

Student Performance Measurement System

content

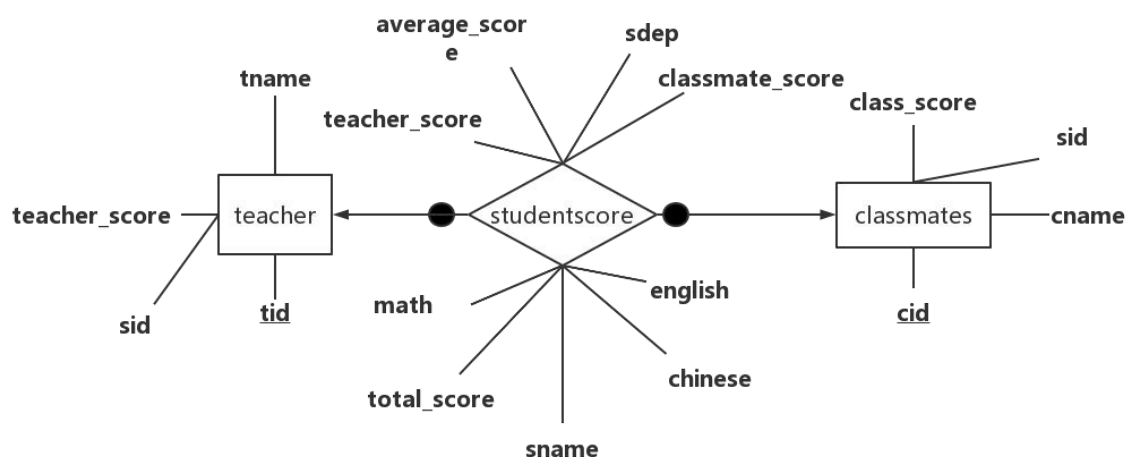
Student Performance Measurement System	1
1、 Requirement analysis	2
2、 Functional design and analysis	2
①Access to data using database	2
(1) Use PyMySQL module to operate the database to access the data	2
(2) Create database school, create data tables student_sore, teacher_login	3
(3) Use xlrd module to read data from Excel file to database	6
(4) Define a PyMySQL tool class PyMySQLUtils for adding, deleting, and checking	6
②Use desktop form interface for interaction	8
(1) Use Tkinter module to realize GUI design with graphical interface	8
(2) Start interface: teacher registration, teacher login, and exit system	8
(3) Teacher registration interface: enter account number, enter password, confirm account, OK, return	8
(4) teacher login interface: account, password, OK, back	9
(5) teacher operation interface: add, modify, query, delete	9
③to achieve all relevant information to add, modify, query, delete and other functions	10
(1) Adding student grade information	10
(2) Modify student's grade information	11
(3) Query student grade information	12
(4) Delete student grade information	12
(5) Clear the content of the input box	13
(6) Write to Excel file	13
3、 Summary and experience	14
①Course design summary	14
②Curriculum design experience	14
③Database ER diagram	2
4、 Running results	15
①Start screen	15
②Teacher registration interface	16
③Teacher login screen	17
④Teacher operation interface	18

1、 Requirement analysis

- (1) The knowledge of class to achieve student grade information (student number, name, department, three course grades, exam average, peer rating, teacher rating, total comprehensive assessment score, where the total comprehensive assessment score consists of: exam average score 70%, peer rating 10%, teacher rating 20%).
- (2) The ability to save and read student grade information (using a database to access the data).
- (3) Realizing the functions of inputting, outputting, finding, deleting and modifying all relevant information.
- (4) The system interface should implement at least a console interface (using a desktop form interface for interaction).
- (5) Reading and writing Excel files through the xlrd and xlwt modules, for the convenience of teachers to print student results (direct use of the database teacher to export results and view results will be more inconvenient).

