第 3 章

数据库应用——智力问答测试

3.1 智力问答测试功能介绍

智力问答测试，内容涉及历史、经济、风情、民俗、地理、人文等古今中外各方面的知识，让玩家在轻松娱乐、益智、搞笑的同时不知不觉地增长知识。在答题过程中对做对、做错进行实时跟踪，测试完成后能根据玩家的答题情况给出成绩。程序运行界面如图 3-1 所示。

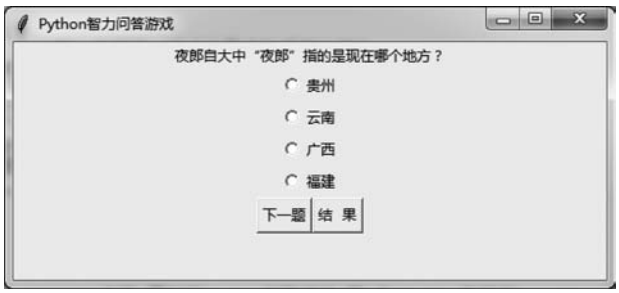

图 3-1 智力问答测试程序的运行界面

3.2 程序设计的思路

程序使用了一个 SQLite 试题库 test2.db，其中每个智力问答由题目、4 个选项和正确答案组成(question、Answer_A、Answer_B、Answer_C、Answer_D、right_Answer)。在测试前，程序从试题库 test2.db 读取试题信息，存储到 values 列表中。在测试时，顺序从 values 列表读出题目显示在 GUI 界面中供用户答题。在进行界面设计时，智力问答题目是标签控件，4 个选项是单选按钮控件，在“下一题”按钮单击事件中实现题目切换和对错判断，如果正确则得分 score 加 10 分，错误不加分，并判断用户是否做完。在“结果”按钮单击

事件中实现得分 score 的显示。


视频讲解

3.3 关键技术

Python 从 2.5 版本以上就内置了 SQLite3，所以在 Python 中使用 SQLite 不需要安装任何东西，直接使用。SQLite3 数据库使用 SQL 语言。SQLite 作为后端数据库，可以制作有数据存储需求的工具。Python 标准库中的 SQLite3 提供该数据库的接口。

3.3.1 访问数据库的步骤

从 Python 2.5 开始，SQLite3 就成了 Python 的标准模块，这也是 Python 中的唯一一个数据库接口类模块，大大方便了用户使用 Python SQLite 数据库开发小型数据库应用系统。

Python 的数据库模块有统一的接口标准，所以数据库操作有统一的模式，操作数据库 SQLite3 主要分为以下几步。

❶ 导入 Python SQLite 数据库模块

Python 标准库中带有 sqlite3 模块，可直接导入：

```
import sqlite3
```

❷ 建立数据库连接，返回 Connection 对象

使用数据库模块的 connect()函数建立数据库连接，返回连接对象 con。

```
con = sqlite3.connect(connectstring)        # 连接到数据库，返回 sqlite3.connection 对象
```

说明：connectstring 是连接字符串。对于不同的数据库连接对象，其连接字符串的格式不同，sqlite 的连接字符串为数据库的文件名，例如“E:\test.db”。如果指定连接字符串为 memory，则可创建一个内存数据库。例如：

```
import sqlite3
con = sqlite3.connect("E:\\test.db")
```

如果 E 盘下的 test.db 存在，则打开数据库；否则在该路径下创建数据库 test.db 并打开。

❸ 创建游标对象

使用游标对象能够灵活地对从表中检索出的数据进行操作，就本质而言，游标实际上是一种能从包括多条数据记录的结果集中每次提取一条记录的机制。

调用 con.cursor()创建游标对象 cur：

```
cur = con.cursor()                          # 创建游标对象
```

❹ 使用 Cursor 对象的 execute()方法执行 SQL 命令返回结果集

调用 cur.execute()、cur.executemany()、cur.executescript()方法查询数据库。

- cur.execute(sql)：执行 SQL 语句。

- cur.execute(sql, parameters)：执行带参数的 SQL 语句。
- cur.executemany(sql, seq_of_pqrameters)：根据参数执行多次 SQL 语句。
- cur.executescript(sql_script)：执行 SQL 脚本。

例如创建一个表 category。

```
cur.execute("create table category(id primary key, sort, name)")
```

此时将创建一个包含 3 个字段 id、sort 和 name 的表 category。下面向表中插入记录：

```
cur.execute("insert into category values(1, 1, 'computer')")
```

