

# 使用資料庫 (PostgreSQL)

- 主要參考文件
  - Database in a Docker container — how to start and what's it about
    - <https://wkrzywiec.medium.com/database-in-a-docker-container-how-to-start-and-whats-it-about-5e3ceea77e50>
  - Using PostgreSQL through SQLAlchemy
    - <https://www.compose.com/articles/using-postgresql-through-sqlalchemy/>
  - PostgreSQL: Documentation: 10: Chapter 25. Backup and Restore
    - <https://www.postgresql.org/docs/10/backup.html>
- 先透過圖形化資料庫管理工具，DBeaver，來認識並試操作資料庫
- 藉由python第三方模組(SQLAlchemy)的幫助，執行程式對資料庫進行操作，程式有三種寫法
  - Raw SQL (最底層，sql的原始語法)
  - SQLAlchemy Expression Language (中間層，python的一般語法)
  - SQLAlchemy ORM (最高層，python的物件導向語法)
- 最終測試
  - 用python程式讀取csv檔案，將檔案內的資料轉移到個人電腦上的資料庫
  - 下linux指令，將存放在個人電腦上的資料庫複製到架設在威騰的nas上的資料庫

# 資料庫管理軟體的名稱 (DBeaver)

DBeaver 21.0.0 - 協和馬祖珠山\_A1\_26-16094\_119-93583.csv

文件(F) 编辑(E) 导航(N) Search SQL 编辑器 数据库(D) 窗口(W) 帮助(H)

60.249.144.60 public@windy\_powerplants Auto 60.249.144.60 windy\_powerplants 模式 public 表 協和馬祖珠山\_A1\_26-16094\_119-93583.csv

数据库导航 项目

输入表格名称的一部分

60.249.144.60 (60.249.144.60:5433) **連線網址**

- postgres
  - windy\_powerplants **資料庫名稱**
    - 模式
      - public
        - films 16K
        - persons 32K
        - 協和\_A1\_25-16100\_121-74080.csv 776K
        - 協和\_A2\_25-16183\_121-743716.csv 776K
        - 協和\_A3\_25-160329\_121-747333.csv 776K
        - 協和\_B\_25-170528\_121-73925.csv 776K
        - 協和\_放流口\_25-157744\_121-743672.csv 776K
        - 協和馬祖珠山\_A1\_26-16094\_119-93583.csv 776K**
        - 協和馬祖珠山\_A2\_26-16010\_119-938578.csv 776K
        - 協和馬祖珠山\_A3\_26-158164\_119-940275.csv 776K
        - 協和馬祖珠山\_B\_26-169983\_119-936261.csv 776K
        - 協和馬祖珠山\_放流口\_26-15644\_119-935624.csv 776K
        - 南部\_A1\_22-59363\_120-30057.csv 776K
        - 南部\_A2\_22-60510\_120-295366.csv 776K
        - 南部\_A3\_22-60623333\_120-2950833.csv 776K

某表格

Project - General

Name DataSource

- Bookmarks
- ER Diagrams
- Scripts

協和馬祖珠山\_A1\_26-16094\_119-93583.csv

输入一个 SQL 表达式来过滤结果 (使用 Ctrl+Space)

