



資料庫應用期末報告

使用Python實作SQL Server資料庫 的基礎操作

許宏韜

900032762

114年2月4日

簡報大綱

1

緣起

2

Try it

3

程式碼重點
解說

緣起

1. 113年上學期修習空大開設「Python程式設計與實務應用」課程，深感python的有趣與強大，故想繼續學習。
2. 爰以python實作SQL server的創建資料庫、建立資料表、增刪查改（CRUD）及刪除資料庫等功能，做為期末報告的題目。

Try it

下載程式碼：

https://github.com/taipeihugo/python_sql_server

```
>pip install pyodbc, customtkinter
```



安裝模組

```
>pip install pyodbc, customtkinter
```

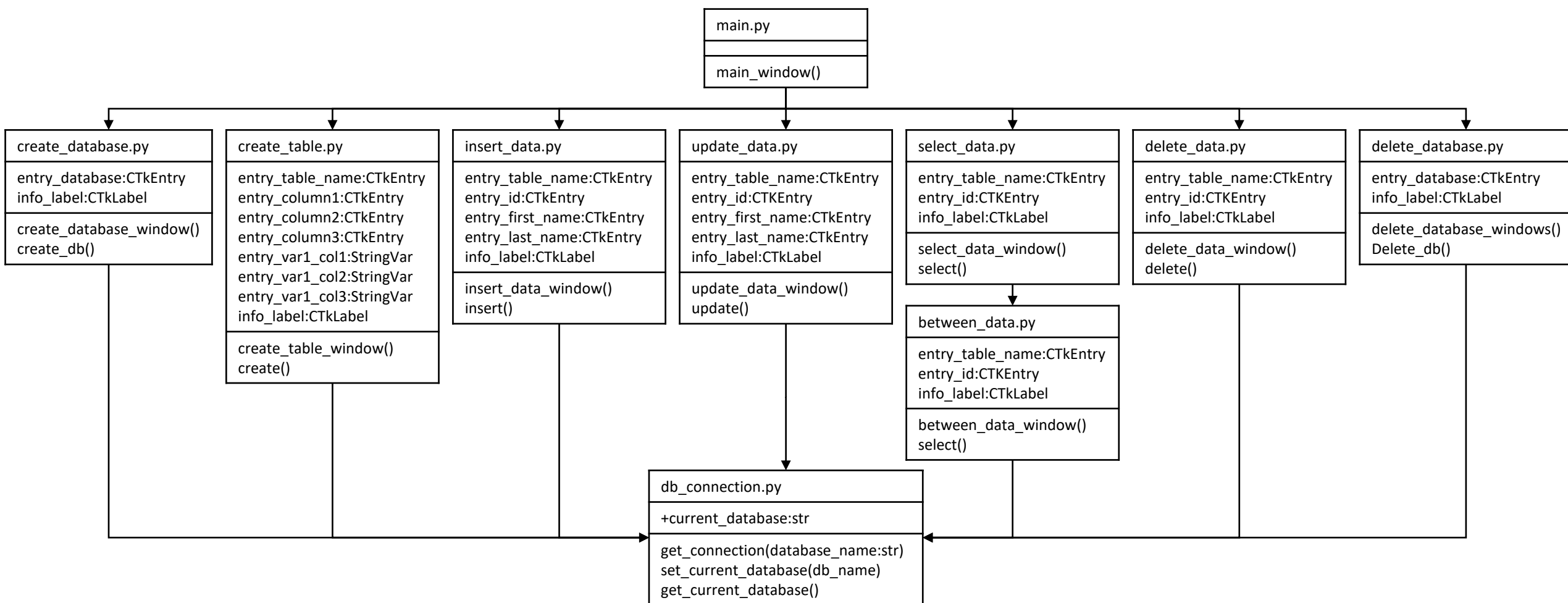
pyodbc是一個開源Python模組，可讓存取ODBC資料庫變得簡單。

customtkinter是一個基於Tkinter的Python桌面UI（User Interface）函式庫，它提供現代外觀和可自訂小工具的功能。

參考資料：

1. <https://pypi.org/project/pyodbc/>
2. <https://customtkinter.tomschimansky.com/>
3. <https://steam.oxxostudio.tw/category/python/tkinter/index.html>

