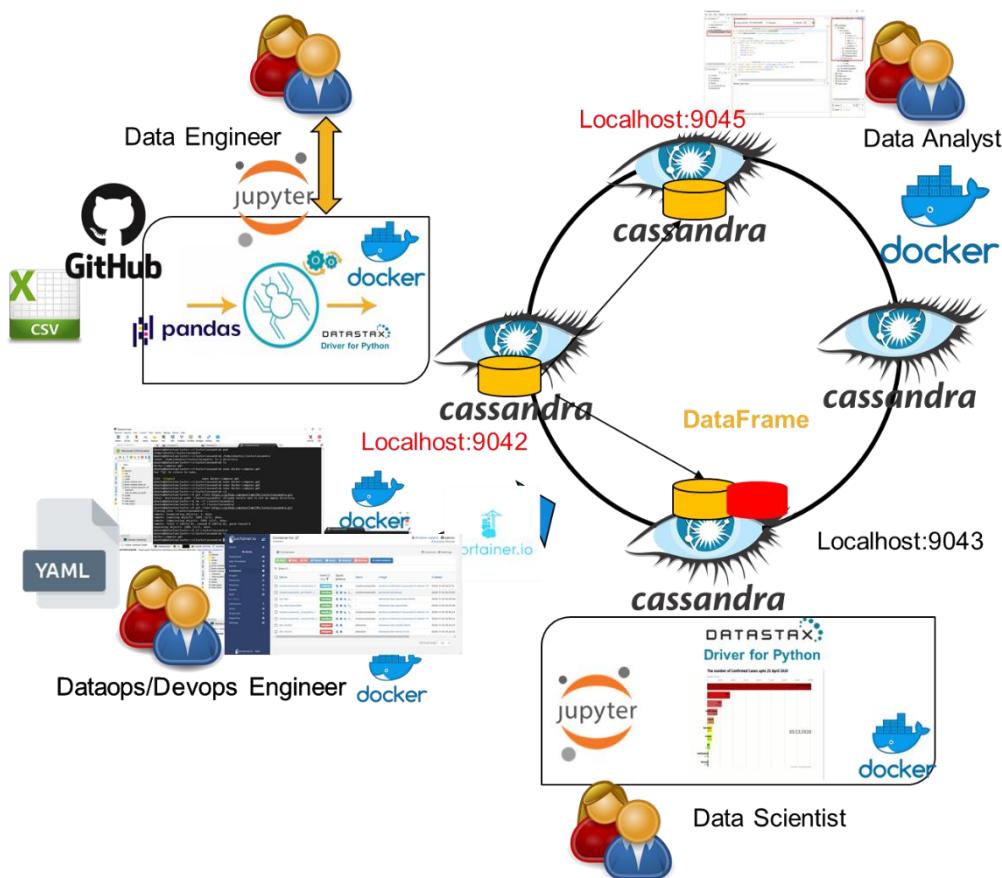


BASE DE DONNEES NO SQL Cassandra

Formateur : Sellami Mokhtar
mokhtar.sellami@gmail.com

Atelier 2 : Mise en place d'un Cluster Cassandra



Objectifs

Après avoir terminé cet atelier, vous serez en mesure de :

- Installation de Docker et Docker Composer
- Configurer le Cluster avec un fichier YAML
- Lancer le Cluster
- Vérifier que des nœuds du Cluster
- Exécution des requêtes CQL Via DevCenter sur différents Nœuds
- Ajouter et Supprimer des Nœuds de Cluster

Installation de Docker et Docker Compose

#Installation Docker Compose

BASE DE DONNEES NO SQL Cassandra

Formateur : Sellami Mokhtar
mokhtar.sellami@gmail.com

```
sudo curl -L "https://github.com/docker/compose/releases/download/1.27.4/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
sudo chmod +x /usr/local/bin/docker-compose
sudo ln -s /usr/local/bin/docker-compose /usr/bin/docker-compose
```

#Installation Docker

```
sudo apt-get update
sudo apt-get install apt-transport-https ca-certificates curl
gnupg-agent software-properties-common
sudo add-apt-repository "deb [arch=amd64]
https://download.docker.com/linux/ubuntu \
$(lsb_release -cs) \
stable"
sudo apt-get update
sudo apt-get install docker-ce docker-ce-cli containerd.io
```

Récupérer et Editer le fichier de Configuration YAML via Git Clone

Lancer un Shell et Cloner le projet à partir du GitHub

```
$ cd /home/fitec/
$ git clone https://github.com/msellamiTN/clusterCassandra.git
$ cd clusterCassandra
```

```
ubuntu@datastaxcluster:~$ git clone https://github.com/msellamiTN/clusterCassandra.git
Cloning into 'clusterCassandra'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.
ubuntu@datastaxcluster:~$ cd clusterCassandra/
ubuntu@datastaxcluster:~/clusterCassandra$ ls
docker-compose.yml
ubuntu@datastaxcluster:~/clusterCassandra$ █
```

Editer le Fichier docker-compose.yml pour configurer votre nœuds de Cluster

```
$ gedit docker-compose.yml
```

BASE DE DONNEES NO SQL Cassandra

Formateur : Sellami Mokhtar
mokhtar.sellami@gmail.com

```

version: '2'
services:
  cassandra:
    image: docker.io/bitnami/cassandra:3-debian-10
    ports:
      - 7000:7000
      - 9042:9042
    volumes:
      - cassandra_data:/bitnami
    environment:
      - CASSANDRA_SEEDS=cassandra,cassandra2
      - CASSANDRA_CLUSTER_NAME=cassandra-cluster
      - CASSANDRA_PASSWORD_SEEDER=yes
      - CASSANDRA_PASSWORD=cassandra
    # By default, Cassandra autodetects the available host memory and takes as much as it can.
    # Therefore, memory options are mandatory if multiple Cassandras are launched in the same node.
      - MAX_HEAP_SIZE=2G
      - HEAP_NEWSIZE=200M
  cassandra2:
    image: docker.io/bitnami/cassandra:3-debian-10
    ports:
      - 7001:7000
      - 9043:9042
    volumes:
      - cassandra2_data:/bitnami
    environment:
      - CASSANDRA_SEEDS=cassandra,cassandra2
      - CASSANDRA_CLUSTER_NAME=cassandra-cluster
      - CASSANDRA_PASSWORD=cassandra
    # By default, Cassandra autodetects the available host memory and takes as much as it can.
    # Therefore, memory options are mandatory if multiple Cassandras are launched in the same node.
      - MAX_HEAP_SIZE=2G
      - HEAP_NEWSIZE=200M
volumes:
  cassandra_data:
    driver: local
  cassandra2_data:
    driver: local

```

Spécifier les des nœuds d'amorçage

Configuration de premier Noeud

Renvoie de ports

Configuration deuxième Noeud

Nom du Cluster

Ajouter des volumes de disques

Enregistrer vos fichiers et taper la commande suivante pour lancer le cluster

```
$ sudo docker-compose up -d
```

Il faut avoir un résultat similaire à l'écran suivant :

```

ubuntu@datastaxcluster:~/clusterCassandra$ sudo docker-compose up -d
Creating network "clustercassandra_default" with the default driver
Creating volume "clustercassandra_cassandra_data" with local driver
Creating volume "clustercassandra_cassandra2_data" with local driver
Pulling cassandra (docker.io/bitnami/cassandra:3-debian-10)...
3-debian-10: Pulling from bitnami/cassandra
58212c1109c5: Pull complete
96824dc62a8e: Pull complete
d1fc77bdbd0d: Pull complete
12ad5de9ad28: Pull complete
8155d7751d4a: Pull complete
8e60f03cafa5: Pull complete
elc5595ff8dc: Pull complete
a53fce7fa23b: Pull complete
0e079e2c5e6e: Pull complete
bf028f87464b: Pull complete
ab687316fb33: Pull complete
be514576d31e: Pull complete
8aff291030a7: Pull complete
c9cdf3d6c4b5: Pull complete
Digest: sha256:8a5de434943387c0bd5ce859b6d20ac0868fdfa458a0ea29d36fab12a9902e18
Status: Downloaded newer image for bitnami/cassandra:3-debian-10
Creating clustercassandra_cassandra2_1 ... done
Creating clustercassandra_cassandra_1 ... done
ubuntu@datastaxcluster:~/clusterCassandra$ sudo docker ps

