**电 子 科 技 大 学**

**实 验 报 告**

**学生姓名： 学 号： 指导教师：**

**一、实验室名称：**

**二、实验项目名称：简单应用系统实现**

**三、实验原理： 结构化查询语言SQL**

**四、实验目的：**

**1、熟悉数据库的设计，并掌握数据库设计的基本方法和步骤，了解数据库设计各阶段的任务等；**

**2、巩固SQL语言命令、练习E-R图的绘制、体会数据库应用与数据库理论知识的衔接；**

**3、通过前几个实验的练习，对数据库的基本概念有了一定的掌握。本实验综合所掌握的内容加以应用，选定自己熟悉的一种开发工具设计实现一个简单的数据库应用系统，包括数据的插入、删除、修改、查询、统计等功能，做到界面友好、使用方便。**

1. **实验内容：**

选择自己熟悉的某行业领域，或事务流程，加以分析，进行功能和需求分析，在此基础上设计相应的概念模型和数据库系统，并通过测试数据进行评价。

（可供参考的应用模型：课程管理子系统、图书管理子系统、通讯录子系统、教室管理子系统、宿舍管理子系统、个人小帐务管理子系统、个人事务管理子系统、食堂管理子系统、卫生科医疗管理子系统、游戏管理子系统、库房入库出库管理子系统、小商店销售管理子系统、其它管理子系统）

具体实验内容及要求如下：

1．对一小型数据库应用系统进行需求分析（文字说明）；

2．绘制E-R图（E-R图）；

3．将E-R模型向数据模型转换（写出关系模式R(U,F)，及相应的数据依赖集F）；

4．创建数据库，定义基本表；

5．开发应用系统，能满足用户的基本需求；

**六、实验器材（设备、元器件）：**

**硬件平台：PC；**

**软件平台：Windows；**

**MICROSOFT SQL SERVER；**

**或MySQL**

**七、实验步骤：**

开发数据库应用程序的第一步是设计系统的数据库和数据表结构，数据库设计包括数据库的需求分析、概念设计和逻辑实现等。图书信息管理系统作为数据库管理系统的一种应用，数据库的设计非常重要，下面是图书信息管理系统的数据库设计过程。

**7.1 系统介绍**

校园图书馆为学校师生提供了便捷的文献查找、借阅等服务，是学校中重要的文化服务部门。一套稳定、高效的图书信息管理系统可以提高图书馆的对图书的管理效率。

本图书信息管理系统的目标是：信息收集标准化，信息加工简单化，信息传递规范化，信息内容系统化，信息存储条目化及信息工作专业化。

本图书信息管理系统包括了图书预约、图书借阅、图书续借和图书归还等基本部分。用户总体业务构造如下：

（1）对图书信息的管理：图书馆管理人员需要对馆内图书进行管理。图书包括的信息主要有：书号，属性，作者，出版社，图书状态。这里的图书状态包括：被借阅、被预约、可借等状态

（2）对学生信息的管理：图书馆管理人员还需要管理学生的信息。学生包含的信息主要有：学号，姓名，学别，班级，学院，借阅许可等信息。其中借阅许可表示该学生是否有资格借阅图书，规定只要是在校本科生，都可以借阅图书，且一个学生的已借阅图书不能超过五本。

（3）图书预约和借阅包括：学生可以根据图书管理系统查找可借的书籍，并可以对图书进行借阅和预约操作。系统应做到图书借阅和预约的无缝连接，以及时掌握图书状态，方便使用，最大程度地利用图书馆的资源。

（4）图书续借和归还包括：学生在图书到期前可以对自己已借阅的图书进行续借或归还操作。系统应对学生借阅的书籍到期或延期时给与相应的提示和处理。

（5）系统应能做到数据库的维护工作，能执行数据库的备份和恢复，具有一定的安全性、稳定性，并具有一定的兼容性（即可以在图书馆现有的设备上运行）。

**7.2 需求分析**

**7.2.1 需求调查**

设计人员首先需要对用户需求调查，进一步表达和分析用户的需求，进而设计出高效的数据库系统。进过调查分析，我们得到下列用户需求：

（1）需要有学生信息和图书信息：

学生信息包括：学号、姓名、性别、班级、学院

图书信息包括：书号、书名、作者、出版社、出版日期、图书种类、图书状态

当图书被学生借阅时，系统要改变图书的状态，不再允许借阅，但允许预约，并且每本书只能被一人借阅或同时被一人预约。学生可以在系统内查阅自己的借阅情况，归还自己借阅的图书。系统的用例包括：查询图书、借阅图书、归还图书。