在 SQL 语句字符串中可以使用占位符“?”表示参数，传递的参数使用元组。例如：

```
cur.execute("insert into category values (?, ?,?) ",(2, 3, 'literature'))
```

❺ 获取游标的查询结果集

调用 cur.fetchall()、cur.fetchone()、cur.fetchmany()返回查询结果。

- cur.fetchone()：返回结果集的下一行(Row 对象)；无数据时返回 None。
- cur.fetchall()：返回结果集的剩余行(Row 对象列表)，无数据时返回空 List。
- cur.fetchmany()：返回结果集的多行(Row 对象列表)，无数据时返回空 List。

例如：

```
cur.execute("select * from catagory")
print(cur.fetchall())                    #提取查询到的数据
```

返回结果如下：

```
[(1, 1, 'computer'), (2, 2,'literature')]
```

如果使用 cur.fetchone()，则首先返回列表中的第 1 项，再次使用，返回第 2 项，依次进行。用户也可以直接使用循环输出结果，例如：

```
for row in cur.execute("select * from catagory"):
    print(row[0],row[1])
```

❻ 数据库的提交和回滚

根据数据库事务隔离级别的不同，可以提交或回滚。

- con.commit()：事务提交。
- con.rollback()：事务回滚。

❼ 关闭 Cursor 对象和 Connection 对象

最后需要关闭打开的 Cursor 对象和 Connection 对象。

- cur.close()：关闭 Cursor 对象。
- con.close()：关闭 Connection 对象。

3.3.2　创建数据库和表

例 3-1　创建数据库 sales，并在其中创建表 book，表中包含 3 列，即 id、price 和 name，

其中 id 为主键(primary key)。

```
#导入 Python SQLite 数据库模块
import sqlite3
#创建 SQLite 数据库
con = sqlite3.connect("E:\\sales.db")      #若不指定文件夹 E:\,则默认存放在程序所在文件夹
#创建表 book,包含 3 列,即 id(主键)、price 和 name
con.execute("create table book(id primary key, price, name)")
```

说明：Connection 对象的 execute()方法是 Cursor 对象对应方法的快捷方式，系统会创建一个临时 Cursor 对象，然后调用对应的方法，并返回 Cursor 对象。

3.3.3 数据库的插入、更新和删除操作

在数据库表中插入、更新、删除记录的一般步骤如下。

(1) 建立数据库连接。

(2) 创建游标对象 cur，使用 cur.execute(sql)执行 SQL 的 insert、update、delete 等语句完成数据库记录的插入、更新、删除操作，并根据返回值判断操作结果。

(3) 提交操作。

(4) 关闭数据库。

例 3-2 数据库表记录的插入、更新和删除操作。

```
import sqlite3
books = [("021",25,"大学计算机"),("022",30, "大学英语"),("023",18, "艺术欣赏"),
( "024",35, "高级语言程序设计")]
#打开数据库
Con = sqlite3.connect("E:\\sales.db")
#创建游标对象
Cur = Con.cursor()
#插入一行数据
Cur.execute("insert into book(id,price,name) values('001',33,'大学计算机   多媒体')")
Cur.execute("insert into book(id,price,name) values(?,?,?) " ,("002",28,   "数据库基础"))
#插入多行数据
Cur.executemany("insert into book(id,price,name) values (?,?,?) ",books)
#修改一行数据
Cur.execute("Update book set price = ? where name = ? ",(25,"大学英语"))
#删除一行数据
n = Cur.execute("delete from book where price = ?",(25,))
print("删除了",n.rowcount,"行记录")
Con.commit()
Cur.close()
Con.close()
```

运行结果如下：

```
删除了 2 行记录
```

3.3.4　数据库表的查询操作

查询数据库的步骤如下。

(1) 建立数据库连接。

(2) 创建游标对象 cur,使用 cur.execute(sql)执行 SQL 的 select 语句。

(3) 循环输出结果。

```
import sqlite3
#打开数据库
Con = sqlite3.connect("E:\\sales.db")
#创建游标对象
Cur = Con.cursor()
#查询数据库表
Cur.execute("select id,price,name from book")
for row in Cur:
    print(row)
```

运行结果如下:

```
('001', 33, '大学计算机多媒体')
('002', 28, '数据库基础')
('023', 18, '艺术欣赏')
('024', 35, '高级语言程序设计')
```