```

Vérifier que les nœuds sont correctement relancés

```
$ sudo docker ps
```

BASE DE DONNEES NO SQL Cassandra

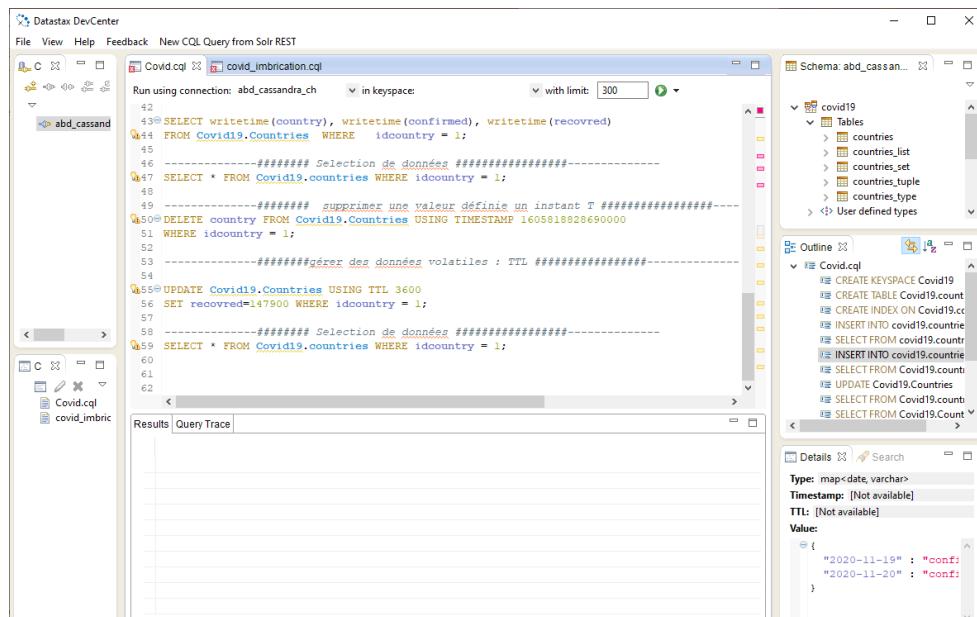
Formateur : Sellami Mokhtar
mokhtar.sellami@gmail.com

CONTAINER ID	IMAGE	NAMES	COMMAND	CREATED	STATUS	PORTS
46c22e7156a4	bitnami/cassandra:3-debian-10	"opt/bitnami/script..."	/opt/bitnami/script..."	54 seconds ago	Up 50 seconds	0.0.0.0:7000
->7000/tcp, 0.0.0.0:9042->9042/tcp	clustercassandra_cassandra_1					
24363bf04722	bitnami/cassandra:3-debian-10	"opt/bitnami/script..."	/opt/bitnami/script..."	54 seconds ago	Up 52 seconds	0.0.0.0:7001
->7000/tcp, 0.0.0.0:9043->9042/tcp	clustercassandra_cassandra_2_1					

Access aux nœuds via DevCenter

Accéder au répertoire d'installation DevCenter et lancer le via la commande suivante

```
./DevCenter
```



Création de la première connexion nomme « CN_Cluster_Node00 »

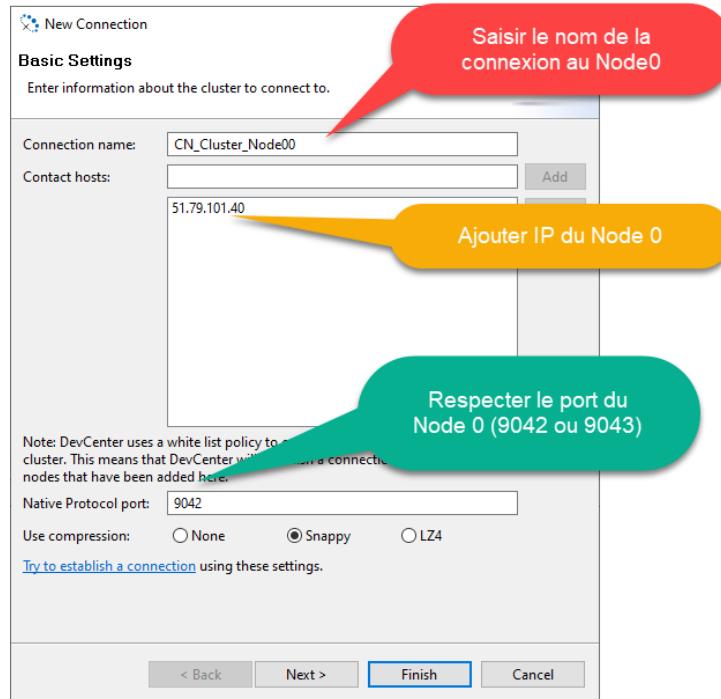
```

version: '2'
services:
  cassandra:
    image: docker.io/bitnami/cassandra:3-debian-10
    ports:
      - 7000:7000
      - 9042:9042
    volumes:

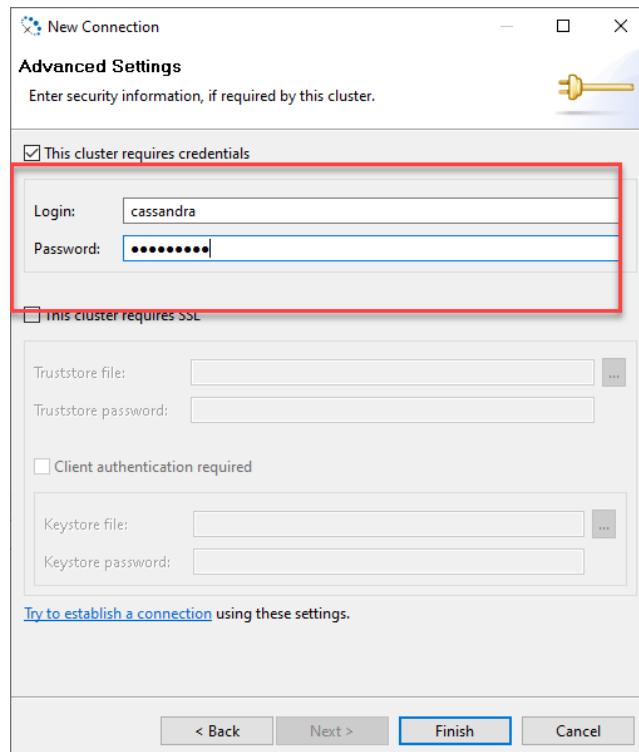
```

BASE DE DONNEES NO SQL Cassandra

Formateur : Sellami Mokhtar
mokhtar.sellami@gmail.com



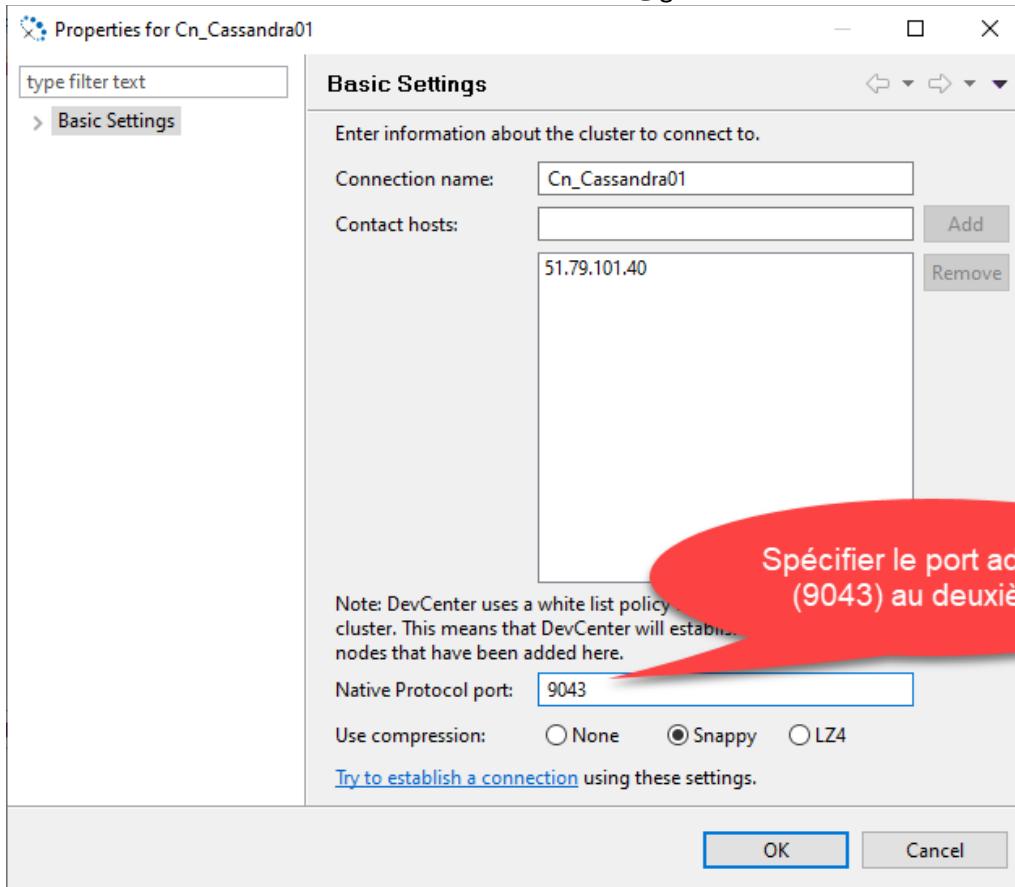
Saisir les paramètres de sécurité 'cassandra : cassandra'



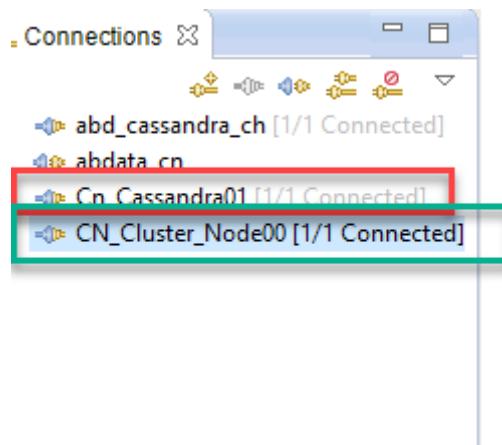
Passer à faire la même chose pour la création de deuxième connexion mais en modifiant cette fois ci le port d'accès en respectant le fichier de configuration docker-compose.yml

