Chaker ALLAOUI
chaker.allaoui@gmail.com

APACHE CASSANDRA

FICHE TECHNIQUE

TABLE DES MATIERES

Contenu

Présentation

HISTORIQUE

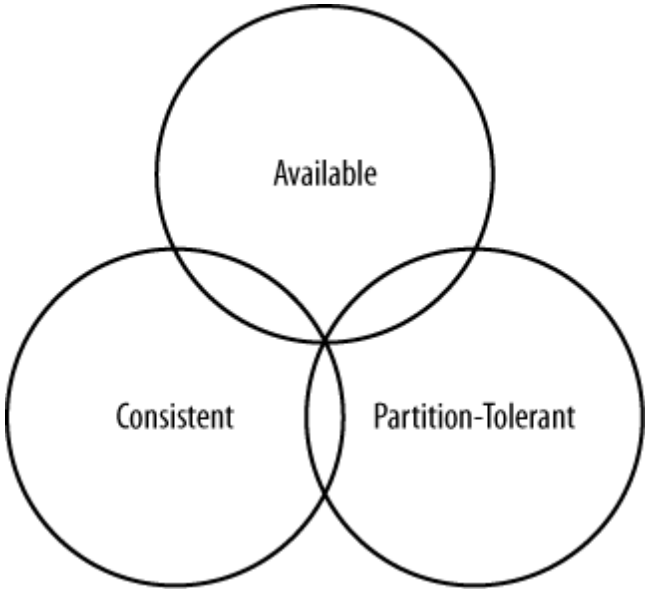
Apache Cassandra est une base de données de la famille NoSQL très en vogue. Elle se classe parmi les bases orientées colonnes tout comme HBase, Apache Accumulo, Big Table. Cette base a été développée à l'origine par des ingénieurs de Facebook pour leurs besoins en interne avant d'être mise à la disposition du grand public en open-source.

CARACTERISTIQUES

- **Tolérance aux pannes** : les données d'un nœud (un nœud est une instance de Cassandra) sont automatiquement répliquées vers d'autres nœuds (différentes machines). Ainsi, si un nœud est hors service les données présentes sont disponibles à travers d'autres nœuds. Le terme de facteur de réplication désigne le nombre de nœuds où la donnée est répliquée. Par ailleurs, l'architecture de Cassandra définit le terme de cluster comme étant un groupe d'au moins deux nœuds et un data center comme étant des clusters délocalisés. Cassandra permet d'assurer la réplication à travers différents data center. Les nœuds qui sont tombés peuvent être remplacés sans indisponibilité du service.
- **Décentralisé** : dans un cluster tous les nœuds sont égaux. Il n'y pas de notion de maitre, ni d'esclave, ni de processus qui aurait à sa charge la gestion, ni même de goulet d'étranglement au niveau de la partie réseau. Le protocole GOSSIP assure la découvert, la localisation et la collecte de toutes les informations sur l'état des nœuds d'un cluster.
- **Modèle de données riche** : le modèle de données proposé par Cassandra basé sur la notion de clé/valeur permet de développer de nombreux cas d'utilisation dans le monde du Web.
- **Élastique** : la scalabilité est linéaire. Le débit d'écriture et de lecture augmente de façon linéaire lorsqu'un nouveau serveur est ajouté dans le cluster. Par ailleurs, Cassandra assure qu'il n'y aura pas d'indisponibilité du système ni d'interruption au niveau des applications.
- **Haute disponibilité** : possibilité de spécifier le niveau de cohérence concernant la lecture et l'écriture. On parle alors de Tuneable Consistency. Apache Cassandra ne dispose pas de transaction. L'écriture des données est très rapide comparée au monde des bases de données relationnelles.

Théorème CAP

Dans le monde des bases de données NoSQL, on entend souvent parler du Théorème CAP. Ce théorème établit 3 paramètres sur lesquels on peut jouer pour configurer une base de données distribuée :



La cohérence (C pour Consistency)

La disponibilité (A pour Availability)

La tolérance aux pannes et aux coupures réseaux (P pour Partition-tolerance)

Le théorème postule que pour toute base de données distribuée, on ne peut choisir que 2 de ces 3 paramètres, jamais les 3 en même temps. En théorie, on peut donc choisir les couples suivants :

a. Cohérence et disponibilité (CA) donc non résistante aux pannes (P)

b. Cohérence et tolérance aux pannes (CP) donc non disponible à 100% (A)

