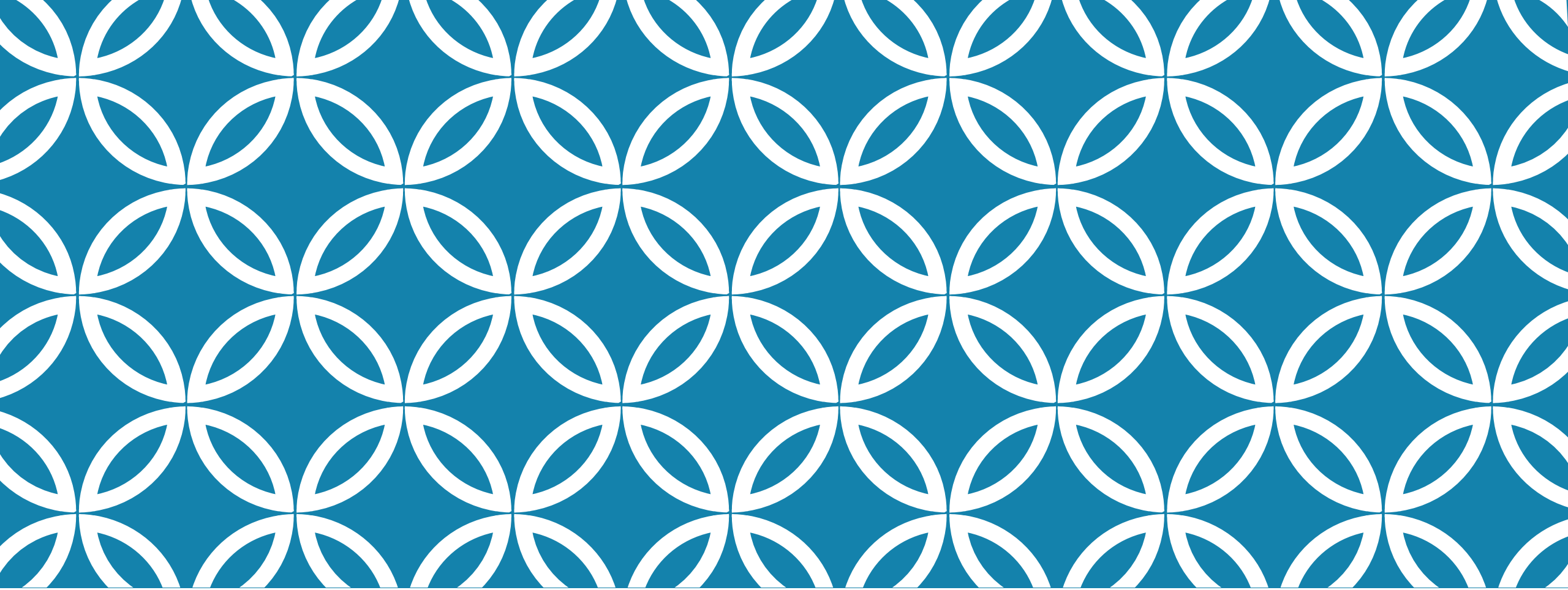


mongoDB®

朱克剛

朱克剛

# 非關連式資料庫

MongoDB 屬於非關連式資料庫（No-SQL）中的文本資料庫

只能儲存 JSON 資料

一份文件：

```
{
    "name": "David"
}
