# Ressource R5.C.05
# Nouveaux paradigmes de base de données
# BD5
# Semaine 41

IUT La Rochelle
BUT INFO3  Développeur IA

M. Sakkari

# Plan

1. Les bases de Données Orientées  Document
2. Les bases de Données Orientées Graphe
3. Les bases de Données Orientées  Clé-Valeur
4. **Les bases de Données Orientées colonnes**

# Les bases de Données Orientées colonnes Cassandra

# I. Cassandra ?

Le cours de cette semaine présente Cassandra, un SGBD conçu pour le stockage de mégadonnées sous forme de tables similaires à celles de SGBDR.

# I. Cassandra ?

- Cassandra est une base de données distribuée NoSQL.
- Elle est open source, non relationnelle et largement distribuée.
- Elle utilise un langage similaire à SQL pour les interroger.
- Elle possède un plugin permet de traiter les données avec Spark.

# II. Puissance et résilience grâce à la distribution

L'un des principaux atouts de Cassandra est que ses bases de données sont distribuées.« Distribué » signifie que :

- Cassandra peut s'exécuter sur plusieurs machines tout en apparaissant aux utilisateurs comme un tout unifié.
- Les données sont réparties sur plusieurs machines, avec ou sans réplication, ce qui signifie que certaines machines peuvent partager les mêmes données.

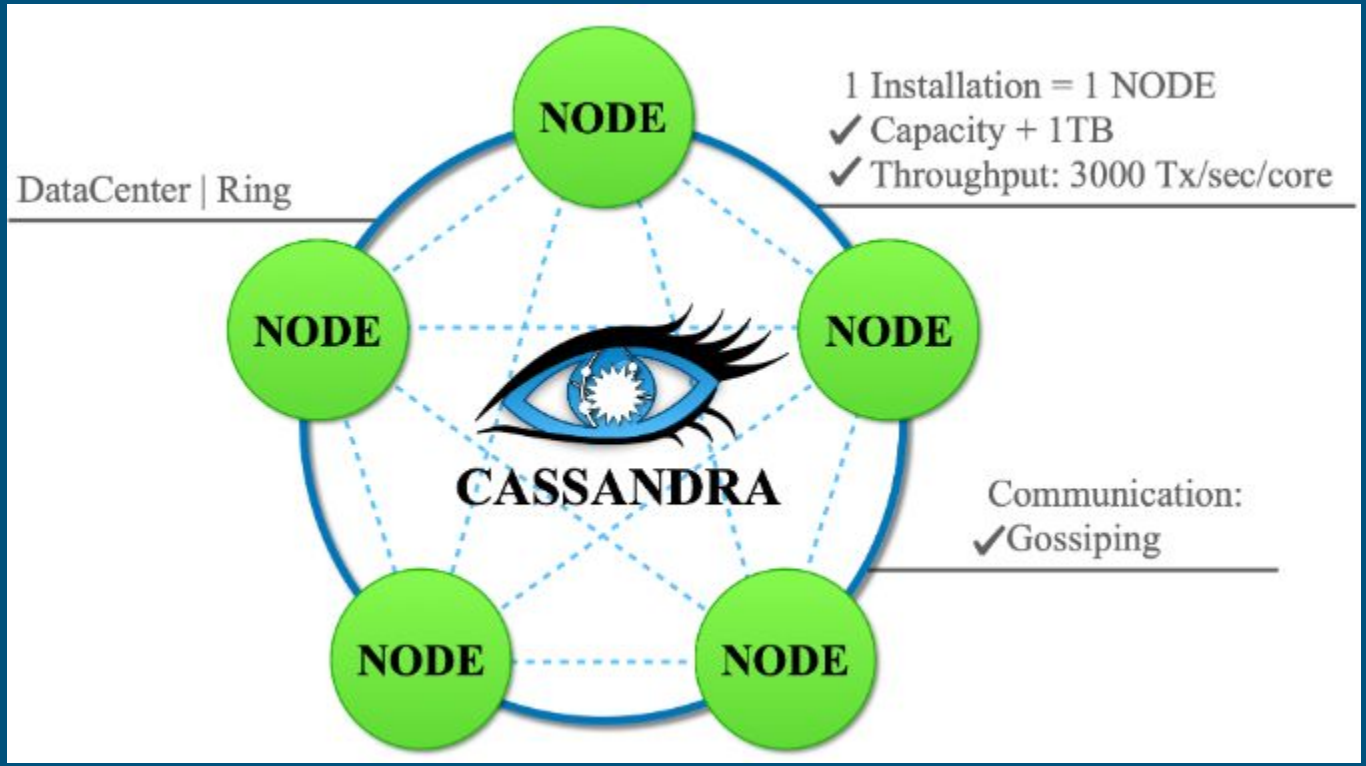
# II. Puissance et résilience grâce à la distribution

