# Sujet du TP – Supervision et synchronisation des Données sous Cassandra

M2 - Data - Ynov

Mohammed El Malki

2024 -2025

## Table des matières

# 1 Introduction

Ce TP se scinde en trois parties distinctes, permettant d'aborder successivement les aspects essentiels de la gestion d'un cluster Cassandra en environnement NoSQL. Chaque partie permettra de mettre en œuvre, tester et documenter les solutions suivantes :

— **Cassandra Reaper** pour l'automatisation des réparations ;
— Une solution de **supervision** basée sur MCAC, Prometheus, Grafana et Alert Manager ;
— **Medusa Cassandra** pour la sauvegarde et la restauration des données.

# 2 Partie 1 : Mise en place de Cassandra Reaper

## 2.1 Rappel des Notions de Réparation et de Synchronisation des Données

Dans un environnement Cassandra, les mécanismes de réplication et de synchronisation sont cruciaux pour garantir la cohérence des données, malgré une architecture distribuée reposant sur la cohérence éventuelle. Les principaux mécanismes sont :

— **Cohérence Éventuelle :** Cassandra privilégie la disponibilité et la tolérance aux pannes en utilisant une réplication asynchrone, ce qui peut entraîner des divergences temporaires entre les copies des données.
— **Read Repair :** Lors d'une lecture, Cassandra compare les versions des données sur plusieurs réplicas. En cas de divergence, le nœud détenant une version obsolète est automatiquement mis à jour.
— **Anti-Entropy Repair (Réparation Manuelle) :** Ce processus planifié compare l'intégralité des données sur les nœuds via des arbres de Merkle pour identifier et corriger les incohérences, évitant ainsi l'accumulation de divergences sur le long terme.

L'automatisation de ces réparations est indispensable en production.
**Cassandra Reaper** intervient en orchestrant les cycles de réparation et en fournissant une interface de suivi, garantissant ainsi la cohérence globale du cluster.

## 2.2 Objectifs et Étapes de la Mise en Place de Cassandra Reaper

**Objectif :** Déployer et configurer Cassandra Reaper pour automatiser et superviser les opérations de réparation du cluster Cassandra.
**Étapes proposées :**

1. **Installation :**
   — Installer Cassandra Reaper (via Docker ou sur une machine dédiée).
   — Vérifier les prérequis (JDK, accès réseau, etc.).
2. **Configuration :**
   — Adapter le fichier de configuration pour définir le cluster Cassandra à surveiller.
   — Définir la périodicité des réparations, la plage d'IP à scanner, et les paramètres de granularité.
3. **Intégration avec le Cluster Cassandra :**
   — S'assurer que le cluster est accessible par Reaper et que les métriques de réparation sont disponibles.
   — Effectuer une première réparation manuelle pour valider la configuration.
4. **Suivi et Reporting :**
   — Utiliser l'interface web de Reaper pour visualiser l'état des réparations et consulter l'historique des tâches.
   — Identifier les métriques importantes pour la partie supervision.

# 3 Partie 2 : Supervision

**Objectif :** Mettre en place un système de supervision complet du cluster Cassandra en utilisant MCAC, Prometheus, Grafana et Alert Manager.
**Étapes proposées :**

1. **Collecte de métriques avec MCAC :**
   — Installer MCAC pour extraire les métriques JMX de Cassandra (latence, erreurs, utilisation CPU/mémoire) et de Cassandra Reaper.
   — Vérifier l'accessibilité des endpoints.
2. **Configuration de Prometheus :**
   — Configurer Prometheus pour scraper les endpoints de MCAC.
   — Définir les jobs de scrape et la fréquence de collecte dans le fichier `prometheus.yml`.
3. **Visualisation avec Grafana :**
   — Connecter Grafana à Prometheus et créer (ou importer) des dashboards adaptés.
4. **Alerting avec Alert Manager :**
   — Définir et tester des règles d'alerte (latence élevée, échec d'une réparation, nœud défaillant, etc.) afin d'envoyer des notifications.

# 4 Partie 3 : Sauvegarde et Restauration

**Objectif :** Implémenter un système de sauvegarde et restauration pour assurer la résilience des données en cas de panne ou de corruption.
**Étapes proposées :**

1. **Installation de Medusa Cassandra :**
   — Installer Medusa (via Docker ou sur une machine dédiée) et vérifier la compatibilité avec la version de Cassandra.
2. **Configuration des Backups :**
   — Paramétrer Medusa pour définir les répertoires de sauvegarde, la fréquence des backups et la destination (stockage local ou cloud, ex. S3).
   — Planifier des sauvegardes automatiques via des scripts ou des tâches cron.
3. **Tests de Restauration :**
   — Réaliser une sauvegarde complète du cluster.
   — Simuler la restauration sur un environnement de test pour vérifier l'intégrité des données.
4. **Documentation et Automatisation :**
   — Documenter les procédures de backup/restauration et envisager des alertes en cas d'échec.

