上海理工大学继续教育学院

2020学年第二学期

课程名称：《Python程序设计》

题目 图书信息管理系统

姓名\_\_\_\_\_\_\_\_\_\_\_\_

学号\_\_\_\_\_\_\_\_\_\_\_\_

专业\_\_\_\_\_\_\_\_\_\_\_\_

层次\_\_\_\_\_\_\_\_\_\_\_\_

1. **项目简介**

该系统主要建立一个基于基本模式的图书管理系统，实现图书信息登记、图书信息删除、图书信息修改、图书信息查询等功能。

1. **项目流程**

该项目模拟了图书信息在图书馆里的登记、删除、修改、查询等基本功能，在此基础上模拟了图书的租借功能、记录图书租借情况和租借时间。用户首先进入功能选择界面。登记图书功能中，需要用户手动输入图书编号、图书名称、图书作者、图书的出版社、图书的租借情况等。输入的信息会在程序中转化成SQL语句，点击创建图书后同步到数据库library.db中。

删除图书则充分利用了图形界面的直观性、尽最大努力让用户用更少的操作获得更优秀的体验、用户在该功能中，图书以表格形式列在界面中、用户可以直接通过鼠标选中表格中的目标图书，经过程序调整，已经设置成为选择一行模式，可以有效减少误解产生。用户选择删除一行操作后，可以点击下方删除按钮、此处的删除功能，以逻辑主键图书编号为索引、自动生成删除SQL语句，删除数据库中的对应图书信息。为了加快程序的执行速度、提高抗鲁棒性、以及为了给用户留下直观的已删除数据印象、防止误操，删除操作作用在数据库中，而不直接修改表格中的目标数据，点击刷新按钮后可以将数据库信息同步回表格中，此时方可看到表格中数据变化情况。

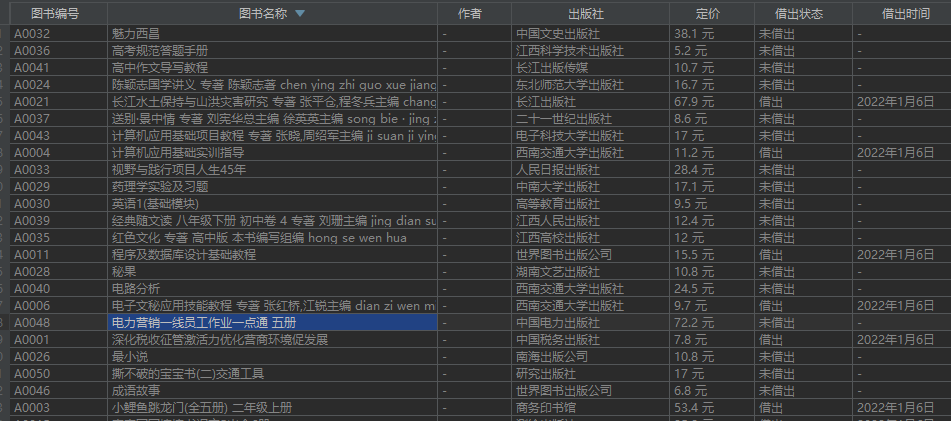
修改图书综合了创建图书和删除图书功能、图书管理系统的添加修改图书功能，发生了质变，功能可以根据使用者的目标不同而起到不同的作用。图书管理系统的添加修改图书功能的出现仍然代表了一定的意义。用户所选选中表格中的数据，可以自动同步到下方的空格中去，既可以简单直接修改部分、也可以通篇全部修改、也可以当做图书租借系统来使用，该功能是程序的核心亮点。

查询功能利用了SQL的select语句进行查询，为了方便用户使用，该查询不设置搜索限制，用户在搜索栏里输入的信息会同时和编号、图书名、作者名、出版社、租借情况、租借时间等所有图书信息比对。

注：图书信息节选自中国信息网2022年1月6日的折扣书单，如有信息错误，请以官方为准。

1. **实现原理**

程序语言为python，以pyqt5第三方库作为GUI库。选用sqlite3作为数据库。数据库中设置图书编号、图书名称、图书作者、图书的出版社、图书价格、图书的租借情况、图书租借时间七项数据库字段。



