

物聯網技術結合AI大數據 分析實作(2)

徐凡耘 
2020-07



DAY2

▶ 上午

- ▶ 安裝樹莓派 **Kafka** 套件
- ▶ **Docker**環境建置
- ▶ 溫濕度資料收集與上傳

▶ 下午

- ▶ 啟動 **Message Queue** (套件 **Kafka/ Ksql**)
- ▶ 啟動 **NoSQL**儲存與**ETL** (套件 **Mongodb/ pyspark**)
- ▶ 查詢結果 (套件 **AdminMongo**)



AIGO業界出題，考驗AI落地

- 官網

- https://aigo.org.tw/ai-plus/home/new_index

- 出題案例

- 【天氣風險管理開發】 AI道路監視器影像判斷降雨：利用影像分析地面、天空、即時天氣資訊
 - 【台灣楓康超市】 自動辨識水果甜度挑選水果方案：以影像方式辨識水果甜度
 - 【邑流微測】 半導體產線AI智動化影像檢測工具：建立自動化影像分析系統連動品管關連性
 - 【佳穎精密】 AOI產線不良品影像自動檢測系統：端子、連接器生產品質
 - 【智慧領航教育科技】 AI頭皮檢測與診療方法判別：透過頭皮顯微影像進行問題分類
 - 【華碼數位】 食物熱量影像辨識：利用影像估算食物熱量

智慧音箱

- 語音辨識教學

- <https://makerpro.cc/2019/04/use-ros-for-speech-recognition/>

- Google AIY

- <https://www.raspberrypi.com.tw/tag/%E8%AA%9E%E9%9F%B3%E6%96%87%E5%AD%97%E8%BD%89%E6%8F%9B/>

關於影像辨識

- 樹莓派影像辨識 (必讀)

- <https://www.youtube.com/watch?v=Cgxsv1riJhl&feature=share>

- 博士實作 (辨識多人)

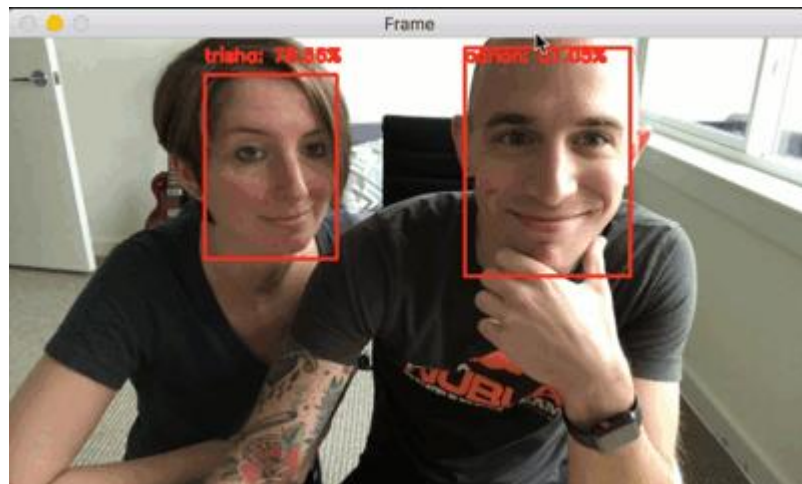
- <https://www.pyimagesearch.com/2018/09/24/opencv-face-recognition/?fbclid=IwAR2MYjcl3EsSZKyMWxDkZl1x8Etuq8NSW7qjdv-Jg9F2DzY7y5MLef1e-uc>

- 下載博士的免費電子書

- https://www.pyimagesearch.com/2018/06/25/raspberry-pi-face-recognition/?fbclid=IwAR3E_uyFLeipDI2314xuMmBh-V6SRCVwzW3lWkO6Mon1NaD-Fbg-0vgOt2g

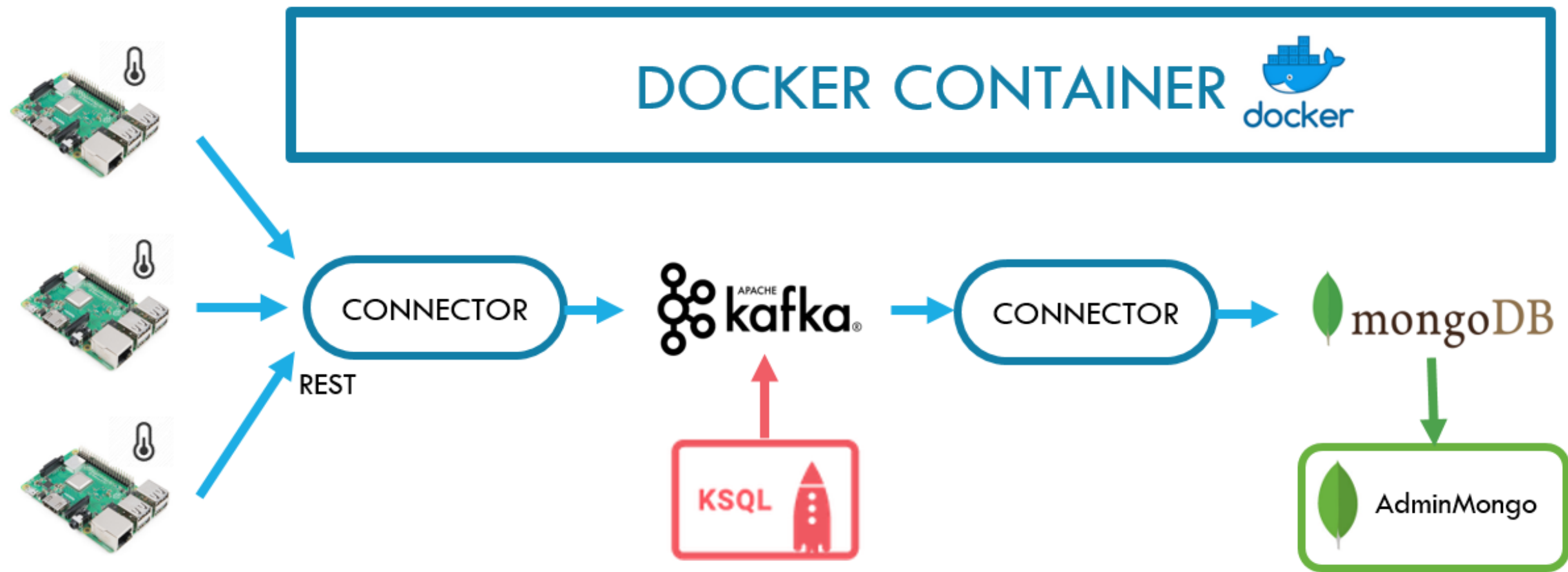
- OpenCV (ver.=4.0.1) + tensorflow 1.13 (pretrain-model)