BASE DE DONNEES NO SQL Cassandra

Formateur : Sellami Mokhtar
mokhtar.sellami@gmail.com



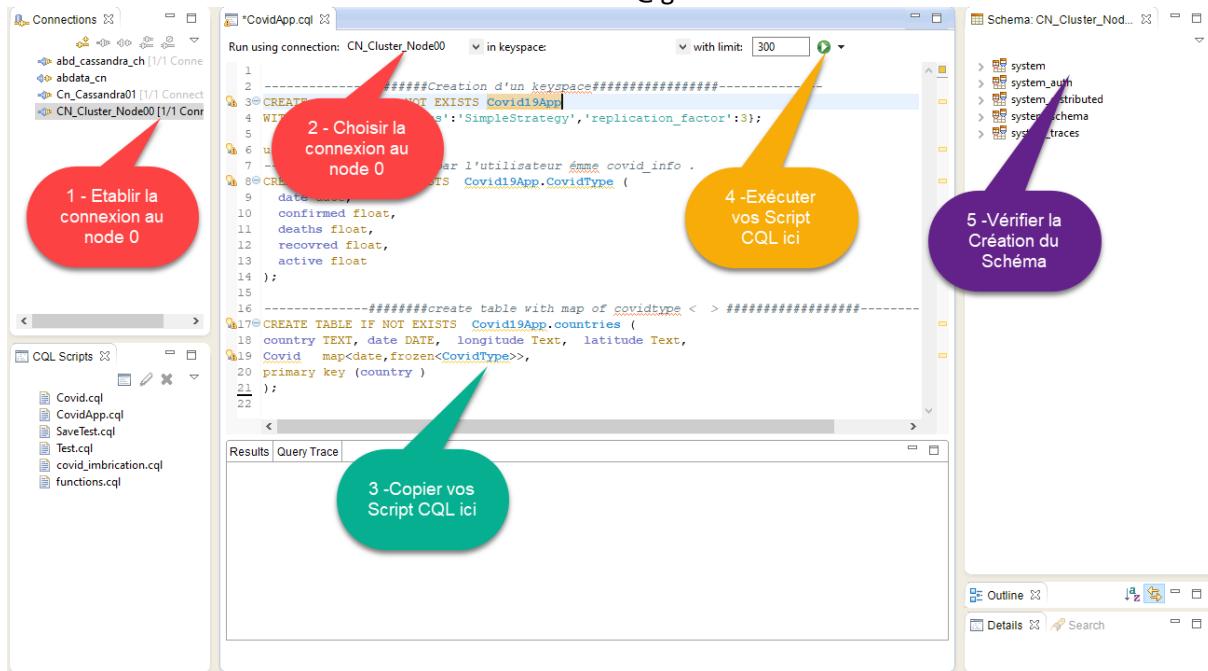
Vérifier que vous avez deux connexions actives au cluster



Passer Maintenant à créer un fichier CQL de création de keyspace et de spécifier la connexion CN_Cluster_Node00

BASE DE DONNEES NO SQL Cassandra

Formateur : Sellami Mokhtar
mokhtar.sellami@gmail.com



Ce Script permettant de créer un Keyspace avec une table countries utilisant une imbrication MAP et UDT – TYPE CovidType

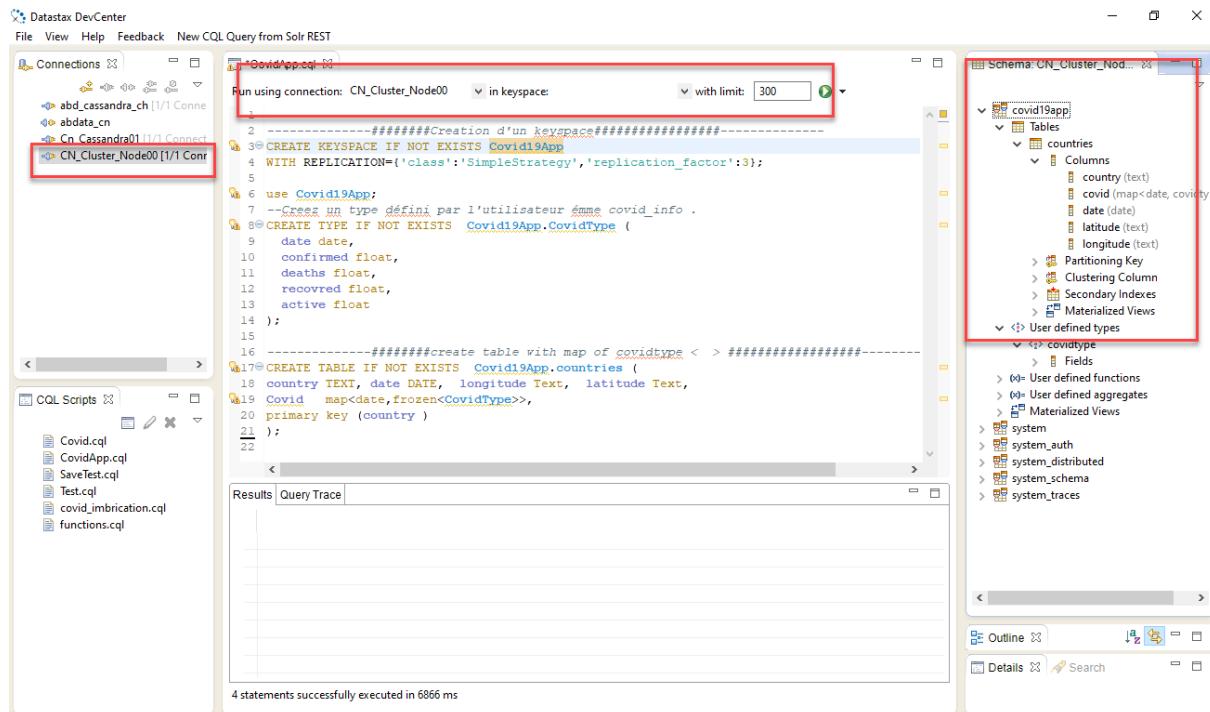
```
-----#####Creation d'un keyspace#####
CREATE KEYSPACE IF NOT EXISTS Covid19App
WITH REPLICATION={'class':'SimpleStrategy','replication_factor':3};

use Covid19App;
--Creez un type défini par l'utilisateur émme covid_info .
CREATE TYPE IF NOT EXISTS Covid19App.CovidType (
    date date,
    confirmed float,
    deaths float,
    recovered float,
    active float
);
-----#####create table with map of covidtype < > #####
CREATE TABLE IF NOT EXISTS Covid19App.countries (
    country TEXT, date DATE, longitude Text, latitude Text,
    Covid map<date,frozen<CovidType>>,
    primary key (country )
);
```

BASE DE DONNEES NO SQL Cassandra

Formateur : Sellami Mokhtar
mokhtar.sellami@gmail.com

Vérifier que la création est bien terminée avec succès au niveau de Cluster Node 0



The screenshot shows the DataStax DevCenter interface. On the left, the 'Connections' sidebar lists 'abd_cassandra_ch [1/1 Conn]', 'abdata_cn', 'Cn_Cassandra01 [1/1 Conn]', and 'CN_Cluster_Node00 [1/1 Conn]' (highlighted with a red box). The main area displays a CQL script named 'CovidApp.cql' with the following code:

```

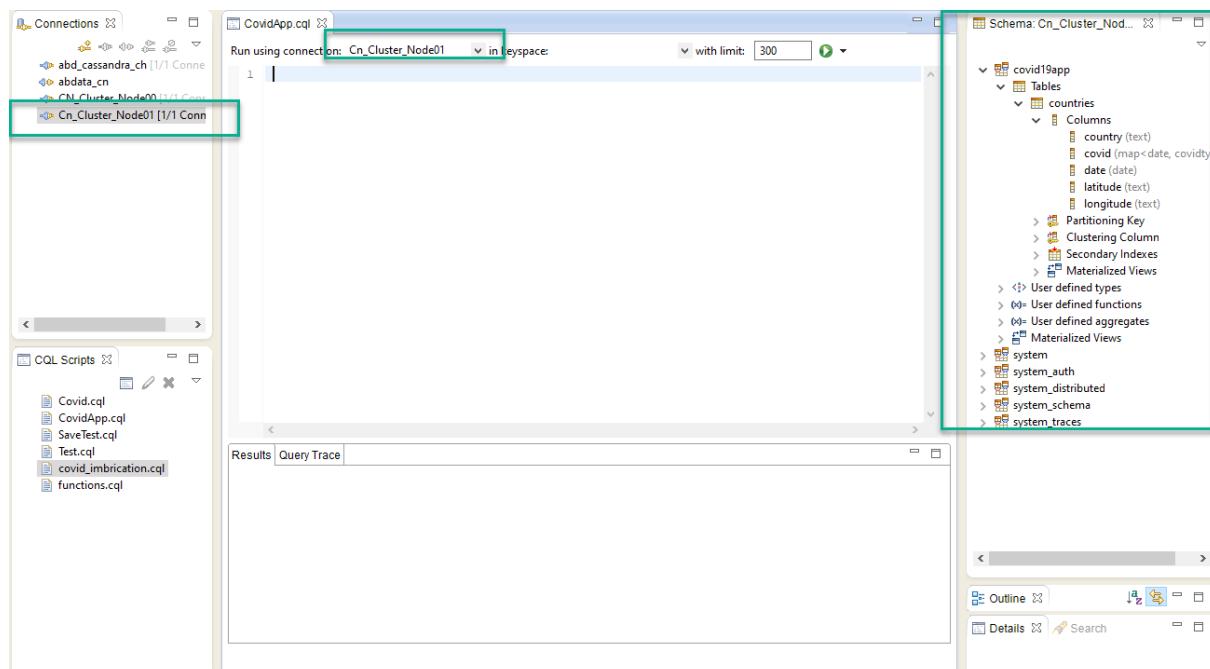
1 1
2 -----#####Creation d'un keyspace#####
3 @CREATE KEYSPACE IF NOT EXISTS Covid19App
4 WITH REPLICATION={'class':'SimpleStrategy','replication_factor':3};
5
6 use Covid19App;
7 --Créer un type défini par l'utilisateur émme covid_info .
8 @CREATE TYPE IF NOT EXISTS Covid19App.CovidType (
9   date date,
10   confirmed float,
11   deaths float,
12   recovered float,
13   active float
14 );
15
16 -----#####create table with map of covidtype < > #####
17 @CREATE TABLE IF NOT EXISTS Covid19App.countries (
18   country TEXT, date DATE, longitude Text, latitude Text,
19   Covid map<date,frozen<${CovidType}>>,
20   primary key (country )
21 );
22