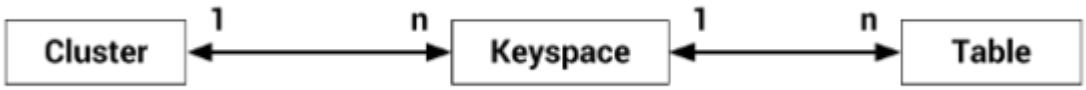
c. Disponibilité et tolérance aux pannes (AP) donc non cohérente à 100% (C)

Ceci est la théorie. En pratique, on se rend compte que le paramètre P est plus ou moins imposé. En effet, les coupures réseaux cela arrive, c'est inévitable. Du coup, le choix se résume en fin de compte à CP ou AP. Cassandra fait clairement le choix de AP pour une tolérance aux pannes et une disponibilité absolue. En contrepartie, Cassandra sacrifie la cohérence absolue (au sens ACID du terme) contre une cohérence finale, c'est à dire une cohérence forte obtenue après une convergence des données (garantie par un ensemble de mécanisme d'anti-entropie).

Architecture

LES CONCEPTS

Apache Cassandra est un système permettant de gérer une grande quantité de données de manière distribuée. Ces dernières peuvent être structurées, semi-structurées ou pas structurées du tout. Cassandra a été conçu pour être hautement scalable sur un grand nombre de serveurs tout en ne présentant pas de Single Point Of Failure (SPOF). Cassandra fournit un schéma de données dynamique afin d'offrir un maximum de flexibilité et de performance. Mais pour bien comprendre cet outil, il faut tout d'abord bien assimiler le vocabulaire de base.



Cluster : un cluster est un regroupement des nœuds qui se communiquent pour la gestion de données.

Keyspace : c'est l'équivalent d'une database dans le monde des bases de données relationnelles. À noter qu'il est possible d'avoir plusieurs « Keyspaces » sur un même serveur.

Colonne (Column) : une colonne est composée d'un nom, d'une valeur et d'un timestamp.

Ligne (Row) : les colonnes sont regroupées en Rows. Une Row est représentée par une clé et une valeur. Il existe deux types de rows : les « wide» et les « skinny» pour le stockage des données.

Une famille de colonnes (Column family) : c'est l'objet principal de données et peut être assimilé à une table dans le monde des bases de données relationnelles. Toutes les rows sont regroupées dans les column family.

PARTITION DES DONNEES

Il est possible de configurer le partitionnement pour une famille de colonnes en précisant que l'on veut que cela soit géré avec une stratégie de type Ordered Partitioners. Ce mode peut, en effet, avoir un intérêt si l'on souhaite récupérer une plage de lignes comprises entre deux valeurs (chose qui n'est pas possible si le hash MD5 des clés des lignes est utilisé).

Cependant, il est conseillé d'utiliser plutôt une deuxième clé d'indexation positionnée sur la colonne contenant les informations voulues. En effet, utiliser la stratégie Ordered Partitionners a les conséquences suivantes : l'écriture séquentielle peut entraîner des hotspots : si l'application tente d'écrire ou de mettre à jour un ensemble séquentiel de lignes, alors l'écriture ne sera pas distribuée dans le cluster ; un overhead accru pour l'administration du load balancer dans le cluster : les administrateurs doivent calculer manuellement les plages de jetons afin de les répartir dans le cluster ; répartition inégale de charge pour des familles de colonnes multiples.

Installation

Une machine virtuelle Java est indispensable pour l'installation d'apache Cassandra.

Vous devez définir votre JAVA_HOME qui doit pointer sur le répertoire de votre JDK.

Après avoir télécharger le code binaire d'apache cassandra depuis le site web : **planetcassandra.org**

Décompressez l'archive téléchargée, dans le répertoire C:\Program Files (x86)\apache-cassandra-x.x.x.x. Nous remarquons la structure suivante très proche des outils fournis par Apache :

bin : contient les scripts d'exécution. Vous y trouverez principalement le script de démarrage de Cassandra (cassandra.bat), le script pour démarrer le client en ligne de commande (cassandra-cli.bat), un script Python pour le client CQL (cqlsh), un script pour l'administration d'un cluster (nodetool.bat).

