后端处理，tikinter库进行窗体界面搭建，OpenGauss提供数据库服务。

**关键词:** OpenGauss；数据库；商品订单管理

目录

[1 需求分析 1](#_Toc170203901)

[**1.1**需求概述 1](#_Toc170203902)

[1.1.1 需求说明 1](#_Toc170203903)

[1.1.2 用户特点 2](#_Toc170203904)

[**1.2**对功能需求的规定 2](#_Toc170203905)

[1.2.1 商品管理模块 2](#_Toc170203906)

[1.2.2 用户管理模块 2](#_Toc170203907)

[1.2.3 订单管理模块 2](#_Toc170203908)

[1.2.4 确定参与者 2](#_Toc170203909)

[**1.3**数据字典 3](#_Toc170203910)

[1.3.1 数据项 3](#_Toc170203911)

[1.3.2 数据项 3](#_Toc170203912)

[**1.4**运行栈与技术环境 3](#_Toc170203913)

[1.4.1 技术 3](#_Toc170203914)

[1.4.2 工具 3](#_Toc170203915)

[1.4.3 语言 3](#_Toc170203916)

[2 概念结构设计 3](#_Toc170203917)

[**2.1**系统结构图 4](#_Toc170203918)

[**2.2**E-R图 4](#_Toc170203919)

[3 逻辑结构设计 4](#_Toc170203920)

[**3.1**关系模式 4](#_Toc170203921)

[**3.2**基本表 5](#_Toc170203922)

[**3.3**数据模型的优化 5](#_Toc170203923)

[4 物理结构设计 5](#_Toc170203924)

[5 系统结构设计 6](#_Toc170203925)

[**5.1**Psycopg2库访问数据库的一般方式 6](#_Toc170203926)

[**5.2**用户信息管理模块 7](#_Toc170203927)

[5.2.1 登陆 7](#_Toc170203928)

[5.2.2 查看用户信息 8](#_Toc170203929)

[5.2.3 编辑用户信息 11](#_Toc170203930)

[5.2.4 检索用户信息 12](#_Toc170203931)

[**5.3**商品信息管理模块 13](#_Toc170203932)

[5.3.1 查看商品信息 13](#_Toc170203933)

[5.3.2 编辑商品信息 15](#_Toc170203934)

[5.3.3 检索用户信息 15](#_Toc170203935)

[**5.4**订单信息管理模块 17](#_Toc170203936)

[5.4.1 查看订单信息 17](#_Toc170203937)

[5.4.2 编辑订单信息 19](#_Toc170203938)

[5.4.3 查询订单信息 19](#_Toc170203939)

[6 效率分析及优化方案 21](#_Toc170203940)

[**6.1**效率分析 21](#_Toc170203941)

[**6.2**优化方案 22](#_Toc170203942)

[7 数据库实施与维护 22](#_Toc170203943)

[**7.1**数据的载入和应用程序的调试 22](#_Toc170203944)

[**7.2**数据库的试运行 22](#_Toc170203945)

[**7.3**数据库运行与维护 22](#_Toc170203946)

[8 心得体会 22](#_Toc170203947)

[9 实验二至实验五代码 23](#_Toc170203948)

[参考文献: 26](#_Toc170203949)

# 需求分析

## 需求概述

### 需求说明

随着经济社会的不断发展，企业对于商品订单管理的需求越来越迫切。商品订单管理系统的目标是帮助企业有效管理订单信息，提供高效的服务。以下是对商品信息管理系统的需求说明：

1.用户管理：系统应该提供用户管理功能，包括管理员和普通用户角色。管理员对数据库中所有表增删改查的权限，普通用户能够访问所有的商品信息，能够访问与修改与自身相关的用户与订单信息。提供权限管理功能，管理员可以配置用户对系统中不同功能模块的访问权限，确保安全性和数据隐私。

2.订单信息录入：系统应提供用户友好的界面，让买家能够轻松录入订单的基本信息，如订单id、商品id、用户id，购买时间，订单量等。系统应支持验证机制，确保输入的数据的准确性和完整性。

3.订单信息检索：系统应提供快速可靠的订单信息检索功能。买家可以根据订单id、商品id或其他标识符快速搜索和查找特定订单的信息。

4.数据安全和隐私保护：系统应采取必要的安全措施，确保订单的安全和隐私。数据传输应使用加密协议进行保护，以防止未经授权的访问和信息泄漏。系统应实施访问控制和权限管理，只有经授权的人员才能访问和修改顾客信息。

5.与其他系统的集成：系统应能够与其他业务系统进行集成，例如销售系统、客户关系管理系统等。可以通过API接口实现数据的共享和交换，确保不同系统之间的数据一致性和流动性

6.用户界面和易用性：系统的用户界面应直观友好，使员工能够快速学习和使用系统。提供简洁明了的操作界面和可点击的菜单，确保买家能够快速找到所需的功能和信息。

7.可扩展性和灵活性：系统应具备良好的可扩展性和灵活性，以满足不断变化的业务需求。系统架构应设计成模块化和可配置的方式，以便支持新的功能扩展和自定义字段添加。

该商品订单管理系统将实现商品订单的高效管理和查询功能，并提供数据分析和报告、数据安全和隐私保护、系统集成、用户界面友好等特点。它将帮助企业提高服务质量和经营效率，满足不断增长的商品订单管理需求。