- Les traitements sont effectués simultanément sur ces machines en fonction des données qu'elles possèdent et des opérations à réaliser.
- Cassandra est également décentralisé, c'est-à-dire qu'aucune machine n'a un rôle particulier, contrairement à Hadoop (namenode, datanode, nodemanager. . .)

# II. Puissance et résilience grâce à la distribution

- Il n'y a pas grand intérêt à exécuter Cassandra en tant que nœud unique, bien qu'il soit très utile de le faire pour vous aider à vous familiariser avec son fonctionnement.
- Mais pour tirer le meilleur parti de Cassandra, vous devez l'exécuter sur plusieurs machines

# III. Architecture Cassandre



# III. Architecture Cassandra

- Apache Cassandra est une solution de base de données NoSQL, **distribuée** et **hautement évolutive**, développée pour gérer les données critiques sans aucune défaillance.
- Apache Cassandra possède une architecture **de cluster Peer to Peer** dans laquelle il n'y a pas de concept de maître et d'esclaves.

# III. Architecture Cassandre

- Dans l'architecture Cassandra, le nœud représente l’unité de base d'un cluster.
- Il est utilisé pour stocker les données .
- Dans le cluster Cassandra, il peut y avoir de nombreux nœuds et, ensemble, ils fournissent la fonctionnalité distribuée de Cassandra.

# III. Architecture Cassandra

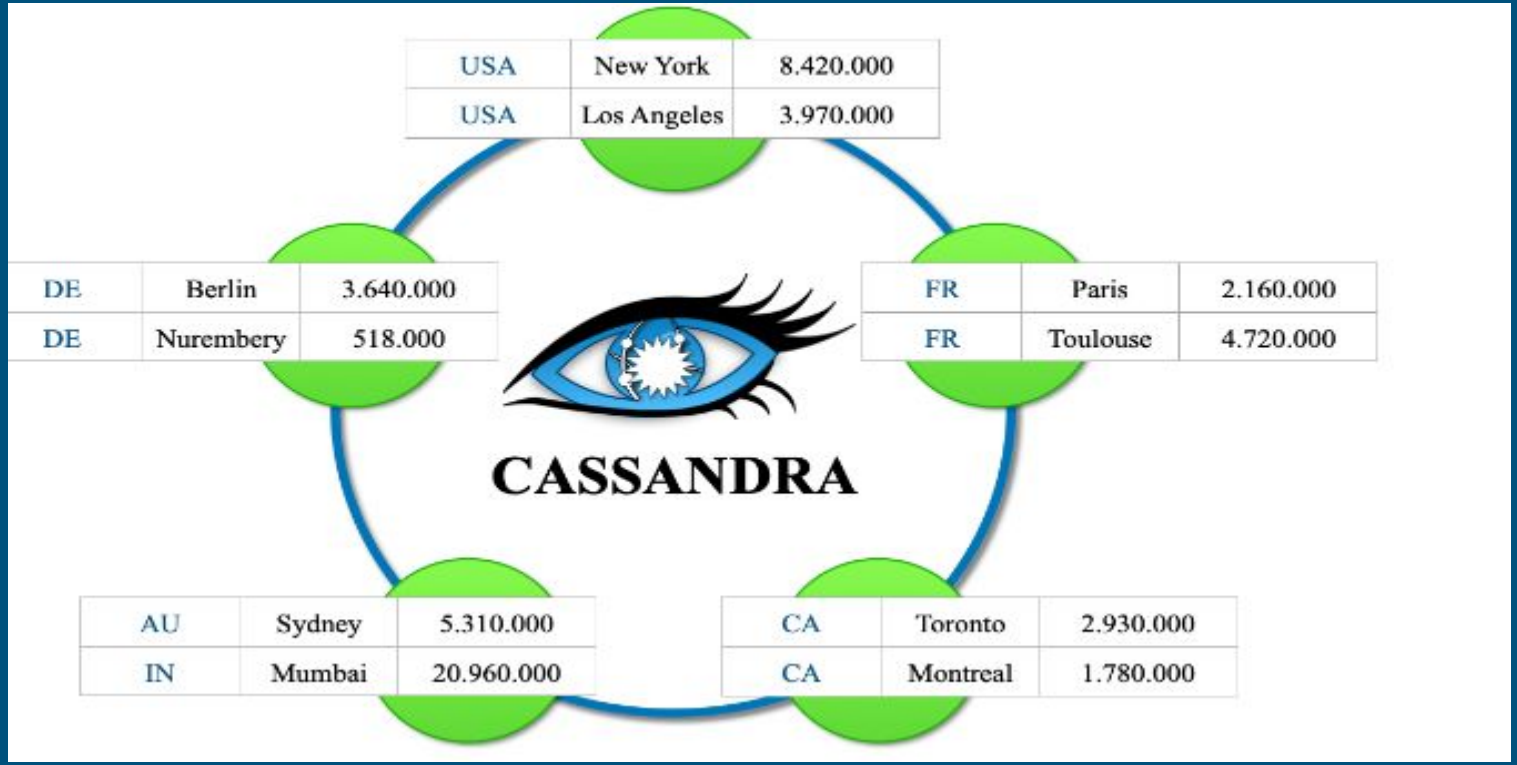
- Chaque ligne de Cassandra possède **une clé de partition**.
- Après avoir appliqué le hachage sur la clé de partition, un **token** unique est généré pour chaque ligne.
- Chaque nœud dispose d'une plage de tokens attribués, de sorte que les lignes de tokens similaires sont stockées sur le même nœud.