程式碼重點解說



`main_Window()` 作為主要入口，連接所有功能視窗，各功能視窗依賴（以箭頭表示）

`db_connection` 管理資料庫連接，並包含特定的 UI 元素與操作方法。



主視窗 main.py

```
import customtkinter as tk
.....

def main_window():
    app = tk.CTk()                #產生視窗物件
    app.geometry("300x400")        #設定視窗大小
    app.title("資料庫管理系統")    #設定視窗標題
    .....

    tk.CTkButton(app,
                  text="創建資料庫",
                  command=create_database_window).pack(pady=10)
    .....

    app.mainloop()
```



pack()基本佈局

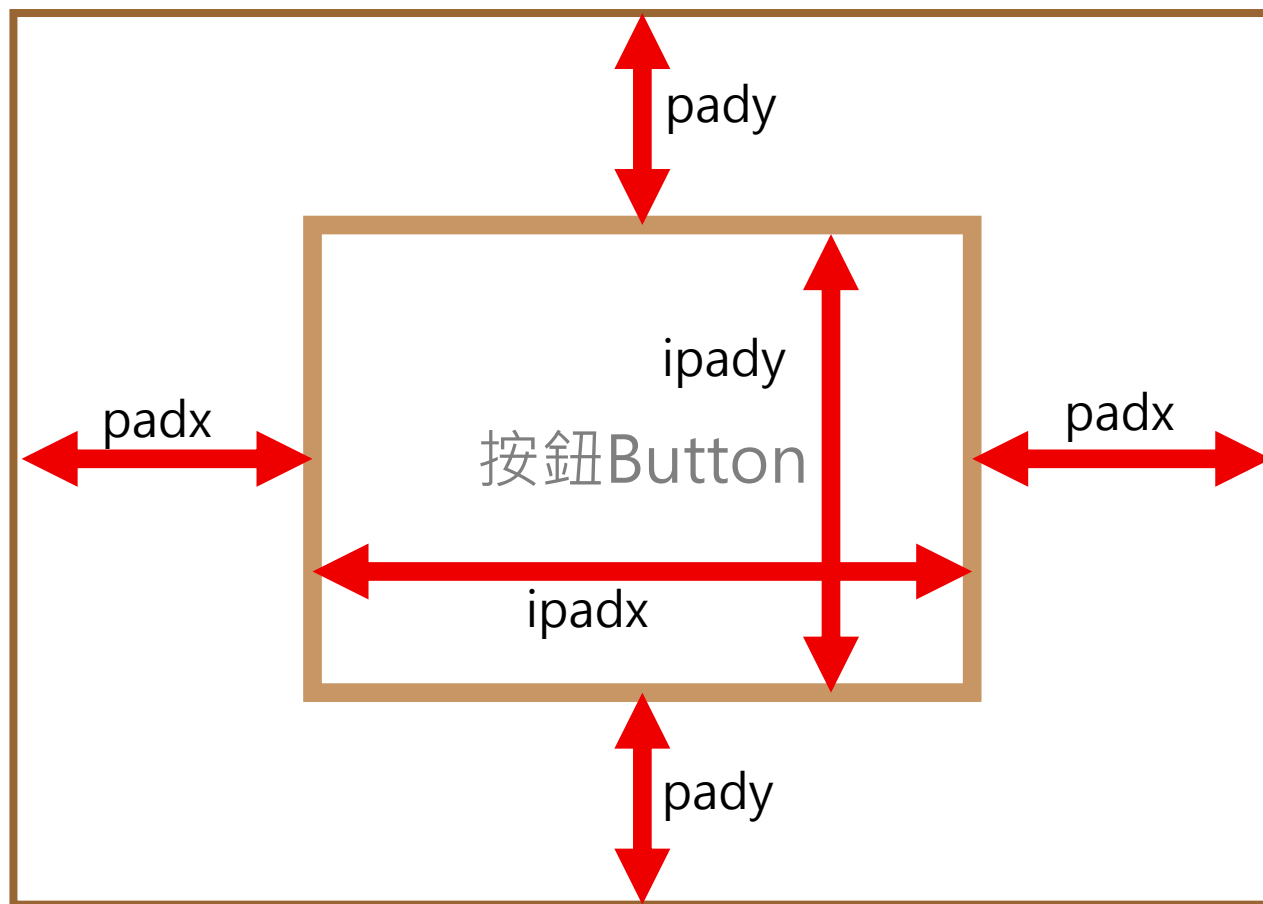
```
tk.CTkButton(app, text="abc").pack(pady=10)
```