abc time	abc Sea temperature	abc currents	abc waves
time	Sea temperature	Currents	Waves
2020/11/02 10:38:02	22°C	NW 0.3 kt	NE 1.5m Period 6 s.
2020/11/02 11:01:24	22°C	NW 0.3 kt	NE 1.5m Period 6 s.
2020/11/02 12:01:24	22°C	NW 0.3 kt	NE 1.5m Period 6 s.
2020/11/02 13:01:34	22°C	NW 0.3 kt	NE 1.4m Period 6 s.
2020/11/02 14:01:25	22°C	NW 0.3 kt	NE 1.4m Period 6 s.
2020/11/02 15:01:27	22°C	NW 0.3 kt	NE 1.4m Period 6 s.
2020/11/02 16:01:26	22°C	NW 0.3 kt	E 1.5m Period 6 s.
2020/11/02 17:01:25	22°C	NW 0.3 kt	E 1.5m Period 6 s.
2020/11/02 18:01:24	22°C	NW 0.3 kt	E 1.5m Period 6 s.
2020/11/02 19:01:34	22°C	NW 0.3 kt	NE 1.8m Period 6 s.
2020/11/02 20:01:24	22°C	NW 0.3 kt	NE 1.8m Period 6 s.
2020/11/02 21:01:24	22°C	NW 0.3 kt	NE 1.8m Period 6 s.
2020/11/02 22:01:25	22°C	NW 0.3 kt	NE 1.9m Period 6 s.
2020/11/02 23:01:25	22°C	NW 0.3 kt	NE 1.9m Period 6 s.
2020/11/03 00:01:27	22°C	NW 0.3 kt	NE 1.9m Period 6 s.
2020/11/03 01:01:33	22°C	NW 0.3 kt	NE 1.8m Period 6 s.
2020/11/03 02:01:26	22°C	NW 0.3 kt	NE 1.8m Period 6 s.
2020/11/03 03:01:25	22°C	NW 0.3 kt	NE 1.8m Period 6 s.
2020/11/03 04:01:25	22°C	NW 0.3 kt	NE 1.8m Period 6 s.
2020/11/03 05:01:26	22°C	NW 0.3 kt	NE 1.8m Period 6 s.
2020/11/03 06:01:28	22°C	NW 0.3 kt	NE 1.8m Period 6 s.
2020/11/03 07:01:36	22°C	NW 0.3 kt	NE 1.9m Period 6 s.
2020/11/03 08:01:27	22°C	NW 0.3 kt	NE 1.9m Period 6 s.
2020/11/03 09:01:27	22°C	NW 0.3 kt	NE 1.9m Period 6 s.
2020/11/03 10:01:26	22°C	NW 0.3 kt	NE 2m Period 6 s.
2020/11/03 11:01:26	22°C	NW 0.3 kt	NE 2m Period 6 s.

数值查看器 Sea temperature

Rows: 200

## 3 種在 SQLAlchemy 模組下的寫程式方式 : 建表並填入一筆資料

```
1  ### (1) Raw SQL ###
2  from sqlalchemy import create_engine
3
4  db_string = "postgres://postgres:postgres@localhost:5432/postgres"
5  db = create_engine(db_string)
6
7  # Create
8  db.execute("CREATE TABLE IF NOT EXISTS films (title text, director text, year text)")
9  db.execute("INSERT INTO films (title, director, year) VALUES ('Doctor Strange', 'Scott Derrickson', '2016')")
10
11 ### (2) the SQLAlchemy Expression Language ###
12 from sqlalchemy import create_engine
13 from sqlalchemy import Table, Column, String, MetaData
14
15 db_string = "postgres://postgres:postgres@localhost:5432/postgres"
16 db = create_engine(db_string)
17
18 meta = MetaData(db)
19 film_table = Table('films', meta,
20                   Column('title', String),
21                   Column('director', String),
22                   Column('year', String))
23
24 with db.connect() as conn:
25
26     # Create
27     film_table.create(checkfirst=True)
28     insert_statement = film_table.insert().values(title="Doctor Strange", director="Scott Derrickson", year="2016")
29     conn.execute(insert_statement)
30
31 ### (3) the SQLAlchemy ORM ###
32 from sqlalchemy import create_engine
33 from sqlalchemy import Column, String
34 from sqlalchemy.ext.declarative import declarative_base
35 from sqlalchemy.orm import sessionmaker
36
37 db_string = "postgres://postgres:postgres@localhost:5432/postgres"
38
39 db = create_engine(db_string)
40 base = declarative_base()
41
42 class Film(base):
43     __tablename__ = 'films'
44
45     title = Column(String, primary_key=True)
46     director = Column(String)
47     year = Column(String)
48
49 Session = sessionmaker(db)
50 session = Session()
51 base.metadata.create_all(db)
52
53 # Create
54 doctor_strange = Film(title="Doctor Strange", director="Scott Derrickson", year="2016")
55 session.add(doctor_strange)
56 session.commit()
```

# 安裝資料庫和相關工具

- 安裝資料庫，以 PostgreSQL 為例（以 docker 容器呈現，避免影響到作業系統）
  - `docker run --name postgres-docker -e POSTGRES_PASSWORD=postgres -p 5432:5432 -d postgres:10`
- 安裝一般的資料庫管理工具
  - DBeaver
- 安裝 SQLAlchemy 和相關驅動程式（給 python 程式語言使用）
  - `pip3 install sqlalchemy`
  - `pip3 install psycopg2`（在我的 ubuntu linux 系統上安裝失敗）
    - `pip3 install psycopg2-binary`（改成直接安裝編譯好的執行檔就成功了）