2、 Functional design and analysis

① Database ER diagram



② Access to data using database

(1) Use PyMySQL module to operate the database to access the data

Install PyMySQL module first: `pip install PyMySQL`

When using it again, import it directly: `import pymysql`

(2) Create database school, create data tables `student_score`, `teacher_login`

can be created using workbench, or the tables can be created using preprocessing statements, if they do not exist, then create them, if they exist, then skip them.

```
def create_table(self):
    utils = PyMySQLUtils()
    # 使用预处理语句创建表，若不存在则创建，若存在则跳过
    sql1 = """CREATE TABLE IF NOT EXISTS student_score(
        sid varchar(32) PRIMARY KEY,
        sname varchar(32) NOT NULL,
        sdept varchar(32) NOT NULL,
        chinese float(5,2) NOT NULL,
        math float(5,2) NOT NULL,
        english float(5,2) NOT NULL,
        average_score float(5,2) NOT NULL,
        classmate_score float(5,2) NOT NULL,
        teacher_score float(5,2) NOT NULL,
        total_score float(5,2) NOT NULL
    ) ENGINE = InnoDB DEFAULT CHARSET = utf8
    """
    utils.execute(sql1)
    # 使用预处理语句创建表，若不存在则创建，若存在则跳过
    sql2 = """CREATE TABLE IF NOT EXISTS teacher_login(
        username varchar(32) NOT NULL,
        password varchar(32) NOT NULL
    ) ENGINE = InnoDB DEFAULT CHARSET = utf8
    """
    utils.execute(sql2)
```

```
utils.execute(sql2)
sql3 = """CREATE TABLE IF NOT EXISTS teacher(
    tid VARCHAR(32) PRIMARY KEY,
    tname VARCHAR(32) NOT NULL,
    sid VARCHAR(32) NOT NULL,
    teacher_score FLOAT(5,2) NOT NULL,
    FOREIGN KEY(sid) REFERENCES students_score(sid)
)ENGINE = InnoDB DEFAULT CHARSET = utf8
"""
utils.execute(sql3)
sql4 = """CREATE TABLE IF NOT EXISTS classmates(
    cid VARCHAR(32) PRIMARY KEY,
    cname VARCHAR(32) NOT NULL,
    sid VARCHAR(32) NOT NULL,
    classmate_score FLOAT(5,2) NOT NULL,
    FOREIGN KEY(sid) REFERENCES students_score(sid)
)ENGINE = InnoDB DEFAULT CHARSET = utf8
"""
utils.execute(sql4)
utils.close()
```

Use the command prompt to view the database creation

管理员: 命令提示符 - mysql -u root -p

2 rows in set (0.00 sec)

mysql> show tables;

Tables_in_school
classmates
student_score
teacher
teacher_login

4 rows in set (0.02 sec)

mysql> desc teacher_login;

Field	Type	Null	Key	Default	Extra
username	varchar(32)	NO		NULL	
password	varchar(32)	NO		NULL	

2 rows in set (0.02 sec)

mysql> desc student_score

-> ;

Field	Type	Null	Key	Default	Extra
sid	varchar(32)	NO	PRI	NULL	
sname	varchar(32)	NO		NULL	
sdept	varchar(32)	NO		NULL	
chinese	float(5,2)	NO		NULL	
math	float(5,2)	NO		NULL	
english	float(5,2)	NO		NULL	
average_score	float(5,2)	NO		NULL	
classmate_score	float(5,2)	NO		NULL	
teacher_score	float(5,2)	NO		NULL	
total_score	float(5,2)	NO		NULL	

10 rows in set (0.00 sec)

mysql> desc teacher;

Field	Type	Null	Key	Default	Extra
tid	varchar(32)	NO	PRI	NULL	
tname	varchar(32)	NO		NULL	
sid	varchar(32)	NO		NULL	
teacher_score	float(5,2)	NO		NULL	

4 rows in set (0.00 sec)

mysql> desc classmates;

Field	Type	Null	Key	Default	Extra
cid	varchar(32)	NO	PRI	NULL	
classmate_score	float(5,2)	NO		NULL	
sid	varchar(32)	NO		NULL	