```

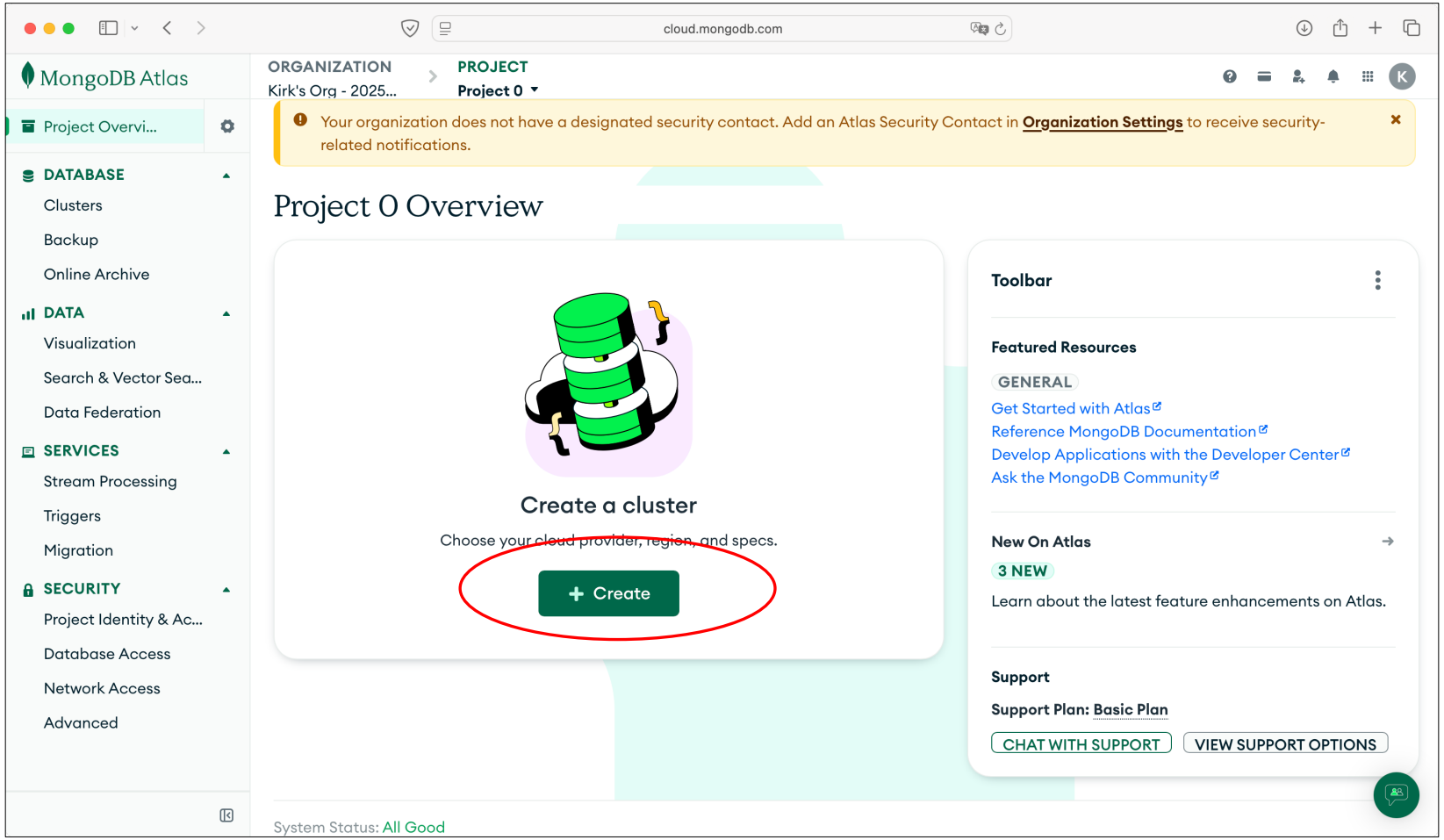
兩份文件：

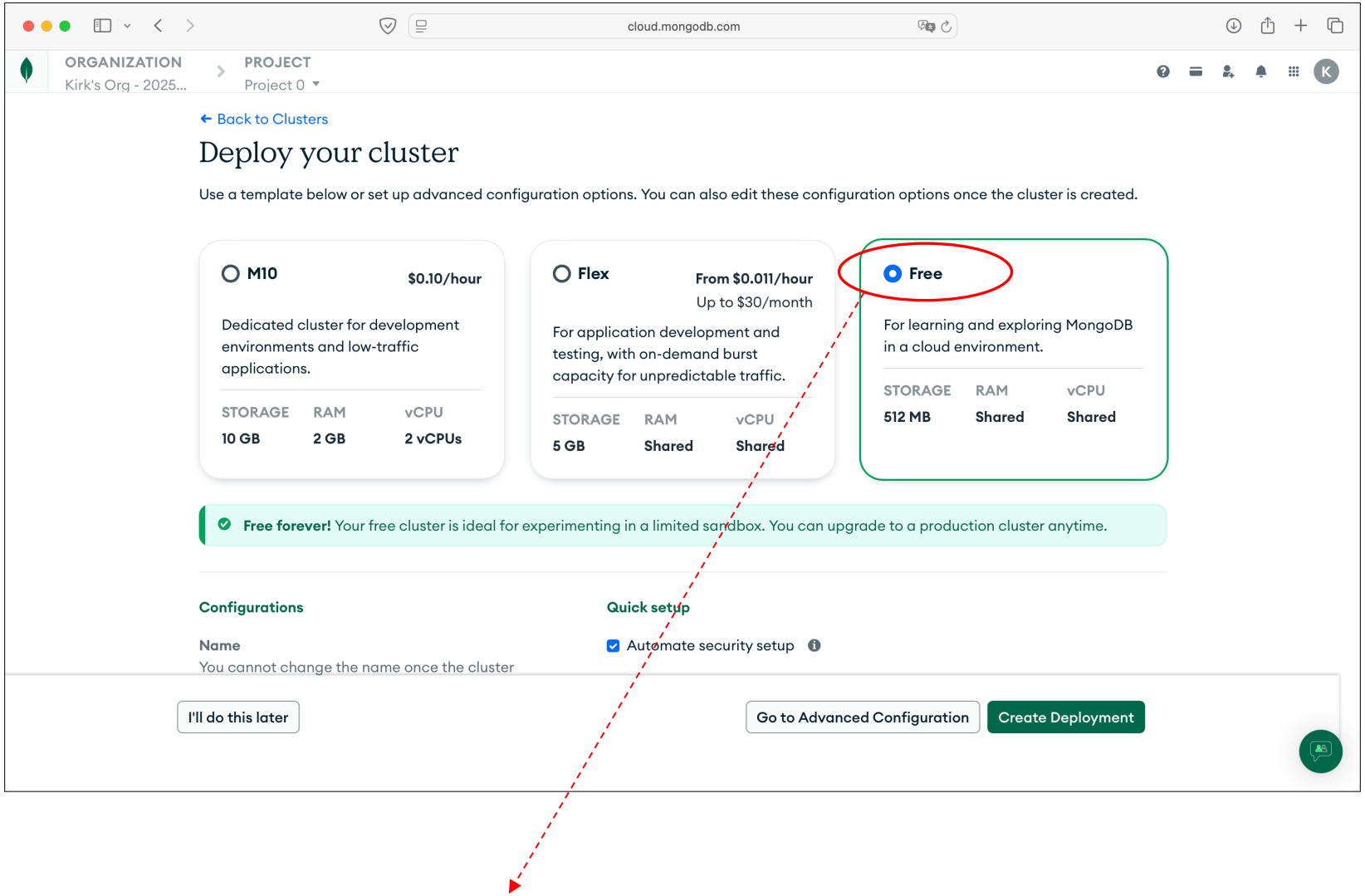
```
[
    { "name": "David" },
    { "name": "Betty" }
]
```