# III. Architecture Cassandre


| COUNTRY | CITY | POPULATION |
|---|---|---|
| USA | New York | 8.420.000 |
| USA | Los Angeles | 3.970.000 |
| FR | Paris | 2.160.000 |
| DE | Berlin | 3.640.000 |
| AU | Sydney | 5.310.000 |
| DE | Nurembery | 518.000 |
| CA | Toronto | 2.930.000 |
| CA | Montreal | 1.780.000 |
| FR | Toulouse | 4.720.000 |
| IN | Mumbai | 20.960.000 |

Partition Key

# III. Architecture Cassandre

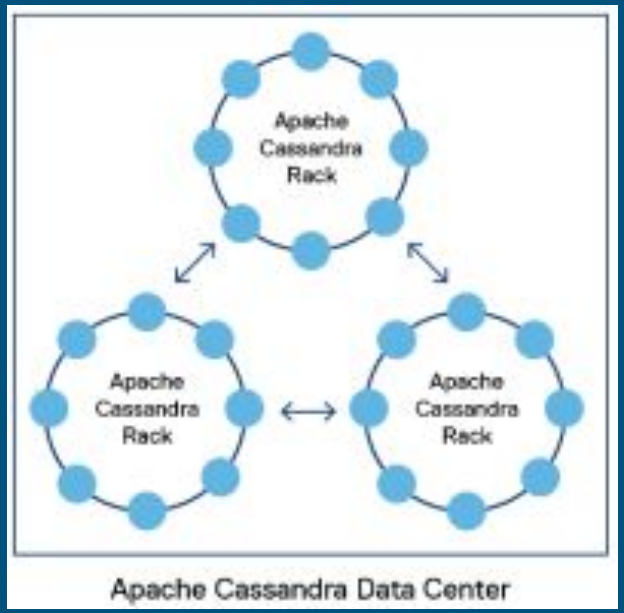


# III. Architecture Cassandre

- Les machines, appelées **nodes**, sont organisés en un **cluster**/**anneau**/**ring**.
- L'architecture de Cassandra ressemble à un anneau dans lequel les nœuds sont distribués logiquement et communiquent entre eux à l'aide du protocole **Gossip** pour échanger l'état des nœuds.

# III. Architecture Cassandra

- Le **Cassandra Data Center** est l'ensemble des **cluster** associés.
- Ils peuvent être des centres de données physiques ou des centres de données logiques et, en fonction de la charge de travail, un centre de données distinct peut être utilisé.