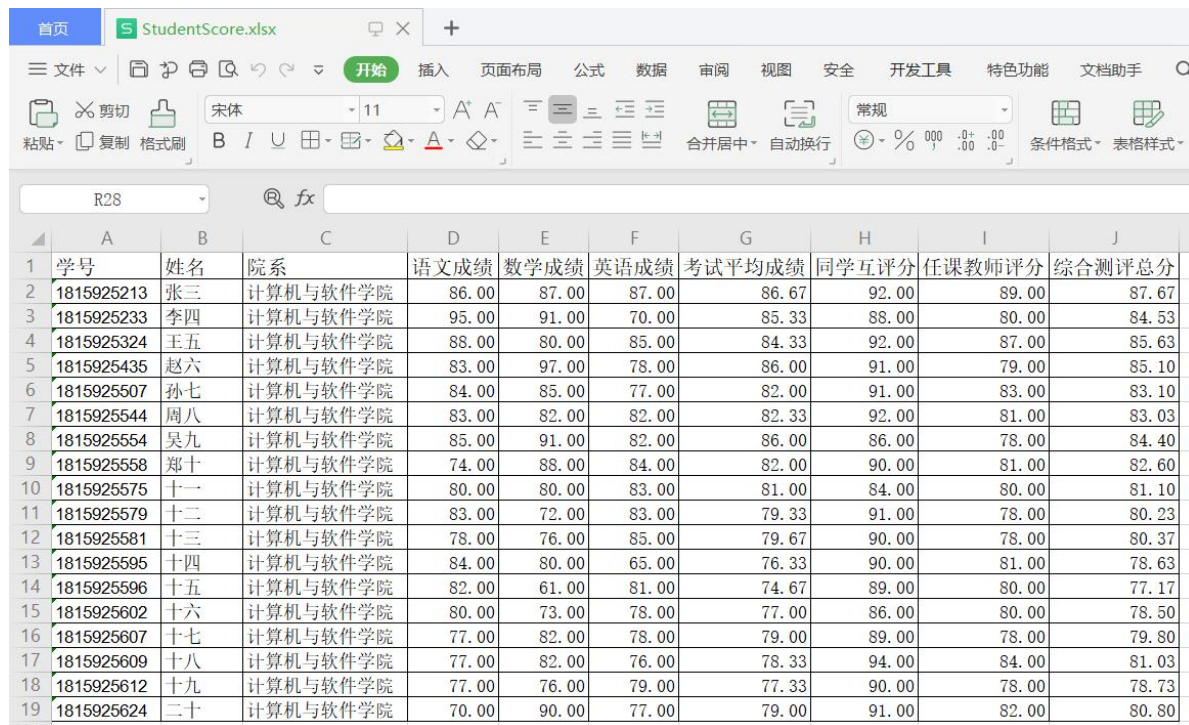
3 rows in set (0.00 sec)

mysql> █

(3) Use xlrd module to read data from Excel file to database

Install: pip install xlrd Import: import xlrd

Open an Excel file, get the sheet table by the index of the sheet, loop to get the value of each cell, and read it to the database table line by line.



The screenshot shows an Excel spreadsheet titled 'StudentScore.xlsx'. The table contains student information and scores across various subjects. The columns are: 学号 (Student ID), 姓名 (Name), 院系 (Department), 语文成绩 (Chinese Score), 数学成绩 (Math Score), 英语成绩 (English Score), 考试平均成绩 (Average Exam Score), 同学互评分 (Peer Evaluation), 任课教师评分 (Teacher Evaluation), and 综合测评总分 (Total Comprehensive Evaluation Score). The data rows list 20 students with their respective scores.

学号	姓名	院系	语文成绩	数学成绩	英语成绩	考试平均成绩	同学互评分	任课教师评分	综合测评总分
1815925213	张三	计算机与软件学院	86.00	87.00	87.00	86.67	92.00	89.00	87.67
1815925233	李四	计算机与软件学院	95.00	91.00	70.00	85.33	88.00	80.00	84.53
1815925324	王五	计算机与软件学院	88.00	80.00	85.00	84.33	92.00	87.00	85.63
1815925435	赵六	计算机与软件学院	83.00	97.00	78.00	86.00	91.00	79.00	85.10
1815925507	孙七	计算机与软件学院	84.00	85.00	77.00	82.00	91.00	83.00	83.10
1815925544	周八	计算机与软件学院	83.00	82.00	82.00	82.33	92.00	81.00	83.03
1815925554	吴九	计算机与软件学院	85.00	91.00	82.00	86.00	86.00	78.00	84.40
1815925558	郑十	计算机与软件学院	74.00	88.00	84.00	82.00	90.00	81.00	82.60
1815925575	十一	计算机与软件学院	80.00	80.00	83.00	81.00	84.00	80.00	81.10
1815925579	十二	计算机与软件学院	83.00	72.00	83.00	79.33	91.00	78.00	80.23
1815925581	十三	计算机与软件学院	78.00	76.00	85.00	79.67	90.00	78.00	80.37
1815925595	十四	计算机与软件学院	84.00	80.00	65.00	76.33	90.00	81.00	78.63
1815925596	十五	计算机与软件学院	82.00	61.00	81.00	74.67	89.00	80.00	77.17
1815925602	十六	计算机与软件学院	80.00	73.00	78.00	77.00	86.00	80.00	78.50
1815925607	十七	计算机与软件学院	77.00	82.00	78.00	79.00	89.00	78.00	79.80
1815925609	十八	计算机与软件学院	77.00	82.00	76.00	78.33	94.00	84.00	81.03
1815925612	十九	计算机与软件学院	77.00	76.00	79.00	77.33	90.00	78.00	78.73
1815925624	二十	计算机与软件学院	70.00	90.00	77.00	79.00	91.00	82.00	80.80

```
def excel_mysql(self):
    book = xlrd.open_workbook("E:\\PycharmProjects\\StudentScoreManagerSystem\\StudentScore.xlsx")
    sheet = book.sheet_by_index(0)
    utils = PyMySQLUtils()
    for i in range(1, sheet.nrows):
        sid = sheet.cell(i, 0).value
        sname = sheet.cell(i, 1).value
        sdept = sheet.cell(i, 2).value
        chinese = sheet.cell(i, 3).value
        math = sheet.cell(i, 4).value
        english = sheet.cell(i, 5).value
        average_score = sheet.cell(i, 6).value
        classmate_score = sheet.cell(i, 7).value
        teacher_score = sheet.cell(i, 8).value
        total_score = sheet.cell(i, 9).value
        utils.execute(
            f"INSERT INTO student_score VALUES('{sid}', '{sname}', '{sdept}', {chinese}, {math}, {english}, "
            f"{average_score}, {classmate_score}, {teacher_score}, {total_score})")
    utils.close()
```