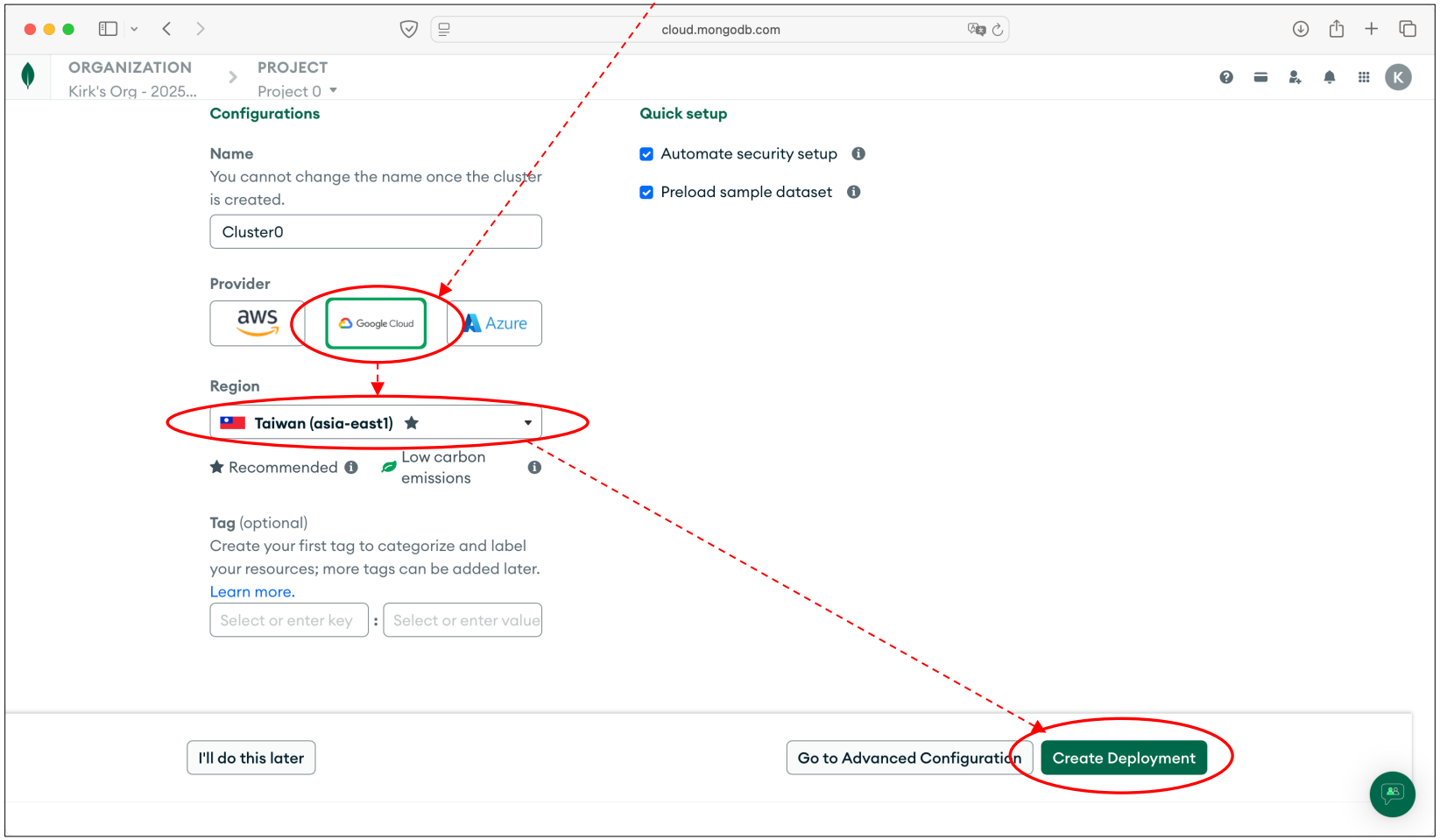
# 與關連式資料庫差異

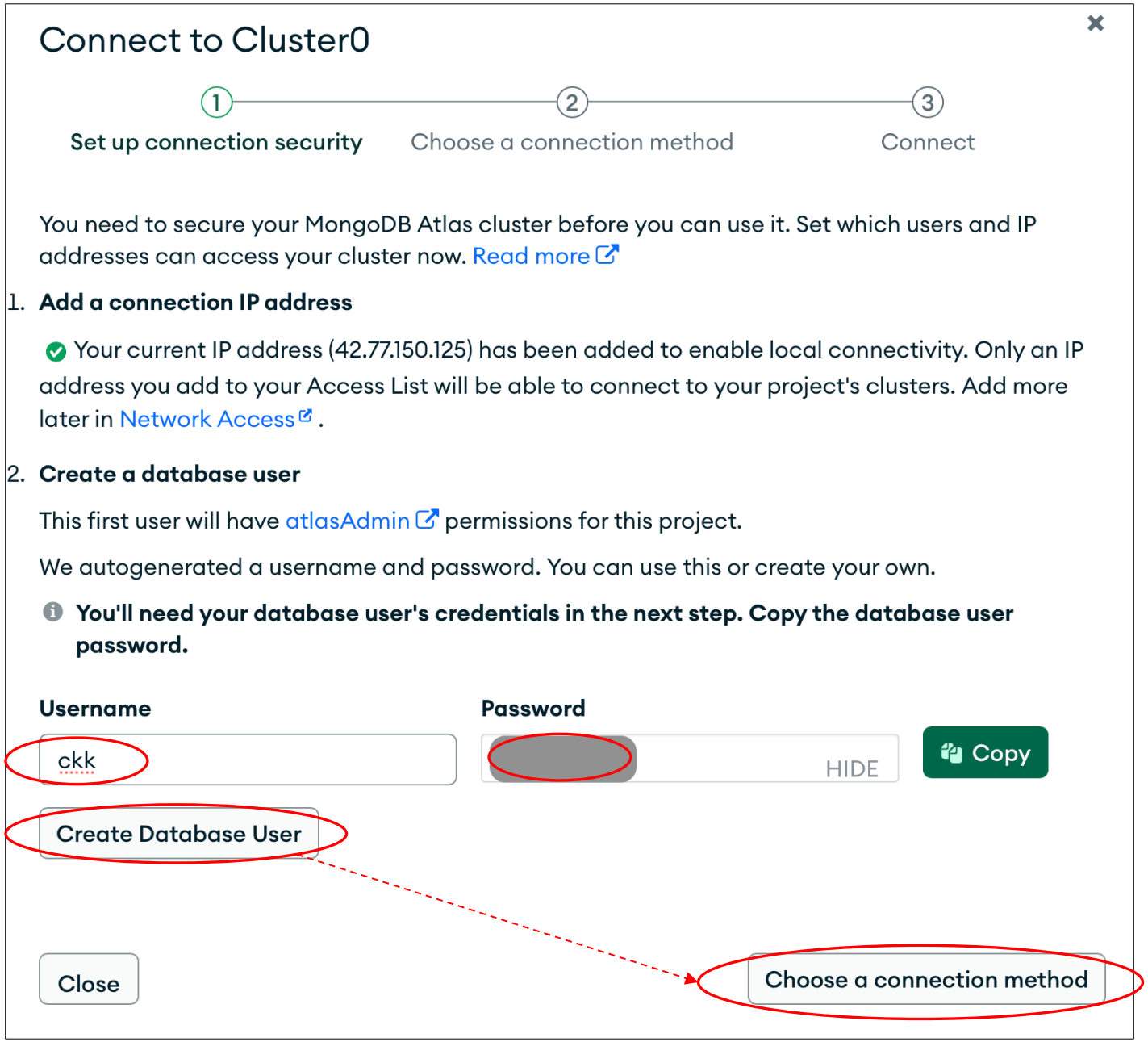
| | MongoDB | 關連式資料庫 |
|---|---|---|
| 資料庫建置時間 | 短 | 長 |
| 資料表能否儲存不同格式資料 | ✅ | ❌ |
| 資料表之間的關連性 | 幾乎沒有 | 大部分都有 |
| 欄位設定 | ❌ | ✅ |
| 正規化 | ❌ | ✅ |
| PK、FK | ❌ | ✅ |
| 預存程序 | ❌ | ✅ |
| 分散式叢集設定 | 易 | 難 |
| 操作指令 | JSON | SQL指令 |
| 儲存空間需求 | 大 | 小 |