```

The 'Results' tab shows the message '4 statements successfully executed in 6866 ms'. To the right, the 'Schema: CN_Cluster_Nod...' browser shows the structure of the 'covid19app' keyspace, including the 'countries' table and its columns.

Passer maintenant à vérifier dans le deuxième Nœud que se passe t'il

Comme vous pouvez noter que la mise à jour de la création du keyspace , table et type sont déjà présent dans le deuxième nœud Node01



The screenshot shows the DataStax DevCenter interface on Node 01. The 'Connections' sidebar lists 'abd_cassandra_ch [1/1 Conn]', 'abdata_cn', 'Cn_Cassandra01 [1/1 Conn]' (highlighted with a green box), and 'CN_Cluster_Node00 [1/1 Conn]'. The main area displays a CQL script named 'CovidApp.cql' with the same code as above. The 'Results' tab is empty. To the right, the 'Schema: Cn_Cluster_Nod...' browser shows the same schema structure as Node 00.

Un Editeur web pour le développement Python

BASE DE DONNEES NO SQL Cassandra

Formateur : Sellami Mokhtar
mokhtar.sellami@gmail.com



Jupyter est une application web utilisée pour programmer dans plus de 40 langages de programmation, dont Python, Julia, Ruby, R, ou encore Scala2. Jupyter est une évolution du projet IPython. Jupyter permet de réaliser des calepins ou notebooks, c'est-à-dire des programmes contenant à la fois du texte en markdown et du code en Julia, Python, R... Ces calepins sont utilisés en science des données pour explorer et analyser des données.

Exécuter un conteneur Jupyter

L'utilisation de l'une des Jupyter Docker Stacks nécessite deux choix:

Quelle image Docker vous souhaitez utiliser

Comment vous souhaitez démarrer les conteneurs Docker à partir de cette image

Cette section fournit des détails sur le second.

Utilisation de la CLI Docker

Vous pouvez lancer un conteneur Docker local à partir des Jupyter Docker Stacks à l'aide de l' interface de ligne de commande Docker . Il existe de nombreuses façons de configurer les conteneurs à l'aide de l'interface de ligne de commande. Voici quelques modèles courants.

Lancement du conteneur jupyter

Cette commande extrait l' image `jupyter/scipy-notebook` balisée `2c80cf3537ca` de Docker Hub si elle n'est pas déjà présente sur l'hôte local. Il démarre ensuite un conteneur exécutant un serveur Jupyter Notebook et expose le serveur sur le port hôte 8888. Les journaux du serveur apparaissent dans le terminal et incluent une URL vers le serveur Notebook.

```
$ sudo docker run --rm -p 8888:8888 -e JUPYTER_ENABLE_LAB=yes -v
"$PWD":/home/ubuntu/cassandra_abdata/ jupyter/scipy-
notebook:2c80cf3537ca
```

Le fait d'appuyer sur `Ctrl-C` arrête le serveur notebook mais laisse le conteneur intact sur le disque pour un redémarrage ultérieur ou une suppression permanente à l'aide de commandes telles que les suivantes :

```
# List containers
```

```
docker ps -a
```

CONTAINER	ID	IMAGE	COMMAND	PORTS	NAMES
CREATED	STATUS				

BASE DE DONNEES NO SQL Cassandra

Formateur : Sellami Mokhtar
mokhtar.sellami@gmail.com

```
d67fe77f1a84      jupyter/base-notebook  "tini -- start-noteb..."  44
seconds ago   Exited (0) 39 seconds ago
cocky_mirzakhani
```

start the stopped container

```
docker start -a d67fe77f1a84
```

Executing the command: jupyter notebook

```
[W 16:45:02.020 NotebookApp] WARNING: The notebook server is listening
on all IP addresses and not using encryption. This is not recommended.
```

...

remove the stopped container

```
docker rm d67fe77f1a84
```

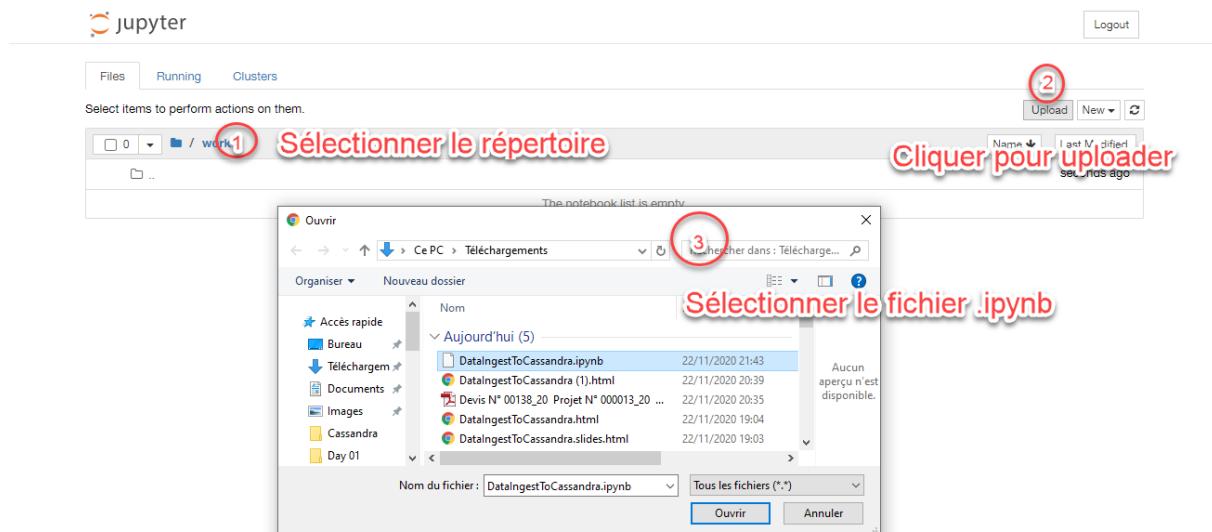
d67fe77f1a84

Lancer Jupyter en cliquant sur le lien dans le Shell

```
Digest: sha256:1457325a4df1803427042686b7b8c99261ec0ae75c6af1d4acbf2df9279c3668
Status: Downloaded newer image for jupyter/scipy-notebook:2c80cf3537ca
Executing the command: jupyter notebook
[I 20:18:59.153 NotebookApp] Writing notebook server cookie secret to /home/jovyan/.local/share/jupyter/runtime/notebook_cookie_secret
[W 20:18:59.635 NotebookApp] WARNING: The notebook server is listening on all IP addresses and not using encryption. This is not recommended.
[I 20:18:59.676 NotebookApp] JupyterLab alpha preview extension loaded from /opt/conda/lib/python3.6/site-packages/jupyterlab
[I 20:18:59.676 NotebookApp] JupyterLab application directory is /opt/conda/share/jupyter/lab
[I 20:18:59.682 NotebookApp] Serving notebooks from local directory: /home/jovyan
[I 20:18:59.683 NotebookApp] 0 active kernels
[I 20:18:59.683 NotebookApp] The Jupyter Notebook is running at:
[I 20:18:59.683 NotebookApp] http://[all ip addresses on your system]:8888/?token=aa9cbf81a8cdaadc789a01f09decb64cb1091c7626720f57
[I 20:18:59.683 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
[C 20:18:59.683 NotebookApp]

Copy/paste this URL into your browser when you connect for the first time,
to login with a token:
    http://localhost:8888/?token=aa9cbf81a8cdaadc789a01f09decb64cb1091c7626720f57
```