（2）需要有系统管理员信息：

管理员信息包括：用户名、联系方式、密码

（3）需要有图书的借阅、预约、还书信息：

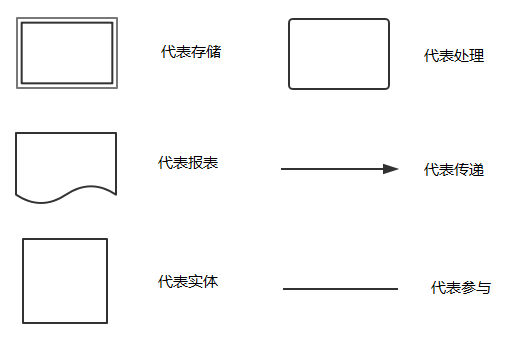
借阅信息包括：借阅号、借阅时间、借阅数目、续借时间、书号、学号

预约信息包括：预约号、预约时间、预约数目、书号、学号

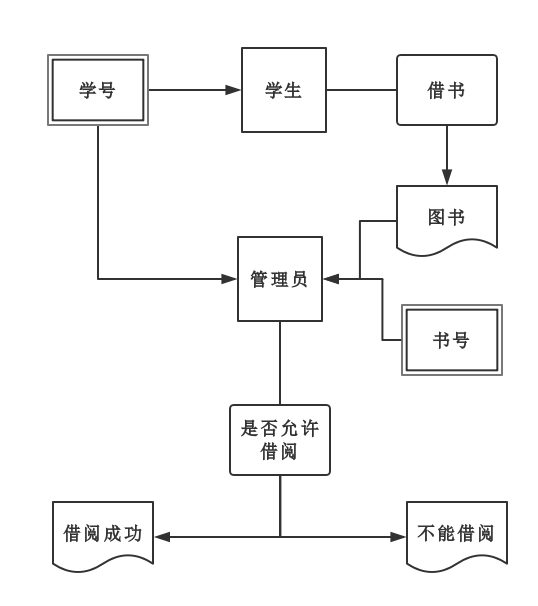
还书信息包括：还书号、还书时间、书号、学号

**7.2.2 数据流程图**

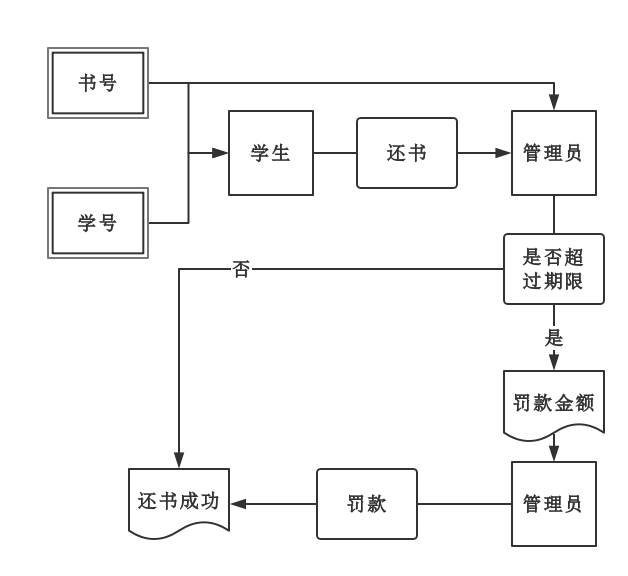
**图例：**

****

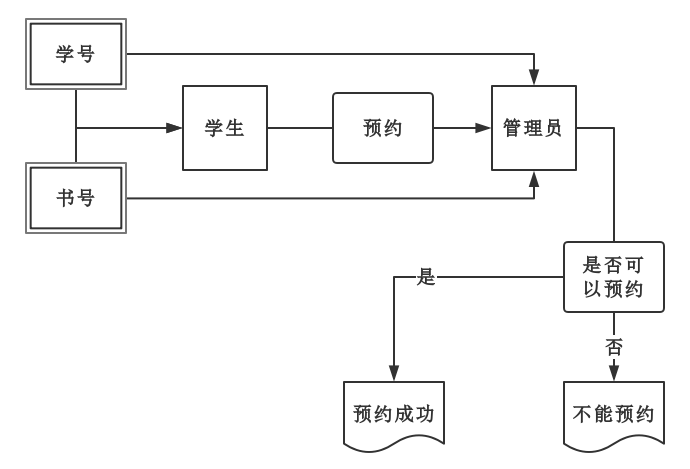
**借书流程：**

****

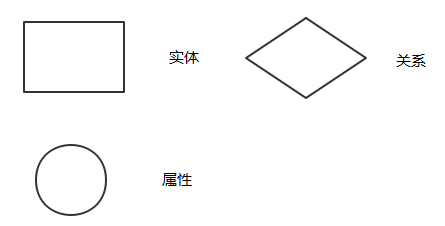
**还书流程：**

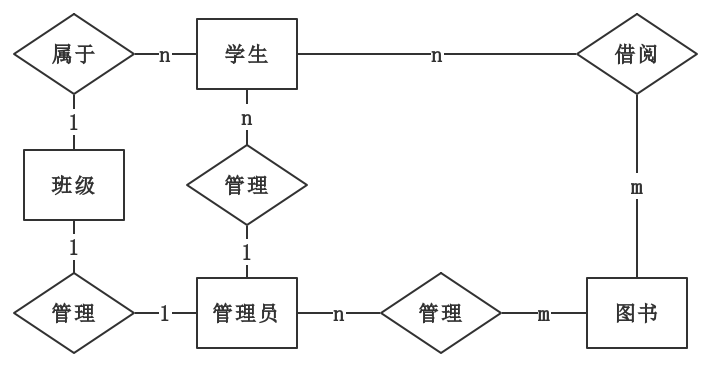
****

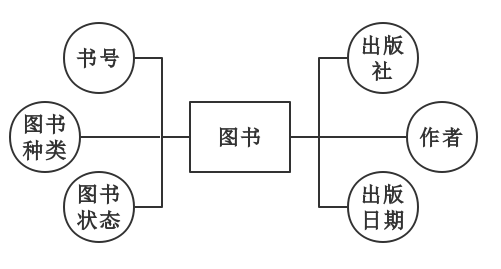
**预约流程:**

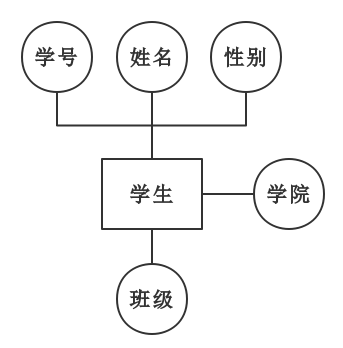
****

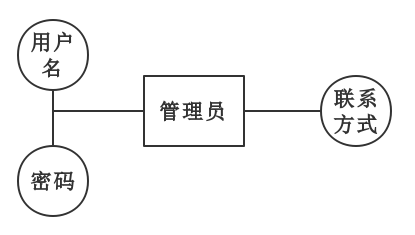
**7.2.3 E-R图**

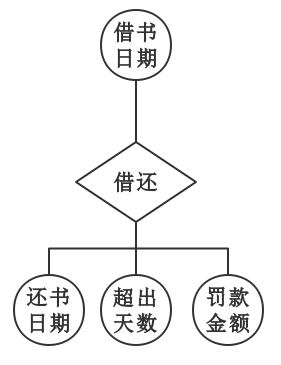
****

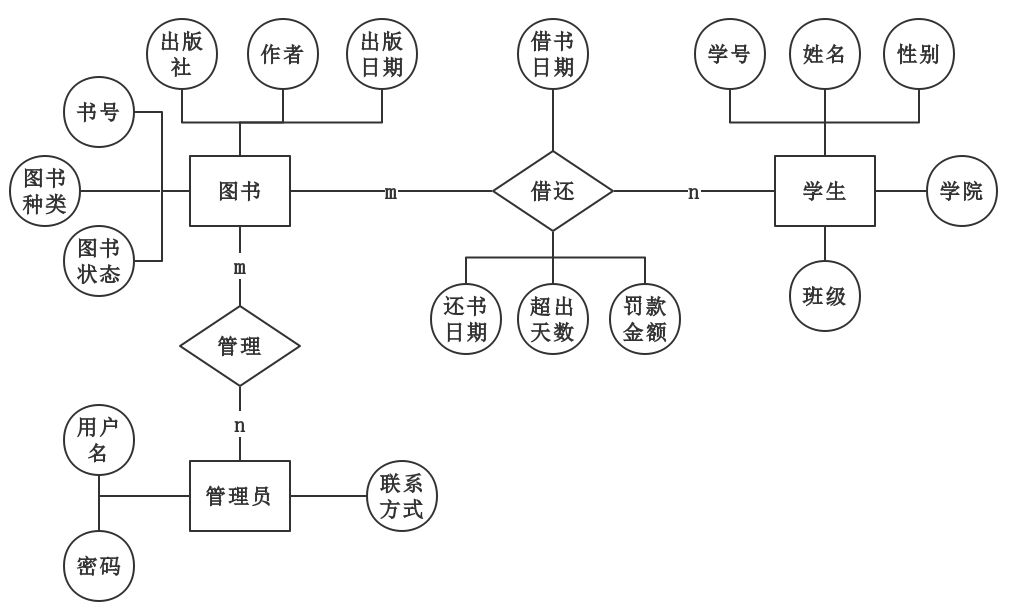












**7.2.4 E-R模型向数据模型转换**

此图书信息管理系统的所有表学生信息表都属于函数依赖集。

**学生信息表:**

U={学号，姓名，性别，班级，学院}

候选码为：学号；

F={学号——>姓名,学号——>性别，学号——>班级，学号——>学院 }

**管理员信息表:**

U={用户名，联系方式，密码}

候选码为：用户名；

F={用户名——>联系方式,用户名——>密码}

**图书信息表:**

U={书号，书名，作者，出版社，出版日期，图书种类，图书状态}

候选码为：书号；

F={书号——>书名,书号——>作者，书号——>出版社，书号——>出版日期，书号——>图书种类, 书号——>图书状态}

**借阅信息表:**

U={借阅号，借阅时间，借阅数目，续借时间，书号，学号}

候选码为：借阅号，书号，学号；

F={借阅号——>借阅时间,借阅号——>借阅数目，借阅号——>续借时间}

**预约信息表:**

U={预约号，预约时间，预约数目，书号，学号}

候选码为：预约号，书号，学号；

F={预约号——>预约时间,预约号——>预约数目}

**还书信息表:**

U={还书号，还书时间，书号，学号 }

候选码为：还书号，书号，学号；

F={还书号——>还书时间}

对于以上6个关系模式，每个属性值都是不可再分的原子值，满足1NF。它们的每个非主属性都完全函数依赖于码，首先R=2NF。它们的每一个非主属性既不部分依赖于码也不传递依赖于码，R=3NF。故以上关系规范化符合第三范式。

**7.2.5 数据字典**

表1.学生用户基本信息表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 数据类型 | 最大长度 | 允许空 | 约束控制 | 备注 |
| StuNo | int |  | 否 | PRI | 学号 |
| StuName | varchar(50) | 50 | 否 |  | 姓名 |
| StuSex | varchar(20) | 20 | 是 | 男，女 | 性别 |
| Class | varchar(50) | 50 | 否 |  | 班级 |
| Department | varchar(50) | 20 | 否 |  | 学院 |