# III. Architecture Cassandra

- Chaque nœud du cluster Cassandra stocke les journaux d'opérations écrits dans un **Commit Log** pour maintenir la durabilité des données.
- Après cela, les données sont écrites dans la structure en mémoire de chaque nœud appelée **memtable.**
- Ainsi, lorsque la memtable est pleine, les données sont écrites dans le fichier de données **SSTables.**

# III. Architecture Cassandre

- L'opération d'écriture dans Cassandra est automatique et elle est répliquée et partitionnée sur l'ensemble du cluster.
- Pour garantir la cohérence des données sur tous les nœuds, Cassandra utilise les mécanismes de réparation **Hinted Handoff, Read Repair, Anti-Entropy Repair .**
- Ces nœuds communiquent entre eux via un protocole appelé **Gossip**.

# III. Architecture Cassandra

- Le protocole **Gossip** dans Cassandra est un mécanisme de communication décentralisé utilisé pour diffuser des informations entre les nœuds du cluster.
- Chaque nœud échange périodiquement des données avec un sous-ensemble aléatoire de nœuds voisins, ce qui permet de propager les informations sur l'état du cluster, comme les ajouts, suppressions ou défaillances de nœuds, de manière efficace et résiliente.

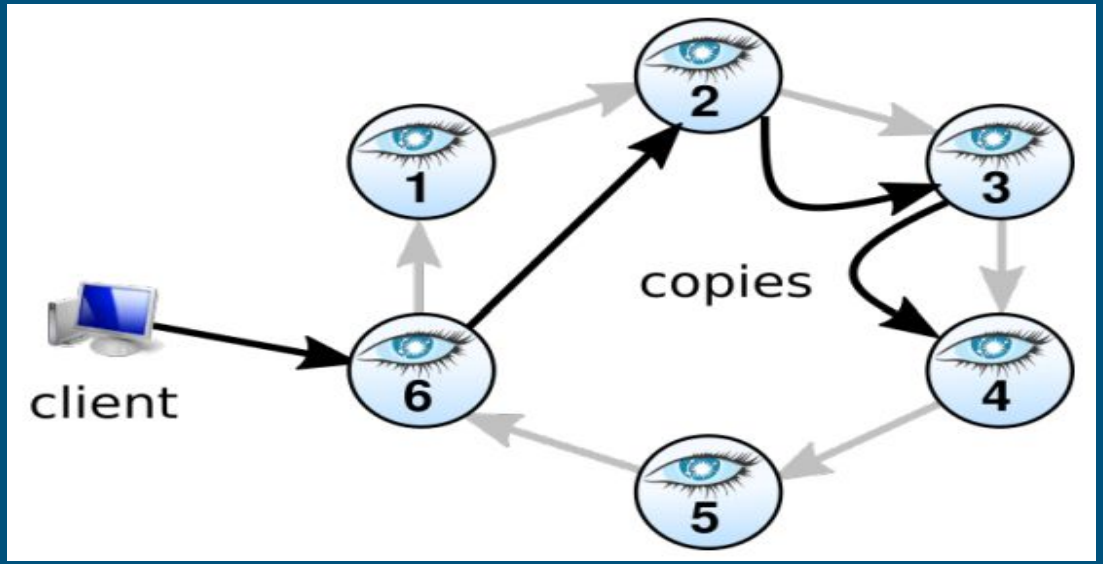
# III. Architecture Cassandra

- Ce protocole assure une cohérence et une disponibilité élevée dans un environnement distribué.
- L'existence principale du protocole Gossip est de garantir que chaque nœud connaît son état et celui de tous les autres nœuds de l'anneau.

https://docs.datastax.com/en/cassandra-oss/3.x/cassandra/architecture/archGossipAbout.html

# III. Architecture Cassandre

- Un client dépose des données sur l'un des nœuds, elles sont dupliquées et envoyées aux nœuds concernés



# III. Architecture Cassandre

- Les mises à jour des données sont donc effectuées de proche en proche, et de manière non synchronisée.
- Ce modèle est sans arbitre central (Master).
- À un moment donné, il est possible que les machines n’aient pas toutes les mêmes valeurs dans les tables, le temps que les mises à jour se propagent.

# III. Architecture Cassandra

Plus de puissance ? Ajoutez plus de nœuds

- L'une des raisons de la popularité de Cassandra est qu'elle permet aux développeurs de faire évoluer leurs bases de données de manière dynamique, à l'aide de matériel standard, sans interruption de service.
- Vous pouvez étendre votre base de données quand vous en avez besoin, et également la réduire si les exigences de l'application le suggèrent.