### 用户特点

本系统的用户最终分为两类：系统管理员和买家用户。

系统管理员拥有对买家信息表，商品信息表，订单信息表增删改查的所有权限。

买家用户可以查看商品信息表，查看并修改个人信息，查看并修改个人的订单记录。

## 对功能需求的规定

### 商品管理模块

1. 商品信息编辑：用户可以根据自身需求插入商品数据，删除商品数据，更改或更新商品数据。以上功能只有具有管理员权限的用户才能使用。
2. 商品信息查看：所有类型的用户均可对商品信息（ID，名称，库存量）进行查看。
3. 商品信息检索：所有用户均可以通过商品ID与商品名称对商品信息进行检索。

### 用户管理模块

1. 用户注册与登陆功能：允许向数据库中添加新用户。添加后用户可根据用户名与对应的密码进行登录。
2. 用户权限管理：提供权限管理功能，允许管理员配置用户对系统中不同功能模块的访问权限。管理员可以定义角色的权限级别，并控制其对敏感信息的访问。
3. 用户信息编辑：允许用户编辑或注销自身信息，如密码，姓名，电话，地址等。用户的访问权限只有管理员能进行编辑。
4. 用户信息检索：提供用户信息检索功能，使管理员能够根据姓名、角色、部门等条件快速搜索和查找用户信息。

### 订单管理模块

1. 订单信息编辑：允许用户创建新的订单记录，也可以修改和删除现有的订单记录。订单记录中的订单量进行修改后商品信息中对应商品的库存量会发生改变，确保系统的一致性与正确性。
2. 订单信息检索：由于订单记录属性较多，向用户提供快速可靠的订单检索功能。用户可以任意组合所需的属性进行查询，同时提供筛选与排序功能，使用户能够根据不同的条件快速获取所需的订单信息。
3. 订单信息查看：用户可以点击订单记录以详细查看订单信息。除展示订单记录表中的基础属性（订单ID、商品ID、用户ID、创建时间、订单量）之外，为便于用户查询，连接了用户信息表中的姓名属性和商品信息表中的商品名称属性供用户同时进行查看。
4. 数据安全和隐私保护：为保证数据安全性和用户数据隐私，每个买家用户只能查看本人的用户信息及订单信息，编辑权限也只限于与本人相关的数据记录，不可查看其他用户的数据信息。管理员用户可以查看和编辑所有数据表的信息。

### 确定参与者

表1 参与者确定

|  |  |
| --- | --- |
| 参与者 | 参与操作 |
| 系统管理员 | 登陆，查看与编辑商品信息、买家信息、订单信息 |
| 买家用户 | 登陆，查看商品信息，查看与编辑自身信息，查看与编辑自身订单信息 |

## 数据字典

数据字典是进行详细的数据收集和数据分析所获得的主要成果。它是关于数据库中数据的描述，即元

数据，而不是数据本身。数据字典通常包括数据项、数据结构、数据流、数据存储和处理过程几个部分。

### 数据项

表2 数据项

|  |  |  |  |
| --- | --- | --- | --- |
| 数据项名 | 类型 | 长度 | 备注 |
| 用户id | Varchar | 255 |  |
| 用户密码 | Varchar | 255 |  |
| 用户姓名 | Varchar | 255 |  |
| 用户电话 | Varchar | 11 |  |
| 用户地址 | Varchar | 255 |  |
| 用户类型 | Varchar | 2 | 0表示管理员 1表示普通用户 |
| 商品id | Varchar | 20 |  |
| 商品名称 | Varchar | 20 |  |
| 库存量 | Int |  | 属性名被误设为price |
| 订单id | Varchar | 20 |  |
| 购买时间 | Varchar | 255 | 格式为Y-M |
| 订购量 | Int |  | 属性名被误设为money |

### 数据项

数据结构反映了数据之间的组合关系。一个数据结构可由若干个数据项组成，也可由若干个数据结构

组成，或由若干个数据项和数据结构混合组成。

表3数据结构

|  |  |  |
| --- | --- | --- |
| 数据结构名 | 含义说明 | 组成 |
| 用户信息 | 所有用户的信息 | 用户id、用户密码、用户姓名、用户电话、用户地址、用户类型 |
| 商品信息 | 所有商品的信息 | 商品id、商品名称、库存量 |
| 订单信息 | 所有订单的信息 | 订单id、商品id、用户id、购买时间、订单量 |