表2.管理员基本信息表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 数据类型 | 最大长度 | 允许空 | 约束控制 | 备注 |
| MgNo | int |  | 否 | PRI | 用户名 |
| Phone | int |  | 否 |  | 联系方式 |
| Password | Varchar(50) | 50 | 否 |  | 密码 |

表3.图书基本信息表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 数据类型 | 最大长度 | 允许空 | 约束控制 | 备注 |
| BookNo | int |  | 否 | PRI | 书号 |
| BookName | varchar(50) | 50 | 否 |  | 书名 |
| Author | varchar(50) | 50 | 否 |  | 作者 |
| Publisherment | varchar(50) | 50 | 否 |  | 出版社 |
| PublishTime | datetime |  | 否 |  | 出版日期 |
| BookFile | varchar(50) | 50 | 否 | 历史，科学，文学 | 图书种类 |
| BookState | varchar(50) | 50 | 否 | 借出，在馆 | 图书状态 |

表4.借阅图书信息表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 数据类型 | 最大长度 | 允许空 | 约束控制 | 备注 |
| RecordNo | int |  | 否 | PRI | 借阅号 |
| StuNo | int |  | 否 | MUL | 学号 |
| BookNo | int |  | 否 | MUL | 书号 |
| RecordTime | datetime |  | 否 |  | 借阅时间 |
| RecordNum | int |  | 否 |  | 借阅数目 |
| ReReturnTime | datetime |  | 否 |  | 借阅时间 |

表5.预约图书信息表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 数据类型 | 最大长度 | 允许空 | 约束控制 | 备注 |
| BookNo | int |  | 否 | MUL | 书号 |
| StuNo | int |  | 否 | MUL | 学号 |
| OrderReportNo | int |  | 否 | PRI | 预约号 |
| OrderReportTime | datetime |  | 否 |  | 预约时间 |
| OrderReportNum | int |  | 否 |  | 预约数目 |

表6.还书信息表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 数据类型 | 最大长度 | 允许空 | 约束控制 | 备注 |
| BookNo | int |  | 否 | MUL | 书号 |
| StuNo | int |  | 否 | MUL | 学号 |
| ReturnNo | int |  | 否 | PRI | 还书号 |
| ReturnTime | datetime |  | 否 |  | 还书时间 |

**7.2.6 数据库实现**

**数据库创建：**

Create Database Library

**创建学生用户基本信息表：**

Create table student(

StuNo int primary key,

StuName varchar(50) not null,

StuSex varchar(20),

Class varchar(50) not null,

Department varchar(50) not null

);

**创建管理员基本信息表：**

Create table Manager(

MgNo int primary key,

Phone varchar(50) not null,

Password varchar(50) not null

)

**创建图书基本信息表：**

Create table book(

BookNo int primary key,

BookName varchar(50) not null,

Author varchar(50) not null,

Publisherment varchar(50) not null,

PublishTime varchar(50) not null,

BookFile varchar(50),

BookState varchar(50) not null

);

**创建借阅图书信息表：**

Create table record(

RecordNo int primary key,

StuNo int,

BookNo int,

ReturnTime datetime not null,

RecordNum int not null,

ReReturnTime datetime not null,

constraint Srecordforeignkey foreign key(StuNo) references student(StuNo),

constraint Brecordforeignkey foreign key(BookNo) references book(BookNo)

);

**创建预约图书信息表：**

Create table orderreport(

OrderReportNo int primary key,

StuNo int,

BookNo int,

OrderReportTime datetime not null,

OrderReportNum int not null,

constraint Sorderreportforeignkey foreign key(StuNo) references student(StuNo),

constraint Borderreportforeignkey foreign key(BookNo) references book(BookNo)

);

**创建还书信息表：**

Create table bookreturn(

ReturnNo int primary key,

StuNo int,

BookNo int,

ReturnTime datetime not null,

constraint Sbookreturnforeignkey foreign key(StuNo) references student(StuNo),

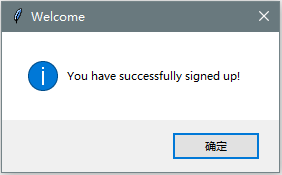
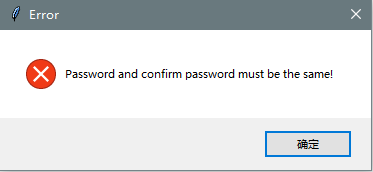
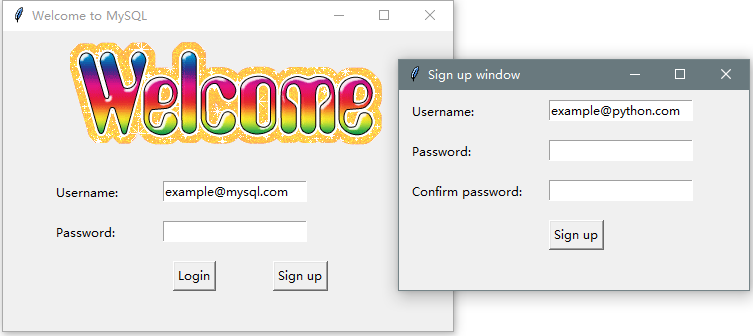
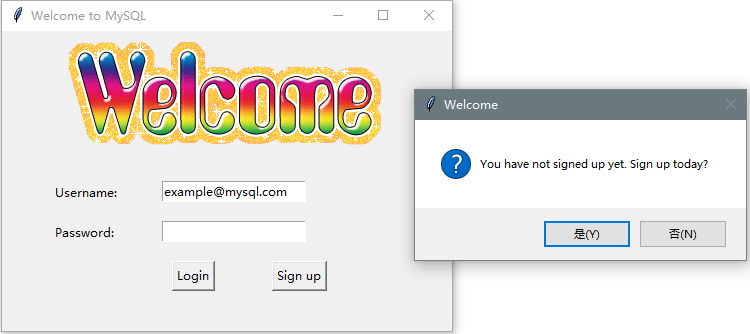
constraint Bbookreturnforeignkey foreign key(BookNo) references book(BookNo)

);

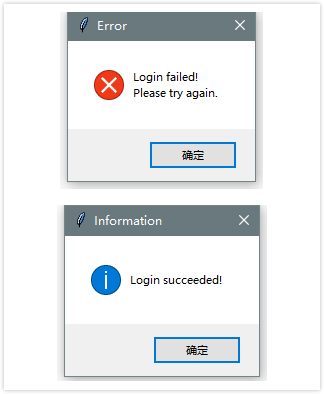
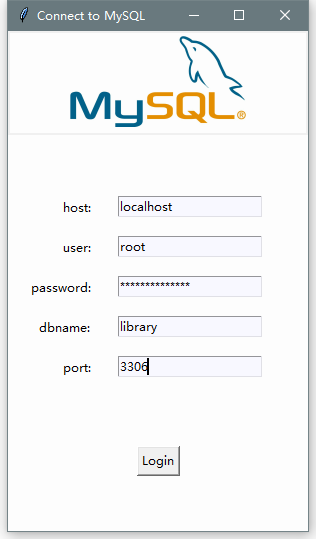
1. **实验数据及结果分析：**

用Python编写图形界面程序，实现了软件登录界面和用户管理的功能，并利用MySQL API，实现了连接数据库和数据库的增删查改的功能（代码见**十二、附录**）

