Correction du TP4 : Configuration et Implémentation d'un Pipeline de Données avec Docker Compose et Outils Big Data

Introduction

Cette correction détaillée du TP4 présente la configuration et l'implémentation d'un pipeline de données complet utilisant Docker Compose et des outils Big Data (Hadoop, Spark, Flink, Kafka, Cassandra). Le pipeline est conçu pour gérer à la fois des traitements batch et temps réel, offrant ainsi une architecture Lambda complète.

La correction est structurée en sept parties principales, couvrant l'ensemble du cycle de vie du pipeline, de la mise en place de l'environnement de développement jusqu'au déploiement en production, en passant par l'implémentation des différents composants et les tests.

Chaque partie contient des explications détaillées, des exemples de code, des configurations et des bonnes pratiques pour vous guider dans la mise en œuvre de votre propre pipeline de données.

Table des matières

1. [Mise en place de l'environnement de développement avec Docker Compose](#)
2. [Implémentation des extracteurs de données avec Kafka et outils Big Data](#)
3. [Implémentation des transformations avec Spark et Flink](#)
4. [Implémentation des chargeurs pour Hadoop et Cassandra](#)
5. [Orchestration avec Airflow dans un environnement conteneurisé](#)
6. [Tests et validation pour environnement Big Data](#)
7. [Documentation et déploiement avec Docker Compose](#)

Commençons par la mise en place de l'environnement de développement.

Correction Partie 1 : Mise en place de l'environnement de développement avec Docker Compose

La mise en place d'un environnement de développement robuste, reproductible et proche de la production est une étape fondamentale pour tout projet d'ingénierie de données. Dans cette partie, nous allons utiliser Docker Compose pour créer un environnement complet intégrant tous les outils Big Data nécessaires à notre pipeline.

Avantages de Docker Compose pour un environnement Big Data

L'utilisation de Docker Compose pour un environnement Big Data présente de nombreux avantages :

1. **Isolation** : Chaque composant (Hadoop, Spark, Flink, Kafka, Cassandra) s'exécute dans son propre conteneur, évitant les conflits de dépendances.
2. **Reproductibilité** : L'environnement est défini comme code, garantissant que tous les développeurs travaillent dans des conditions identiques.
3. **Proximité avec la production** : L'environnement de développement peut être configuré pour refléter fidèlement l'environnement de production.
4. **Facilité de démarrage** : Un simple `docker-compose up` suffit pour démarrer l'ensemble de l'écosystème.
5. **Gestion des ressources** : Les ressources allouées à chaque service peuvent être contrôlées précisément.

Structure du projet

Commençons par définir une structure de projet adaptée à notre pipeline Big Data :

```
data-pipeline/  
├── docker/                                # Configuration Docker Compose  
│   ├── base.yml                          # Configuration de base  
│   └── dev.yml                            # Configuration de  
développement  
│   ├── test.yml                          # Configuration de test  
│   └── prod.yml                          # Configuration de production  
└── config/                                # Fichiers de configuration des services
```

```

├── hadoop/           # Configuration Hadoop
├── spark/           # Configuration Spark
├── flink/           # Configuration Flink
├── kafka/          # Configuration Kafka
├── cassandra/      # Configuration Cassandra
├── airflow/        # Configuration Airflow
├── src/            # Code source du pipeline
│   ├── extractors/ # Extracteurs de données
│   ├── transformers/ # Transformateurs de données
│   │   ├── batch/ # Transformations batch (Spark)
│   │   └── streaming/ # Transformations streaming
├── (Flink)
│   └── loaders/    # Chargeurs de données
├── dags/           # DAGs Airflow
├── notebooks/     # Notebooks Jupyter pour
l'exploration
├── scripts/       # Scripts utilitaires
├── test/          # Tests
│   ├── unit/      # Tests unitaires
│   ├── integration/ # Tests d'intégration
│   └── data/      # Données de test
└── README.md     # Documentation du projet

```

Installation des prérequis

Avant de commencer, assurez-vous que les outils suivants sont installés sur votre machine de développement :

1. **Docker** : Pour exécuter les conteneurs
2. **Docker Compose** : Pour orchestrer les conteneurs
3. **Git** : Pour la gestion du code source

Installation sur Ubuntu

```

# Mise à jour du système
sudo apt update && sudo apt upgrade -y

# Installation de Docker
sudo apt install -y apt-transport-https ca-certificates curl
software-properties-common
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo
apt-key add -
sudo add-apt-repository "deb [arch=amd64] https://
download.docker.com/linux/ubuntu $(lsb_release -cs) stable"
sudo apt update
sudo apt install -y docker-ce

```

```
# Ajout de l'utilisateur courant au groupe docker
sudo usermod -aG docker $USER
newgrp docker

# Installation de Docker Compose
sudo curl -L "https://github.com/docker/compose/releases/
download/v2.15.1/docker-compose-$(uname -s)-$(uname -m)" -o /
usr/local/bin/docker-compose
sudo chmod +x /usr/local/bin/docker-compose

# Installation de Git
sudo apt install -y git
```

Installation sur macOS

```
# Installation de Homebrew (si non installé)
/bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/
Homebrew/install/HEAD/install.sh)"

# Installation de Docker Desktop (inclut Docker Compose)
brew install --cask docker

# Installation de Git
brew install git
```

Installation sur Windows

1. Téléchargez et installez [Docker Desktop](#)
2. Téléchargez et installez [Git](#)

Configuration de l'environnement Docker Compose

Création des fichiers Docker Compose

Commençons par créer les fichiers Docker Compose nécessaires. Créez d'abord le répertoire du projet et les sous-répertoires :

```
mkdir -p data-pipeline/docker/config/
{hadoop,spark,flink,kafka,cassandra,airflow}
cd data-pipeline
```

Fichier `docker/base.yml`

Ce fichier contient la configuration de base commune à tous les environnements :