## 运行栈与技术环境

### 技术

Python实现后台数据处理，tikinter实现图形化窗体界面，OpenGauss数据库管理系统，psycopg2库访问数据库

### 工具

MacOS OpenGauss

### 语言

Python PostgreSQL

# 概念结构设计

将需求分析得到的用户需求抽象为信息结构（即概念模型）的过程就是概念结构设计。

## 系统结构图

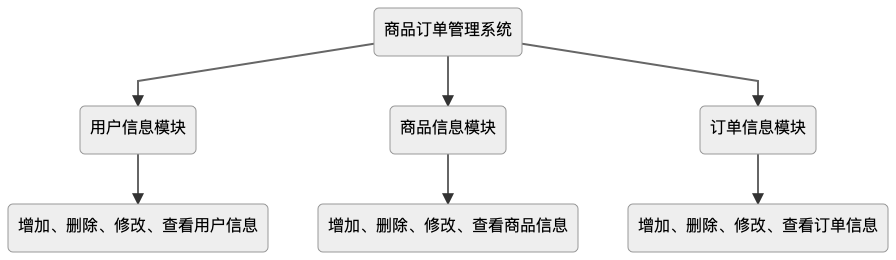


图1 系统结构图

## E-R图

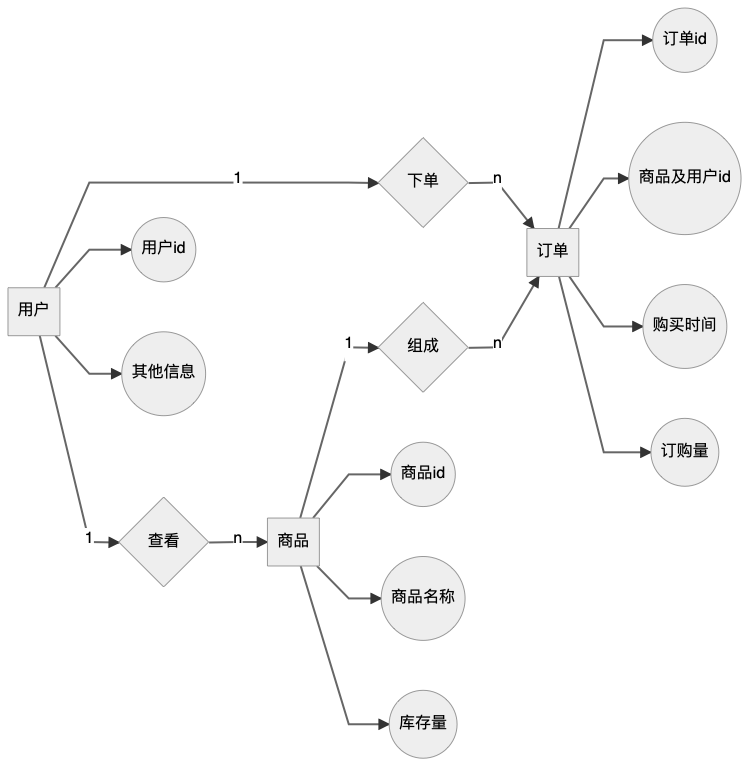


图2 E-R图

# 逻辑结构设计

## 关系模式

用户表（用户id、密码、姓名、电话、地址、类型）

商品表（商品id、名称、库存量）

订单表（订单id、商品id、用户id、购买时间、订购量）

## 基本表

表4 用户表t\_user

|  |  |  |  |
| --- | --- | --- | --- |
| 属性名 | 类型 | 约束 | 字段说明 |
| Id | Varchar(255) | Primary key, not NULL | 用户ID |
| Passwd | Varchar(255) | Not NULL | 密码 |
| Name | Varchar(255) |  | 姓名 |
| Number | Varchar(11) |  | 电话 |
| Addr | Varchar(255) |  | 地址 |
| Type | Varchar(2) | Not NULL | 类型，0表示管理员，1表示普通用户 |

表5 商品表t\_goods

|  |  |  |  |
| --- | --- | --- | --- |
| 属性名 | 类型 | 约束 | 字段说明 |
| Id | Varchar(20) | Primary key, not NULL | 商品id |
| Name | Varchar(20) |  | 商品名称 |
| Price | Int | Not NULL, price >= 0 | 库存量，属性名误设为price |

表6 订单表t\_order

|  |  |  |  |
| --- | --- | --- | --- |
| 属性名 | 类型 | 约束 | 字段说明 |
| Id | Varchar(20) | Not NULL | 订单id |
| Gid | Varchar(20) | Not NULL, references t\_goods(id) | 商品id |
| Uid | Varchar(255) | Not NULL, references t\_user(id) | 用户id |
| Buy\_time | Varchar(255) |  | 购买时间 |
| Money | Int | Not NULL | 订购量，属性名误设为money |

## 数据模型的优化

1. 符合1NF的关系中的每个属性都不可再分
2. 2NF在1NF的基础上，消除了非主属性对于码的部分函数依赖
3. 3NF在2NF的基础上，消除了非主属性对于码的传递函数依赖

# 物理结构设计

* 创建用户表t\_user
* CREATE TABLE t\_user (  
  id *varchar*(255) NOT NULL,  
  passwd *varchar*(255) NOT NULL,  
  name *varchar*(255) DEFAULT NULL,  
  number *varchar*(11) DEFAULT NULL,  
  addr *varchar*(255) DEFAULT NULL,  
  type *varchar*(2) NOT NULL, ---0表示管理员 1表示普通用户  
  PRIMARY KEY (id)  
  );
* 创建商品表t\_goods
* CREATE TABLE t\_goods (  
  id *varchar*(20) NOT NULL,  
  name *varchar*(20) DEFAULT NULL,  
  price *int* NOT NULL CHECK ( price >= 0 ), ---实际上用来表示库存量  
  PRIMARY KEY (id)  
  );
* 创建订单表t\_order
* CREATE TABLE t\_order(  
  id *varchar*(20) NOT NULL,  
  gid *varchar*(20) NOT NULL,  
  uid *varchar*(255) NOT NULL,  
  buy\_time *varchar*(255) DEFAULT NULL,  
  money *int* NOT NULL, ---实际上用来表示订单量  
  PRIMARY KEY (id),  
  FOREIGN KEY (gid) REFERENCES t\_goods(id),  
  FOREIGN KEY (uid) REFERENCES t\_user(id)  
  );

# 系统结构设计

## Psycopg2库访问数据库的一般方式

db = psycopg2.connect(database="db\_rzy", user="rzy", password="Rzy021006", host="1.92.80.33",  
 port=26000) *# 服务器名,账户,密码,数据库名*cursor = db.cursor()  
sql = """  
---此处写入PostgreSQL语句  
"""  
  