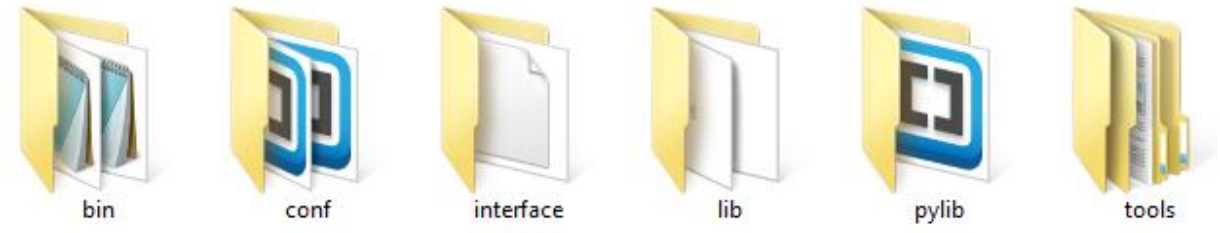
conf : contient les fichiers de configuration pour Cassandra. Vous y trouverez principalement le fichier cassandra.yaml dédié à la configuration de Cassandra et le fichier log4j-server.properties dédié à la configuration des messages logs.

interface : contient la configuration pour Thrift qui assure l'intéropérabilité entre Apache Cassandra et les API clientes. Ainsi grâce à Thrift, Cassandra offre de nombreux connecteurs vers des clients de différents langages (C#, PHP, Java, C++…).

lib : contient les bibliothèques nécessaires à l'exécution d'Apache Cassandra ;

pylib : contient les bibliothèques Python pour Thrift.

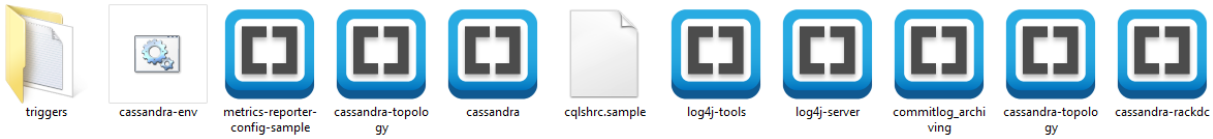
tools : contient des outils tiers et notamment l'outil Cassandra-Stress (cassandra-stress.bat) idéal pour effectuer des benchmarks sur son cluster.



Configuration

Dans un premier temps, ajoutez dans la variable PATH, le chemin du répertoire BIN de Cassandra et définissez votre **CASSANDRA_HOME** pour qu'il pointe sur votre dossier d'installation.

Tous les fichiers de configuration de Cassandra sont sous le dossier « **apache-cassandra-x.x.x.x/conf** ».



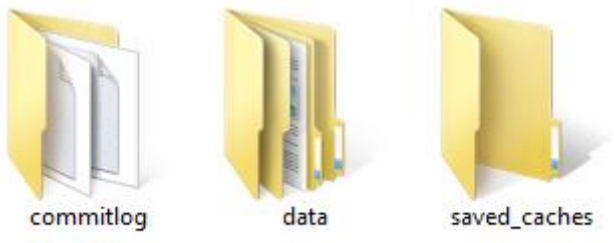
Depuis le fichier de configuration « **conf/cassandra.yaml** », modifiez les entrées suivantes :

data_file_directories : précise le répertoire où Cassandra stockera les données sur le disque. De préférence créer un dossier sur une deuxième partition pour accueillir les données et les logs de Cassandra, la valeur choisie est C: /CassandraData/data ;

commitlog_directory : c'est le dossier où cassandra enregistre ces commit utilisé pour gérer le cache du commit, la valeur choisie est C: /CassandraData/commitlog ;

saved_caches_directory : c'est le dossier des données manipulées par Cassandra dans le cache avant d'être persister, la valeur choisie est C: /CassandraData/saved_caches.

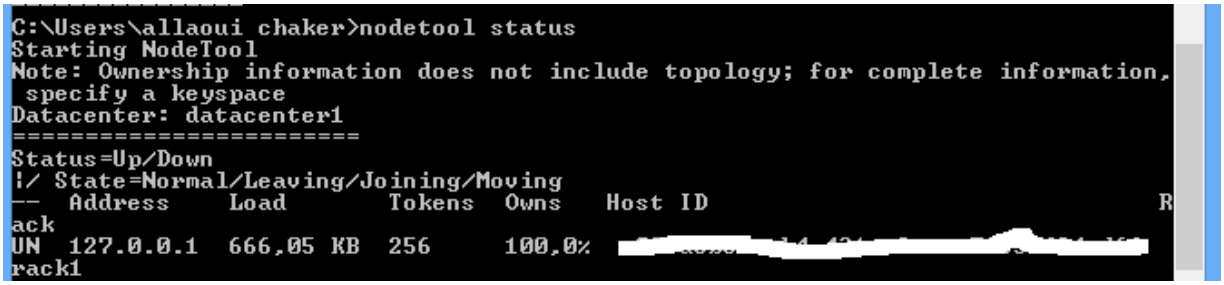
La configuration pour les logs est définie dans le fichier « **log4j-server.properties »**. Vous pouvez par exemple préciser le fichier d'écriture des messages via l'entrée **log4j.appender.R.File,** la valeur choisie est C: /CassandraData/system.log.