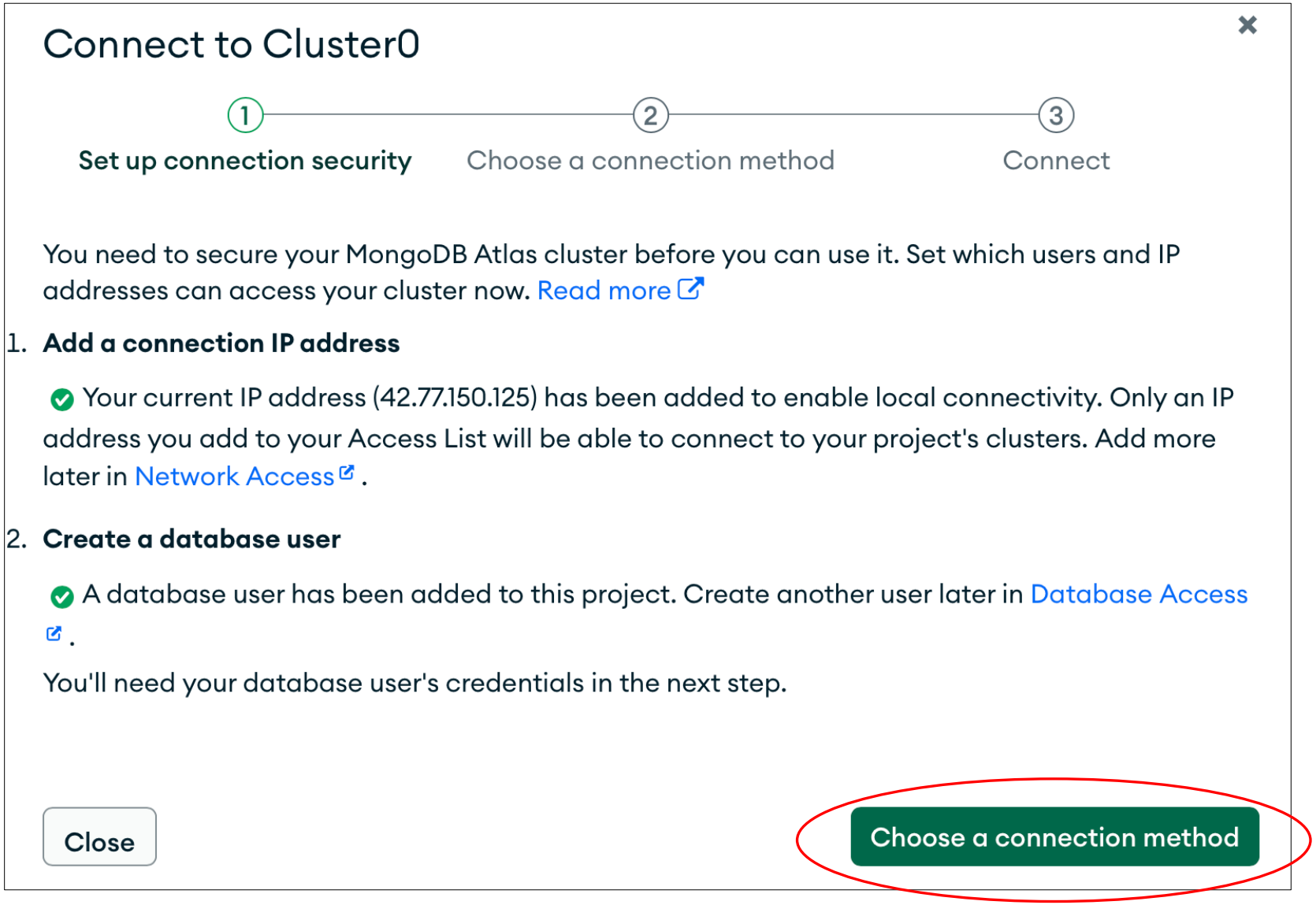
# 安裝雲端SERVER - ATLAS




cloud.mongodb.com
ORGANIZATION
Kirk's Org - 2025...
PROJECT
Project 0
Back to Clusters
Deploy your cluster
Use a template below or set up advanced configuration options. You can also edit these configuration options once the cluster is created.
M10
$0.10/hour
Dedicated cluster for development environments and low-traffic applications.
STORAGE
10 GB
RAM
2 GB
vCPU
2 vCPUs
Flex
From $0.011/hour
Up to $30/month
For application development and testing, with on-demand burst capacity for unpredictable traffic.
STORAGE
5 GB
RAM
Shared
vCPU
Shared
Free
For learning and exploring MongoDB in a cloud environment.
STORAGE
512 MB
RAM
Shared
vCPU
Shared
Free forever! Your free cluster is ideal for experimenting in a limited sandbox. You can upgrade to a production cluster anytime.
Configurations
Name
You cannot change the name once the cluster
Quick setup
Automate security setup
I'll do this later
Go to Advanced Configuration
Create Deployment


cloud.mongodb.com
ORGANIZATION
Kirk's Org - 2025...
PROJECT
Project 0
Configurations
Name
You cannot change the name once the cluster is created.
Cluster0
Provider
aws
Google Cloud
Azure
Region
Taiwan (asia-east1)
Recommended
Low carbon emissions
Tag (optional)
Create your first tag to categorize and label your resources; more tags can be added later.
Learn more.
Select or enter key
Select or enter value
Quick setup
Automate security setup
Preload sample dataset
I'll do this later
Go to Advanced Configuration
Create Deployment