- https://omnixri.blogspot.com/2019/05/aigo2opencv.html?m=1&fbclid=IwAR3cw7B7zGFaV9MDgjON3rKhLH3jOWNgpTHjNQyYkUNPn_Oc_i1DxXUAgo





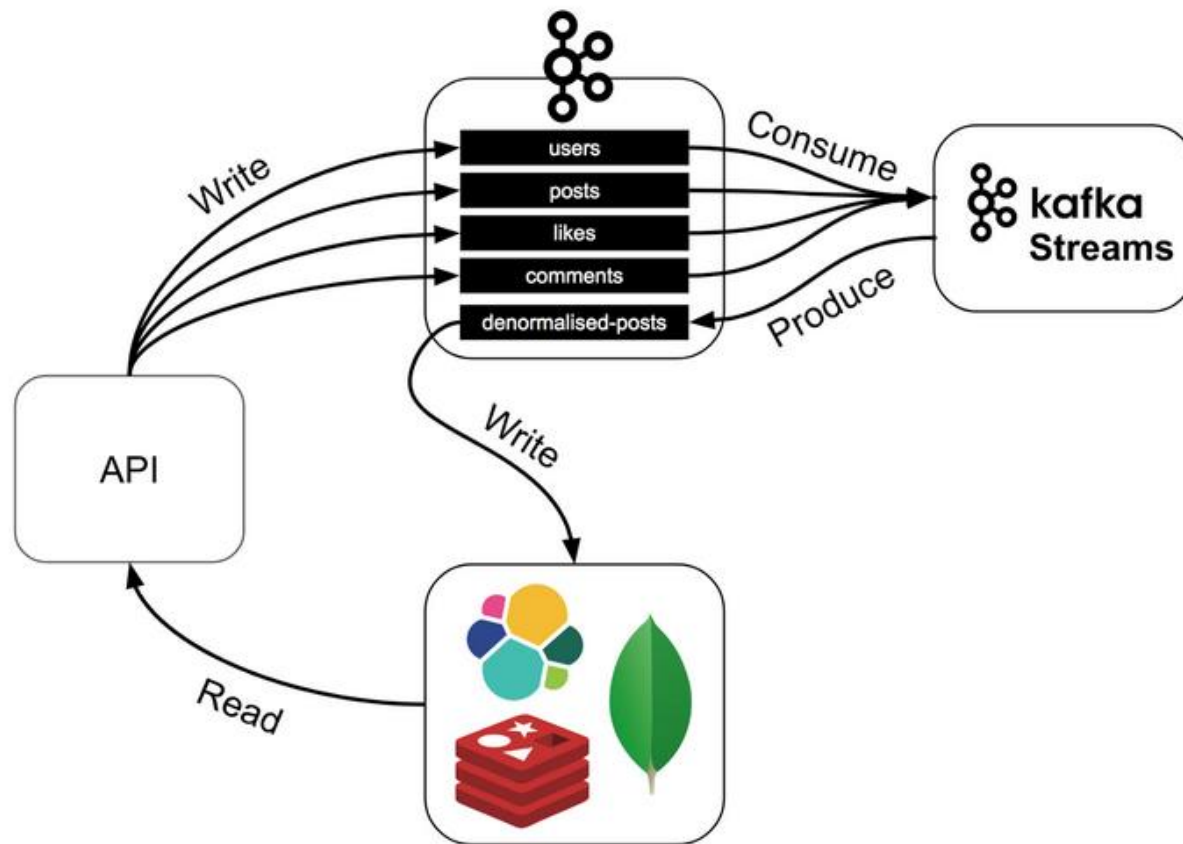
溫濕度資料收集與上傳





KAFKA 介紹

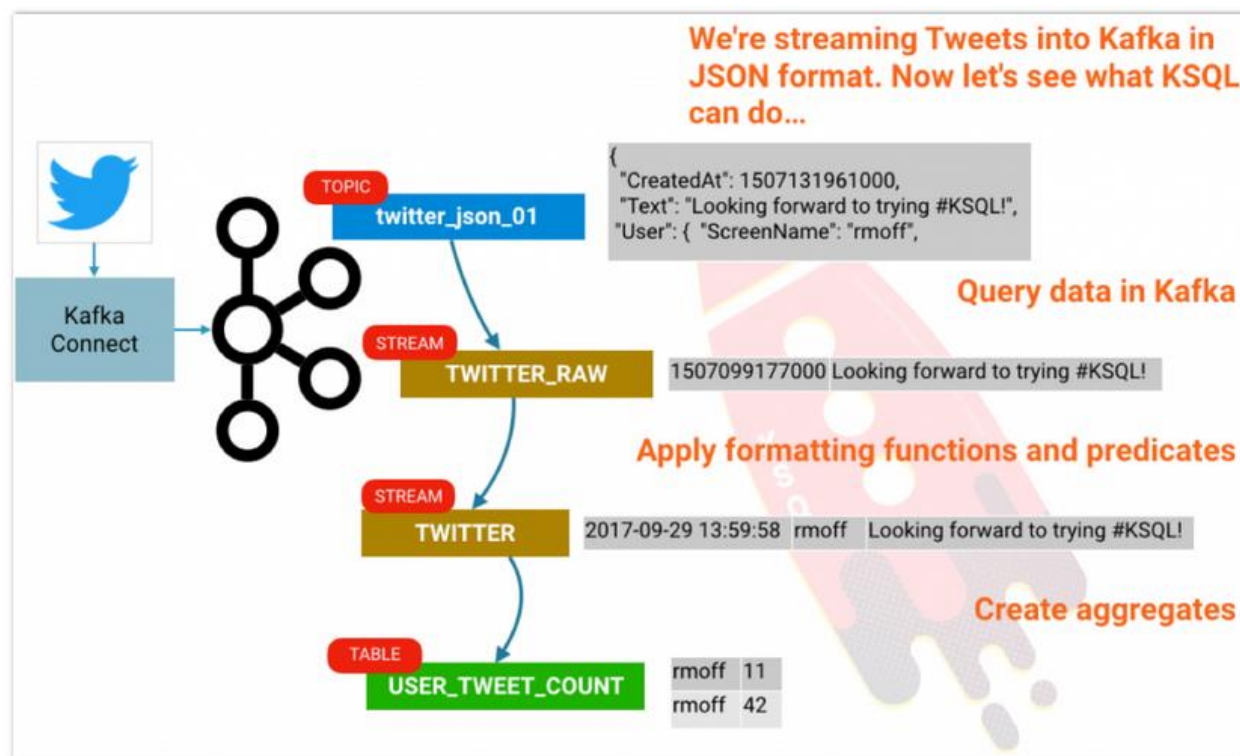
- Message Queue
(發佈/訂閱)
- 可設定為單機或叢集
- 高吞吐率
(普通的電腦上可處理 100K筆資料/每秒)
- 支援離線/即時資料處理
- 可設定個別 topic
(將 streams 資料丟入該 topic 當中)
- 應用:
 - 埋 Code (Page view)
 - 收集機器 Log (CPU、IO使用率)