Charger un notebook DataIngestion-ToCassandra-.ipynb existant sur le disque local

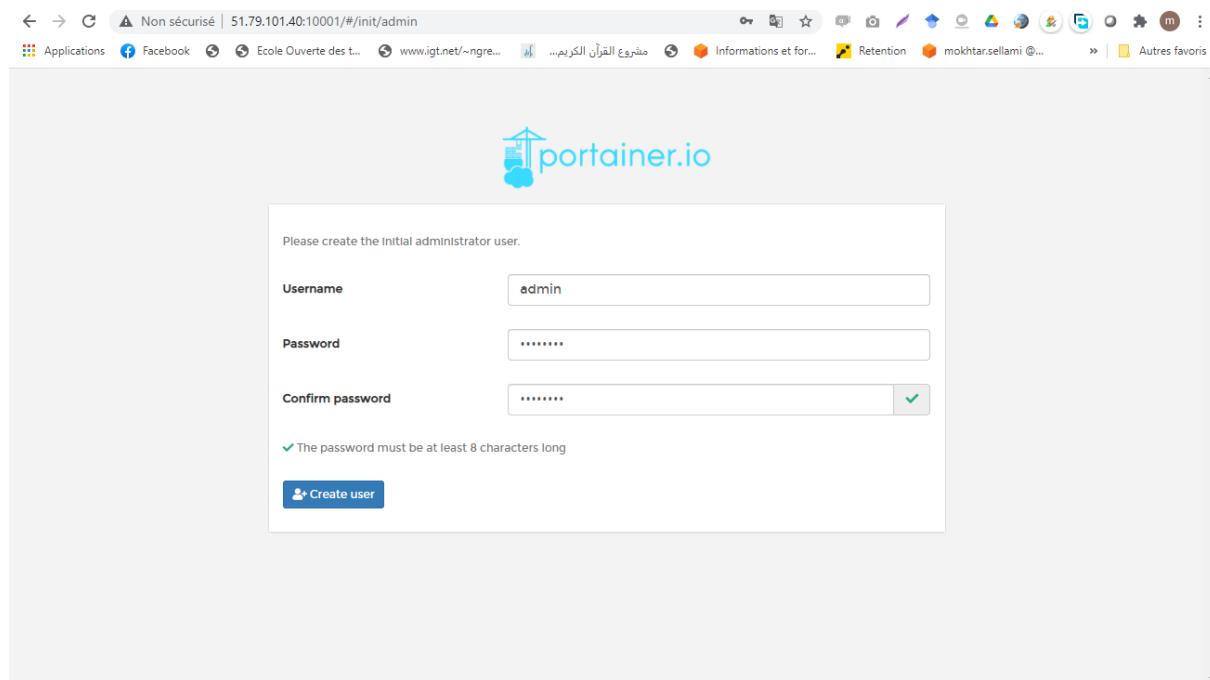


BASE DE DONNEES NO SQL Cassandra

Formateur : Sellami Mokhtar
mokhtar.sellami@gmail.com

Modifier les paramètres de l'adresse IP et le port pour choisir deux nœuds différents pour utiliser l'un pour ingester vos données et l'autres appliquer des analyses en lisant les données répliquées

Administration de vos conteneur dockers



Please create the initial administrator user.

Username: admin

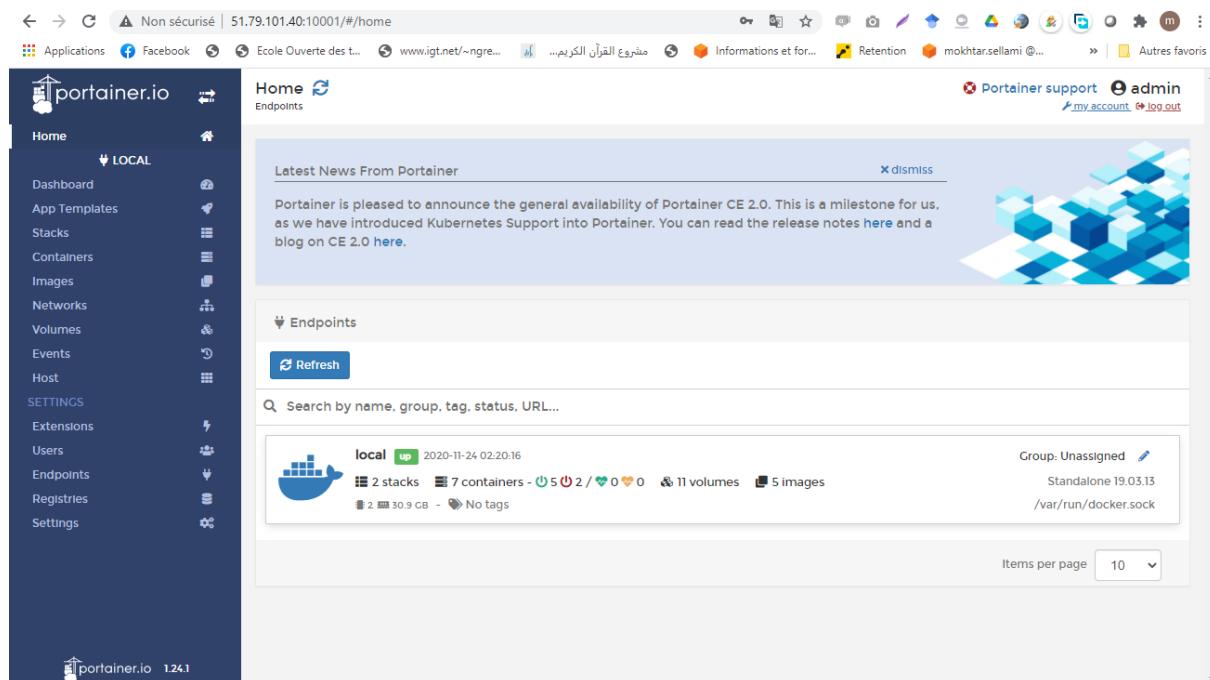
Password:

Confirm password: ✓

The password must be at least 8 characters long

Create user

Choisir vos environnements locaux et valider pour allez vers la page d'accueil



Latest News From Portainer

Portainer is pleased to announce the general availability of Portainer CE 2.0. This is a milestone for us, as we have introduced Kubernetes Support into Portainer. You can read the release notes [here](#) and a blog on CE 2.0 [here](#).

Home

Endpoints

Refresh

Search by name, group, tag, status, URL...

local 2020-11-24 02:20:16

2 stacks 7 containers - 5 up 2 down 0 healthy 0 unhealthy 11 volumes 5 images 30.9 GB 0 tags

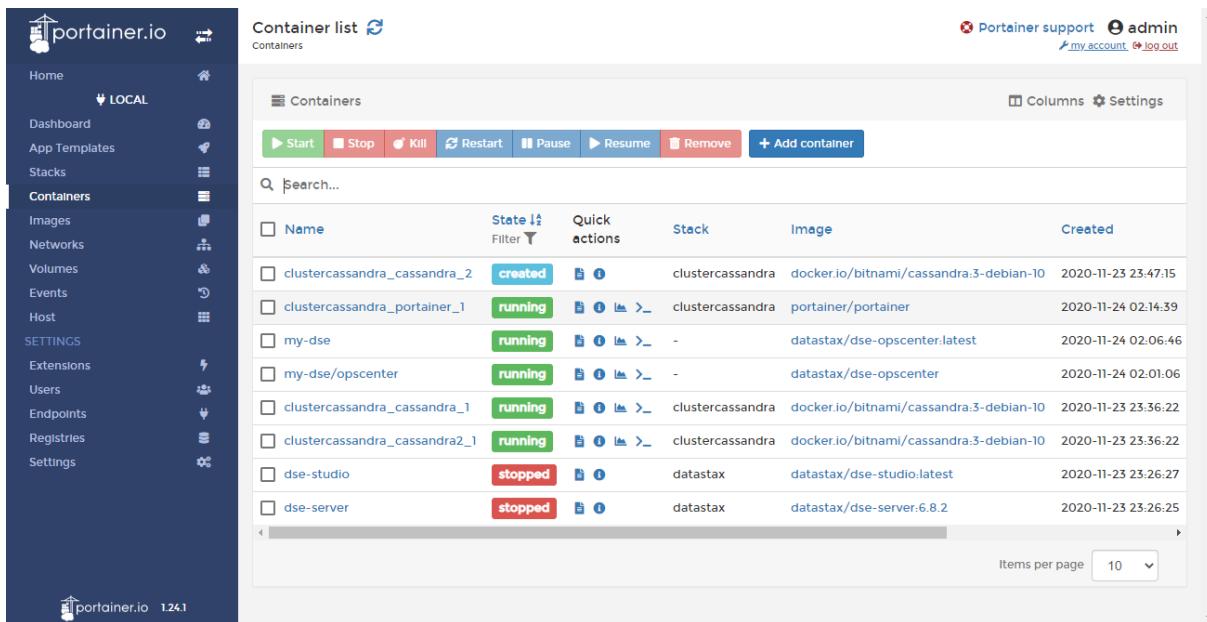
Group: Unassigned Standalone 19.03.13 /var/run/docker.sock