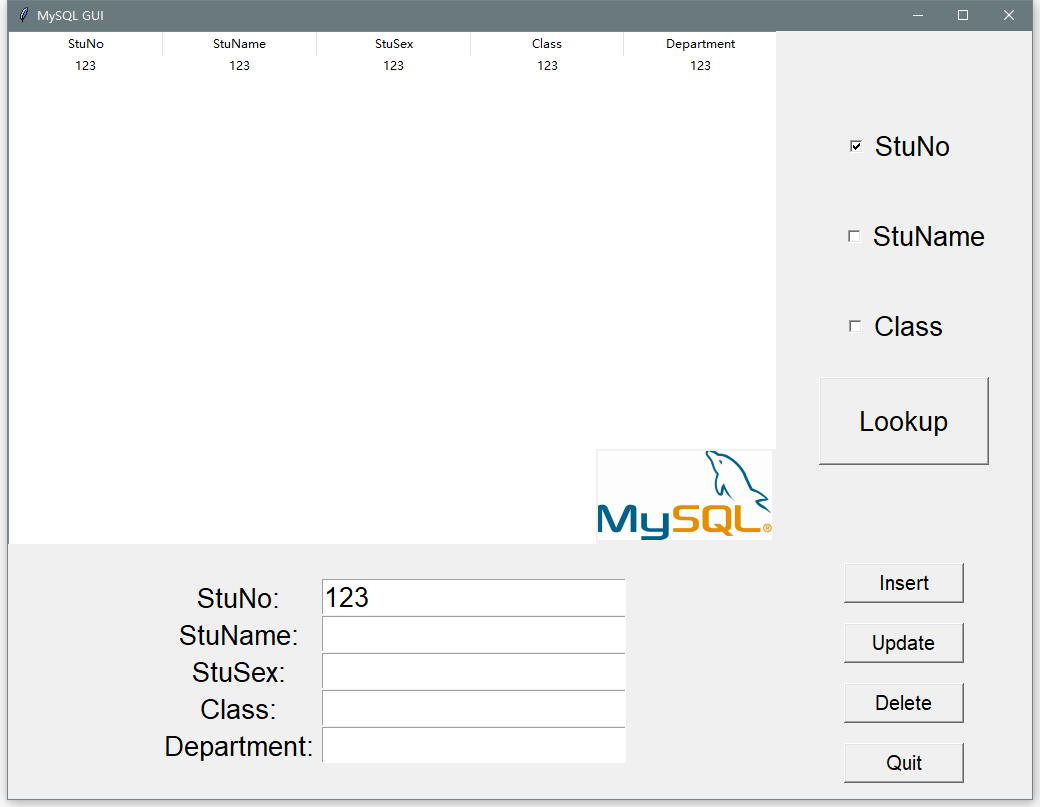




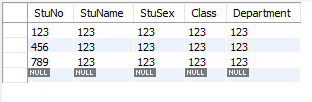
如上面的截图是GUI的第一层窗口，是软件的登录界面（与mysql用户无关），该登录界面可以实现，创建用户，登录用户，并可以把用户信息保存到本地文件，实现了一个简单的用户信息数据库。



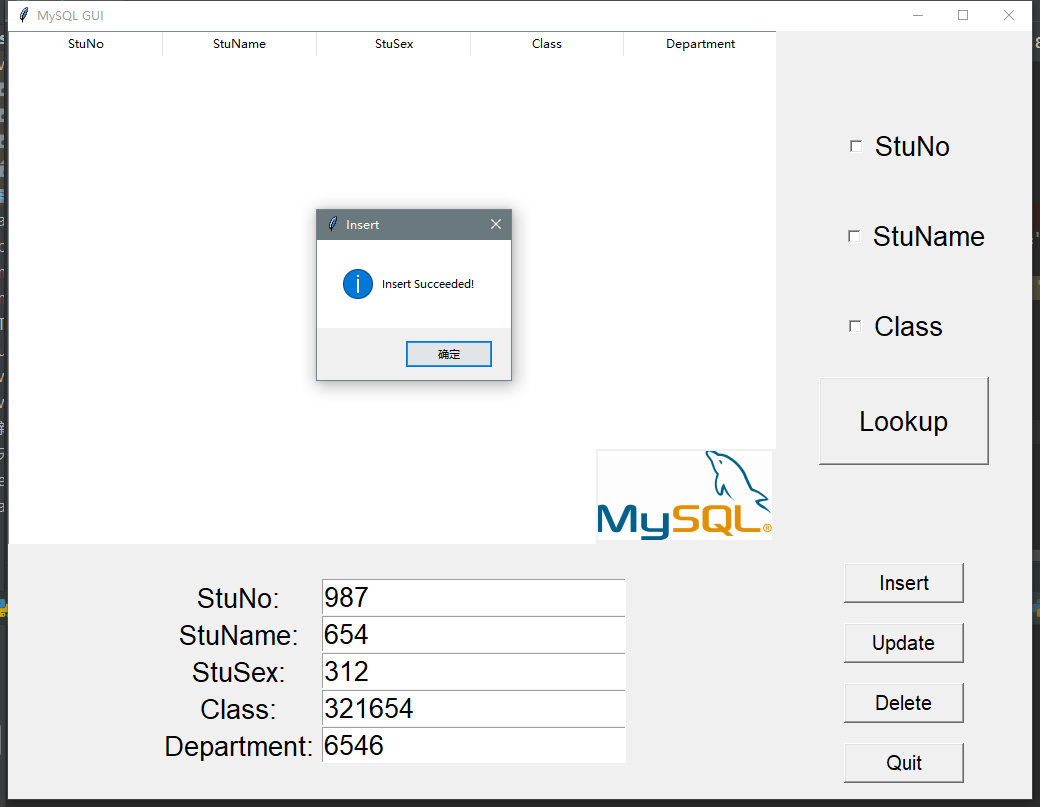
上面图片是GUI的第二层窗口，用户在上一层的软件登录界面输入正确的账号和密码信息后登录，之后会跳转到该界面，这个界面是用户连接软件和MySQL数据库的，通过在上面输入host、user、password、dbname、post等信息，可以连接系统本地的MySQL账户，并可进行后续操作。



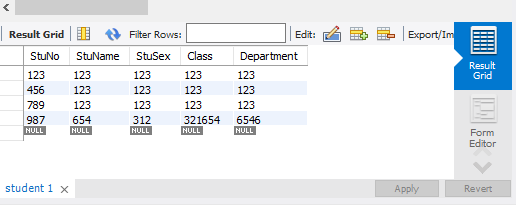
上图是本软件的第三个界面，也是软件的主界面，软件基本分为四个区域，左上角是信息区，用于展示软件操作得到的数据表格信息，右上角是查询区，通过选择要查询的属性，配合左下角的输入区输入的信息，可以进行数据库的查询工作。右下角是插入、更新、删除三个按钮，配合左下角的输入区可实现对应增删改的操作。最后一个按钮的功能是退出程序。