KAFKA SQL 介紹

- 與 Kafka 一起運行
- KSQL 是 Apache Kafka 開源的 Streaming SQL 引擎
- 透過進來的資料，可以設定條件過濾與聚合
- 應用：
 - 統計由 **twitter** 進來的用戶每小時推文數量





DOCKER 環境建置

- 準備一台可以與樹莓派同網段的電腦
- 記憶體: 8GB+
- CPU: 2 cores+
- 硬碟空間: 100 GB



DOCKER 環境建置

- 安裝 git 套件

```
yum install git
```

- 下載 docker，完成之後需要重新開機

```
curl -sSL https://get.docker.com | sh
```

- 重新登入系統後，請啟動 docker 服務

```
systemctl start docker
```



DOCKER 環境建置

- 安裝 docker-compose

```
yum install epel-release
```

```
curl -L "https://github.com/docker/compose/releases/download/1.10.0/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
```

```
chmod +x /usr/local/bin/docker-compose
```

```
docker-compose --version
```



DOCKER 環境建置

- 下載伺服器docker環境

```
cd /root; git clone https://github.com/orozcohsu/2019-ltu-iot.git
```

- 啟動伺服器環境

```
cd /root/2019-ltu-iot; docker-compose up -d
```

- docker容器介紹





DOCKER 環境建置

- 進入 kafka container

```
docker exec -it kafka /bin/bash
```

- 建立 topic 名為 temperature

```
kafka-topics.sh --create --zookeeper zookeeper:2181 --topic temperature --partitions 1 --replication-factor 1
```

- 建立 topic 名為 humidity

```
kafka-topics.sh --create --zookeeper zookeeper:2181 --topic humidity --partitions 1 --replication-factor 1
```



DOCKER 環境建置

- 列出有哪些 topic

```
kafka-topics.sh --list --zookeeper zookeeper:2181
```

- 檢視 temperature topic 狀態

```
kafka-topics.sh --describe --zookeeper zookeeper:2181 --topic temperature
```

- 進入ksql-cli container

```
docker exec -it ksql-cli bash
```



DOCKER 環境建置

- 啟動 ksql

```
ksql http://ksql-server:8088
```

- 查看有哪些 topic

```
SHOW TOPICS;
```



DOCKER 環境建置

- 建立一個 stream 為 temperature_stream

```
CREATE STREAM temperature_stream (device_id varchar, timestamp varchar, \  
temperature double, rd varchar) WITH(kafka_topic='temperature', value_format='json');
```

- 建立一個 stream 為 humidity_stream

```
CREATE STREAM humidity_stream (device_id varchar, timestamp varchar, \  
humidity double, rd varchar) WITH(kafka_topic='humidity', value_format='json');
```




DOCKER 環境建置

- 查看所有 stream

```
SHOW STREAMS;
```

- 查看 temperature_stream 欄位資訊

```
DESCRIBE temperature_stream;
```

- 查看 humidity_stream 欄位資訊

```
DESCRIBE humidity_stream;
```



DOCKER 環境建置

- 建立 stream 名為 temp_humidity_enriched，內容是將 temperature_stream 和 humidity_stream 做 JOIN

```
CREATE STREAM temp_humidity_enriched AS SELECT t.device_id, t.timestamp, \  
temperature, humidity, t.rd FROM temperature_stream t JOIN humidity_stream h \  
WITHIN 1 HOURS ON t.rd= h.rd;
```

- 再次確定是否順利建立 stream

```
ksql> show streams;
```

Stream Name	Kafka Topic	Format
TEMP_HUMIDITY_ENRICHED1	TEMP_HUMIDITY_ENRICHED1	JSON
HUMIDITY_STREAM	humidity	JSON
TEMPERATURE_STREAM	temperature	JSON