## 4.1 Exercice Medusa – Sauvegarde et Restauration

**Objectif :** Réaliser une sauvegarde complète du cluster Cassandra à l'aide de Medusa, puis mettre en œuvre différents scénarios de restauration :

1. Restauration complète sur la même machine.
2. Restauration complète sur une machine différente issue d'un autre cluster.
3. Restauration partielle d'un keyspace ou d'une table à l'aide du *–use-sstableloader*.

### 4.1.1 Jeu de Données et Modèle CQL

Pour cet exercice, nous utiliserons un jeu de données dédié à une application de gestion de commandes en ligne. Le keyspace s'appellera `ecommerce` et contiendra deux tables : `clients` et `commandes`.

Toutefois, Il est possible de vous appuer sur un jeu de données existant ou utiliser cassandra-stress pour générer un workflow de lecture et d'écriture.

```
-- Cr ation du keyspace avec une r plication adapt e pour un environnement
    de test
CREATE KEYSPACE IF NOT EXISTS ecommerce WITH REPLICATION = {
  'class': 'SimpleStrategy',
  'replication_factor': 3
};

-- Table des clients
CREATE TABLE IF NOT EXISTS ecommerce.clients (
  client_id UUID PRIMARY KEY,
  nom TEXT,
  prenom TEXT,
  email TEXT,
  date_inscription TIMESTAMP
);

-- Table des commandes
CREATE TABLE IF NOT EXISTS ecommerce.commandes (
  commande_id UUID PRIMARY KEY,
  client_id UUID,
  date_commande TIMESTAMP,
  montant DECIMAL,
  statut TEXT
);
```

Listing 1 – Modèle CQL

**Données d'Exemple :**

```
-- Insertion de clients
INSERT INTO ecommerce.clients (client_id, nom, prenom, email, date_inscription)
VALUES (uuid(), 'Durand', 'Alice', 'alice.durand@example.com', toTimestamp(now
    ()));

INSERT INTO ecommerce.clients (client_id, nom, prenom, email, date_inscription)
VALUES (uuid(), 'Martin', 'Bob', 'bob.martin@example.com', toTimestamp(now()));

-- Insertion de commandes
INSERT INTO ecommerce.commandes (commande_id, client_id, date_commande, montant
    , statut)
VALUES (uuid(), (SELECT client_id FROM ecommerce.clients LIMIT 1), toTimestamp(
    now()), 99.99, 'En cours');

INSERT INTO ecommerce.commandes (commande_id, client_id, date_commande, montant
    , statut)
VALUES (uuid(), (SELECT client_id FROM ecommerce.clients LIMIT 1), toTimestamp(
    now()), 49.50, 'Livr e');
```

Listing 2 – Insertion de Données

Pour chaque scénario, les étudiants devront :

— Documenter les commandes utilisées, les paramètres de configuration et les logs obtenus.

— Discuter des avantages et des limites d'une restauration complète versus partielle.

# 5 Installation et Configuration des Outils (Linux/CentOS)

Ce guide détaille, étape par étape, l'installation et la configuration des outils utilisés dans le TP. **L'installation se fera sur CentOS uniquement.**

## 5.1 1. Apache Cassandra

### 5.1.1 Prérequis

— Installer JDK 8 ou supérieur et définir la variable d'environnement `JAVA_HOME`.
— Prévoir suffisamment de ressources (RAM, CPU, stockage).

### 5.1.2 Installation

1. **Via Package Manager (CentOS) :**

```
sudo yum update -y
sudo yum install cassandra

```

2. **Via Tarball :**

```
# T l charger depuis https://cassandra.apache.org/
tar -xzf apache-cassandra-X.Y.Z-bin.tar.gz
cd apache-cassandra-X.Y.Z

```

### 5.1.3 Configuration

— Modifier le fichier `conf/cassandra.yaml` :
  — Définir `cluster_name` (ex. "ecommerce_cluster").
  — Configurer `seed_provider` avec les adresses IP des nœuds (un nœud unique est suffisant pour le TP).
  — Régler `listen_address` et `rpc_address` sur l'IP de l'hôte.
— Vérifier que JMX est activé (par défaut activé) pour permettre la collecte des métriques par MCAC et Reaper.