(4) Define a PyMySQL tool class PyMySQLUtils for adding, deleting, and checking

1) def __init__(self) to get the connection.

Open a connection to the database and use the cursor() method to get the operation cursor.


```

# 数据库操作的工具类
class PyMySQLUtils:
    # 获取连接
    def __init__(self):
        # 打开数据库连接
        self.db = pymysql.connect("localhost", "root", "123456", "school")
        # 使用cursor()方法获取操作游标
        self.cursor = self.db.cursor()

```

2) def fetchall(self, sql) query to get multiple data.

Executing SQL statements using the execute() method and fetching multiple data using the fetchall() method.

```

# 查询获取多条数据
def fetchall(self, sql):
    # 使用execute()方法执行SQL语句
    self.cursor.execute(sql)
    # 使用fetchall()方法获取多条数据
    results = self.cursor.fetchall()
    return results

```

3) def fetchone(self, sql) query to fetch a single piece of data.

Execute the SQL statement using the execute() method and fetch a single piece of data using the fetchone() method.

```

# 查询获取单条数据
def fetchone(self, sql):
    # 使用execute()方法执行SQL语句
    self.cursor.execute(sql)
    # 使用fetchone()方法获取单条数据
    result = self.cursor.fetchone()
    return result

```

4) def execute(self, sql) add delete update operation.

Execute the SQL statement using the execute() method, commit it to the database for execution and roll back in case of errors.

```

# 添加删除更新操作
def execute(self, sql):
    try:
        # 使用execute()方法执行SQL语句
        self.cursor.execute(sql)
        # 提交到数据库执行
        self.db.commit()
    except:
        # 发生错误时回滚
        self.db.rollback()

```

5) def close(self) closes the connection.

Closes the cursor and closes the database connection.

```

# 关闭连接
def close(self):
    # 关闭游标
    self.cursor.close()
    # 关闭数据库连接
    self.db.close()

```

③Use desktop form interface for interaction

(1) Use Tkinter module to realize GUI design with graphical interface

To use it, you can directly import

```
from tkinter import *
```

```
from tkinter import ttk
```

```
import tkinter.font as tkFont
```

```
import tkinter.messagebox as messagebox
```

(2) Start interface: teacher registration, teacher login, and exit system

class StartMenu: (destroy the previous window) initialize a root window window; add a Label control to display "Student Grade Management System" in a single line of text; add three Button controls and bind them to the associated functions lambda: TeacherRegister, lambda: TeacherLogin, window.destroy; wait for the user to trigger an event response in the main event loop.

(3) Teacher registration interface: enter account number, enter password, confirm account, OK, return

class TeacherRegister: (destroy the previous window) initialize a root window window; add Label control to display "Teacher Registration Page", "Enter Account Number", "Enter Password", "Confirm Account Number" in a single line of text; add three Entry controls to display user input text, add two Button button controls and bind them to the associated functions

register and back respectively; wait for the user to trigger an event response in the main event loop.

```
def register(self):
    utils = PyMySQLUtils()
    result = utils.fetchone("SELECT * FROM teacher_login WHERE username = '%s'" % self.my_username.get())
    if self.my_username.get() == "" or self.my_password.get() == "" or self.re_password.get() == "":
        messagebox.showerror(title="showerror", message="注册失败! 注册信息不完整! ")
    elif self.my_password.get() != self.re_password.get():
        messagebox.showerror(title="showerror", message="注册失败! 两次密码不相同! ")
    elif result:
        messagebox.showerror(title="showerror", message="注册失败! 该账号已被注册过! ")
    else:
        utils.execute("INSERT INTO teacher_login(username, password) VALUES('%s', '%s')" % (
            self.my_username.get(), self.my_password.get()))
        utils.close()
        messagebox.showinfo(title="showinfo", message="注册成功! 欢迎您登录使用! ")
        TeacherLogin(self.window)
```