DOCKER KAFKA REST PROXY 測試

- 登入 Kafka REST 容器

```
docker exec -it restproxy /bin/bash
```

- 查看目前有多少個 topic

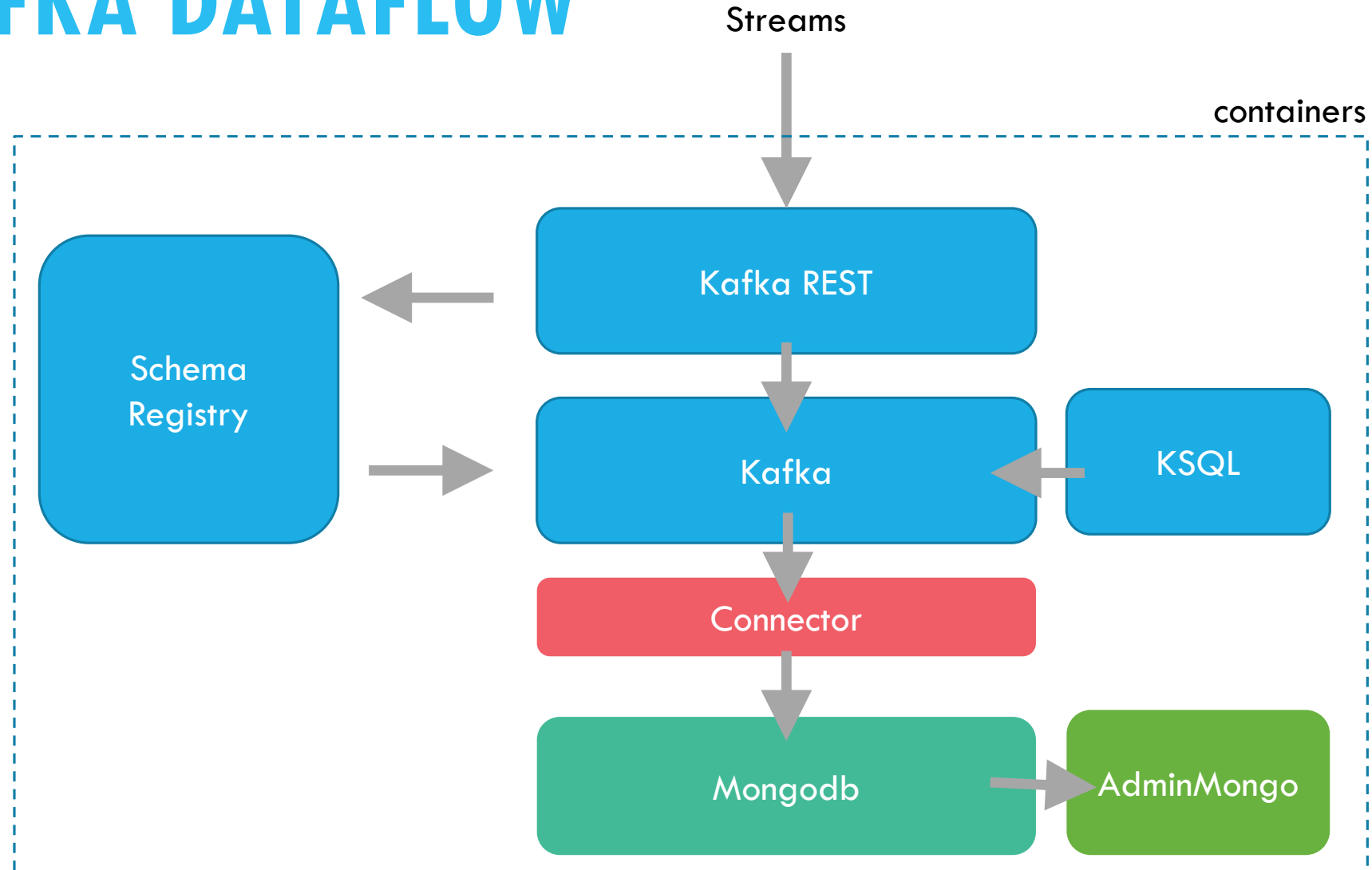
```
curl http://localhost:8082/topics
```

- 查看 temperature 相關資訊

```
curl http://localhost:8082/topics/temperature
```



KAFKA DATAFLOW





MONGODB

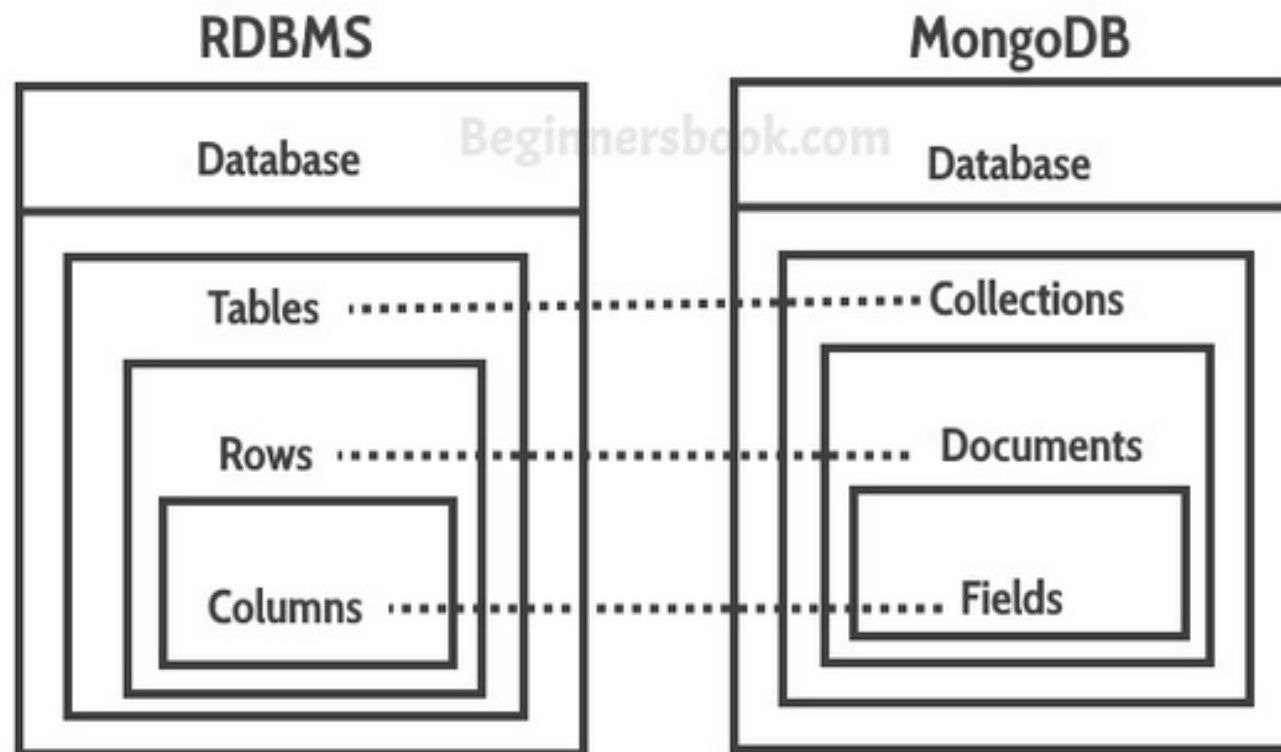
- MongoDB 屬於文件資料庫 (Document Database) ，以文本方式儲存
- 可設定為單機或叢集
- 本身沒有 Schema ，所以在架構上很好調整
- 資料樣貌

```
{  
  _id: "948794777",  
  name: "Robby",  
  age: 30,  
  email: "Robby",  
  skill: [  
    'javascript',  
    'java'  
  ]  
}
```



MONGODB

- 與結構化資料庫的術語比較





DOCKER 環境建置

- 開啟 jupyter 網頁 (IP是Type1的IP)

<http://IP:8889>

- 開啟 consumer 程式

`kafka_consumer.ipynb`





ADMINMONGO

- Mongodb (<http://IP:1234>)
- 網頁查詢平台 (連線輸入: mongodb://mongodb:27017)

 adminMongo 🔌 Connections

MongoDB Connections

No connections. Please add one below

Connection name	Connection string	Connection options (See docs)	Action
ltu	<input type="text" value="mongodb://mongodb:27017"/>	1 {}	<button>Add connection</button>