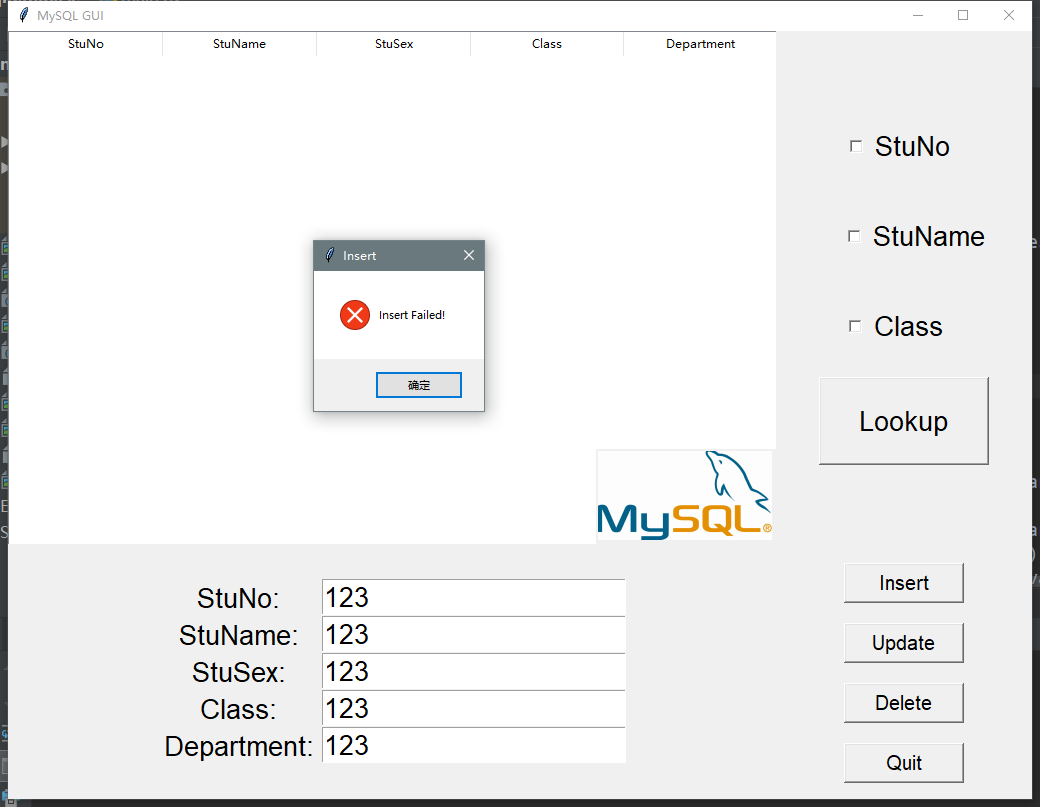
首先我们可以在MySQL Workbench中查看到当前数据库有如上图的三个元组信息。



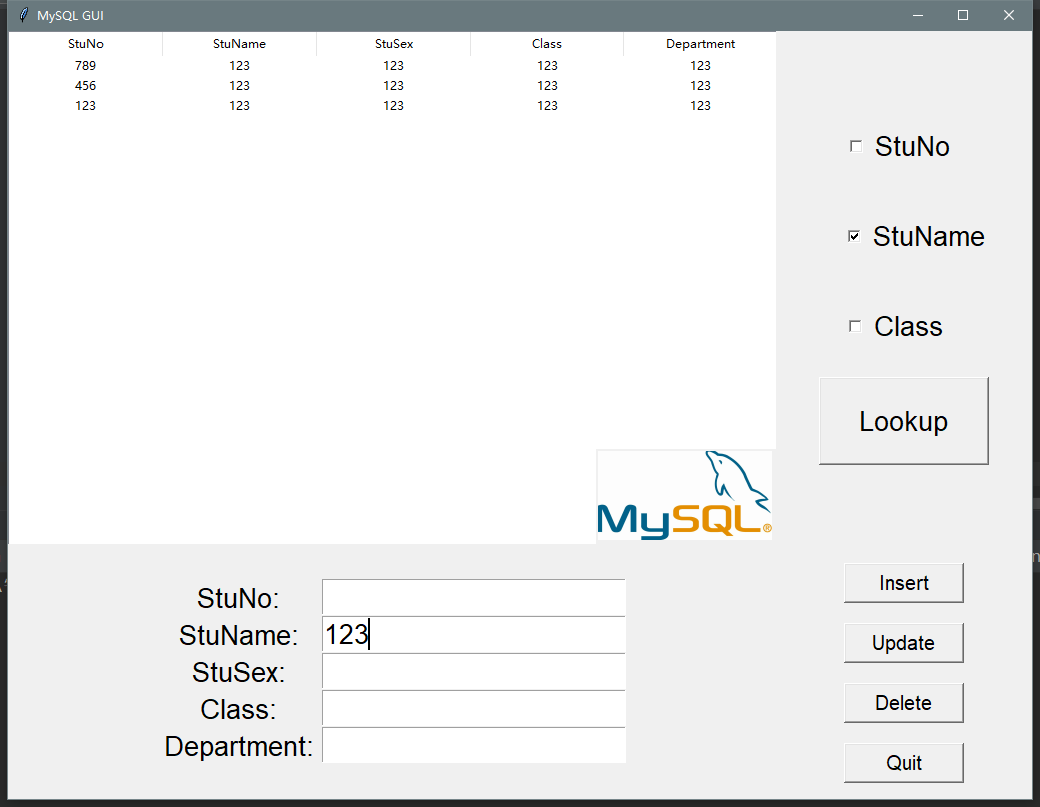
此时我们再软件中进行插入操作，插入如上图所示数据，界面提示插入成功，接着在MySQL Workbench中检验结果，发现数据已被插入：



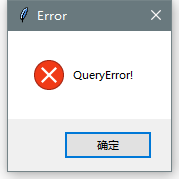
此时，如果要插入一组已经存在的数据，由于数据库的表中存在的约束，会出现插入错误提示，如下图所示：



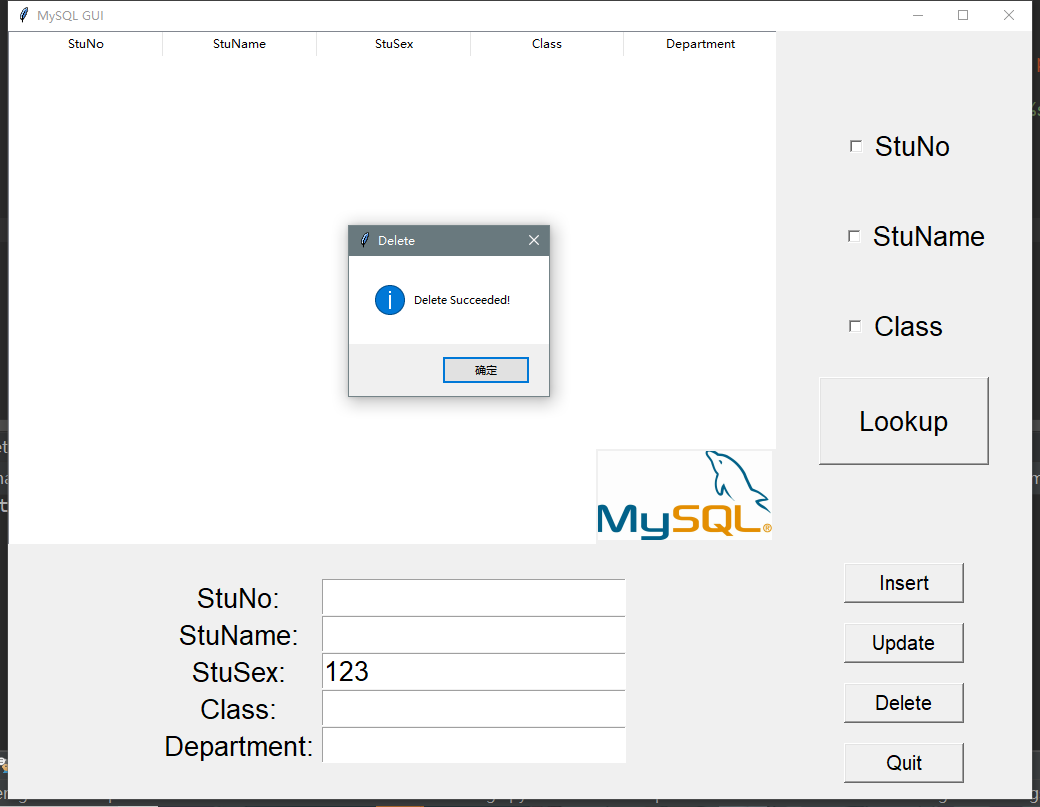
此时进行查询操作，查询StuName为123的数据，在软件中得到如下结果，显示了所有查询到的结果：