Items per page 10

BASE DE DONNEES NO SQL Cassandra

Formateur : Sellami Mokhtar
mokhtar.sellami@gmail.com

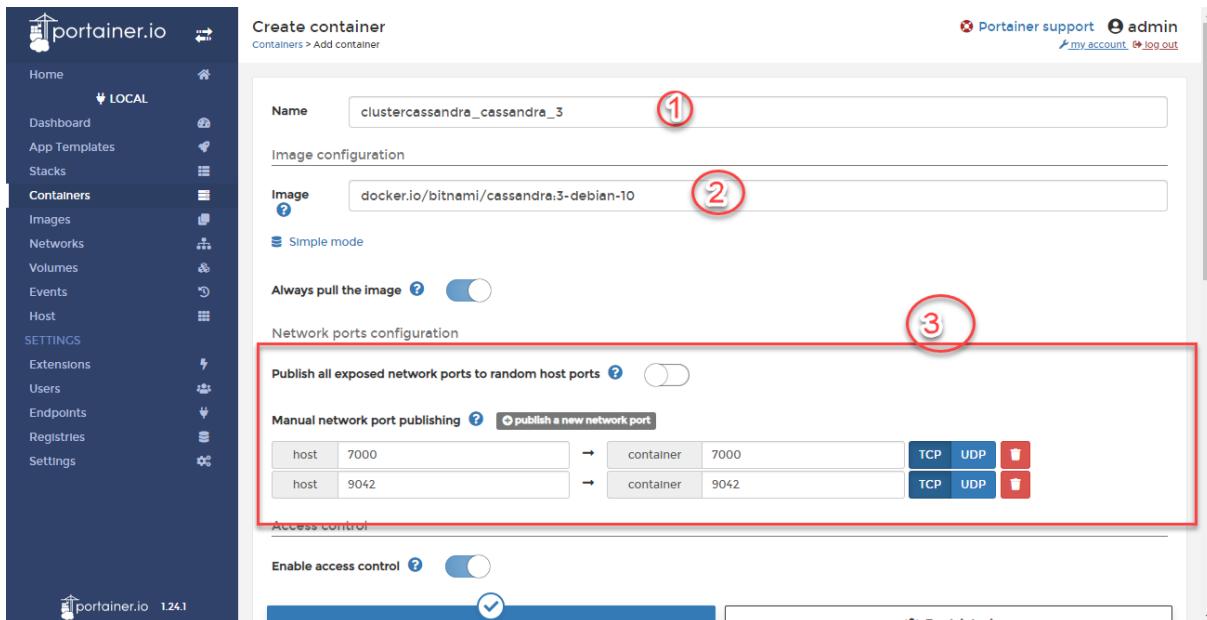
Vérifier vos conteneurs et leur état



The screenshot shows the Portainer.io interface for managing Docker containers. On the left is a sidebar with navigation links like Home, Dashboard, App Templates, Stacks, Containers, Images, Networks, Volumes, Events, Host, SETTINGS, Extensions, Users, Endpoints, Registries, and Settings. The main area is titled "Container list" and shows a table of containers. The columns are Name, State, Quick actions, Stack, Image, and Created. There are 10 items per page. The table contains the following data:

Name	State	Quick actions	Stack	Image	Created
clustercassandra_cassandra_2	created		clustercassandra	docker.io/bitnami/cassandra:3-debian-10	2020-11-23 23:47:15
clustercassandra_portainer_1	running		clustercassandra	portainer/portainer	2020-11-24 02:14:39
my-dse	running		-	datastax/dse-opscenter.latest	2020-11-24 02:06:46
my-dse/opscenter	running		-	datastax/dse-opscenter	2020-11-24 02:01:06
clustercassandra_cassandra_1	running		clustercassandra	docker.io/bitnami/cassandra:3-debian-10	2020-11-23 23:36:22
clustercassandra_cassandra2_1	running		clustercassandra	docker.io/bitnami/cassandra:3-debian-10	2020-11-23 23:36:22
dse-studio	stopped		datastax	datastax/dse-studio.latest	2020-11-23 23:26:27
dse-server	stopped		datastax	datastax/dse-server:6.8.2	2020-11-23 23:26:25

Créer un nœud Cassandra pour scale le cluster



The screenshot shows the "Create container" dialog in Portainer.io. The sidebar on the left is identical to the previous screenshot. The main form has the following fields:

- Name:** clustercassandra_cassandra_3 (circled in red 1)
- Image:** docker.io/bitnami/cassandra:3-debian-10 (circled in red 2)
- Network ports configuration:** A section with a red border containing:
 - A toggle switch for "Publish all exposed network ports to random host ports" (circled in red 3)
 - A "Manual network port publishing" section with two entries:

host	7000	→	container	7000	TCP	UDP	
host	9042	→	container	9042	TCP	UDP	
- Access control:** A section with a toggle switch for "Enable access control" and a dropdown menu set to "Restricted".

1 Spécifier le nom du conteneur

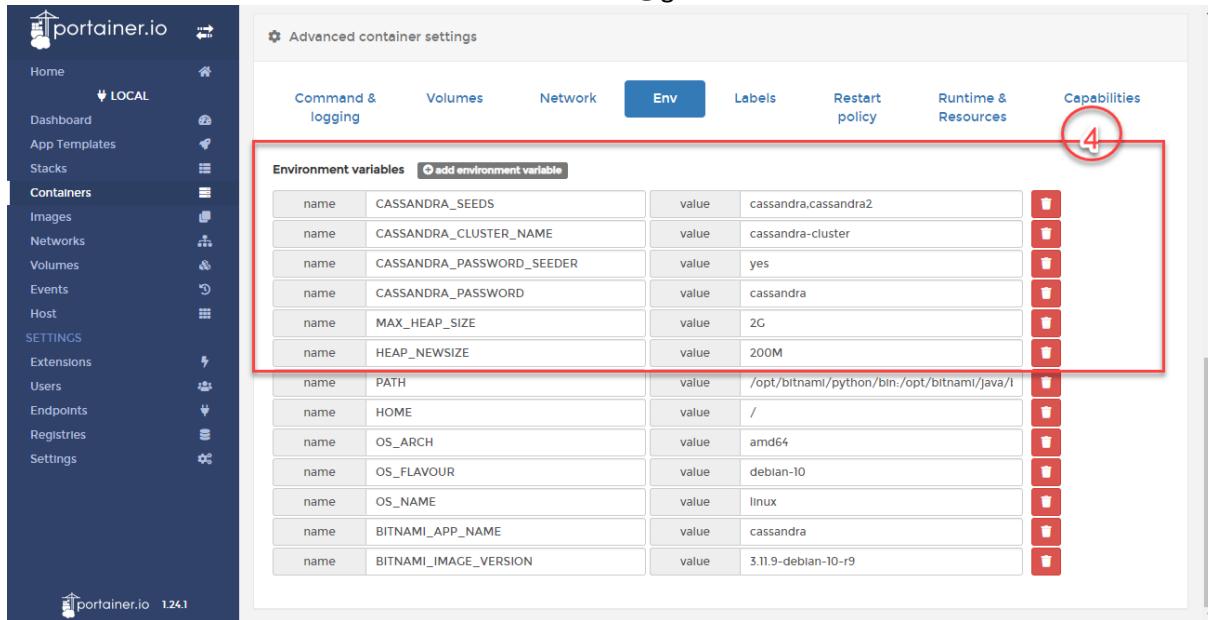
2 choisir la bonne image

3 configurations de ports et leur renvois 'host-container'

4 -configurer les variables d'environnement

BASE DE DONNEES NO SQL Cassandra

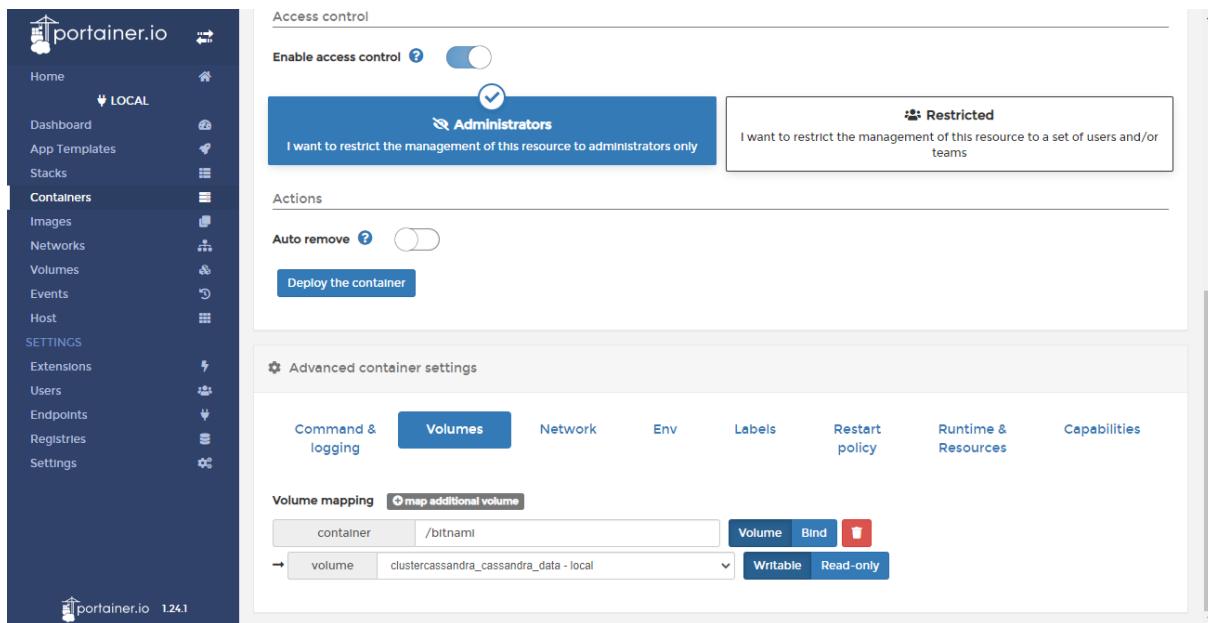
Formateur : Sellami Mokhtar
mokhtar.sellami@gmail.com