ADMINMONGO

Database Objects

ltu

test

data_20190630

temp_humidity

Connections

ltu

Monitoring

Connections

Database: test / Collection - data_20190630

Home / ltu (connection) / test (database) / data_20190630 (collection)

New document

Indexes

Search

Query

Reset

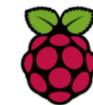
Docs per page

5

Total records: 42

Delete all

<pre>{ "_id": "5d1b8a6cbe567be7cc90350d", "device_id": "001", "timestamp": "2019-07-03 00:36:58", "Temperature": 27, "Humidity": 65 }</pre>	<div>DeleteLinkEdit</div>
<pre>{ "_id": "5d1b8a6cbe567be7cc90350e", "device_id": "001", "timestamp": "2019-07-03 00:37:06", "Temperature": 27, "Humidity": 66 }</pre>	<div>DeleteLinkEdit</div>
<pre>{ "_id": "5d1b8a6cbe567be7cc90350f", "device_id": "001", "timestamp": "2019-07-03 00:37:22", "Temperature": 27, "Humidity": 66 }</pre>	<div>DeleteLinkEdit</div>



建立DATABASE

- Database = Itu_demo



The screenshot displays the adminMongo web interface. On the left sidebar, under 'Database Objects', the 'config' database is expanded, showing 'system.sessions'. The 'Itu_demo' database is also listed. The main panel shows the 'Server status' section with the following details:

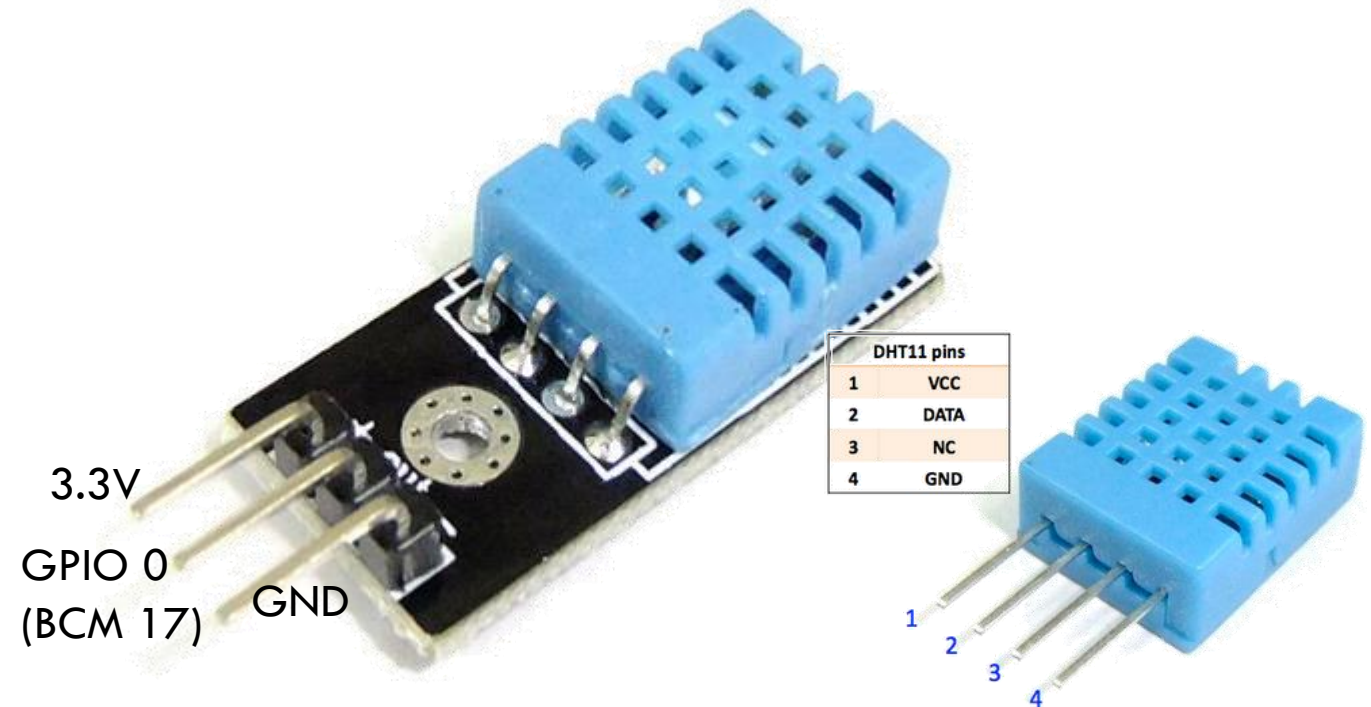
- Version: 4.0.10
- Host: mongoddb
- PID: 1
- Uptime: 28 minutes
- Local time: Tue Jul 02 2019 16:59:01 GMT+0000 (Coordinated Universal Time)

Below the server status, the 'Database maintenance' section includes a 'Create new database' form with a text input containing 'Itu_demo' and a green 'Create' button. There is also a 'Delete database' section with a dropdown menu showing 'config' and a red 'Delete' button.