3.3.5　数据库使用实例——学生通讯录

设计一个学生通讯录,可以添加、删除、修改里面的信息。

```
import sqlite3
#打开数据库
def opendb():
        conn = sqlite3.connect("E:\\mydb.db")
        cur = conn.execute("create table if not exists tongxinlu(usernum
            integer primary key,username varchar(128), password varchar(128),
address varchar(125), telnum varchar(128))")
        return cur, conn
#查询全部信息
def showalldb():
        print("--------------------处理后的数据--------------------")
        hel = opendb()
        cur = hel[1].cursor()
        cur.execute("select * from tongxinlu")
        res = cur.fetchall()
        for line in res:
                for h in line:
                        print(h,end = ","),
```

```
                print()
        cur.close()
#输入信息
def into():
        usernum = input("请输入学号:")
        username1 = input("请输入姓名:")
        password1 = input("请输入密码:")
        address1 = input("请输入地址:")
        telnum1 = input("请输入联系电话:")
        return usernum,username1, password1, address1, telnum1
#往数据库中添加内容
def adddb():
        welcome = """ ---------- 欢迎使用添加数据功能 ---------- """
        print(welcome)
        person = into()
        hel = opendb()
        hel[1].execute("insert into tongxinlu(usernum,username, password, address, telnum)
        values(?,?,?,?,?)",(person[0], person[1], person[2], person[3],person[4]))
        hel[1].commit()
        print ("---------- 恭喜你,数据添加成功 ----------")
        showalldb()
        hel[1].close()
#删除数据库中的内容
def deldb():
        welcome = "---------- 欢迎使用删除数据库功能 ----------"
        print(welcome)
        delchoice = input("请输入想要删除的学号:")
        hel = opendb()                  #返回游标 conn
        hel[1].execute("delete from tongxinlu where usernum  = " + delchoice)
        hel[1].commit()
        print ("---------- 恭喜你,数据删除成功 ----------")
        showalldb()
        hel[1].close()
#修改数据库的内容
def alter():
        welcome = "---------- 欢迎使用修改数据库功能 ----------"
        print(welcome)
        changechoice = input("请输入想要修改的学生的学号:")
        hel = opendb()
        person = into()
        hel[1].execute("update tongxinlu set usernum = ?,username = ?,
        password = ?,address = ?,telnum = ? where usernum = " + changechoice,
        (person[0], person[1], person[2], person[3],person[4]))
        hel[1].commit()
        showalldb()
        hel[1].close()
#查询数据
def searchdb():
        welcome = "---------- 欢迎使用查询数据库功能 ----------"
        print(welcome)
```

```
        choice = input("请输入要查询的学生的学号:")
        hel = opendb()
        cur = hel[1].cursor()
        cur.execute("select * from tongxinlu where usernum = " + choice)
        hel[1].commit()
              print("---------------- 恭喜你,你要查找的数据如下 ----------------")
        for row in cur:
              print(row[0],row[1],row[2],row[3],row[4])
            cur.close()
        hel[1].close()
#是否继续
def conti():
        choice = input("是否继续?(y or n):")
        if choice == 'y':
              a = 1
        else:
              a = 0
        return a
if __name__ == "__main__":
        flag = 1
        while flag:
              welcome = " ---------- 欢迎使用数据库通讯录 ---------- "
              print(welcome)
              choiceshow = """
请选择您的进一步选择:
(添加)往通讯录数据库中添加内容
(删除)删除通讯录中的内容
(修改)修改通讯录的内容
(查询)查询通讯录的内容
选择您想要进行的操作:
"""
              choice = input(choiceshow)
              if choice == "添加":
                    adddb()
                    flag = conti()
              elif choice == "删除":
                    deldb()
                    flag = conti()
              elif choice == "修改":
                    alter()
                    flag = conti()
              elif choice == "查询":
                    searchdb()
                    flag = conti()
              else:
                    print("你输入错误,请重新输入")
```