The screenshot shows the Portainer.io interface for managing a container. On the left is a sidebar with various options like Home, Dashboard, App Templates, Stacks, Containers, Images, Networks, Volumes, Events, Host, SETTINGS, Extensions, Users, Endpoints, Registries, and Settings. The main area is titled 'Advanced container settings' for a specific container. It has tabs for Command & logging, Volumes, Network, Env (which is selected), Labels, Restart policy, Runtime & Resources, and Capabilities. The Env tab displays a list of environment variables with their values. A red circle with the number '4' is drawn around the Capabilities tab.

name	value
CASSANDRA_SEEDS	cassandra.cassandra2
CASSANDRA_CLUSTER_NAME	cassandra-cluster
CASSANDRA_PASSWORD_SEEDER	yes
CASSANDRA_PASSWORD	cassandra
MAX_HEAP_SIZE	2G
HEAP_NEWSIZE	200M
PATH	/opt/bitnami/python/bin:/opt/bitnami/java/bin
HOME	/
OS_ARCH	amd64
OS_FLAVOUR	debian-10
OS_NAME	linux
BITNAMI_APP_NAME	cassandra
BITNAMI_IMAGE_VERSION	3.11.9-debian-10-r9

5 -Ajouter un volume a ton conteneur



This screenshot shows the 'Advanced container settings' page for a container, with the 'Volumes' tab selected. It includes sections for Access control (with 'Enable access control' turned on, showing options for Administrators and Restricted), Actions (with 'Auto remove' turned off and a 'Deploy the container' button), and Volume mapping. A red circle highlights the 'Volume mapping' section.

Access control

- Enable access control:
- Administrators**: I want to restrict the management of this resource to administrators only
- Restricted**: I want to restrict the management of this resource to a set of users and/or teams

Actions

- Auto remove:
- Deploy the container

Advanced container settings

Volumes tab selected.

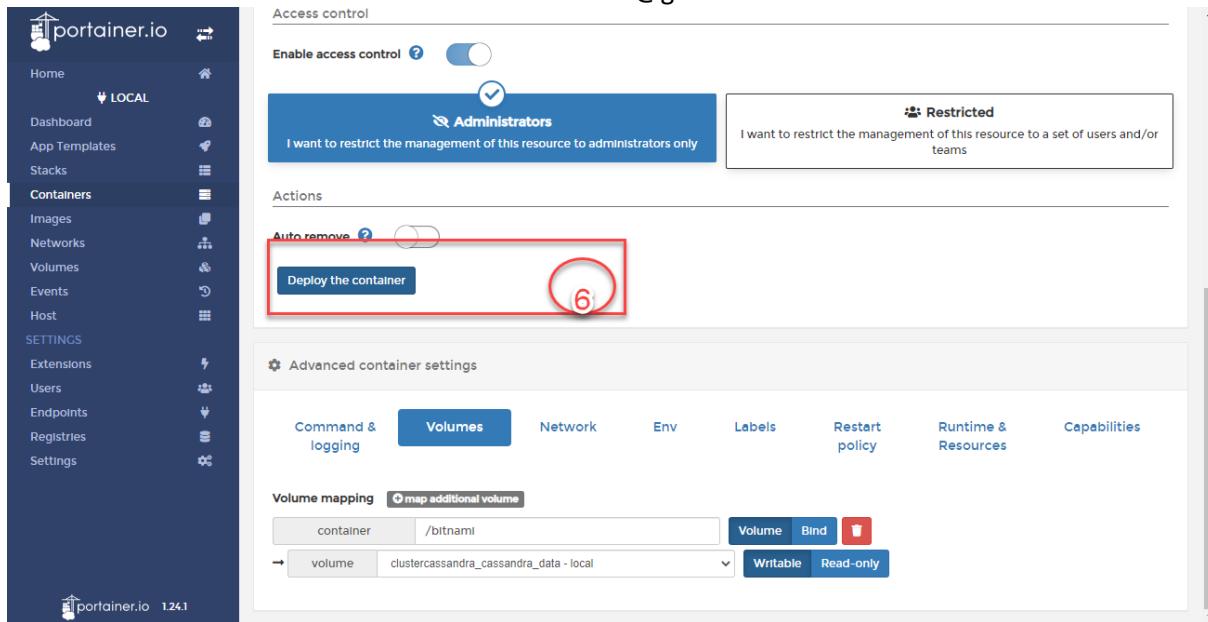
Volume mapping

container	volume	Volume	Bind
/bitnami	clustercassandra_cassandra_data - local	Writable	Read-only

6- déployer ton conteneur

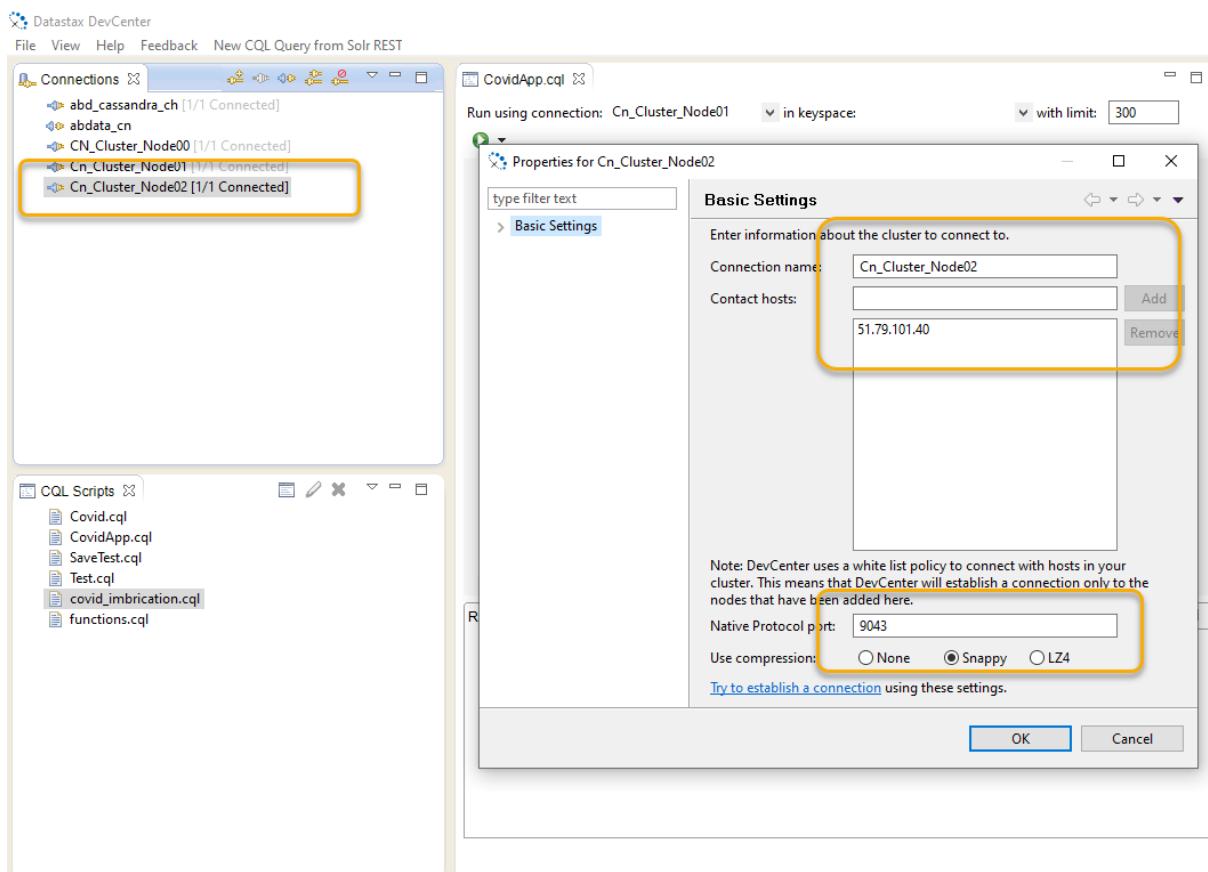
BASE DE DONNEES NO SQL Cassandra

Formateur : Sellami Mokhtar
mokhtar.sellami@gmail.com



The screenshot shows the Portainer.io interface for managing Docker containers. On the left is a sidebar with various options like Home, Dashboard, App Templates, Stacks, Containers (which is selected), Images, Networks, Volumes, Events, Host, SETTINGS, Extensions, Users, Endpoints, Registries, and Settings. The main area is titled 'Access control' with a toggle switch for 'Enable access control'. Below it are two sections: 'Administrators' (selected) and 'Restricted'. Under 'Administrators', there's a note: 'I want to restrict the management of this resource to administrators only'. Under 'Restricted', there's a note: 'I want to restrict the management of this resource to a set of users and/or teams'. At the bottom of this section is a large blue button labeled 'Deploy the container' with a circled number '6' above it. Below this are tabs for 'Command & logging', 'Volumes' (which is selected), Network, Env, Labels, Restart policy, Runtime & Resources, and Capabilities. Under the 'Volumes' tab, there's a 'Volume mapping' section with a 'map additional volume' button. It shows a mapping from a container volume '/bitnami' to a host volume 'clustercassandra_cassandra_data - local' with 'Bind' type, 'Writable' status, and 'Read-only' checked.