(4) teacher login interface: account, password, OK, back

class TeacherLogin: (destroy the previous window) initialize a root window window; add two Entry input controls for displaying user input text, add two Button controls, bind them to the associated functions login and back respectively; wait for the user to trigger an event response in the main event loop.

```
def login(self):
    utils = PyMySQLUtils()
    result = utils.fetchone("SELECT * FROM teacher_login WHERE username = '%s'" % self.my_username.get())
    if self.my_username.get() == "" or self.my_password.get() == "":
        messagebox.showerror(title="showerror", message="登录失败! 登录信息不完整! ")
    elif result:
        utils.close()
        if self.my_password.get() == result[1]:
            messagebox.showinfo("showinfo", "登录成功! 欢迎您使用! ")
            TeacherMenu(self.window)
        else:
            messagebox.showerror("showerror", "登录失败! 输入的密码错误! ")
    else:
        messagebox.showerror("showerror", "登录失败! 输入的账号有误! ")
```

(5) teacher operation interface: add, modify, query, delete

class TeacherMenu: (destroy the last window) initialize a root window window; add three Frame frame controls for frame grouping self.frame_center, self.frame_left, frame_right; add ttk.Treeview tree view in the center area Scrollbar Scrollbar control, set the columns, table headers, define the list of stored data, get the table content from the database, set the table content, bind the function tree_sort_column to the table headers, click to sort, bind the click event tree_click to the table, get the clicked entries; add eight in the left area Label label control in the left area for single line text display; add eight Entry input controls for displaying user input text, add six Button controls and bind them to the associated functions respectively; wait for the user to trigger an event response in the main event loop.

```

# 从数据库获取表格内容
utils = PyMySQLUtils()
results = utils.fetchall("SELECT * FROM student_score")
for row in results:
    self.list_sid.append(row[0])
    self.list_sname.append(row[1])
    self.list_sdept.append(row[2])
    self.list_chinese.append(row[3])
    self.list_math.append(row[4])
    self.list_english.append(row[5])
    self.list_average_score.append(row[6])
    self.list_classmate_score.append(row[7])
    self.list_teacher_score.append(row[8])
    self.list_total_score.append(row[9])
utils.close()

# 设置表格内容
for i in range(min(len(self.list_sid), len(self.list_sname), len(self.list_sdept), len(self.list_chinese),
    len(self.list_math), len(self.list_english), len(self.list_average_score),
    len(self.list_classmate_score), len(self.list_teacher_score), len(self.list_total_score))):
    self.tree.insert("", i,
        values=(self.list_sid[i], self.list_sname[i], self.list_sdept[i], self.list_chinese[i],
            self.list_math[i], self.list_english[i], self.list_average_score[i],
            self.list_classmate_score[i], self.list_teacher_score[i],
            self.list_total_score[i]))

# 绑定函数使表头可排序
for col in self.columns:
    self.tree.heading(col, text=col, command=lambda _col=col: self.tree_sort_column(self.tree, _col, False))

# 绑定点击事件
self.tree.bind('<ButtonRelease-1>', self.tree_click)

```

```

# 点击表头排序
def tree_sort_column(self, tv, col, reverse):
    l = [(tv.set(k, col), k) for k in tv.get_children('')]
    l.sort(reverse=reverse)
    for index, (val, k) in enumerate(l):
        # 根据排序后索引移动
        tv.move(k, '', index)
    # 重写标题，使之成为再点倒序的标题
    tv.heading(col, command=lambda: self.tree_sort_column(tv, col, not reverse))

# 获取被点击的条目
def tree_click(self, event):
    row = self.tree.identify_row(event.y)
    row_info = self.tree.item(row, 'values')
    self.var_sid.set(row_info[0])
    self.var_sname.set(row_info[1])
    self.var_sdept.set(row_info[2])
    self.var_chinese.set(row_info[3])
    self.var_math.set(row_info[4])
    self.var_english.set(row_info[5])
    self.var_classmate_score.set(row_info[7])
    self.var_teacher_score.set(row_info[8])

```

④to achieve all relevant information to add, modify, query, delete and other functions