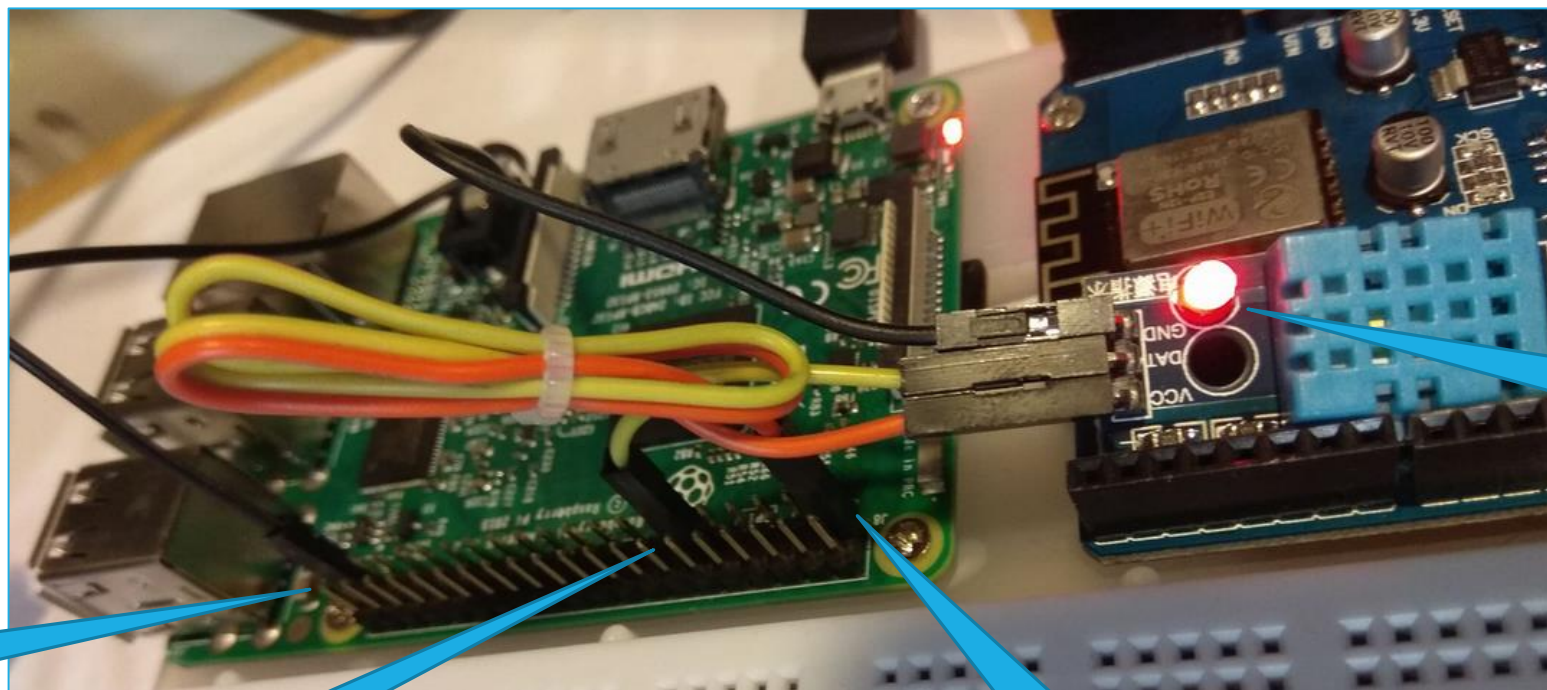
溫濕度資料收集與上傳

DHT11	DHT22
	
3 - 5.5V	3.3 - 6V
20 - 80 %	0 - 100 %
5%	5%
0 - 50 °C	-40 - 125 °C
+/- 2 °C	+/- 0,5 °C
1s	2s





溫濕度資料收集與上傳



第39隻腳
(接地)

第11隻腳
(GPIO0)

第1隻腳
(3.3V)

指示燈



樹莓派上執行 (感測器是 DHT22)

- 安裝 DHT22 的感測器驅動程式
- git clone https://github.com/adafruit/Adafruit_Python_DHT.git
- cd Adafruit_Python_DHT
- sudo python setup.py install

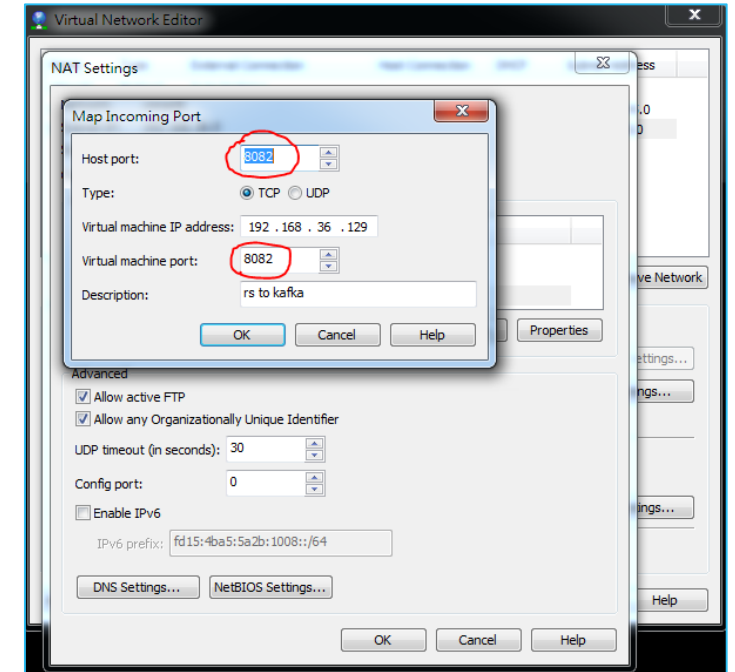
Windows 主機 IP

```
import Adafruit_DHT
```

```
#topic  
temp="http://192.168.43.196:8082/topics/temperature"  
humd="http://192.168.43.196:8082/topics/humidity"
```

```
#DHT22 (GPIO0=BCM17)  
humidity, temperature = Adafruit_DHT.read_retry(22, 17)
```

開啟 sensor_kafka_dht22.py



新增 NAT 內容



樹莓派上執行

- 開啟 jupyter 上的 kafka_consumer 程式，確定以下資訊
 - database = ltu_demo
 - collection = db.data_20191103
- 於樹莓派上執行
 - python sensor_kafka_dht22.py

```
if __name__ == "__main__":  
  
    # 與 MongoDB連線  
    client = pymongo.MongoClient(host="mongodb", port=27017)  
    # 指定為 test 資料庫  
    db = client.ltu_demo  
    # 指定 temp_humidity 集合, MongoDB的每個資料庫又包含許多集  
    collection = db.data_20190630
```

```
root@raspberrypi:/home/pi/2019-ltu-airbox# python sensor_kafka.py  
temperature updated  
humidity updated  
wait next 5 seconds  
temperature updated  
humidity updated  
wait next 5 seconds  
temperature updated  
humidity updated  
wait next 5 seconds  
temperature updated  
humidity updated  
wait next 5 seconds
```

實際執行狀況

- 執行狀況

- <https://www.dropbox.com/s/p0ytcxy3ntdhblu/video-1594314243.mp4?dl=0>

SPARK (PYSPARK)