*try*:  
 cursor.execute(sql)  
 db.commit()  
*except*:  
 messagebox.showinfo('警告！', '触发器创建失败')  
cursor.close() *# 关闭游标*db.close() *# 关闭数据库连接*

## 用户信息管理模块

### 登陆



用户输入自身用户明及密码即可登陆。



系统通过前端获取用户名后，在t\_user表中检索与输入用户名相同的表项。然后判断输入的账号密码是否与输入的一致。若一致，则登陆成功，根据检索到的表项的类型属性判断用户属于管理员还是普通用户，跳转到不同的用户菜单中。

### 查看用户信息

对于管理员用户，登陆后可以查看到所有的用户信息，点击表项可以在左上角查看具体的属性。点击一次具体的属性名可以按照该属性进行升序排序，再点击一次可以进行降序排序。



对于普通用户，登陆后只能查看自己的用户信息。



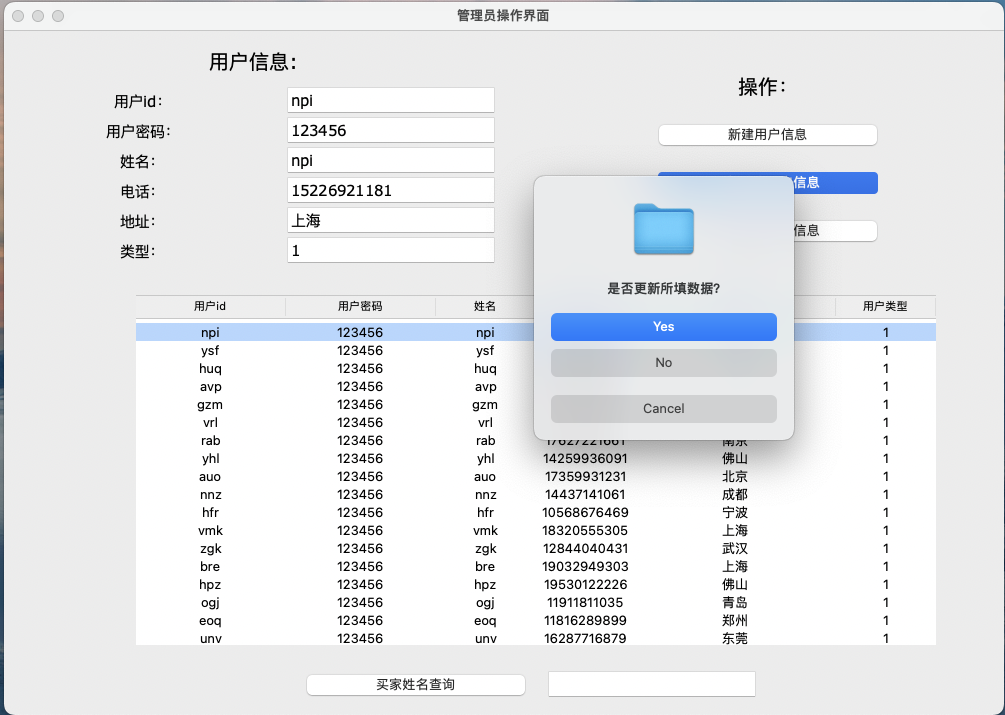
核心实现代码：



管理员与普通用户的查询语句区别在于普通用户加上了id = login\_id的查询条件，此处不再展示。