(1) Adding student grade information

insert: judge whether the student number in the input box is in the list of stored student numbers, if it is, then warn "The student's grade information already exists! If it is not, add the data in the

input box to the database, then add it to the list of stored data, and finally add it to the table content.

```
# 添加学生成绩信息
def insert(self):
    if messagebox.askyesnocancel("askyesnocancel", "是否该添加学生成绩信息? "):
        sid = self.var_sid.get()
        sname = self.var_sname.get()
        sdept = self.var_sdept.get()
        chinese = round(float(self.var_chinese.get()), 2)
        math = round(float(self.var_math.get()), 2)
        english = round(float(self.var_english.get()), 2)
        average_score = round(((chinese + math + english) / 3), 2)
        classmate_score = round(float(self.var_classmate_score.get()), 2)
        teacher_score = round(float(self.var_teacher_score.get()), 2)
        total_score = round((average_score * 0.7 + classmate_score * 0.1 + teacher_score * 0.2), 2)
        if sid in self.list_sid:
            messagebox.showwarning("showwarning", "该学生成绩信息已存在! ")
        else:
            utils = PyMySQLUtils()
            utils.execute(
                f"INSERT INTO student_score VALUES('{sid}', '{sname}', '{sdept}', {chinese}, {math}, {english}, "
                f"{average_score}, {classmate_score}, {teacher_score}, {total_score})")
            utils.close()
            self.list_sid.append(sid)
            self.list_sname.append(sname)
            self.list_sdept.append(sdept)
            self.list_chinese.append(chinese)
            self.list_math.append(math)
            self.list_english.append(english)
            self.list_average_score.append(average_score)
            self.list_classmate_score.append(classmate_score)
            self.list_teacher_score.append(teacher_score)
            self.list_total_score.append(total_score)
            self.tree.insert('', 'end', values=(sid, sname, sdept, chinese, math, english, average_score,
                                                classmate_score, teacher_score, total_score))
            self.tree.update()
            messagebox.showinfo("showinfo", "添加学生成绩信息成功! ")
```

(2) Modify student's grade information

update: judge whether the student number in the input box is in the list of stored student numbers or not, if it is not, then warn "The student grade information does not exist! If it is, the data in the database will be modified according to the student number in the input box, then the data in the list of stored data will be deleted according to the index of the stored student number in the input box, and finally the data in the table content will be deleted.

```

# 修改学生成绩信息
def update(self):
    if messagebox.askyesnocancel("askyesnocancel", "是否修改该学生成绩信息?"):
        sid = self.var_sid.get()
        sname = self.var_sname.get()
        sdept = self.var_sdept.get()
        chinese = round(float(self.var_chinese.get()), 2)
        math = round(float(self.var_math.get()), 2)
        english = round(float(self.var_english.get()), 2)
        average_score = round(((chinese + math + english) / 3), 2)
        classmate_score = round(float(self.var_classmate_score.get()), 2)
        teacher_score = round(float(self.var_teacher_score.get()), 2)
        total_score = round((average_score * 0.7 + classmate_score * 0.1 + teacher_score * 0.2), 2)
        if sid not in self.list_sid:
            messagebox.showwarning("showwarning", "该学生成绩信息不存在!")
        else:
            utils = PyMySQLUtils()
            utils.execute(
                f"UPDATE student_score SET sname = '{sname}', sdept = '{sdept}', chinese = {chinese}, math = {math}, "
                f"english = {english}, average_score = {average_score}, classmate_score = {classmate_score}, "
                f"teacher_score = {teacher_score}, total_score = {total_score} WHERE sid = '{sid}'")
            sid_index = self.list_sid.index(sid)
            self.list_sname[sid_index] = sname
            self.list_sdept[sid_index] = sdept
            self.list_chinese[sid_index] = chinese
            self.list_math[sid_index] = math
            self.list_english[sid_index] = english
            self.list_average_score[sid_index] = average_score
            self.list_classmate_score[sid_index] = classmate_score
            self.list_teacher_score[sid_index] = teacher_score
            self.list_total_score[sid_index] = total_score
            self.tree.item(self.tree.get_children()[sid_index], values=(
                sid, sname, sdept, chinese, math, english, average_score, classmate_score, teacher_score, total_score))
            messagebox.showinfo("showinfo", "修改学生成绩信息成功!")