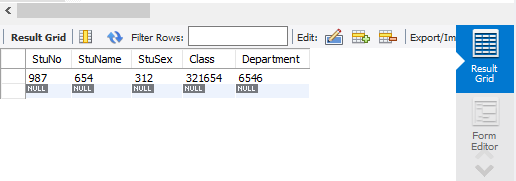


如果使用违规的查询操作会得到如下图的错误提示：

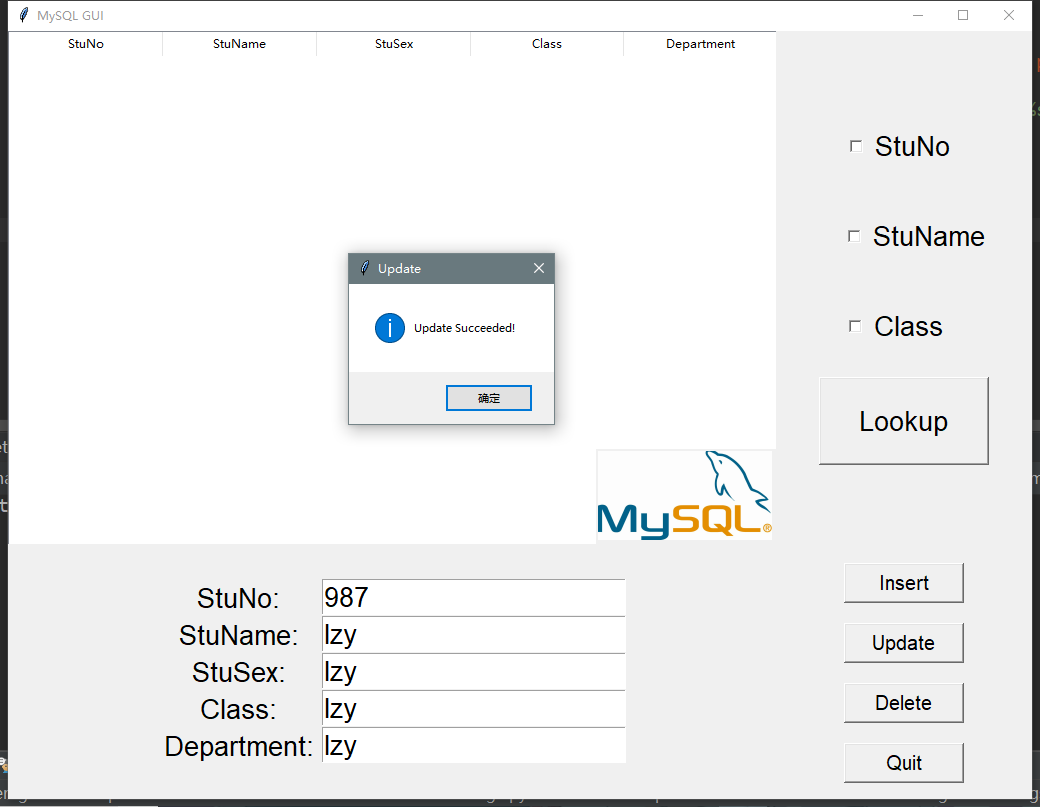


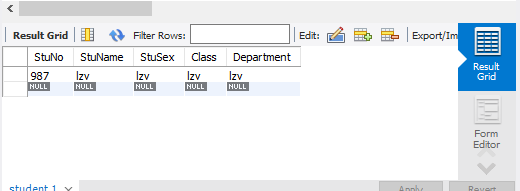
下面进行删除操作，如下图所示，删除StuSex=123的元组，并在MySQL Workbench中检验结果，正确：



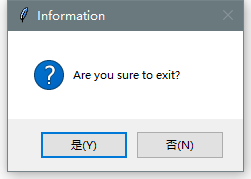


接着对StuNo=987的元组进行更新操作，将其StuName，StuSex，Class，Department属性都改为lzy，并在MySQL Workbench中检验结果，正确：





最后是软件的退出功能：



**九、实验结论：**

数据库的设计需要根据数据库理论进行设计，必须满足数据库理论中的一些条件，才能使数据库安全、稳定。并且在数据库的设计中要尽可能做到优化结构。

**十、总结及心得体会：**

通过本次实验，我初步掌握了数据库设计的相关知识，并通过实践，实际设计出了一个简单的小型数据库，这让我对数据库设计的体会更深了。比如为了避免数据容易和增查删改的操作异常，要通过模式分解来解决。并且要通过范式的规定，来使自己的数据库达到范式的要求。

通过这次实验，我还加深了对编程开发的理解。学习了Python中的tkinter图形界面模块，和Python中用于连接MySQL数据库的pymysql模块，进一步加深了对软件开发中前端和后台的理解，并且进一步体会了SQL作为一种结构化查询语言，与其他编程语言的方便的接口功能。

**十一、对本实验过程及方法、手段的改进建议：**

可以尝试让学生多使用几款不同的DBMS，体会不同DBMS在数据库操作和与编程语言对接上的差异。

**十二、附录**

MySQL图形化界面的python程序代码：

# MySQL GUI by Tera

#!/usr/bin/python

# -\*- coding: UTF-8 -\*-

import tkinter as tk

from tkinter import messagebox # import this to fix messagebox error

import pickle

import pymysql

from tkinter import ttk

# part1: software login ###################

# software welcome window

window = tk.Tk()

window.title('Welcome to MySQL')

window.geometry('450x300')

window.resizable(width = False, height = False)

# welcome image

canvas = tk.Canvas(window, height=200, width=500)

image\_file = tk.PhotoImage(file='welcome.gif')

image = canvas.create\_image(225, 0, anchor='n', image=image\_file)

canvas.pack(side='top')

# user information

tk.Label(window, text='Username: ').place(x=50, y= 150)

tk.Label(window, text='Password: ').place(x=50, y= 190)

var\_usr\_name = tk.StringVar()

var\_usr\_name.set('example@mysql.com')

entry\_usr\_name = tk.Entry(window, textvariable=var\_usr\_name)

entry\_usr\_name.place(x=160, y=150)

var\_usr\_pwd = tk.StringVar()

entry\_usr\_pwd = tk.Entry(window, textvariable=var\_usr\_pwd, show='\*')

entry\_usr\_pwd.place(x=160, y=190)

def usr\_login():

usr\_name = var\_usr\_name.get()

usr\_pwd = var\_usr\_pwd.get()

try:

with open('usrs\_info.pickle', 'rb') as usr\_file:

usrs\_info = pickle.load(usr\_file)

except FileNotFoundError:

with open('usrs\_info.pickle', 'wb') as usr\_file:

usrs\_info = {'admin': 'admin'}

pickle.dump(usrs\_info, usr\_file)

if usr\_name in usrs\_info:

if usr\_pwd == usrs\_info[usr\_name]:

tk.messagebox.showinfo(title='Welcome', message='How are you? ' + usr\_name)

window.destroy()

else:

tk.messagebox.showerror(message='Error, your password is wrong, try again.')

else:

is\_sign\_up = tk.messagebox.askyesno('Welcome',

'You have not signed up yet. Sign up today?')

if is\_sign\_up:

usr\_sign\_up()

def usr\_sign\_up():

def sign\_to\_mysql():

np = new\_pwd.get()

npf = new\_pwd\_confirm.get()

nn = new\_name.get()

with open('usrs\_info.pickle', 'rb') as usr\_file:

exist\_usr\_info = pickle.load(usr\_file)

if np != npf:

tk.messagebox.showerror('Error', 'Password and confirm password must be the same!')

elif nn in exist\_usr\_info:

tk.messagebox.showerror('Error', 'The user has already signed up!')

else:

exist\_usr\_info[nn] = np

with open('usrs\_info.pickle', 'wb') as usr\_file:

pickle.dump(exist\_usr\_info, usr\_file)

tk.messagebox.showinfo('Welcome', 'You have successfully signed up!')

