

**Lien git du projet**

Le projet se situe ici : <https://github.com/niry1/weather_docker_cassandra>

L’objectif du projet est d’injecter dans la base de données Cassandra les données météorologiques française provenant de l’API <https://openweathermap.org/api> .

Pour ce faire il faut réaliser en python ces deux étapes :

1. Récupérer les données en utilisant python et l’insérer dans un fichier CSV « weatherOpenMap.csv »

Voir github : <https://github.com/niry1/weather>

1. Récupérer les données dans weatherOpenMap.csv et les injecter dans Cassandra (en mode cluster)

Après avoir créé le fichier CSV, i faut réaliser les étapes suivantes, afin d’ingester les données dans la base de données Cassandra :

* 1. Créer une connexion à Cassandra

On définit l’host, le port, et toutes les informations pour se connecter à Cassandra

CASSANDRA\_HOST = ["cassandra"] CASSANDRA\_PORT = 9042 CASSANDRA\_DB = "weather" CASSANDRA\_TABLE = "France" CASSANDRA\_LOGIN = "cassandra" CASSANDRA\_PWD = "cassandra"

Avec les informations ci-dessus, nous allons nous connecter à la BDD

cluster = Cluster(protocol\_version=3,contact\_points=CASSANDRA\_HOST,load\_balancing\_policy=None,port=CASSANDRA\_PORT, auth\_provider=auth\_provider)

session =cluster.connect()

* 1. Créer un key space France

session.execute("CREATE KEYSPACE IF NOT EXISTS france WITH REPLICATION={'class':'SimpleStrategy','replication\_factor':1};")

* 1. Créer la table ville

session.execute("CREATE TABLE IF NOT EXISTS  france.villes (city text, country text, temp float, temp\_max float,  temp\_min float, humidity float, pressure float, sky text, sunrise text, sunset text, wind float, cloudiness int, dt text, primary key (city));")

* 1. Créer une variable préparant les données pour être injecté à Cassandra

query\_insert="INSERT INTO france.villes  (city, country, temp, temp\_max,temp\_min, humidity,pressure, sky,sunrise, sunset,  wind,cloudiness,dt) VALUES ($${}$$, '{}', {}, {},{}, {}, {}, '{}','{}', '{}', {}, {},'{}');"

float, sky text, sunrise text, sunset text, wind float, cloudiness int, dt text, primary key (city));")

La variable df\_weather récupère les données du fichier csv en utilisant Pandas

df\_weather = pd.read\_csv("./weatherOpenMap.csv", sep = ",")

* 1. Injecter les données dans Cassandra

La partie [ct] permet d’avoir tous les indices de df\_weather et donc de récupérer pour chaque villes les informations de "temp\_max", "temp\_min",etc …

for ct in df\_weather.index:

CQL\_query = query\_insert.format(df\_weather["city"][ct].replace("'", ""),df\_weather["country"][ct],df\_weather['temp'][ct],\

df\_weather["temp\_max"][ct],df\_weather["temp\_min"][ct],df\_weather['humidity'][ct],df\_weather["pressure"][ct],\

df\_weather["sky"][ct],df\_weather['sunrise'][ct],df\_weather['sunset'][ct],df\_weather['wind'][ct],df\_weather["cloudiness"][ct],\

df\_weather["dt"][ct])

Afin de réaliser ce projet, j’ai réalisé deux dockers :

1. Le premier docker : « docker\_weather »

Ce docker contiendra le code python weather.py, requirements.txt et city.list.json.  
docker\_weather sera géré par un Dockerfile qui va :

* 1. Créer une image python

FROM python:3.6-slim

* 1. Copier toutes les dépendances liées à weather.py dans le répertoire de travail /app

WORKDIR /app/

COPY weather.py requirements.txt city.list.json /app/

* 1. Installer tous les requirements utilisé par weather.py

RUN pip install -r requirements.txt

1. Le deuxième docker qui est en mode cluser contiendra « cassandra », « cassandra1 » et « cassandra2 »

Ce docker contiendra la partie base de données NoSQL de Cassandra. Dès que les donnés seront ingesté dans le 1er cluster cassandra, les données seront répliquées dans cassandra1 et cassandra2

Le docker docker\_weather et le docker en mode cluster cassandra vont être orchestré par docker-compose.yml :

docker-compose.yml est composé de 4 services (3 services cassandra et 1 service weather), il sera décomposé en création de service et de volume :

Service :

1. Services cassandra (1 seul service sera présenté car les 3 configurations sont identiques) :
   1. Configuration de(s) port(s)

Cassandra sera dispo sur le port 9042 et les échanges des données entre cluster se fera via le port 7000

ports:

- 7000:7000

- 9042:9042

* 1. Configuration des volumes

volumes:

- cassandra\_data:/bitnami

* 1. Configuration de l’environnement cassandra

environment:

- CASSANDRA\_SEEDS=cassandra,cassandra2,cassandra3

- CASSANDRA\_CLUSTER\_NAME=cassandra-cluster

- CASSANDRA\_PASSWORD\_SEEDER=yes

- CASSANDRA\_PASSWORD=cassandra

- MAX\_HEAP\_SIZE=2G

- HEAP\_NEWSIZE=200M

ports:

- 7000:7000

- 9042:9042

* 1. Vérification de l’état de santé des services

Cette partie consiste à faire un test de vie (healthcheck) pour vérifier que les bdd du cluster cassandra sont bien accessibles. Le test s'effectue toute les 5 secondes et tombent en timeout au bout de 10s. Au bout de 6 retry en KO ou en timeout, le service est statué comme étant "unealthy". Cela permet de conditionner le démarrage du service "weather". En effet le container de ce dernier ne démarre uniquement que si les services cassandra sont statués comme étant healthy par le test de vie (healthcheck)

healthcheck:

test: cqlsh -u cassandra -p cassandra

interval: 5s

timeout: 10s

retries: 6

1. Service weather
   1. Configuration des dépendances

Avant de lancer le service weather il faut attendre que le cluster cassandra soit bien lancé

depends\_on:

cassandra:

condition: service\_healthy

cassandra2:

condition: service\_healthy

cassandra3:

condition: service\_healthy

Volume :

Création des volumes des cluster cassandra

volumes:

cassandra\_data:

driver: local

cassandra2\_data:

driver: local

cassandra3\_data:

driver: local