Projet individuel - Real Time Data Streaming

Contexte

Notre entreprise a récemment lancé une application de monitoring météo en temps réel pour permettre aux utilisateurs de suivre l'évolution de la météo à travers différentes régions et villes. Nous utiliserons Kafka pour collecter les données et Spark Streaming pour leur traitement.

Objectifs

1. Collecter des données en temps réel à partir de l'API OpenWeatherMap dans un topic Kafka nommé topic-weather.

2. Traiter les données en temps réel avec Spark Streaming pour produire de nouvelles variables et stocker les résultats dans un nouveau topic Kafka nommé topic-weather-final.

3. Fournir un pipeline fonctionnel et prêt à l'emploi.

Installation et Configuration

Création du dépôt

1. Créer un dépôt GitHub nommé projet-esme :

2. Ouvrir un codespace sur le dépôt créé.

Installation de Java

1. Mettre à jour les paquets :

```
sudo apt-get update
```

2. Installer Java JDK 11 :

```
sudo apt-get install openjdk-11-jdk-headless
```

3. Vérifiez l'installation :

```
java --version
```

Configuration de Java

Ajouter Java à votre variable d'environnement :

```
echo 'export JAVA_HOME=/usr/lib/jvm/java-11-openjdk-amd64' >> ~/.bashrc
echo 'export PATH=$JAVA_HOME/bin:$PATH' >> ~/.bashrc
source ~/.bashrc
```

Installation de Kafka

Téléchargez et extrayez Kafka :

```
wget https://archive.apache.org/dist/kafka/2.6.0/kafka_2.12-2.6.0.tgz
tar -xzf kafka_2.12-2.6.0.tgz
```

Lancement de Zookeeper et Kafka

1. Lancer Zookeeper :

```
./kafka_2.12-2.6.0/bin/zookeeper-server-start.sh ./kafka_2.12-2.6.0/config/zookeeper.properties
```

2. Lancer Kafka :

```
./kafka_2.12-2.6.0/bin/kafka-server-start.sh ./kafka_2.12-2.6.0/config/server.properties
```

Création de Topics Kafka

1. Créer un topic nommé topic-weather :

```
./kafka_2.12-2.6.0/bin/kafka-topics.sh --create --bootstrap-server localhost:9092 --replication-factor 1 --partitions 1 --topic topic-weather
```

2. Créer un topic nommé topic-weather-final :

```
./kafka_2.12-2.6.0/bin/kafka-topics.sh --create --bootstrap-server localhost:9092 --replication-factor 1 --partitions 1 --topic topic-weather-final
```

Installation de Spark

Téléchargez et extrayez Spark :

```
wget https://archive.apache.org/dist/spark/spark-3.2.3/spark-3.2.3-bin-hadoop2.7.tgz
tar -xvf spark-3.2.3-bin-hadoop2.7.tgz
```

Configurez Spark :

```
export SPARK_HOME=/workspaces/<votre-repertoire>/spark-3.2.3-bin-hadoop2.7
```

```
export PATH=$SPARK_HOME/bin:$PATH
```

Installation de Python 3.10

1. Mettre à jour Python :

```
sudo apt update
sudo apt install -y python3.10 python3.10-venv python3.10-distutils
```

2. Installer la bibliothèque python-kafka :

```
python3.10 -m pip install kafka-python
```

Exécution des Scripts

1. Producteur Kafka :

```
python3.10 producer.py
```

2. Traitement Spark Streaming :

```
$SPARK_HOME/bin/spark-submit --packages org.apache.spark:spark-sql-kafka-0-10_2.12:3.2.3 spark.py
```

Travail à Compléter

1. Compléter le producteur Kafka (producer.py) pour récupérer les données météo depuis l'API OpenWeatherMap et les envoyer vers topic-weather.

Projet individuel - Real Time Data Streaming

2. Compléter le traitement Spark (spark.py) pour générer de nouvelles variables et envoyer les résultats dans topic-weather-final.

Évaluation

1. Code fonctionnel : Le pipeline doit fonctionner de bout en bout.

2. Documentation : le répo et le code doivent être bien documentés.

3. Présentation orale : Présenter le projet. Des questions vous seront posées.

Bonne chance !