Connect to Cluster0
1
2
3
Set up connection security
Choose a connection method
Connect
You need to secure your MongoDB Atlas cluster before you can use it. Set which users and IP addresses can access your cluster now. Read more
1. Add a connection IP address
Your current IP address (42.77.150.125) has been added to enable local connectivity. Only an IP address you add to your Access List will be able to connect to your project's clusters. Add more later in Network Access .
2. Create a database user
This first user will have atlasAdmin permissions for this project.
We autogenerated a username and password. You can use this or create your own.
You'll need your database user's credentials in the next step. Copy the database user password.
Username
Password
ckk
HIDE
Copy
Create Database User
Close
Choose a connection method


Connect to Cluster0
1
2
3
Set up connection security
Choose a connection method
Connect
You need to secure your MongoDB Atlas cluster before you can use it. Set which users and IP addresses can access your cluster now. Read more
1. Add a connection IP address
Your current IP address (42.77.150.125) has been added to enable local connectivity. Only an IP address you add to your Access List will be able to connect to your project's clusters. Add more later in Network Access .
2. Create a database user
A database user has been added to this project. Create another user later in Database Access .
You'll need your database user's credentials in the next step.
Close
Choose a connection method

# Connect to Cluster0

Set up connection security — 2 Choose a connection method — 3 Connect

## Connect to your application

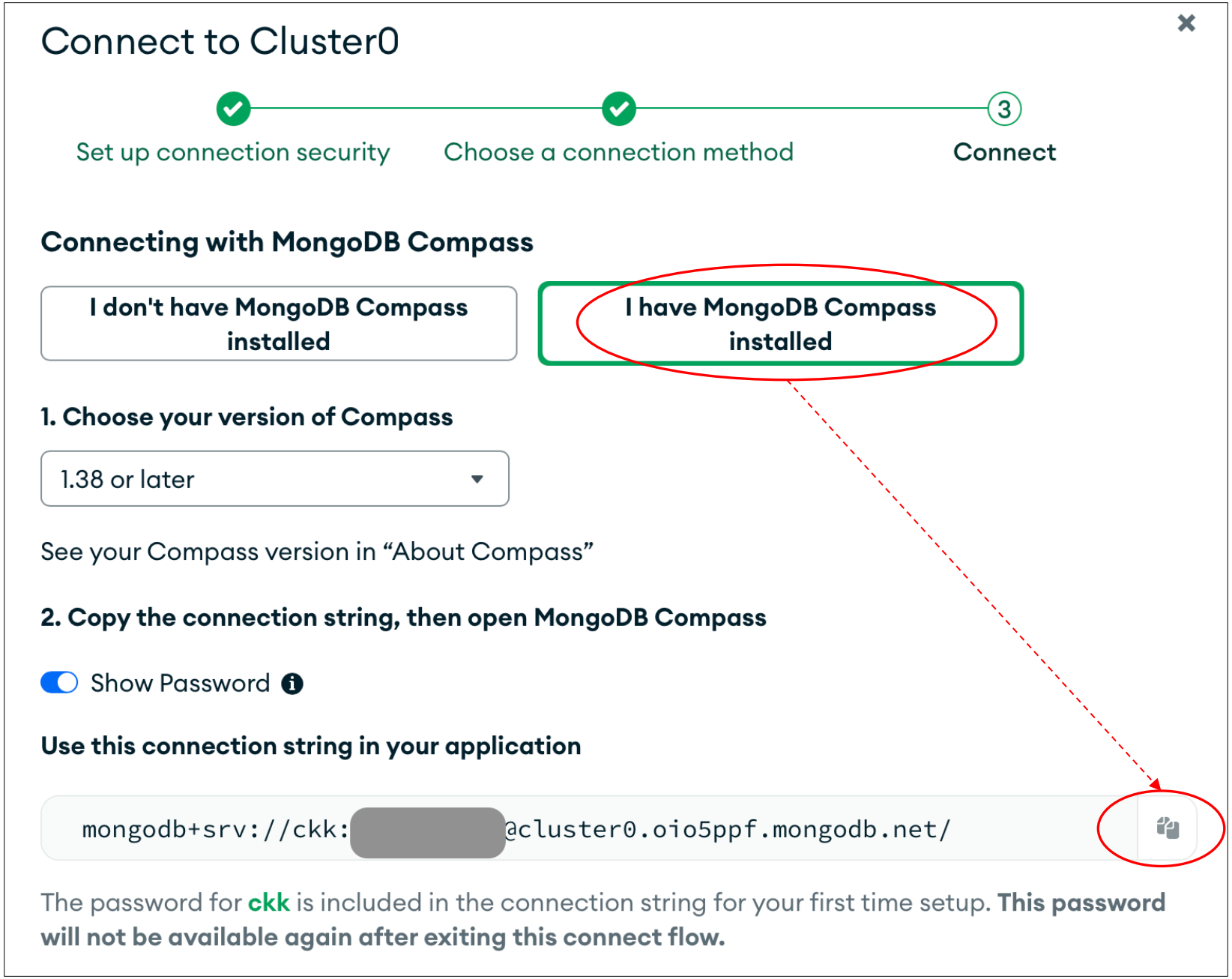
**Drivers**
Access your Atlas data using MongoDB's native drivers (e.g. Node.js, Go, etc.)

## Access your data through tools

**Compass**
Explore, modify, and visualize your data with MongoDB's GUI

**Shell**
Quickly add & update data using MongoDB's Javascript command-line interface

**MongoDB for VS Code**


Connect to Cluster0
Set up connection security
Choose a connection method
3
Connect
Connecting with MongoDB Compass
I don't have MongoDB Compass installed
I have MongoDB Compass installed
1. Choose your version of Compass
1.38 or later
See your Compass version in “About Compass”
2. Copy the connection string, then open MongoDB Compass
Show Password
Use this connection string in your application
mongodb+srv://ckk:[redacted]@cluster0.oio5ppf.mongodb.net/
The password for ckk is included in the connection string for your first time setup. This password will not be available again after exiting this connect flow.