window\_sign\_up.destroy()

window\_sign\_up = tk.Toplevel(window)

window\_sign\_up.geometry('350x200')

window\_sign\_up.title('Sign up window')

new\_name = tk.StringVar()

new\_name.set('example@python.com')

tk.Label(window\_sign\_up, text='Username: ').place(x=10, y= 10)

entry\_new\_name = tk.Entry(window\_sign\_up, textvariable=new\_name)

entry\_new\_name.place(x=150, y=10)

new\_pwd = tk.StringVar()

tk.Label(window\_sign\_up, text='Password: ').place(x=10, y=50)

entry\_usr\_pwd = tk.Entry(window\_sign\_up, textvariable=new\_pwd, show='\*')

entry\_usr\_pwd.place(x=150, y=50)

new\_pwd\_confirm = tk.StringVar()

tk.Label(window\_sign\_up, text='Confirm password: ').place(x=10, y= 90)

entry\_usr\_pwd\_confirm = tk.Entry(window\_sign\_up, textvariable=new\_pwd\_confirm, show='\*')

entry\_usr\_pwd\_confirm.place(x=150, y=90)

btn\_comfirm\_sign\_up = tk.Button(window\_sign\_up, text='Sign up', command=sign\_to\_mysql)

btn\_comfirm\_sign\_up.place(x=150, y=130)

# login and sign up button

btn\_login = tk.Button(window, text='Login', command=usr\_login)

btn\_login.place(x=170, y=230)

btn\_sign\_up = tk.Button(window, text='Sign up', command=usr\_sign\_up)

btn\_sign\_up.place(x=270, y=230)

window.mainloop()

# part2: mysql login #####################

# login window

window1 = tk.Tk()

window1.title('Connect to MySQL')

window1.geometry('300x500')

window1.config(bg='#FDFDFD')

window1.resizable(width = False, height = False)

# mysql image

c1 = tk.Canvas(window1, height=100, width=300, bg='#FDFDFD')

image\_file1 = tk.PhotoImage(file='mysql.gif')

image1 = c1.create\_image(150, 50, anchor='center', image=image\_file1)

c1.pack(side='top')

# mysql user information

dy = 15

tk.Label(window1, text=' host: ', bg='#FDFDFD').place(x=20, y=150+dy)

tk.Label(window1, text=' user: ', bg='#FDFDFD').place(x=20, y=190+dy)

tk.Label(window1, text='password: ', bg='#FDFDFD').place(x=20, y=230+dy)

tk.Label(window1, text=' dbname: ', bg='#FDFDFD').place(x=20, y=270+dy)

tk.Label(window1, text=' port: ', bg='#FDFDFD').place(x=20, y=310+dy)

var\_usr\_host1 = tk.StringVar()

var\_usr\_name1 = tk.StringVar()

var\_usr\_pwd1 = tk.StringVar()

var\_usr\_dbn1 = tk.StringVar()

var\_usr\_port1 = tk.StringVar()

entry\_usr\_host1 = tk.Entry(window1, textvariable=var\_usr\_host1, bg='GhostWhite')

entry\_usr\_name1 = tk.Entry(window1, textvariable=var\_usr\_name1, bg='GhostWhite')

entry\_usr\_pwd1 = tk.Entry(window1, textvariable=var\_usr\_pwd1, show='\*', bg='GhostWhite')

entry\_usr\_dbn1 = tk.Entry(window1, textvariable=var\_usr\_dbn1, bg='GhostWhite')

entry\_usr\_port1 = tk.Entry(window1, textvariable=var\_usr\_port1, bg='GhostWhite')

entry\_usr\_host1.place(x=110, y=150+dy)

entry\_usr\_name1.place(x=110, y=190+dy)

entry\_usr\_pwd1.place(x=110, y=230+dy)

entry\_usr\_dbn1.place(x=110, y=270+dy)

entry\_usr\_port1.place(x=110, y=310+dy)

# mysql log in button

def mysqllog():

vhost = var\_usr\_host1.get()

vname = var\_usr\_name1.get()

vpwd = var\_usr\_pwd1.get()

vdbn = var\_usr\_dbn1.get()

vport = var\_usr\_port1.get()

try:

conn = pymysql.Connect(

host=vhost,

port=int(vport),

user=vname,

passwd=vpwd,

db=vdbn,

charset='utf8'

)

except:

tk.messagebox.showerror(title='Error', message='Login failed!\nPlease try again.')

else:

tk.messagebox.showinfo(title='Information', message='Login succeeded!')

conn.close() # close the connection

window1.destroy()

btn\_mysqllog = tk.Button(window1, text='Login', command=mysqllog)

btn\_mysqllog.place(x=150, y=430, anchor='center')

window1.mainloop()

# part3: main window ############################

# window2

window2 = tk.Tk()

window2.title('MySQL GUI')

window2.geometry('1024x768')

# frames

frm1=tk.Frame(window2, width=768, height=512)

frm1.place(x=0, y=0, anchor='nw')

frm2=tk.Frame(window2, width=256, height=512)

frm2.place(x=768, y=0, anchor='nw')

frm3=tk.Frame(window2, width=768, height=256)

frm3.place(x=0, y=513, anchor='nw')

frm4=tk.Frame(window2, width=256, height=256)

frm4.place(x=768, y=512, anchor='nw')

# ctest1 = tk.Canvas(frm1, bg='blue', width=768, height=512).pack()

# ctest2 = tk.Canvas(frm2, bg='green', width=256, height=512).pack()

# ctest3 = tk.Canvas(frm3, bg='yellow', width=768, height=256).pack()

# ctest4 = tk.Canvas(frm4, bg='red', width=256, height=256).pack()

# frame 1

tree = ttk.Treeview(frm1, height=512, show="headings")

tree["columns"] = ("StuNo", "StuName", "StuSex", "Class", "Department")

tree.column("StuNo", width=154, anchor="center")

tree.column("StuName", width=153, anchor="center")

tree.column("StuSex", width=154, anchor="center")

tree.column("Class", width=153, anchor="center")

tree.column("Department", width=154, anchor="center")

tree.heading("StuNo", text="StuNo")

tree.heading("StuName", text="StuName")

tree.heading("StuSex", text="StuSex")

tree.heading("Class", text="Class")

tree.heading("Department", text="Department")

tree.pack()

def delete\_tree():

items = tree.get\_children()

for item in items:

tree.delete(item)

# frame2

c\_background = tk.Canvas(window2, width=176, height=91)

image\_file12 = tk.PhotoImage(file='mysql.gif')

image12 = c\_background.create\_image(0, 0, anchor='nw', image=image\_file12)

c\_background.place(x=768, y=513, anchor='se')

btn\_change=tk.Button(frm2, text='Books', width=10, height=2).place(x=43, y=100, anchor='nw')

var\_r\_stuno = tk.IntVar()

var\_r\_stuname = tk.IntVar()

var\_r\_class = tk.IntVar()

def lookup():

no = v\_fr3\_stuno.get()

name = v\_fr3\_stuname.get()

stuclass = v\_fr3\_class.get()

a1 = var\_r\_stuno.get()

a2 = var\_r\_stuname.get()

a3 = var\_r\_class.get()

if a1 == 1 and a2 == 0 and a3 == 0:

db = pymysql.connect(host="localhost", user="root", password="lzy19980324xyz", db="library", port=3306)

cur = db.cursor()

sql = "select \* from student where StuNo='%s'" % no

try:

cur.execute(sql)

results = cur.fetchall()

for row in results:

result\_stuno = row[0]

result\_stuname = row[1]

result\_stusex = row[2]

result\_class = row[3]

relust\_department = row[4]

delete\_tree()

tree.insert("", 0, text='data1', values=(result\_stuno, result\_stuname, result\_stusex, result\_class, relust\_department))

except Exception as e:

raise e

finally:

db.close()

if a1 == 0 and a2 == 1 and a3 == 0:

db = pymysql.connect(host="localhost", user="root", password="lzy19980324xyz", db="library", port=3306)

cur = db.cursor()

sql = "select \* from student where StuName='%s'" % name

try:

cur.execute(sql)

results = cur.fetchall()

for row in results:

result\_stuno = row[0]

result\_stuname = row[1]

result\_stusex = row[2]

result\_class = row[3]

relust\_department = row[4]

delete\_tree()

tree.insert("", 0, text='data1',

values=(result\_stuno, result\_stuname, result\_stusex, result\_class, relust\_department))

except Exception as e:

raise e

finally:

db.close()

if a1 == 0 and a2 == 1 and a3 == 0:

db = pymysql.connect(host="localhost", user="root", password="lzy19980324xyz", db="library", port=3306)

cur = db.cursor()

sql = "select \* from student where Class='%s'" % stuclass

try:

cur.execute(sql)

results = cur.fetchall()

for row in results:

result\_stuno = row[0]

result\_stuname = row[1]

result\_stusex = row[2]

result\_class = row[3]

relust\_department = row[4]

delete\_tree()

tree.insert("", 0, text='data1',

values=(result\_stuno, result\_stuname, result\_stusex, result\_class, relust\_department))

except Exception as e:

raise e

finally:

db.close()

else:

tk.messagebox.showerror(title='Error',message='QueryError!')