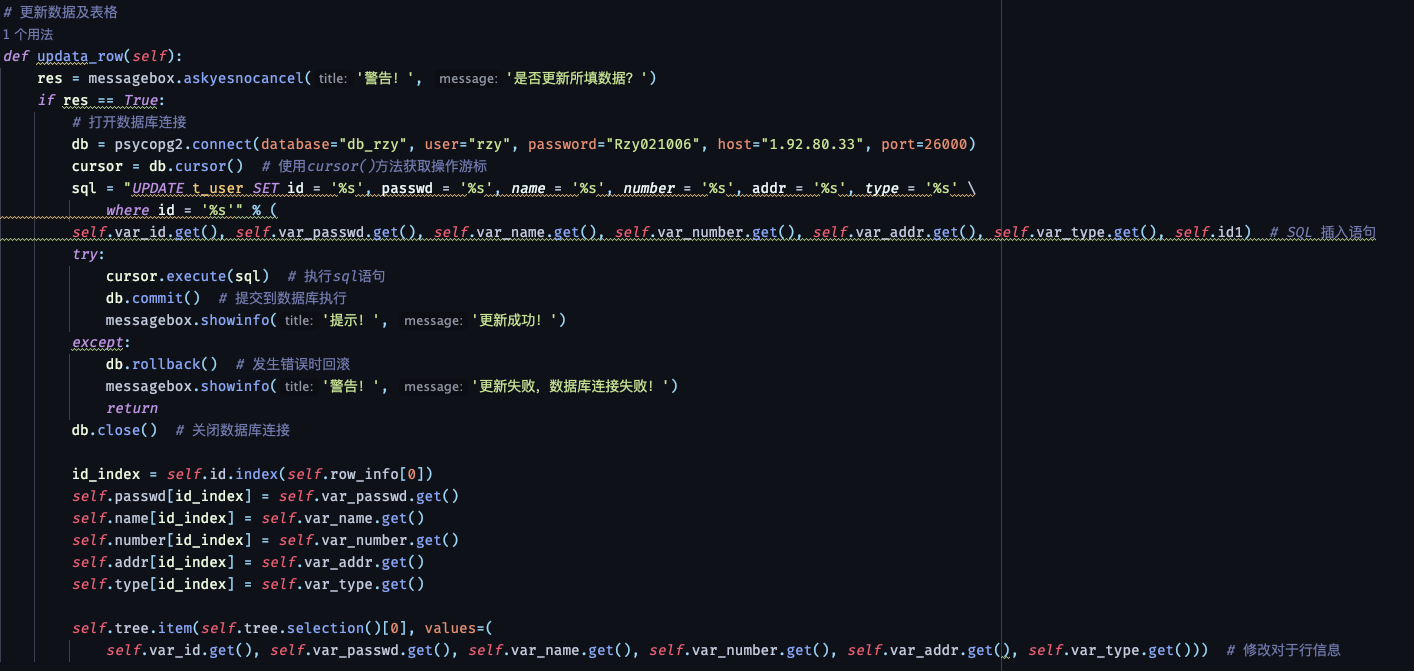
### 编辑用户信息

管理员可以对所有用户的信息进行增加、删除、修改。在左上角的输入框中填入所有要编辑的信息后，根据操作类型选择右边的按钮即可进行对应操作。此处只展示更新提示界面。



普通用户只能删除自身信息或修改除用户id与类型之外的属性，界面与5.2.2中相同，不再展示。

核心代码（此处只展示更新代码）：



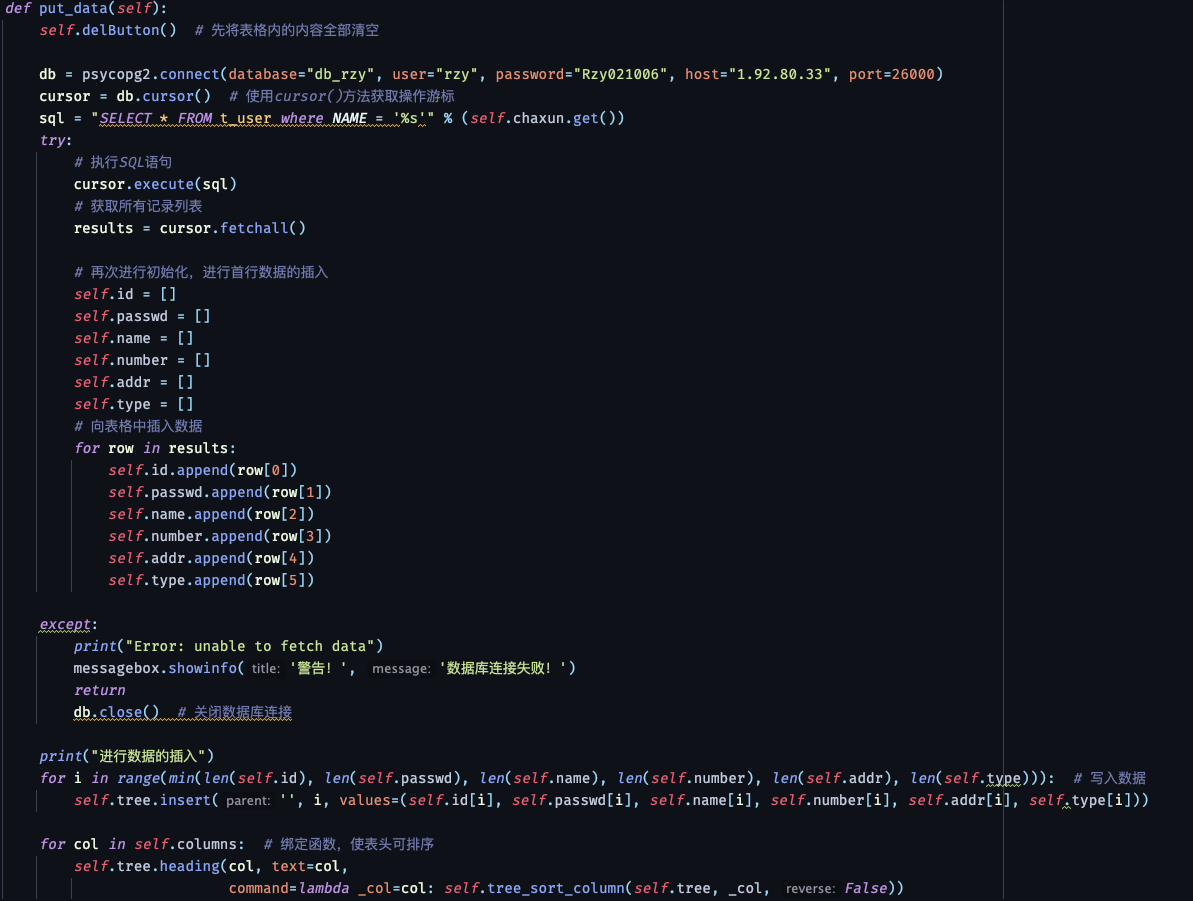
### 检索用户信息

用户信息表较为简单，表项较少，不在此设计较为复杂的查询。只提供用户进行姓名查询的方式。用户在下方输入框中输入买家姓名后点击查询按钮即可查询到用户信息。



普通用户只能对自身信息进行查询，无法检索其他用户的信息。

核心代码：



## 商品信息管理模块

### 查看商品信息

对于管理员用户与普通用户，登陆后均可以查看到所有的商品信息，点击表项可以在左上角查看具体的属性。点击一次具体的属性名可以按照该属性进行升序排序，再点击一次可以进行降序排序。普通用户对商品表只有可读权限，因此左上角属性框为不可输入框（灰色）。