# 本地安裝 SERVER

## ◆Windows

下載 .msi 檔後執行（過程中有一個步驟要選擇「不要安裝Compass」）

預設安裝路徑為 C:\Program Files\MongoDB\Server\[版本]\bin

- 建議將此路徑加到環境變數 PATH 中

## ◆Mac / Linux

下載 .tgz 並解開（tar zxvf 檔名）後將 mongo、mongod、mongos 檔案移到 /usr/local/bin

建立資料儲存目錄

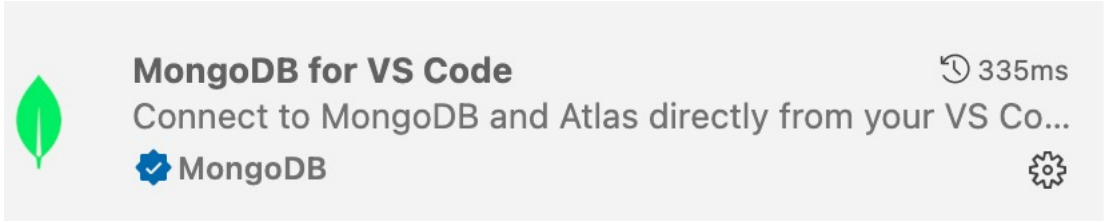
- mkdir -p ~/data/db （資料放哪裡都可以）

# 管理方式

Compass – GUI 介面

mongosh – 純文字介面

VS Code – 寫程式專用

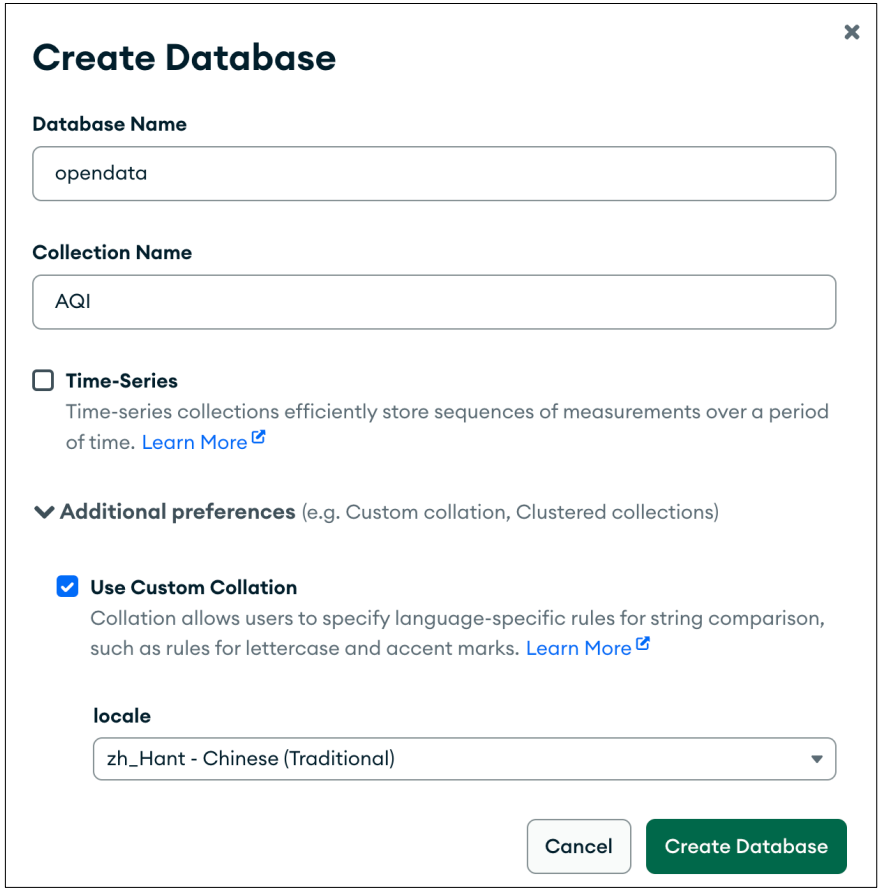



Compass 與 VS Code 都有 AI 可以輔助

# 準備資料

資料來源：https://github.com/kirkchu/mongodb/blob/main/aqitutorial.json

建立 opendata （小寫）資料庫，再建立 AQI（大寫） 資料表，匯入資料




MongoDB稱為Collection

改資料表定序，中文排序可按照筆畫數排

# _ID

MongoDB 會為每一筆資料自動加上 _id 欄位，透過 unique index 限制值唯一

類似關連式資料庫的 Primary Key，但可以接受 NULL

```
_id: ObjectId('680af1903b5100522be1a7af')
sitename : "基隆"
county : "基隆市"
```

預設 ObjectId 包含了時間，可透過特定函數知道該筆資料何時建立

# 基本查詢 – FIND

◆查詢 AQI 資料表中所有資料

```
db.AQI.find({})
```

◆只顯示部分欄位：1表示要，0表示不要

```
db.AQI.find({}, {"county": 1, "aqi": 1 , "_id": -1})
```

相當於 where

相當於 select

# 查詢條件

find() 列出所有符合條件的資料
findOne() 列出所有符合條件的第一筆

◆只列出臺中市的資料

```
db.AQI.find({"county": "臺中市"})
```

◆列出雲林縣與新北市資料

```
db.AQI.find({
    "$or": [
        {"county": "雲林縣"},
        {"county": "新北市"}
    ]
})
```

◆條件為同一個欄位時也可以使用 $in

```
db.AQI.find({"county": {"$in": ["雲林縣", "新北市"]}})
```

# 比較運算

◆查詢 aqi 大於等於 30 的資料

```
db.AQI.find({"aqi": {"$gte": 30}})
```

◆查詢 aqi 為30到50的資料

```
{"aqi": {"$gte": 30, "$lte": 50}}
```

| 運算子 | 說明 |
|---|---|
| $gt | > |
| $gte | ≥ |
| $lt | < |
| $lte | ≤ |
| $ne | ≠ |

# 模糊查詢

- 使用正規表示法做模糊查詢
- 查詢 county 開頭為「新」的資料

```
db.AQI.find({
    "county": {"$regex": "^新"}
})
```