```

(3) Query student grade information

select: judge whether the student number in the input box is in the list of stored student numbers, if not, then warn "The student's grade information does not exist!" If it is, the index of the student number in the input box in the list of stored numbers will be used to query the data in the list of stored data and set the data in the input box.

```

# 按学号查询某个学生成绩信息
def select(self):
    if messagebox.askyesnocancel("askyesnocancel", "是否查询该学生成绩信息?"):
        sid = self.var_sid.get()
        if sid not in self.list_sid:
            messagebox.showwarning("showwarning", "该学生成绩信息不存在!")
        else:
            sid_index = self.list_sid.index(sid)
            self.var_sname.set(self.list_sname[sid_index])
            self.var_sdept.set(self.list_sdept[sid_index])
            self.var_chinese.set(self.list_chinese[sid_index])
            self.var_math.set(self.list_math[sid_index])
            self.var_english.set(self.list_english[sid_index])
            self.var_classmate_score.set(self.list_classmate_score[sid_index])
            self.var_teacher_score.set(self.list_teacher_score[sid_index])
            messagebox.showinfo("showinfo", "查询学生成绩信息成功!")

```

(4) Delete student grade information

delete: judge whether the student number in the input box is in the list of stored student numbers or not, if not, the warning "The student grade information does not exist! If it is, delete the data in the database according to the student number in the input box, then delete the data in the list of stored data according to the index of the stored student number in the input box, and finally delete the data in the table content.


```
# 删除学生成绩信息
def delete(self):
    if messagebox.askyesnocancel("askyesnocancel", "是否删除该学生成绩信息?"):
        sid = self.var_sid.get()
        if sid not in self.list_sid:
            messagebox.showwarning("showwarning", "该学生成绩信息不存在!")
        else:
            utils = PyMySQLUtils()
            utils.execute(f"DELETE FROM student_score WHERE sid = '{sid}'")
            utils.close()
            sid_index = self.list_sid.index(sid)
            del self.list_sid[sid_index]
            del self.list_sname[sid_index]
            del self.list_sdept[sid_index]
            del self.list_chinese[sid_index]
            del self.list_math[sid_index]
            del self.list_english[sid_index]
            del self.list_average_score[sid_index]
            del self.list_classmate_score[sid_index]
            del self.list_teacher_score[sid_index]
            del self.list_total_score[sid_index]
            self.tree.delete(self.tree.get_children()[sid_index])
            messagebox.showinfo("showinfo", "删除学生成绩信息成功!")
```

(5) Clear the content of the input box

clear: set the content of the input box to empty directly through the StringVar.set() method

```
# 清空输入框的内容
def clear(self):
    self.var_sid.set("")
    self.var_sname.set("")
    self.var_sdept.set("")
    self.var_chinese.set("")
    self.var_math.set("")
    self.var_english.set("")
    self.var_classmate_score.set("")
    self.var_teacher_score.set("")
```

(6) Write to Excel file

Install: pip install xlwt Import: import xlwt

Create a new Excel file, add a sheet named sheet1, set allow rewrite override, get data from the database, write data to the sheet in a loop, and finally save the file.

```
class mysql_excel:
    def __init__(self, path):
        if messagebox.askyesnocancel("askyesnocancel", "是否写入到Excel文件?"):
            utils = PyMySQLUtils()
            results = utils.fetchall("SELECT * FROM student_score")
            book = xlwt.Workbook(encoding="utf-8")
            sheet = book.add_sheet("sheet1", cell_overwrite_ok=True)
            table_head = ["学号", "姓名", "院系", "语文成绩", "数学成绩", "英语成绩", "考试平均成绩", "同学互评分", "任课教师评分", "综合测评总分"]
            for i in range(len(table_head)):
                sheet.write(0, i, table_head[i])
            for row in range(len(results)):
                for col in range(len(results[row])):
                    print(results[row][col])
                    sheet.write(row+1, col, results[row][col])
            book.save(path)
            messagebox.showinfo("showinfo", "成功写入到Excel文件!")
```

3、 Summary and experience

①Course design summary

The student grade management system is still mainly to achieve the function of adding, deleting, checking and accessing, and it is easy to achieve this by using PyMySQL to operate the database operations, and a tool class PyMySQLUtils can be defined to simplify the code to avoid redundancy. The graphical interface is designed with Tkinter, importing the tkinter module, creating the root form, adding human-computer interaction controls and writing the corresponding functions, waiting for the user to trigger an event response in the main event loop; the placement of controls requires several trials and changes to find a suitable and relatively beautiful layout, grid is positioned in the form of a table, place() is to give precise coordinates to locate, pack() will be arranged in the way of up and down; relatively difficult is to get the data of the input box, you need to change the type in order to calculate the average score of the test and the number of bits retained and so on; the function of adding, deleting, changing and checking is to first operate the data in the database according to the student number in the input box, then operate the index of the stored data according to the index of the stored student number list in the input box Treeview tree view window control is to add, delete, change, and check the data in the database, the list, and the ttk. Sorting function is to bind the tree_sort_column function to the table header, using the list.sort() method, according to the sorted index shift, rewrite the table header so that it becomes the table header again point reverse order.