Monitoring

NODETOOL

L'utilitaire nodetool est un outil en ligne de commande qui permet de superviser et d'administrer Apache Cassandra. Cet utilitaire est fourni avec la distribution de Cassandra via l'exécutable nodetool.bat depuis le répertoire bin de Cassandra.Vous pouvez par exemple avoir un état complet d'un cluster, faire un snapshot et donner des statistiques sur des keyspaces et ses familles de colonnes.

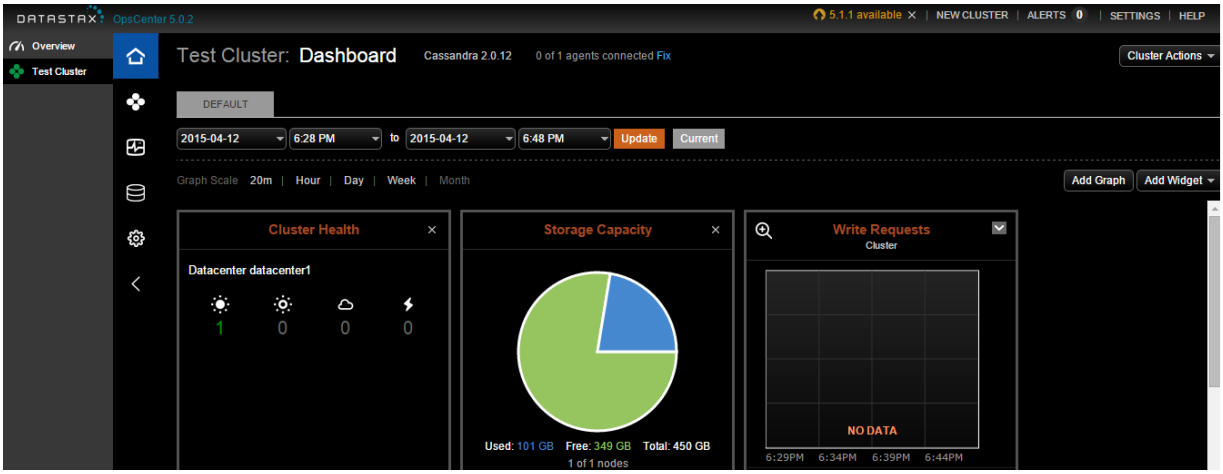


OPSCENTER

OpsCenter est un outil Web de supervision fourni par la société DataStax. L'outil est gratuit dans sa distribution communautaire. Dans sa version Windows, OpsCenter n'est pas disponible unitairement en téléchargement. Il faudra passer par le package Community pour obtenir une version.

OpsCenter fournit tout un ensemble de fonctionnalités pour la gestion d'un cluster (visualisation d'un cluser, métriques…).

Il peut également s'occuper de visualiser les données et fournit des interfaces graphiques pour modifier le schéma.



CQL3.1

PRESENTATION

CQL veut dire Cassandra Query Language, et nous sommes à la version 3. La première version a été une tentative expérimentale d'introduire un langage de requêtage pour Cassandra. La deuxième version de CQL a été conçue pour requêter les wide rows mais n'était pas assez flexible pour s'adapter à tous les types de modélisation qui existent dans Apache Cassandra.

DDL (DATE DEFINITION LANGUAGE)

```
CREATE TABLE developer(
        developer_id bigint,
        firstname text,
        lastname text,
        age int,
        task varchar,
        PRIMARY KEY(developer_id));
```

Le mot clé PRIMARY KEY permet de définir la clé primaire (clé de partition) dans Cassandra. Dans l'exemple c'est `developer_id`.

Outre la création de table, il est possible de modifier et de supprimer les tables avec la syntaxe ALTER TABLE / DROP TABLE.

DML (DATA MANIPULATION LANGUAGE)

Pour manipuler les données, CQL3 propose la même chose que SQL pour les 3 types d'opérations : insertion, mise à jour et suppression.

```
INSERT INTO developer(developer_id, firstname, lastname, age, task)
VALUES(0123, 'Toto', 'XYZ', 34, 'DataBase Design');

UPDATE developer SET age = 35 WHERE developer_id = 0123;

DELETE FROM developer WHERE developer_id=0123;
```

Liens utiles

DONWLOAD

http://planetcassandra.org/download

DOCUMENTATION

http://docs.datastax.com/en/index.html

DRIVERS

http://docs.datastax.com/en/developer/driver-matrix/doc/common/driverMatrix.html

OPSCENTER

http://docs.datastax.com/en/opscenter/5.1/opsc/about_c.html

CQL3.1

http://docs.datastax.com/en/cql/3.1/cql/cql_intro_c.html