padx 左、右**外**邊距

pady 上、下**外**邊距

ipadx 左、右**內**邊距

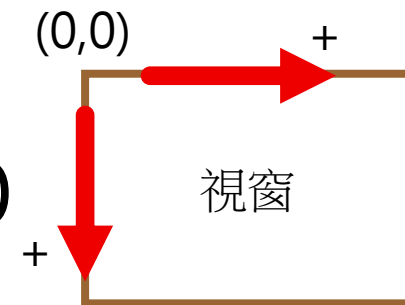
ipady 上、下**內**邊距



place()位置佈局

絕對位置

```
tk.CTkButton(app, text="A").place(x=50, y=50)
```



相對位置

```
tk.CTkButton(app, text="A").place(relx=0.5, rely=0.5)
```



連接資料庫 db_connection.py

```
import pyodbc
.....

def get_connection(database_name):
.....

pyodbc.connect(
    'DRIVER={SQL Server};' +
    'SERVER=主機名稱;' +
    f'DATABASE={database_name};' +
    'Trusted_Connection=True;')
.....
```



創建資料庫 create_database.py

```
Import pyodbc
```

```
.....
```

```
Def create_db():
```

```
.....
```

```
connection.autocommit = True
```

```
db_name = entry_database.get()
```

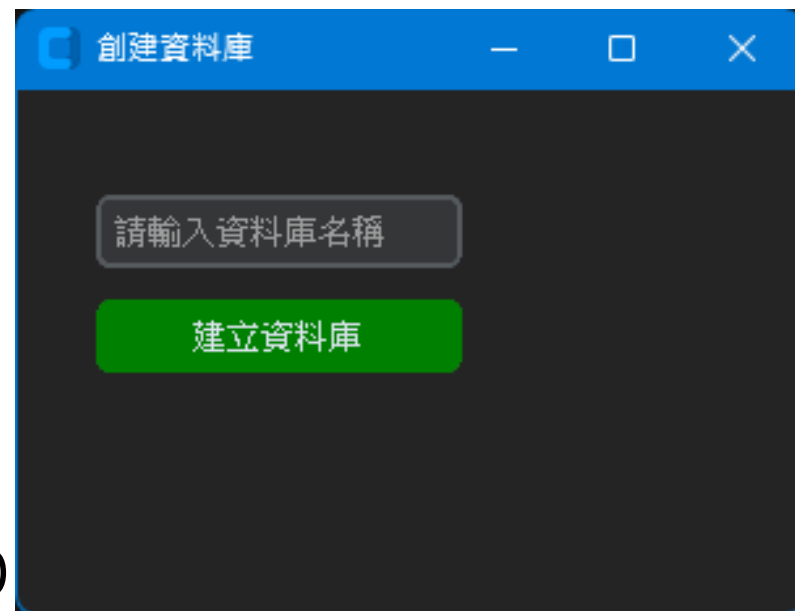
ch14

```
connection.execute(f"CREATE DATABASE {db_name}")
```

```
set_current_database(db_name)
```

```
info_label.configure(text="創建資料庫成功，當前資料庫： " + db_name)
```

```
.....
```