1. **界面设计**

图书添加



图书删除



图书修改



图书查询



1. 代码

from PyQt5.QtWidgets import QWidget, QHBoxLayout, QTableWidget, QPushButton, QApplication, QVBoxLayout, QTableWidgetItem, QCheckBox, QAbstractItemView, QHeaderView, QLabel, QFrame # 使用的界面库为PYQT5

from Hello import Ui\_Dialog # 导入ui件中的类#ss是ui转换后的py文件，Ui\_Form是文件中的类名

import sys # 创建和关闭进程

import sqlite3 # python内置数据库

class win(QWidget, Ui\_Dialog):

def \_\_init\_\_(self):

super().\_\_init\_\_()

# self.resize(300,300)

self.setupUi(self) # 执行类中的setupUi函数

self.exe = " 执行: "

self.output = " 输出: "

self.conn = sqlite3.connect("library.db")

self.cursor = self.conn.cursor()

self.tableName = "一些图书信息"

print("数据库打开成功")

self.pushButton.clicked.connect(self.add\_book)

self.pushButton\_2.clicked.connect(self.Query)

self.pushButton\_3.clicked.connect(self.del\_book)

self.pushButton\_4.clicked.connect(self.change\_book)

self.pushButton\_5.clicked.connect(self.ReFalsh)

self.pushButton\_6.clicked.connect(self.ReFalsh)

self.tableWidget\_3.clicked.connect(self.chang\_line\_edit)

def \_\_del\_\_(self):

print("释放数据库实例")

self.cursor.close()

self.conn.close()

# 增删改查等功能

def \_\_Insert(self, fieldNames, fieldValues):

"""

插入数据

:param fieldNames: 字段list

:param fieldValues: 值list

"""

# 通过fieldNames解析出字段名

try:

names = "" # 字段名，用于插入数据

nameTypes = "" # 字段名及字段类型，用于创建表

for index in range(fieldNames.\_\_len\_\_()):

if index != fieldNames.\_\_len\_\_() - 1:

names += fieldNames[index] + ","

nameTypes += fieldNames[index] + " text,"

else:

names += fieldNames[index]

nameTypes += fieldNames[index] + " text"

# 通过fieldValues解析出字段对应的值

values = ""

for index in range(fieldValues.\_\_len\_\_()):

cell\_value = str((fieldValues[index]))

if isinstance(fieldValues[index], float):

cell\_value = str((int)(fieldValues[index]))

if index != fieldValues.\_\_len\_\_() - 1:

values += "\'" + cell\_value + "\',"

else:

values += "\'" + cell\_value + "\'"

print(values)

# 将fieldValues解析出的值插入数据库

sql = 'insert into %s(%s) values(%s)' % (self.tableName, names, values)

print(self.exe + sql)

try:

self.cursor.execute(sql)

except:

print("插入失败，检查sql执行情况")

self.conn.commit()

except:

print("插入失败，检查\_\_Insert函数")

def add\_book(self):

try:

fieldNames = ['图书编号', '图书名称', '作者', '出版社', '定价', '借出状态', '借出时间']

borrowed = "借出" if self.checkBox.isChecked() else "未借出"

time = self.lineEdit\_7.text() if self.checkBox.isChecked() else "-"

fieldValues = [self.lineEdit.text(), # 编号

self.lineEdit\_2.text(), # 名称

self.lineEdit\_3.text(), # 作者

self.lineEdit\_4.text(), # 出版社

self.lineEdit\_6.text(), # 定价

borrowed, # 借出状态

time # 借出时间

]

# print(fieldValues)

self.\_\_Insert(fieldNames, fieldValues) # 数据插入到数据库中

except:

print("插入失败，add\_book函数")

def del\_book(self):

tableName = "一些图书信息"

try:

row = self.tableWidget.currentRow() # 获取选中文本所在的行

print(self.tableWidget.item(row,0).text())

except:

print("检查tableWidgetd的选择功能")

try:

sql = 'DELETE FROM %s WHERE 图书编号 = %s ' % (tableName,self.tableWidget.item(row,0).text())

self.cursor.execute(sql)

print(self.exe + sql) # 删除目标信息

except:

print("检查删除sql语句执行情况")

self.conn.commit()

def chang\_line\_edit(self):

tableName = "一些图书信息"

try:

row = self.tableWidget\_3.currentRow() # 获取选中文本所在的行

self.lineEdit\_8.setText(self.tableWidget\_3.item(row, 0).text())

self.lineEdit\_9.setText(self.tableWidget\_3.item(row, 1).text())

self.lineEdit\_10.setText(self.tableWidget\_3.item(row, 2).text())

self.lineEdit\_11.setText(self.tableWidget\_3.item(row, 3).text())

self.lineEdit\_12.setText(self.tableWidget\_3.item(row, 4).text())

self.checkBox\_2.setChecked(False) if self.tableWidget\_3.item(row,5).text() == "未借出" else self.checkBox\_2.setChecked(True)

self.lineEdit\_13.setText(self.tableWidget\_3.item(row, 6).text())