### 5.1.4 Démarrage

```
sudo systemctl start cassandra
nodetool status
```

## 5.2 2. MCAC (Metrics Collector for Apache Cassandra)

### 5.2.1 Installation

```
git clone https://github.com/thelastpickle/cassandra-mcac.git
cd cassandra-mcac
```

### 5.2.2 Configuration

Modifier le fichier de configuration (`mcac.properties` ou `application.yml`) pour définir :
— L'adresse et le port JMX de Cassandra (par défaut : 7199).
— Le port d'exposition des métriques (ex. 8080).

### 5.2.3 Démarrage

```
java -jar mcac.jar --config mcac.properties
```

Vérifier l'accès via : `http://localhost:8080/metrics`

## 5.3 3. Cassandra Reaper

### 5.3.1 Option 1 : Installation avec Docker (Recommandée)

```
docker run -d --name cassandra-reaper \
  -p 8081:8080 \
  -e "CASSANDRA_REAPER_STORAGE_TYPE=memory" \
  -e "CASSANDRA_REAPER_JMX_USERNAME=your_username" \
  -e "CASSANDRA_REAPER_JMX_PASSWORD=your_password" \
  thelastpickle/cassandra-reaper
```

### 5.3.2 Option 2 : Installation Manuelle

1. Télécharger le package release depuis le GitHub de Cassandra Reaper.
2. Modifier le fichier `reaper.yaml` pour spécifier les détails de connexion à Cassandra.
3. Démarrer Reaper :

```
java -jar cassandra-reaper.jar --config path/to/reaper.yaml

```

4. Accéder à l'interface web via : `http://localhost:8081`

## 5.4 4. Prometheus

### 5.4.1 Téléchargement

Télécharger Prometheus depuis Prometheus.io et décompresser l'archive.

### 5.4.2 Configuration

Modifier le fichier `prometheus.yml` pour ajouter les jobs de scrape :

```
global:
  scrape_interval: 15s

scrape_configs:
  - job_name: 'mcac'
    static_configs:
      - targets: ['localhost:8080']
  - job_name: 'cassandra_reaper'
    static_configs:
      - targets: ['localhost:8081']
```

### 5.4.3 Démarrage

```
./prometheus --config.file=prometheus.yml
```

Accéder à l'interface via : `http://localhost:9090`

## 5.5 5. Grafana

### 5.5.1 Installation

Télécharger Grafana depuis Grafana.com ou installer via le package manager CentOS :

```
sudo yum install grafana
```

### 5.5.2 Démarrage

```
sudo systemctl start grafana-server
```

Accéder à l'interface via : `http://localhost:3000` (identifiants par défaut : admin/admin)

### 5.5.3 Configuration

Ajouter Prometheus comme source de données (URL : `http://localhost:9090`) et créer/importer des dashboards adaptés.

## 5.6 6. Alert Manager

### 5.6.1 Téléchargement

Télécharger Alertmanager depuis Prometheus Alertmanager.

### 5.6.2 Configuration

Créer un fichier `alertmanager.yml`, par exemple :

```
global:
  smtp_smarthost: 'smtp.example.com:587'
  smtp_from: 'alertmanager@example.com'
  smtp_auth_username: 'your_username'
  smtp_auth_password: 'your_password'
route:
  receiver: 'email'
receivers:
  - name: 'email'
    email_configs:
      - to: 'admin@example.com'
```

### 5.6.3 Démarrage

```
./alertmanager --config.file=alertmanager.yml
```

Intégrer Alertmanager dans le fichier `prometheus.yml` :

```
alerting:
  alertmanagers:
    - static_configs:
        - targets: ['localhost:9093']
```

## 5.7 7. Medusa Cassandra

### 5.7.1 Installation

Télécharger Medusa depuis le repository GitHub de Medusa Cassandra.
Installation via pip (Python) :

```
pip install medusa-cassandra
```

### 5.7.2 Configuration

Créer un fichier de configuration (`medusa.yml`) :

```
cassandra:
  host: "localhost"
  port: 9042
  user: "cassandra"
  password: "cassandra"
backup:
  backup_location: "/var/backups/medusa"
  backup_name: "ecommerce_backup"
  storage_type: "local"
```

### 5.7.3 Exécution

— **Sauvegarde complète :**

```
medusa backup --backup-name ecommerce_backup

```

— **Restauration complète sur la même machine :**

```
medusa restore --backup-name ecommerce_backup --restore-strategy full

```

— **Restauration partielle (exemple pour la table `commandes`) :**

```
medusa restore --backup-name ecommerce_backup --restore-strategy partial
    --tables ecommerce.commandes

```

Pour restaurer sur une machine différente, copier le dossier de sauvegarde sur la nouvelle machine et exécuter la commande de restauration.

# 6 Livrable

— **Dashboard en format json**
— **Les différents Fichiers de configuration**