^新：字串開頭為「新」
新$：字串結尾為「新」
新|南：有「新」或者「南」

# 排序

◆按照 aqi 順向排序

```
db.AQI.find().sort({"aqi": 1})
```

◆逆向排序

```
db.AQI.find().sort({"aqi": -1})
```

◆按照中文筆畫數對 sitename 做排序（預設為 Unicode 排序）

```
db.AQI.find().sort({"sitename": 1})
              .collation({"locale": "zh_Hant"})
```

# AGGREGATION – 聚合查詢

- 相當於關連式資料庫的「子查詢」與「群組運算」
- 根據 county 做群組運算，計算每個 county 的平均aqi，四捨五入取整數，最後結果排序，並僅顯示 county 與平均aqi

```
select
  county,
  round(avg(aqi), 0) as average_aqi
from AQI
group by county
order by average_aqi
```

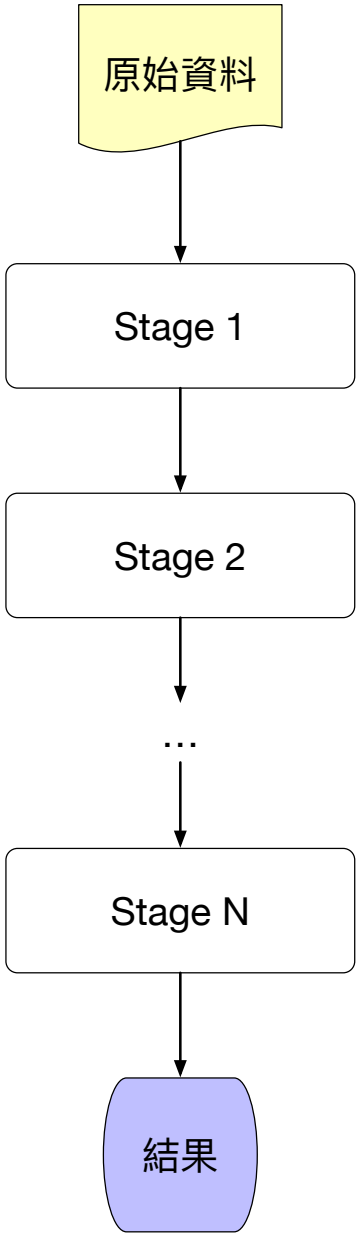
pipeline

```
[
  { $group: {
      _id: "$county", average_aqi: { $avg: "$aqi" }
    }
  },
  { $project: {
      county: "$_id", average_aqi: { $round: ["$average_aqi", 0] }, _id: 0
    }
  },
  { $sort: { average_aqi: 1 } }
]
```

# PIPELINE

- 將資料經過各階段的處理產生最後要的結果，下一階段的輸入就是上一階段的輸出
- 限制資料大小的 stage 放前面
  - 例如：列出 county 為新北市與臺北市的平均aqi

$match $group



# _ID 為常數或NULL

- _id 為群組欄位，如果內容為常數或 null，表示所有資料一個群組
- 例如：計算共有多少資料

```
[
  { $count: "totalRecords" }
]
```

- 計算全部 aqi 平均值

```
[
  { $group: {
      _id: null,
      avg_aqi: { $avg: "$aqi" }
    }
  }
]
```

# 桶型分類 – $bucket

◆根據 aqi 進行桶型分類，區間為：0, 51, 101, 151, 201, 301, 1000

```
[
  { $bucket: {
      groupBy: "$aqi",
      boundaries: [ 0, 51, 101, 151, 201, 301, 1000 ],
      default: "Others",
      output: { count: {
          $sum: 1
        }
      }
    }
  }
]
```

無法分類的資料要放在哪一桶中

# 左側外部連結 – $LOOKUP

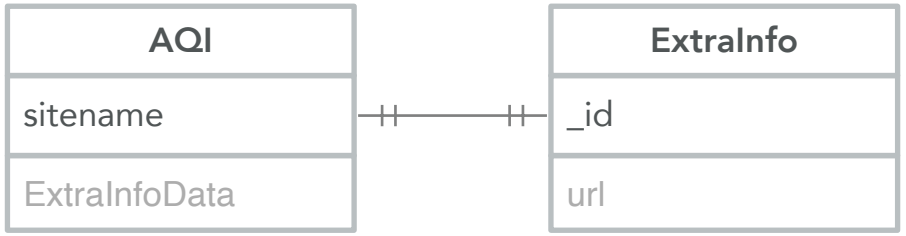
◆如果有兩個資料表且有關連時，可以使用左側外部連結

◆若 opendata 中有另外一個資料表 ExtraInfo，內有一筆文件

```
{
  "_id": "汐止",
  "url": "https://info.tw/汐止"
}
```

◆依據 AQI.sitename = ExtraInfo._id 將 AQI 與 ExtraInfo 合併，合併後查詢 county 為新北市資料

```
db.AQI.aggregate([
  { $match: { county: "新北市" } },
  { $lookup: {
      from: "ExtraInfo",
      localField: "sitename",
      foreignField: "_id",
      as: "ExtraInfoData"
    }
  }
])
```



# 極端值查詢

- 查詢 aqi 值最大的資料（使用 $lookup 指令），需考慮最大值有多筆
- AI 生成的作法很多，有些有錯，因為最大值資料可能多筆

```
[
  { $group: {
      _id: null,
      maxAqi: { $max: "$aqi" }
    }
  },
  { $lookup: {
      from: "aqi",
      localField: "maxAqi",
      foreignField: "aqi",
      as: "maxAqiDocs"
    }
  }
]
```

# 地理座標查詢 1/3

- 將經緯度值轉成 geojson 格式放入 geometry 欄位，並且輸出到 geoaqi 資料表
- 對 geoaqi 中的 geometry.coordinates 建立 2D sphere 索引
- 查詢以虎尾科技大學為中心，半徑30公里內的資料

使用 find()

```
{
  "geometry.coordinates": {
    "$geoWithin": {
      "$centerSphere": [ [120.4374, 23.7077], 30 / 6378.1 ]
    }
  }
}
```