#print(self.tableWidget\_3.item(row, 0).text())

except:

print("检查tableWidgetd\_3的选择功能")

def change\_book(self):

tableName = "一些图书信息"

try:

row = self.tableWidget\_3.currentRow() # 获取选中文本所在的行

print(self.tableWidget\_3.item(row,0).text())

except:

print("检查tableWidgetd的选择功能")

try:

sql = 'DELETE FROM %s WHERE 图书编号 = %s ' % (tableName,self.tableWidget\_3.item(row,0).text())

self.cursor.execute(sql)

#print(self.exe + sql) # 删除目标信息

except:

print("检查修改sql语句执行情况")

self.conn.commit()

try:

fieldNames = ['图书编号', '图书名称', '作者', '出版社', '定价', '借出状态', '借出时间']

borrowed = "借出" if self.checkBox\_2.isChecked() else "未借出"

time = self.lineEdit\_13.text() if self.checkBox.isChecked() else "-"

fieldValues = [self.lineEdit\_8.text(), # 编号

self.lineEdit\_9.text(), # 名称

self.lineEdit\_10.text(), # 作者

self.lineEdit\_11.text(), # 出版社

self.lineEdit\_12.text(), # 定价

borrowed, # 借出状态

time # 借出时间

]

# print(fieldValues)

self.\_\_Insert(fieldNames, fieldValues) # 数据插入到数据库中

except:

print("修改失败，change\_book函数")

def Query(self): #

"""

查找数据表中的数据

:param tableName:表名

"""

tableName = "一些图书信息"

x = str(self.lineEdit\_5.text())

try:

print("查询表 " + tableName)

sql = 'select \* from %s' % tableName

self.cursor.execute(sql)

if x != "":

sql = 'select \* from %s' % tableName

else:

sql = 'select \* from %s WHERE %s in 图书名称' % (tableName, x)

print(self.exe + sql)

except:

print("检查查询函数的sql语句执行情况")

results = self.cursor.fetchall() # 获取所有记录列表

cheat = []

if x != "":

for i in results:

for j in i:

if x in j or j in x:

cheat.append(i)

break

# print(cheat)

results = cheat

index = 0

self.tableWidget\_2.clearContents()

for i in range(0, self.tableWidget\_2.rowCount())[::-1]:

self.tableWidget\_2.removeRow(i)

# print(i)

for row in results:

try:

row\_tw = self.tableWidget\_2.rowCount()

self.tableWidget\_2.insertRow(row\_tw)

for r in range(len(row)):

self.tableWidget\_2.setItem(row\_tw, r, QTableWidgetItem(row[r]))

except:

print("检查qt5 数据插入")

print(self.output + "index=" + index.\_\_str\_\_() + " detail=" + str(row)) # 打印结果

index += 1

print(self.output + "共计" + results.\_\_len\_\_().\_\_str\_\_() + "条数据")

def ReFalsh(self):

tableName = "一些图书信息"

sql = 'select \* from %s' % tableName

self.cursor.execute(sql)

results = self.cursor.fetchall() # 获取所有记录列表

print(self.exe + sql)

index = 0

self.tableWidget.clearContents()

self.tableWidget\_3.clearContents()

for i in range(0, self.tableWidget.rowCount())[::-1]:

self.tableWidget.removeRow(i)

self.tableWidget\_3.removeRow(i)

# print(i)

for row in results:

try:

row\_tw = self.tableWidget.rowCount()

self.tableWidget.insertRow(row\_tw)

self.tableWidget\_3.insertRow(row\_tw)

for r in range(len(row)):

self.tableWidget.setItem(row\_tw, r, QTableWidgetItem(row[r]))

self.tableWidget\_3.setItem(row\_tw, r, QTableWidgetItem(row[r]))

except:

print("检查qt5 数据插入")

print(self.output + "index=" + index.\_\_str\_\_() + " detail=" + str(row)) # 打印结果

index += 1

print(self.output + "共计" + results.\_\_len\_\_().\_\_str\_\_() + "条数据")

pass

if \_\_name\_\_ == '\_\_main\_\_':

app = QApplication(sys.argv)

w = win()

w.show()

sys.exit(app.exec\_())

1. 附件
2. library.db数据库
3. 用excel数据创建数据库的python程序