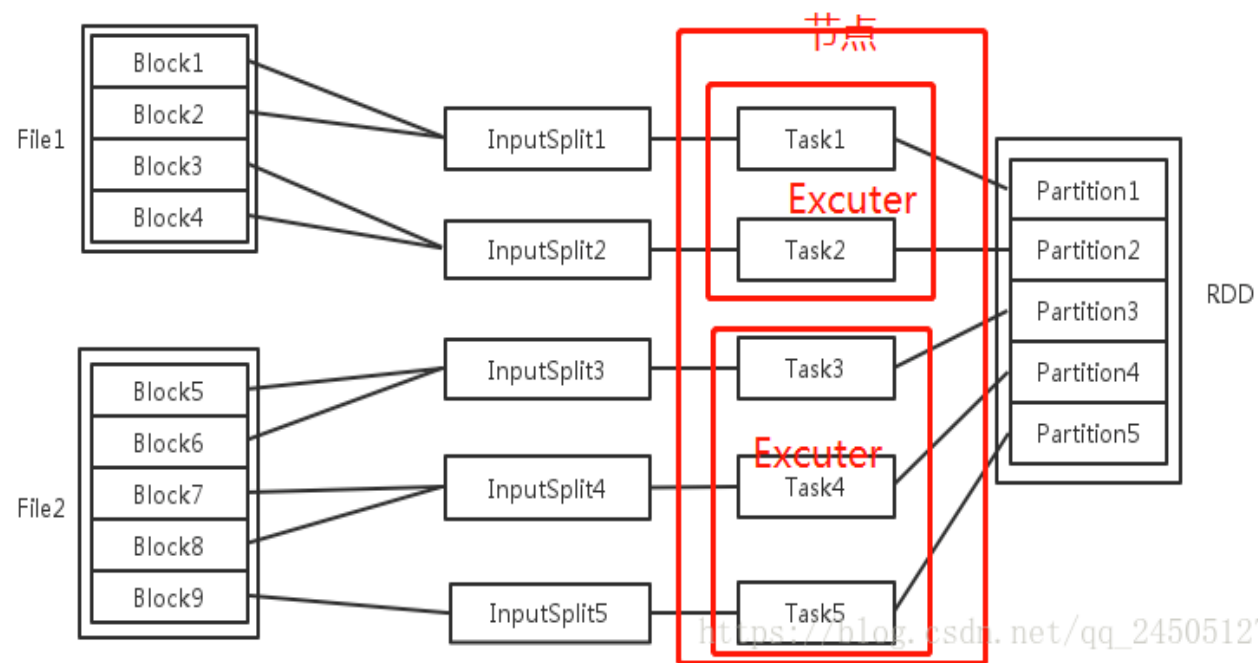


- 分散式運算框架，資料透過記憶體叢集完成
- 為 Hadoop MapReduce 的取代品
- 能處理大於記憶體叢集總和之資料量
- 支援 Spark (Scala)、PySpark、SparkR 工具
- Spark SQL、Spark MLlib、Spark Streaming、Spark GraphX



RDD

- 資料儲存於記憶體叢集之資料格式
- 透過 Transformation 轉變為一個新的 RDD
- 透過 Action 將 RDD 計算後，求得一個新的值 (純量或陣列)





PYTHON (PANDAS 套件)

- Python 常用套件之一
- 適合用來整理資料
- 資料格式為 DataFrame，與 Spark SQL、RDD 可以互轉



執行 PYSPARK

- 開啟 Spark jupyter

<http://IP:8890>

- 執行Pyspark程式

pyspark-read-mysql.ipynb

pyspark_mongodb_read1.ipynb

pyspark_mongodb_read2.ipynb

pyspark_mysql_mongodb_join_etl.ipynb



查看最終結果

Database: *Not selected* / Collection - *Not selected*

Server status

Version: 4.0.10

Host: mongodb

PID: 1

Uptime: 34 minutes

Local time: Tue Jul 02 2019 17:05:01 GMT+0000 (Coordinated Universal Time)

Database maintenance

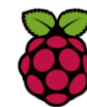
Create new database

Create

Connection statistics

config		
system.sessions	Storage: 396 Bytes	Documents: 4
ltu_demo		
data_2019063020	Storage: 20.2 KB	Documents: 182
final	Storage: 32.2 KB	Documents: 179
test		
data_20190630	Storage: 4.66 KB	Documents: 42
temp_humidity	Storage: 4.44 KB	Documents: 40





查看最終結果

Database: **ltu_demo** / Collection - final

Home / ltu (connection) / ltu_demo (database) / final (collection)