管理员界面



普通用户界面

核心实现代码：



### 编辑商品信息

管理员可以对所有商品的信息进行增加、删除、修改。在左上角的输入框中填入所有要编辑的信息后，根据操作类型选择右边的按钮即可进行对应操作。界面与5.2.3中类似，不再展示。

普通用户不可编辑商品信息。

核心代码（此处只展示更新代码）：

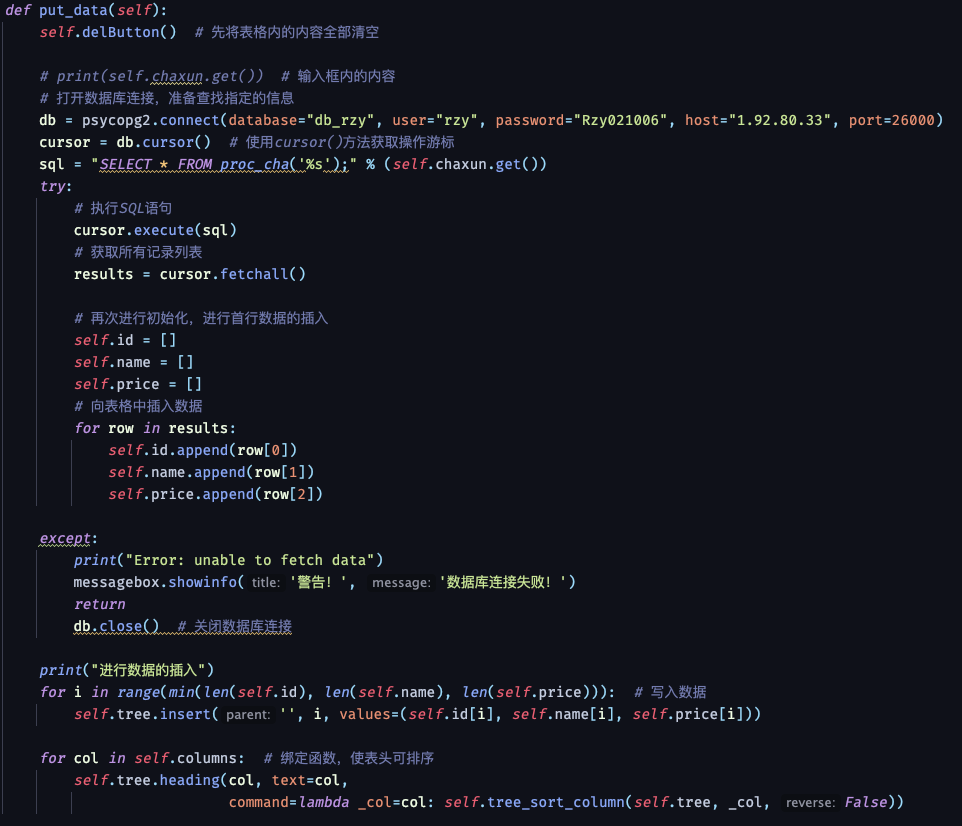


### 检索用户信息

商品信息表较为简单，表项较少，不在此设计较为复杂的查询。只提供用户进行商品名称查询的方式。用户在下方输入框中输入商品名称后点击查询按钮即可查询到用户信息。管理员用户与普通用户均可以对所有商品信息进行查询。



核心代码：



## 订单信息管理模块

### 查看订单信息

管理员可以查看所有订单的信息，普通用户只能查看自身的订单信息。管理员与普通用户查看订单的界面类似。点击表项后弹窗中会显示订单表的详细属性（订单ID、商品ID、用户ID、订单创建时间、订购量），并会连接查询显示商品ID与用户ID对应的商品名称与用户姓名。下方按钮集成了翻页功能，一页显示500条数据，点击不同按钮可实现不同翻页效果。以下展示管理员用户的界面及核心代码：



基本界面



查看弹窗

核心代码：







### 编辑订单信息

管理员用户可以增加、修改、删除所有订单的信息，普通用户只能编辑自身的订单信息，且不可更改订单的uid属性（用户ID），编辑方式与5.2.3及5.3.2中类似，此处不再展示界面。