创建工作表 create_table.py

```
def create():
```

```
.....
```

```
connection = get_connection(current_db)
```

```
if connection:
```

```
★ connection.autocommit = True
```

```
sql_stmt = ( f"CREATE TABLE {entry_table_name.get()} ("
```

ch14

```
    f"{entry_column1.get()} {radio_var_col1.get()}, "
```

```
    f"{entry_column2.get()} {radio_var_col2.get()}, "
```

```
    f"{entry_column3.get()} {radio_var_col3.get()} ) " )
```

```
connection.execute(sql_stmt)
```

```
info_label.configure(text="創建資料表成功")
```

```
.....
```

```
CREATE TABLE students ( id INTEGER, first_name VARCHAR(50), last_name VARCHAR(50) )
```



新增資料 insert_data.py

```
def insert():
```

```
.....
```

```
cursor = connection.cursor()
```

```
cursor.execute(
```

```
f"SELECT COUNT(*)
```

ch11

```
FROM {entry_table_name.get()}
```

```
WHERE id = {entry_id.get()}")
```

★ if cursor.fetchone()[0] > 0: # 防呆

```
    info_label.configure(text="資料已存在，請勿重複新增")
```

```
    return
```

```
.....
```

```
SELECT COUNT(*) FROM stu WHERE id = 1
```



新增資料 insert_data.py

```
def insert():
.....
connection.autocommit = True
connection.execute(
f"INSERT INTO {entry_table_name.get()}
VALUES ({entry_id.get()},
' {entry_first_name.get()} ',
' {entry_last_name.get()} ')"
)
info_label.configure(text="新增資料成功")
.....
```

ch13

```
INSERT INTO students VALUES (1, '小明', '王')
```



修改資料 update_data.py

```
def update():
.....

connection.autocommit = True
connection.execute(
    f"UPDATE {entry_table_name.get()} "
    f"SET first_name = '{entry_first_name.get()}',
        last_name = '{entry_last_name.get()}' "
    f"WHERE id = {entry_id.get()}"
)
info_label.configure(text="新增資料成功")
.....
```

ch13

```
UPDATE stu SET first_name = '大明', last_name = '許' WHERE id = 1
```



查詢資料 select_data.py

```
def select():
```

```
.....
```

① `cursor = connection.cursor()`

```
cursor.execute(f"SELECT *
```

ch11

```
FROM {entry_table_name.get()}
```

```
WHERE id = {entry_id.get()} AND {entry_id_2.get()}")
```

```
result = ""
```

```
for data in cursor:
```

```
    result += f"{data[0]} {data[1]} {data[2]}\n"
```

```
info_label.configure(text=result if result else "查無資料")
```

```
.....
```

②

```
SELECT * FROM stu WHERE id = 1 AND 2
```




刪除資料 delete_data.py

```
def delete():  
.....  
  
connection.autocommit = True  
connection.execute(  
    f"DELETE FROM {entry_table_name.get()}  
    WHERE id = {entry_id.get()}")  
info_label.configure(text="刪除成功")  
.....
```

ch13

```
DELETE FROM stu WHERE id = 1
```



刪除資料庫 delete_database.py

```
def delete_db():
```

```
.....
```

```
connection.autocommit = True
```

```
db_name = entry_database.get()
```

❶ `connection.execute(`

```
f"""ALTER DATABASE {db_name}  # 設為單一用戶模式以便刪除
```

```
SET SINGLE_USER WITH ROLLBACK IMMEDIATE;""")
```

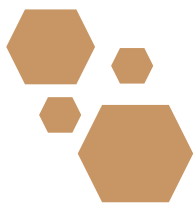
❷ `connection.execute(f"DROP DATABASE {db_name}")`

```
.....
```

```
ALTER DATABASE [test] SET SINGLE_USER WITH ROLLBACK IMMEDIATE;  
DROP DATABASE test
```



ch14



感謝聆聽
敬請指教