Vérification de la connexion au nouveau Node via DevCenter

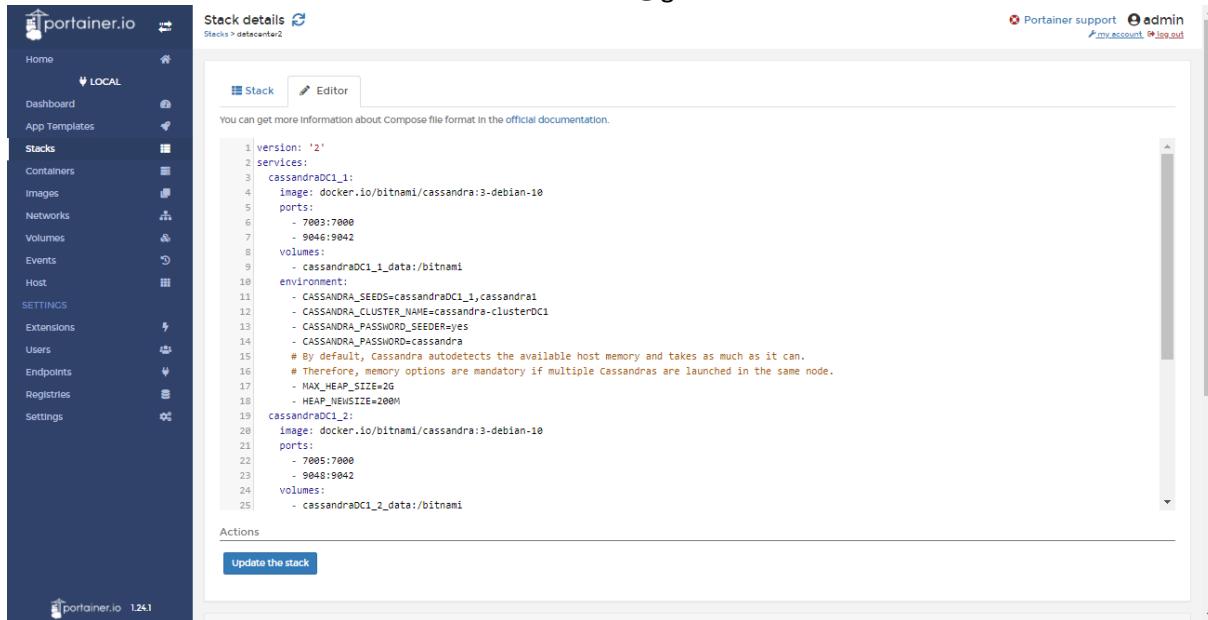


The screenshot shows the DataStax DevCenter application. On the left, the 'Connections' pane lists several connections: 'abd_cassandra_ch [1/1 Connected]', 'abdata_cn', 'CN_Cluster_Node00 [1/1 Connected]', 'Cn_Cluster_Node01 [1/1 Connected]', and 'Cn_Cluster_Node02 [1/1 Connected]'. The 'CQL Scripts' pane on the right lists several CQL script files: Covid.cql, CovidApp.cql, SaveTest.cql, Test.cql, covid_imbrication.cql, and functions.cql. In the center, a dialog box is open for the 'Properties for Cn_Cluster_Node02' connection. The 'Basic Settings' tab is selected. It contains fields for 'Connection name' (set to 'Cn_Cluster_Node02'), 'Contact hosts' (containing '51.79.101.40'), and 'Native Protocol port' (set to '9043'). There are also options for 'Use compression' (with radio buttons for 'None', 'Snappy', and 'LZ4') and a link to 'Try to establish a connection'. The entire 'Contact hosts' input field is highlighted with a yellow box.

Créer un DataCenter 2 via Stack

BASE DE DONNEES NO SQL Cassandra

Formateur : Sellami Mokhtar
mokhtar.sellami@gmail.com

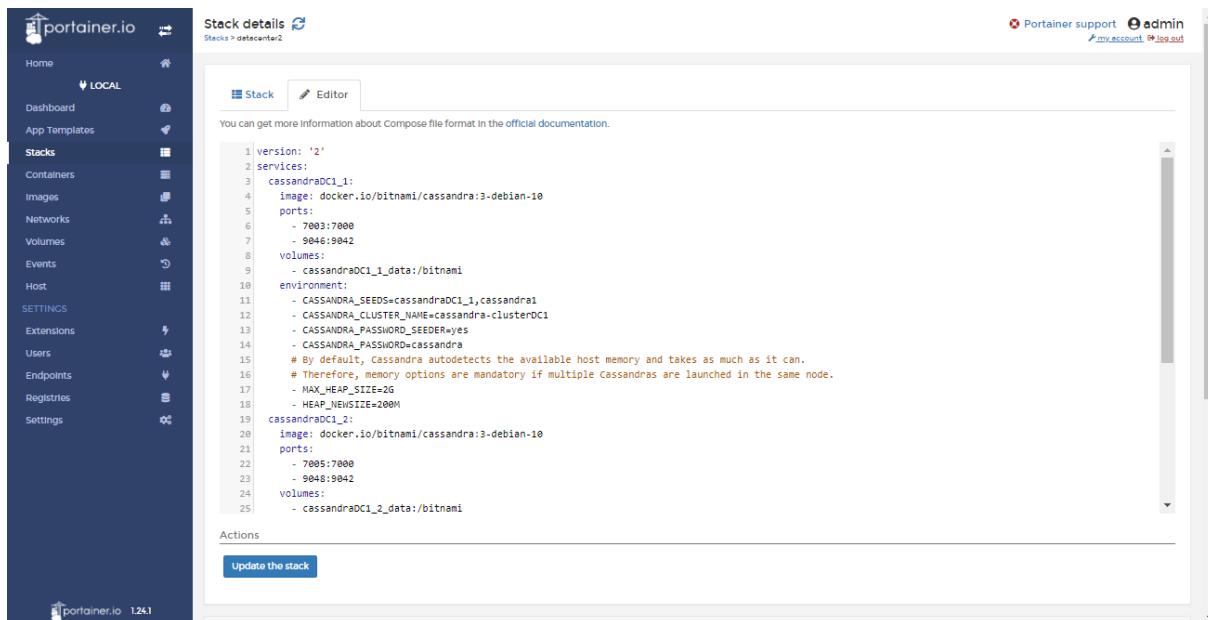


```

version: '2'
services:
  cassandraDC1_1:
    image: docker.io/bitnami/cassandra:3-debian-10
    ports:
      - 7003:7000
      - 9046:9042
    volumes:
      - cassandraDC1_1_data:/bitnami
    environment:
      - CASSANDRA_SEEDS=cassandraDC1_1,cassandra1
      - CASSANDRA_CLUSTER_NAME=cassandra-clusterDC1
      - CASSANDRA_PASSWORD_SEEDER=yes
      - CASSANDRA_PASSWORD=cassandra
    # By default, Cassandra autodetects the available host memory and takes as much as it can.
    # Therefore, memory options are mandatory if multiple Cassandras are launched in the same node.
    - MAX_HEAP_SIZE=2G
    - HEAP_NEWSIZE=20M
  cassandraDC1_2:
    image: docker.io/bitnami/cassandra:3-debian-10
    ports:
      - 7005:7000
      - 9048:9042
    volumes:
      - cassandraDC1_2_data:/bitnami

```

Configure ce DataCenter et le déployer



```

version: '2'
services:
  cassandraDC1_1:
    image: docker.io/bitnami/cassandra:3-debian-10
    ports:
      - 7003:7000
      - 9046:9042
    volumes:
      - cassandraDC1_1_data:/bitnami
    environment:
      - CASSANDRA_SEEDS=cassandraDC1_1,cassandra1
      - CASSANDRA_CLUSTER_NAME=cassandra-clusterDC1
      - CASSANDRA_PASSWORD_SEEDER=yes
      - CASSANDRA_PASSWORD=cassandra
    # By default, Cassandra autodetects the available host memory and takes as much as it can.
    # Therefore, memory options are mandatory if multiple Cassandras are launched in the same node.
    - MAX_HEAP_SIZE=2G
    - HEAP_NEWSIZE=20M
  cassandraDC1_2:
    image: docker.io/bitnami/cassandra:3-debian-10
    ports:
      - 7005:7000
      - 9048:9042
    volumes:
      - cassandraDC1_2_data:/bitnami

```