核心代码（只展示更新代码）：

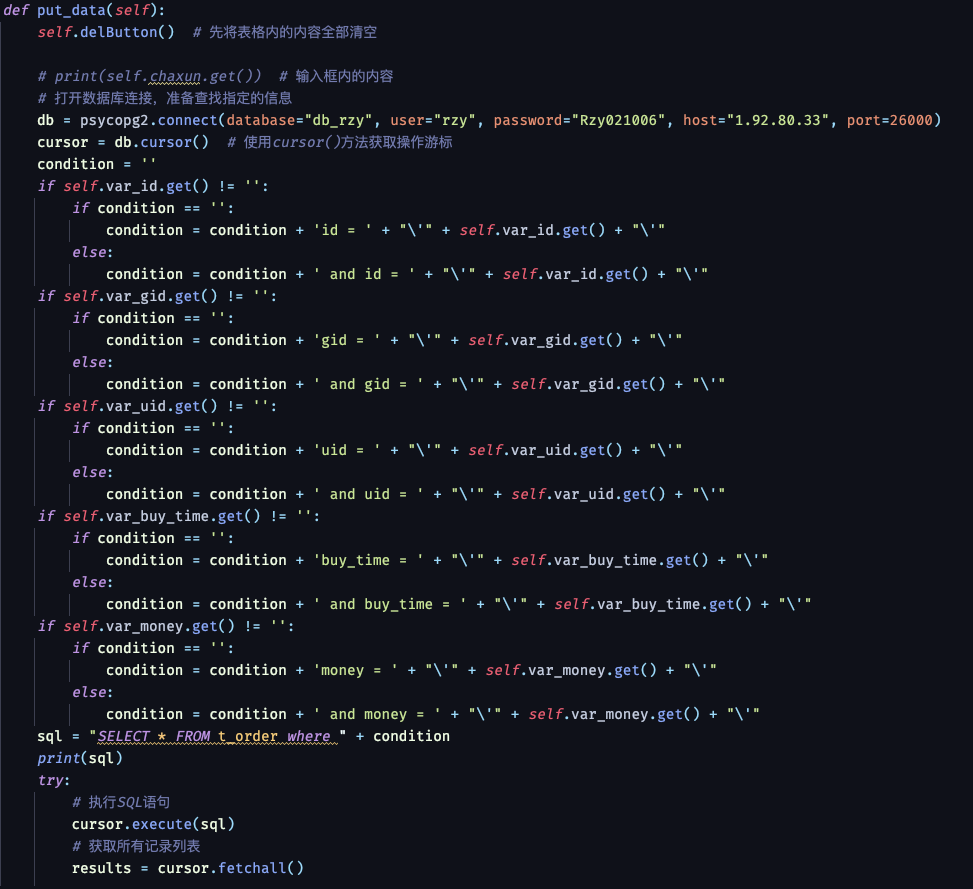


### 查询订单信息

由于订单表属性较多，且数据量较大（1000000条），设置了比前两个表更灵活的查询方式。用户可任意组合输入所需的查询条件，无需将左上角的输入框全部填满，输入部分条件后，点击右上方的查询按钮即可实现查询。



核心代码：



# 效率分析及优化方案

## 效率分析

特色：

1)        在对各表的增加、删除、修改等编辑操作上遵循老师意见才用了存储过程，以此来提高效率

2)        商品表中的“库存量”这一属性是一个需要经常变动的属性。如当订单表中有某订单的订购量发生变化时，商品表中的库存量也要进行相应的增减才能保证库存量总量不变。为了快速、高效率地自动实现这一变化，在修改订单表表项时没有赋予多条SQL进行修改，而是采用了触发器，对订单表上订购量这一属性进行修改或有记录增减时作出相应，提高了对库存量进行修改的效率。

3)        由于订单表表项数量大（1000000条），且普通用户在进行订单管理时只能查询自身的订单，后台查询的效率较低，故在订单表的“用户ID”属性上建立了哈希索引，极大提高了针对具体用户检索订单的效率。

不足：

商品信息表和订单信息表中，id这一字段的数据类型为varchar，varchar类型存储变长字符数据，需要额外的存储空间来存储字符长度信息。相比之下，int类型是固定长度的，不需要额外的空间。这意味着在存储大量数据时，varchar类型会占用更多的存储空间，导致磁盘IO成本增加。

## 优化方案

将关系模式t\_goods和t\_order的id字段改为int类型并且创建索引是优化的关键。通过在id字段上创建索引，可以使数据库更快地定位和检索具有特定id值的记录。

# 数据库实施与维护

## 数据的载入和应用程序的调试

数据库实施阶段包括数据的载入与应用程序的编码和调试。数据库的载入是指将数据从外部源（如文件、其他数据库等）导入到数据库中的过程。这可以包括创建数据库表格和字段结构，将数据插入表格中，并确保导入的数据符合数据库的完整性和一致性要求。应用信息的调试是指在开发或操作数据库应用程序时，通过运行和调试代码来验证和修复与数据库交互的问题。这包括检查数据库查询和操作的正确性、数据传输的准确性，以及确保应用程序正确处理数据库连接、数据处理和错误处理等方面的功能.通过进行数据库的载入和应用信息的调试，开发人员和数据库管理员可以确保数据的正确导入和应用程序与数据库的正常交互。这对于系统的稳定性、数据的完整性和应用程序的功能性都至关重要。

## 数据库的试运行

通过进行试运行，可以预先发现和解决数据库系统可能存在的问题，提高系统的可靠性和稳定性。试运行还可以为数据库管理员和用户提供对系统运行情况的深入了解，以及对后续操作和维护计划的指导。一旦通过试运行阶段的测试和验证，数据库可以被认为是稳定可靠的，并准备好正式投入使用。

## 数据库运行与维护

数据库的运行和维护是指对数据库系统进行日常操作和管理的过程，以确保数据库的正常运行、数据的完整性和可靠性。数据库的运行和维护是一个持续的过程，需要数据库管理员密切关注数据库的运行状态，及时应对问题，并进行必要的优化和改进，以确保数据库系统的稳定和高效运行。

# 心得体会

在大学的学习生涯中，到大三这个阶段，意味着我们将要面临更多的挑战和压力。在这个关键时刻，我非常荣幸能够完成《数据库原理与应用》这门课程的设计任务，并且独立使用Python和PostgreSQL编写了一个商品订单管理系统。通过这个项目，我不仅掌握了更多的技术知识，还收获了宝贵的经验教训。

在这个课程设计中，我的任务是设计和实现一个完整的商品订单管理系统。这个系统需要支持创建、编辑、查询和删除订单记录，而且需要支持各种数据操作和检索操作。为了完成这个任务，我首先需要掌握数据库的基本原理和设计技巧。

在数据库设计方面，我采用了关系型数据库的设计方法，将订单记录存储在一个关系型数据库中。这样，我可以方便地使用SQL语言对数据库进行操作，包括数据的插入、查询、更新和删除等。同时，我还需要学习并使用Python编程语言和PostgreSQL数据库管理系统，来实现系统的各种功能。