rbtn\_stuno = tk.Checkbutton(frm2, width=10, height=2, text='StuNo ', variable=var\_r\_stuno, onvalue=1, offvalue=0

, font=('Arial', 20)).place(x=43, y=76, anchor='nw')

rbtn\_stuname = tk.Checkbutton(frm2, width=10, height=2, text='StuName', variable=var\_r\_stuname, onvalue=1, offvalue=0

, font=('Arial', 20)).place(x=43, y=166, anchor='nw')

rbtn\_class = tk.Checkbutton(frm2, width=10, height=2, text='Class ', variable=var\_r\_class, onvalue=1, offvalue=0

, font=('Arial', 20)).place(x=43, y=256, anchor='nw')

btn\_select = tk.Button(frm2, width=10, height=2, text='Lookup'

, font=('Arial', 20), command=lookup).place(x=43, y=346, anchor='nw')

# frame3

idx = 70

l\_fr3\_stuno = tk.Label(frm3, text='StuNo:', width=10, height=1, font=('Arial', 20)).place(x=384-idx, y=35, anchor='ne') # 160\*37

l\_fr3\_stuname= tk.Label(frm3, text='StuName:', width=10, height=1, font=('Arial', 20)).place(x=384-idx, y=72, anchor='ne')

l\_fr3\_stusex= tk.Label(frm3, text='StuSex:', width=10, height=1, font=('Arial', 20)).place(x=384-idx, y=109, anchor='ne')

l\_fr3\_class= tk.Label(frm3, text='Class:', width=10, height=1, font=('Arial', 20)).place(x=384-idx, y=146, anchor='ne')

l\_fr3\_department= tk.Label(frm3, text='Department:', width=10, height=1, font=('Arial', 20)).place(x=384-idx, y=183, anchor='ne')

v\_fr3\_stuno = tk.StringVar()

v\_fr3\_stuname = tk.StringVar()

v\_fr3\_stusex = tk.StringVar()

v\_fr3\_class = tk.StringVar()

v\_fr3\_department = tk.StringVar()

e\_fr3\_stuno = tk.Entry(frm3, font=('Arial', 20), textvariable=v\_fr3\_stuno).place(x=384-idx, y=35, anchor='nw')

e\_fr3\_stuname = tk.Entry(frm3, font=('Arial', 20), textvariable=v\_fr3\_stuname).place(x=384-idx, y=72, anchor='nw')

e\_fr3\_stusex = tk.Entry(frm3, font=('Arial', 20), textvariable=v\_fr3\_stusex).place(x=384-idx, y=109, anchor='nw')

e\_fr3\_class = tk.Entry(frm3, font=('Arial', 20), textvariable=v\_fr3\_class).place(x=384-idx, y=146, anchor='nw')

e\_fr3\_department = tk.Entry(frm3, font=('Arial', 20), textvariable=v\_fr3\_department).place(x=384-idx, y=183, anchor='nw')

# frame4

def set\_empty():

v\_fr3\_stuno.set('')

v\_fr3\_stuname.set('')

v\_fr3\_stusex.set('')

v\_fr3\_class.set('')

v\_fr3\_department.set('')

def insert():

no = v\_fr3\_stuno.get()

name = v\_fr3\_stuname.get()

sex = v\_fr3\_stusex.get()

stuclass = v\_fr3\_class.get()

department = v\_fr3\_department.get()

db = pymysql.connect(host="localhost", user="root",password="lzy19980324xyz", db="library", port=3306)

cur = db.cursor()

sql\_insert = r"insert into student(StuNo,StuName,StuSex,Class,Department) values('%s','%s','%s','%s','%s')" % (no, name, sex, stuclass, department)

try:

cur.execute(sql\_insert)

db.commit()

tk.messagebox.showinfo(title='Insert', message='Insert Succeeded!')

set\_empty()

except:

db.rollback()

tk.messagebox.showerror(title='Insert', message='Insert Failed!')

set\_empty()

finally:

db.close()

def update():

no = v\_fr3\_stuno.get()

name = v\_fr3\_stuname.get()

sex = v\_fr3\_stusex.get()

stuclass = v\_fr3\_class.get()

department = v\_fr3\_department.get()

# delete

db = pymysql.connect(host="localhost", user="root", password="lzy19980324xyz", db="library", port=3306)

cur = db.cursor()

sql\_delete = "delete from student where StuNo='%s'" % no

try:

cur.execute(sql\_delete)

db.commit()

except:

db.rollback()

tk.messagebox.showerror(title='Update', message='Update Failed!')

finally:

db.close()

# insert

db = pymysql.connect(host="localhost", user="root", password="lzy19980324xyz", db="library", port=3306)

cur = db.cursor()

sql\_insert = r"insert into student(StuNo,StuName,StuSex,Class,Department) values('%s','%s','%s','%s','%s')" % (

no, name, sex, stuclass, department)

try:

cur.execute(sql\_insert)

db.commit()

tk.messagebox.showinfo(title='Update', message='Update Succeeded!')

set\_empty()

except:

db.rollback()

tk.messagebox.showerror(title='Update', message='Update Failed!')

set\_empty()

finally:

db.close()

def delete():

no = v\_fr3\_stuno.get()

name = v\_fr3\_stuname.get()

sex = v\_fr3\_stusex.get()

stuclass = v\_fr3\_class.get()

department = v\_fr3\_department.get()

db = pymysql.connect(host="localhost", user="root", password="lzy19980324xyz", db="library", port=3306)

cur = db.cursor()

sql\_delete = "delete from student where StuNo='%s' or StuName='%s' or StuSex='%s' or Class='%s' or Department='%s'" % (no, name, sex, stuclass, department)

try:

cur.execute(sql\_delete)

db.commit()

delete\_tree()

tk.messagebox.showinfo(title='Delete', message='Delete Succeeded!')

set\_empty()

except:

db.rollback()

tk.messagebox.showerror(title='Delete', message='Delete Failed!')

set\_empty()

finally:

db.close()

def quit():

a = tk.messagebox.askyesno(title='Information', message='Are you sure to exit?')

if a:

window2.quit()

idy = 12

btn\_insert = tk.Button(frm4, width=10, height=1, text='Insert', font=('Arial', 15)

, command=insert).place(x=68, y=8+idy, anchor='nw') # 120\*60

btn\_update = tk.Button(frm4, width=10, height=1, text='Update', font=('Arial', 15)

, command=update).place(x=68, y=68+idy, anchor='nw')

btn\_delete = tk.Button(frm4, width=10, height=1, text='Delete', font=('Arial', 15)

, command=delete).place(x=68, y=128+idy, anchor='nw')

btn\_quit = tk.Button(frm4, width=10, height=1, text='Quit', font=('Arial', 15)

, command=quit).place(x=68, y=188+idy, anchor='nw')

window2.mainloop()