# IV. Cassandra Query Language (CQL)

- Cassandra fournit le langage de requête Cassandra **CQL** , un langage de type SQL, pour créer, modifier et supprimer des schémas de base de données, ainsi que pour accéder aux données.
- CQL est similaire à SQL qui propose d'effectuer les opérations DML/DDL sur l'ensemble de données.

# IV. Cassandra Query Language (CQL)

- Une fois que le client établit une connexion avec l'un des nœuds Cassandra, ce nœud devient le coordinateur et agit comme un proxy entre le client et le nœud qui contient les données.
- Pour insérer, mettre à jour et supprimer des lignes et des colonnes dans les tables Cassandra, CQL fournit les instructions INSERT, UPDATE et DELETE

https://cassandra.apache.org/doc/latest/cassandra/developing/cql/index.html

# V. Vocabulaire de Cassandra

- **Keyspace**:: définit la manière dont un ensemble de données est répliqué, par centre de données. La réplication correspond au nombre de copies enregistrées par cluster. Les espaces de clés contiennent des tables.
- **Table** : les tables sont composées de lignes et de colonnes. Les tables sont partitionnées en fonction des colonnes fournies dans la clé de partition. Les tables Cassandra peuvent ajouter de nouvelles colonnes aux tables de manière flexible sans aucun temps d'arrêt.

# V. Vocabulaire de Cassandra

- **Partition** : définit la partie obligatoire de la clé primaire que toutes les lignes de Cassandra doivent posséder pour identifier le nœud d'un cluster où la ligne est stockée. Toutes les requêtes performantes fournissent la clé de partition dans la requête.
- **Row :** contient une collection de colonnes identifiées par une clé primaire unique.
- **Column :** une donnée unique avec un type qui appartient à une ligne.

# VI. Labs


Lab n°1 :

1. Rendez-vous sur https://www.datastax.com/learn/cassandra-fundamentals/inserts-updates-deletes

2. Connectez-vous avec LinkedIn.

3. Choisissez "Start with Cassandra DB".

4. Pour chaque lab, veuillez fournir une interprétation détaillée de chaque étape de chaque lab, accompagnée de captures d'écran

# VI. Labs

Lab n°2 :

1. Rendez-vous sur https://www.datastax.com/learn/data-modeling-by-example/investment-data-model

# VI. Labs

Lab n°3 :

1. Rendez-vous sur https://www.datastax.com/learn/data-modeling-by-example/messaging-data-model

# VI. Labs

Lab n°4 :

1. Rendez-vous sur

https://www.datastax.com/learn/data-modeling-by-example/digital-library-data-model

# VII. Installation de l'image Docker

- Pour obtenir la dernière image, utiliser :

  docker pull cassandra:latest

- Démarrez Cassandra avec une docker run commande :

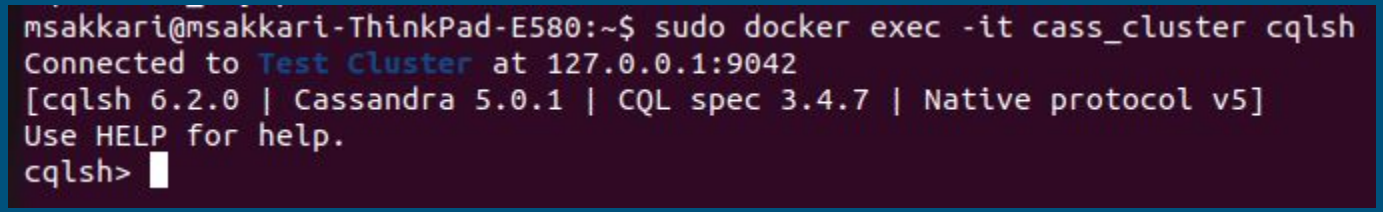
  docker run --name cass_cluster cassandra:latest

- Démarrez le shell CQL cqlsh  pour interagir avec le nœud Cassandra créé.

  docker exec -it cass_cluster cqlsh

# VII. Installation de l'image Docker

- En utilisant la commande sudo docker exec -it cass_cluster cqlsh, vous accédez à l'interface CQL de Cassandra à l'intérieur de votre conteneur Docker. Cela vous permet d'exécuter des requêtes CQL et de gérer votre base de données Cassandra directement depuis le conteneur



# Fin

Merci Pour Votre Attention