在这个课程设计中，我遇到了很多挑战和困难。其中最大的一个问题就是如何将Python编程语言和PostgreSQL数据库管理系统结合起来，以实现系统的各种功能。在这个过程中，我阅读了大量的相关文献和教程，并在互联网上寻找相关的资源和代码。通过不断地尝试和实践，我最终成功地将Python和PostgreSQL集成在一起，实现了系统的各种功能。

通过这个课程设计，我深刻地体会到了数据库设计和开发的重要性。同时，我也学习到了如何使用Python和PostgreSQL来实现一个商品订单管理系统。在课程设计的过程中，我遇到了一些困难和挑战，但是通过不断地学习和实践，我最终成功地完成了任务。

总的来说，这个课程设计让我收获颇丰。我不仅学习到了数据库设计和开发的相关知识，还掌握了一些实用的编程技巧和方法。最重要的是，我意识到了自学的重要性，并且学会了如何独立高效地自学计算机方面编程知识。

在未来的学习和工作中，我将继续努力学习和提高自己的技能水平，以更好地适应社会的需求和发展。我将继续关注数据库技术和编程语言的发展趋势，学习最新的技术和工具，以提高自己的竞争力。同时，我也会注重团队合作和沟通能力的提升，以更好地适应团队合作的环境。

最后，我要感谢我的老师和同学们在课程设计过程中给予我的帮助和支持。没有你们的指导和鼓励，我不可能如此顺利地完成这个任务。我将继续努力学习和实践，以回报大家的期望和信任。

# 实验二至实验五代码

为保证实验训练的完整性，我在完成实验六之余也完成了实验二至实验五。在此给出实验一至实验五的代码。

* 2-1
* *select s*.sno, sname  
  *from s*, sc  
  *where s*.sno = sc.sno *and* cno = '1';
* 2-2
* *select s*.sno, sname  
  *from s*, sc, *c  
  where s*.sno = sc.sno *and c*.cno = sc.cno *and* cname = '数据结构';
* 2-3
* *select s*.sno, sname  
  *from s*, sc  
  *where s*.sno = sc.sno *and* cno != '1';
* 2-4
* *select s*.sname  
  *from s  
  where not exists*(  
   *select* \*  
   *from c  
   where not exists*(  
   *select* \*  
   *from* sc  
   *where* sc.sno = *s*.sno *and* sc.cno = *c*.cno  
   )  
  );
* 2-5
* *select* sno, *avg*(grade)  
  *from* sc scx  
  *where* scx.sno *not in*(  
   *select* sno  
   *from* sc scy  
   *where* scy.cno <> '1' *and* grade < 60  
   )  
  *group by* sno  
  *order by avg*(grade) *desc*;
* 2-6
* *select s*.sname  
  *from s*, sc, *c  
  where s*.sno = sc.sno *and c*.cno = sc.cno *and* cname = '数据库原理'  
  *order by* grade *desc  
  limit* 1 *offset* 1;
* 2-7
* *select s*.sname  
  *from s*, sc  
  *where s*.sno = sc.sno  
  *and* cno *in* (*select* cno *from c where* ccredit = 3)  
  *and* grade >= 80  
  *group by s*.sno  
  *having count*(\*) >= 3;
* 2-8
* *select* sno  
  *from* sc scx  
  *group by* sno  
  *having count*(\*) *not in*(  
   *select count*(\*)  
   *from* sc scy  
   *group by* scy.sno  
   *having* scy.sno <> scx.sno  
   );
* 3-1
* *update* sc  
  *set* grade = grade \* 1.1  
  *where* cno = 1;
* 3-2
* *delete from* sc  
  *where* cno *in*(  
   *select* cno  
   *from c  
   where* cname = '数据结构'  
   );
* 3-3
* *insert into s*(sno, sname, ssex, sbirthdate, sdept)  
  *values* ('202415122', '王佳佳', '女', '2001-12-07', '数据科学与大数据技术');  
    
  *insert into* sc(sno, cno, grade, semester, teachingclass)  
  *values* ('202415122', '2', 98, '20192', '81001-01');  
    
  *delete from* sc  
  *where* sno = '202415122';  
    
  *delete from s  
  where* sno = '202415122';
* 4-1
* *create view* boy(sno, sname, cname, grade)  
  *as  
  select s*.sno, *s*.sname, *c*.cname, sc.grade  
  *from s*, sc, *c  
  where s*.ssex = '男' *and s*.sno = sc.sno *and c*.cno = sc.cno;
* 4-2
* *select* sno, sname  
  *from* boy  
  *group by* sno, sname  
  *having avg*(grade) > 80;
* 5-1
* *select* sno, *count*(\*), *avg*(sc.grade)  
  *from* sc  
  *where* grade *is not null  
  group by* sno;
* 5-2
* *grant all privileges on table s*, *c*, sc  
  *to* quinn;

参考文献:

1. 王珊，萨师煊. 数据库系统概论. 第五版[M]. 北京: 高等教育出版社，2014
2. ODBC:https://baike.baidu.com/item/%E5%BC%80%E6%94%BE%E6%95%B0%E6%8D%AE%E5%BA%93%E4%BA%92%E8%BF%9E/10418782?fromtitle=ODBC&fromid=759553&fr=aladdin.
3. SQL Server 创建表及其约束条件：https://blog.csdn.net/csdn9\_14/article/details/52612466