AI產生的座標需要核對，不一定精準

# 地理座標查詢 2/3

AI產生的座標需要核對，不一定精準

◆查詢新竹以北的所有資料

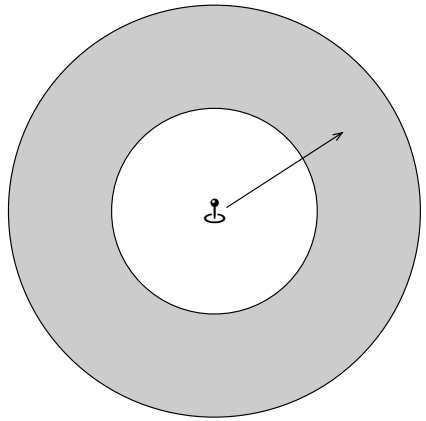
使用 find()

```
{ "geometry.coordinates.1": { $gte: 24.8138 } }
```

◆以台灣地理中心為準，使用nearSphere函數查詢半徑50-100公里範圍內的資料

使用 find()

```
{
  "geometry.coordinates": {
    $nearSphere: {
      $geometry: {
        type: "Point",
        coordinates: [120.973881, 23.583234]
      },
      $minDistance: 50000,
      $maxDistance: 100000
    }
  }
}
```



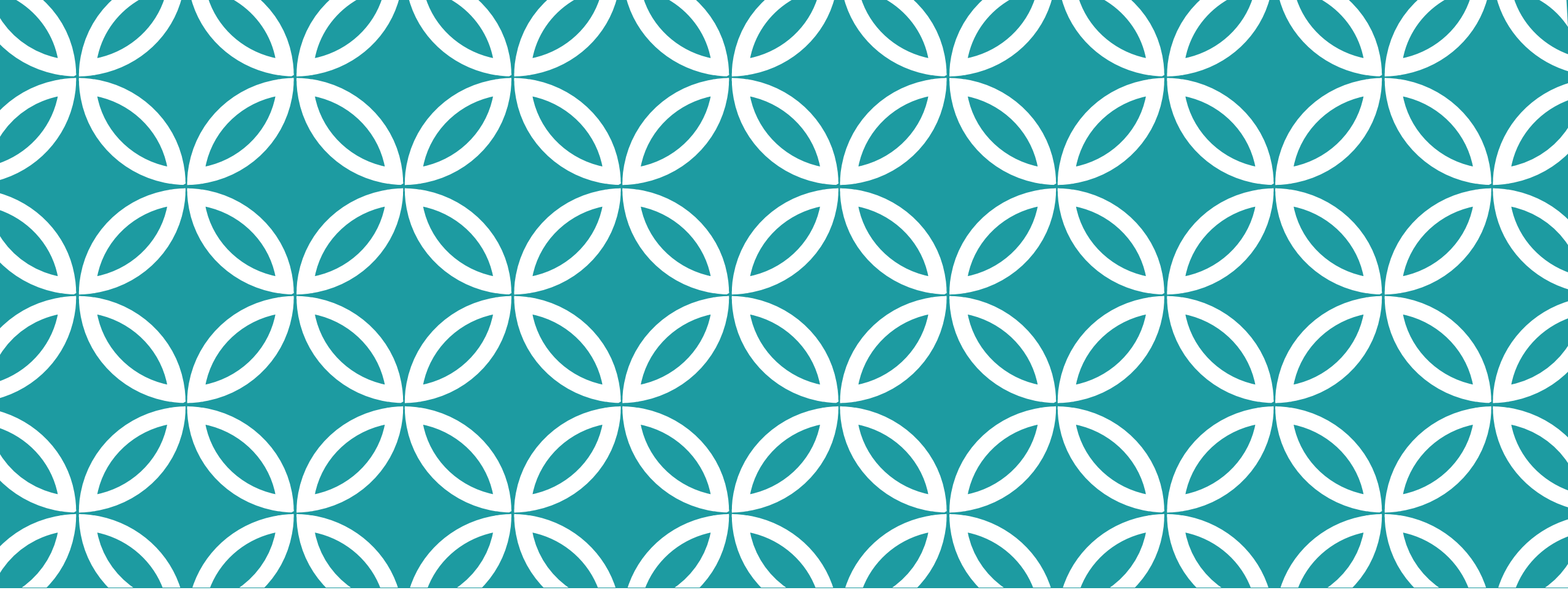
# 地理座標查詢 2/3

◆根據geometry欄位，找出離「淡水」最遠的資料

使用 aggregate()

```
[
  { $geoNear: {
      near: {
        type: "Point",
        coordinates: [121.445, 25.167]
      },
      distanceField: "distance",
      spherical: true
    }
  },
  { $sort: { distance: -1 } },
  { $limit: 1 }
]
```

會自動找有 2D sphere 索引的欄位來計算



# PYTHON

# 安裝 PYTHON LIBRARY

$ pip install pymongo

$ pip install "pymongo[srv]"

使用 MongoDB 雲端服務 Atlas 時要裝

# 函數對映

find => find

findOne => find_ond

[insert | update | delere | replace]One => [insert | update | delere | replace] _one

[insert | update | delere | replace] Many => [insert | update | delere | replace] _many

# 範例 – FIND

◆查詢 opendata 資料庫 AQI 資料表中 sitename 為「淡水」的資料，並將指令轉為 python 程式

```
from pymongo import MongoClient

client = MongoClient("mongodb://localhost:27017/")
db = client["opendata"]
collection = db["AQI"]

query = {"sitename": "淡水"}
result = collection.find(query)

for document in result:
    print(document)
```

# 範例 – AGGREGATION

◆根據 opendata 資料庫中的 AQI 資料表，計算每個county的平均aqi，並將指令轉為 python 程式

```
from pymongo import MongoClient

client = MongoClient("mongodb://localhost:27017/")
db = client["opendata"]
collection = db["AQI"]

pipeline = [
    {"$group": {"_id": "$county", "average_aqi": {"$avg": "$aqi"}}}
]
result = collection.aggregate(pipeline)

for document in result:
    print(f"County: {document["_id"]}, Average AQI: {document["average_aqi"]}")
```

# 開發 WEB API

下面兩行為 AI Prompt

◆將結果以 JSON 字串形式輸出

◆使用 Flask 建立 WebAPI，網址為 /query/<sitename>

執行後就可以開 postman 或瀏覽器打網址試試看了

http://127.0.0.1:5000/query/淡水