New document

Indexes

Search

Query

Reset

Docs per page

5

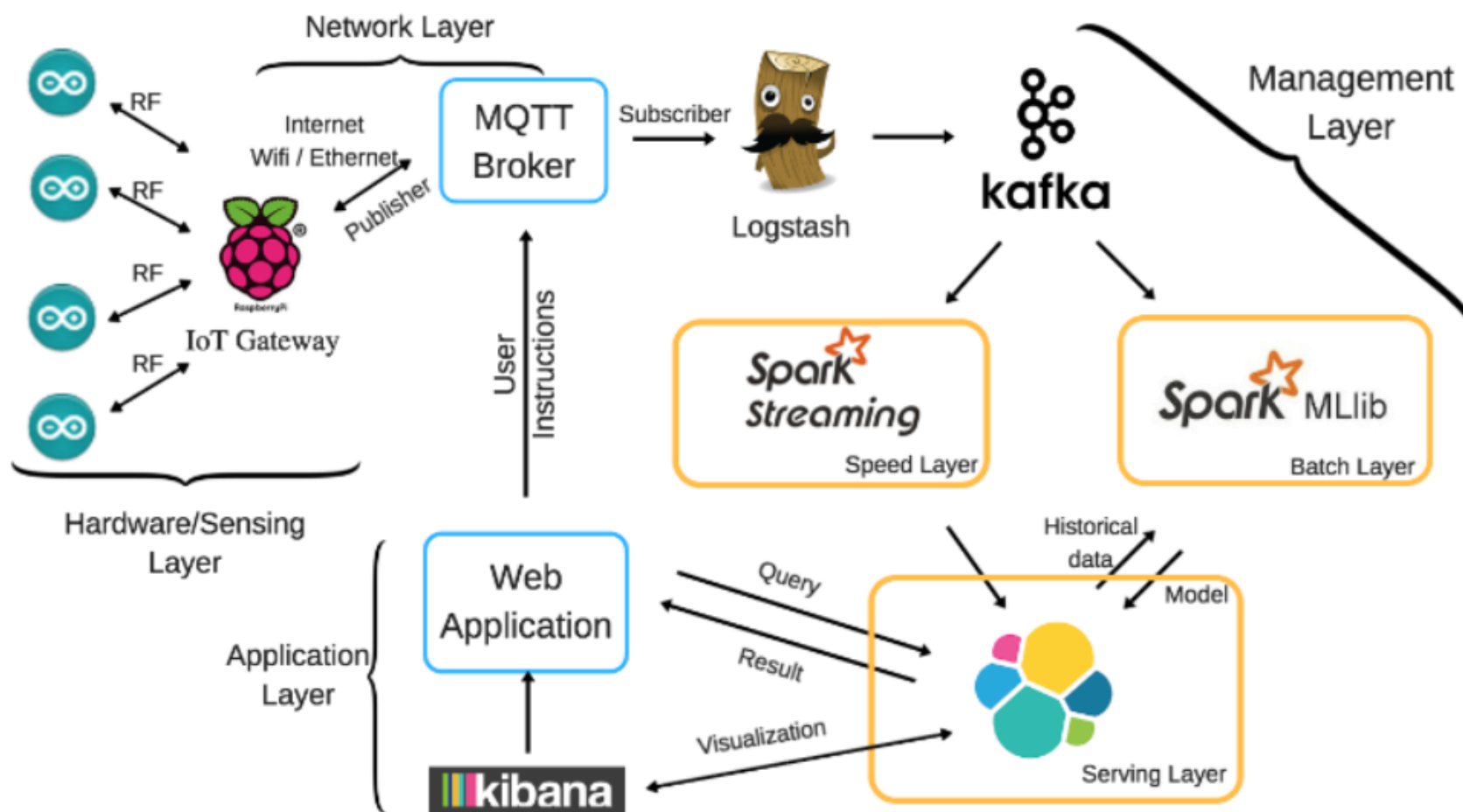
Total records: 179

Delete all

<pre>{ "_id": "5d1b8d40be567be7cc90353b", "device_id": "001", "location": "Taichung", "description": "Semiconductor-Machine-01", "Humidity": 66 }</pre>	<div>Delete</div> <div>Link</div> <div>Edit</div>
<pre>{ "_id": "5d1b8d40be567be7cc90353c", "device_id": "001", "location": "Taichung", "description": "Semiconductor-Machine-01", "Humidity": 66 }</pre>	<div>Delete</div> <div>Link</div> <div>Edit</div>
<pre>{ "_id": "5d1b8d40be567be7cc90353d", "device_id": "001", "location": "Taichung", "description": "Semiconductor-Machine-01", "Humidity": 64 }</pre>	<div>Delete</div> <div>Link</div> <div>Edit</div>



大數據物聯網系統架構回顧 (總結)



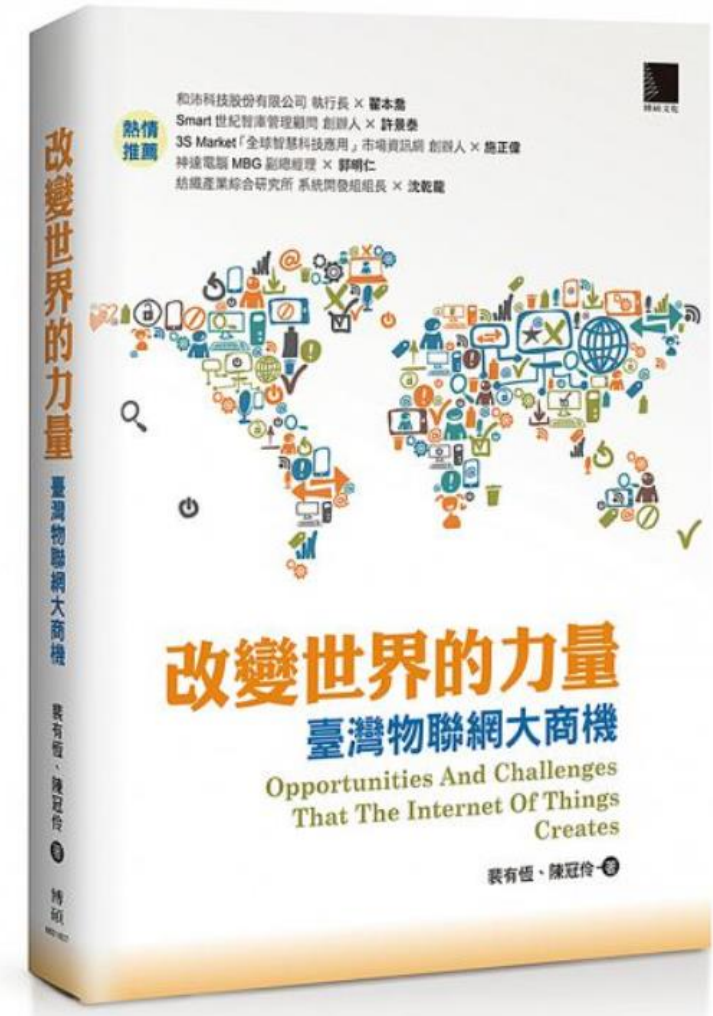
作業

- 練習 pandas merge 與 concat
- <http://violin-tao.blogspot.com/2017/06/pandas-2-concat-merge.html>



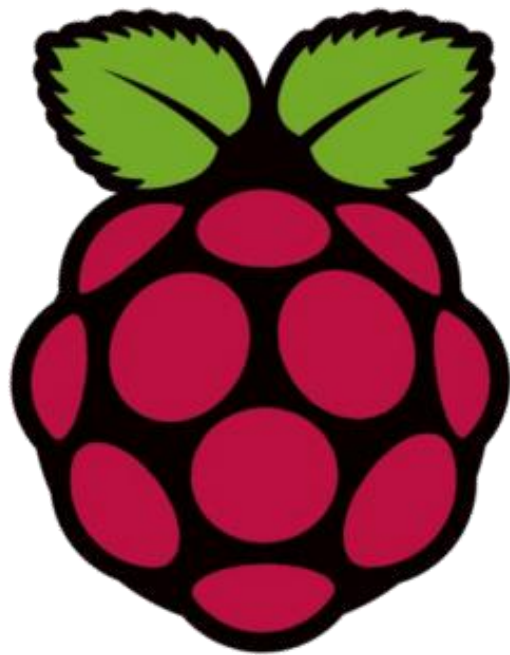
參考資料

- ✓ AIoT產業-多組學的精準醫療
- ✓ AIoT產業-從 Google I/O 跟微軟 Build 談 Edge computing 發展的迫切性
- ✓ AIoT產業-微軟和 Google 為何都在台灣佈局人工智慧-微軟篇
- ✓ AIoT產業-Google 的人工智慧將在台灣往下一個世代開始佈局
- ✓ AIoT產業-AIoT 服務沒有資安，會出這三個大問題



<http://rich4innovation.blogspot.tw>

享受開源的樹莓派應用



歡迎聯繫數數科技資訊社
orozcohsu@hotmail.com