②Curriculum design experience

In general, this course design is a bit tight in terms of time, many functions that I wanted to achieve were not written, originally intended to be divided into: teacher registration and login operations, student login operations, some functions are not open to students, but the code is repeated a lot and omitted. This course design reviewed and deeply understood the basic features of Python language, data file reading methods and object-oriented thinking, and combined them with the database knowledge learned in this semester to realize the interaction using desktop form interface and access to data using database, and basically mastered the use of PyMySQL and Tkinter modules.

4、Running results

①Start screen



②Teacher registration interface

学生成绩管理系统-教师版

— □ ×

欢迎教师注册

输入账号：

输入密码：

确认密码：

确定

返回

showerror

×

 注册失败！注册信息不完整！

确定

showerror

×

 注册失败！该账号已被注册过！

确定

showerror

×

 注册失败！两次密码不相同！

确定

showinfo

×

 注册成功！欢迎您登录使用！

确定

③Teacher login screen

学生成绩管理系统-教师版

— □ ×

欢迎教师登录

账号：

密码：

确定

返回

showerror

×

×

登录失败！登录信息不完整！

确定

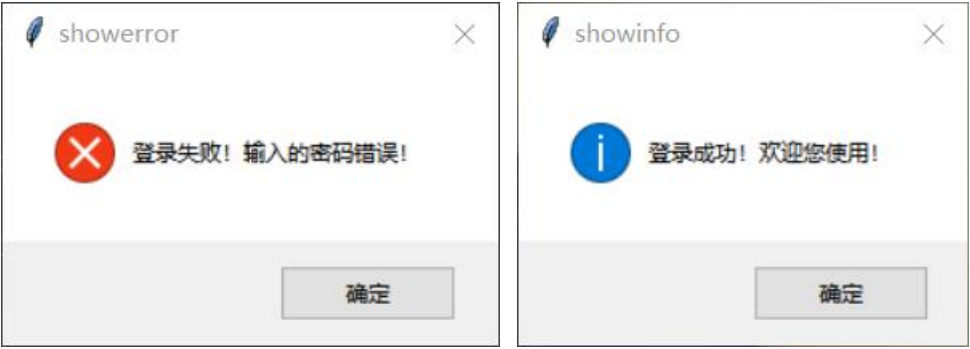
showerror

×

×

登录失败！输入的账号有误！

确定



④Teacher operation interface

学生成绩管理系统-教师版

学生成绩管理系统

学号	姓名	院系	语文成绩	数学成绩	英语成绩	考试平均成绩	同学互评分	任课教师评分	综合测评总分
1815925213	张三	计算机与软件	86.0	87.0	87.0	86.67	92.0	89.0	87.67
1815925233	李四	计算机与软件	95.0	91.0	70.0	85.33	88.0	80.0	84.53
1815925324	王五	计算机与软件	88.0	80.0	85.0	84.33	92.0	87.0	85.63
1815925435	赵六	计算机与软件	83.0	97.0	78.0	86.0	91.0	79.0	85.1
1815925507	孙七	计算机与软件	84.0	85.0	77.0	82.0	91.0	83.0	83.1
1815925544	周八	计算机与软件	83.0	82.0	82.0	82.33	92.0	81.0	83.03
1815925554	吴九	计算机与软件	85.0	91.0	82.0	86.0	86.0	78.0	84.4
1815925558	郑十	计算机与软件	74.0	88.0	84.0	82.0	90.0	81.0	82.6
1815925575	十一	计算机与软件	80.0	80.0	83.0	81.0	84.0	80.0	81.1
1815925579	十二	计算机与软件	83.0	72.0	83.0	79.33	91.0	78.0	80.23
1815925581	十三	计算机与软件	78.0	76.0	85.0	79.33	91.0	78.0	80.37
1815925595	十四	计算机与软件	84.0	80.0	65.0	76.0	91.0	78.0	78.63
1815925596	十五	计算机与软件	82.0	61.0	81.0	74.67	91.0	78.0	77.17
1815925602	十六	计算机与软件	80.0	73.0	78.0	77.0	91.0	78.0	78.5
1815925607	十七	计算机与软件	77.0	82.0	78.0	79.0	91.0	78.0	79.8

学号: 1815925001

姓名: 二十一

院系: 计算机与软件

语文成绩: 66

数学成绩: 77

英语成绩: 88

同学互评分: 88

任课教师评分: 77

修改学生成绩信息

查询学生成绩信息

删除学生成绩信息

写入到Excel文件

showinfo

添加学生成绩信息成功!

确定