程序运行界面和添加记录界面如图 3-2 所示。

图 3-2　程序运行界面和添加记录界面


视频讲解

3.4　程序设计的步骤

3.4.1　生成试题库

建立数据库 test2.db 的代码如下：

```
import sqlite3                                  #导入 SQLite 驱动
#连接到 SQLite 数据库,数据库文件是 test2.db
#如果文件不存在,会自动在当前目录创建
conn = sqlite3.connect('test2.db')
cursor = conn.cursor()                          #创建一个 Cursor
#cursor.execute("delete from exam")
#执行一条 SQL 语句,创建 exam 表
cursor.execute('CREATE TABLE [exam] ([question] VARCHAR(80) NULL,[Answer_A] VARCHAR(50) NULL,
[Answer_B] VARCHAR(50) NULL,[Answer_C] VARCHAR(50) NULL,[Answer_D] VARCHAR(50) NULL,[right_
Answer] VARCHAR(1) NULL)')
#继续执行一条 SQL 语句,插入一条记录
cursor.execute("insert into exam (question, Answer_A, Answer_B, Answer_C,   Answer_D, right_
Answer) values ('哈雷彗星的平均周期为', '54 年', '56 年', '73 年', '83 年', 'C')")
cursor.execute("insert into exam (question, Answer_A, Answer_B, Answer_C,   Answer_D, right_
Answer) values('夜郎自大中"夜郎"指的是现在哪个地方?', '贵州', '云南', '广西', '福建', 'A')")
cursor.execute("insert into exam (question, Answer_A, Answer_B, Answer_C,   Answer_D, right_
Answer) values('在中国历史上是谁发明了麻药', '孙思邈', '华佗', '张仲景', '扁鹊', 'B')")
cursor.execute("insert into exam (question, Answer_A, Answer_B, Answer_C,   Answer_D, right_
Answer) values('京剧中花旦是指', '年轻男子', '年轻女子', '年长男子', '年长女子', 'B')")
cursor.execute("insert into exam (question, Answer_A, Answer_B, Answer_C,   Answer_D, right_
Answer) values('篮球比赛每队几人?', '4', '5', '6', '7', 'B')")
cursor.execute("insert into exam (question, Answer_A, Answer_B, Answer_C,   Answer_D, right_
Answer) values('在天愿作比翼鸟,在地愿为连理枝.讲述的是谁的爱情故事?',   '焦仲卿和刘兰芝',
'梁山伯与祝英台', '崔莺莺和张生', '杨贵妃和唐明皇', 'D')")
print(cursor.rowcount)                          #通过 rowcount 获得插入的行数
cursor.close()                                  #关闭 Cursor
conn.commit()                                   #提交事务
conn.close()                                    #关闭 Connection
```

3.4.2　读取试题信息

读取试题信息的代码如下：

```
conn = sqlite3.connect('test2.db')
cursor = conn.cursor()
#执行查询语句
cursor.execute('select * from exam')
#获得查询结果集
values = cursor.fetchall()
cursor.close()
conn.close()
```

以上代码完成数据库 test2.db 的试题信息的读取，存储到 values 列表中。

3.4.3　界面和逻辑设计

callNext()用于判断用户选择的正误，正确则加 10 分，错误不加分；并判断用户是否做完，如果没做完，则将下一题的题目信息显示到 timu 标签，4 个选项显示到 radio1～radio4 这 4 个单选按钮上。

```
import tkinter
from tkinter import *
from tkinter.messagebox import *
def callNext():
    global k
    global score
    useranswer = r.get()                    #获取用户的选择
    print(r.get())                          #获取被选中单选按钮变量值
    if useranswer == values[k][5]:
        showinfo("恭喜","恭喜你对了!")
        score += 10
    else:
        showinfo("遗憾","遗憾你错了!")
    k = k + 1
    if k >= len(values):                    #判断用户是否做完
        showinfo("提示","题目做完了")
        return
    #显示下一题
    timu["text"] = values[k][0]             #题目信息
    radio1["text"] = values[k][1]           #A 选项
    radio2["text"] = values[k][2]           #B 选项
    radio3["text"] = values[k][3]           #C 选项
    radio4["text"] = values[k][4]           #D 选项
    r.set('E')

def callResult():
    showinfo("你的得分",str(score))
```

以下是界面布局代码：

```
root = tkinter.Tk()
root.title('Python 智力问答游戏')
root.geometry("500x200")
r = tkinter.StringVar()                    #创建 StringVar 对象
r.set('E')                                 #设置初始值为'E',初始没选中
k = 0
score = 0
timu = tkinter.Label(root,text = values[k][0])     #题目
timu.pack()
f1 = Frame(root)                           #创建第 1 个 Frame 组件
f1.pack()
radio1 = tkinter.Radiobutton(f1,variable = r,value = 'A',text = values[k][1])
radio1.pack()
radio2 = tkinter.Radiobutton(f1,variable = r,value = 'B',text = values[k][2])
radio2.pack()
radio3 = tkinter.Radiobutton(f1,variable = r,value = 'C',text = values[k][3])
radio3.pack()
radio4 = tkinter.Radiobutton(f1,variable = r,value = 'D',text = values[k][4])
radio4.pack()
f2 = Frame(root)                           #创建第 2 个 Frame 组件
f2.pack()
Button(f2,text = '下一题',command = callNext).pack(side = LEFT)
Button(f2,text = '结果',command = callResult).pack(side = LEFT)
root.mainloop()
```