TP : Analyse de logs Apache avec Spark sur HDFS

Objectifs du TP

Ce TP a pour but de manipuler Spark avec PySpark et HDFS afin d'effectuer une analyse simple de logs de serveurs web. Les objectifs sont :

- Charger un fichier de logs Apache/NGINX depuis HDFS dans un RDD.
- Transformer les données brutes pour extraire des informations pertinentes.
- Effectuer des agrégations et des analyses sur les données.

Avant de commencer

Un fichier `docker-compose.yml` vous a été fourni pour configurer l'environnement. Ce fichier est issu de la fusion des configurations utilisées dans des anciens TP. Il regroupe les services nécessaires pour exécuter un environnement incluant à la fois HDFS et Spark.

Avant de l'utiliser, assurez-vous de vérifier les points suivants :

- **Réseaux et dépendances** : vérifier que tous les services sont correctement connectés au réseau et que leurs dépendances sont définies.
- **Ports exposés** : s'assurer qu'aucun conflit n'existe avec d'autres services.
- **Volumes** : confirmer que les chemins montés correspondent aux répertoires attendus.
- **Commandes exécutées** : lire et comprendre les commandes lancées dans chaque service.

1 Chargement des données sur HDFS

1. Téléchargez le fichier `web_server.log` depuis Arche.
2. Avant de commencer l'analyse, chargez ce fichier dans HDFS :

Voir les instructions dans le TP sur HDFS.

2 Le Script PySpark

a) Création du fichier

Dans le répertoire /apps, Créer un fichier `analyze_logs.py` et ajouter le squelette suivant :

```
from pyspark import SparkContext

# Initialiser le contexte Spark
sc = SparkContext(appName="AnalyseLogsApacheRDD")

# Charger les logs depuis HDFS
log_file = "hdfs://hdfs-namenode:9000/web_server.log"
logs_rdd = sc.textFile(log_file)

# Afficher les 10 premieres lignes
print("Exemple de lignes du fichier de logs :")
for line in logs_rdd.take(10):
    print(line)
```

Remarque importante : l'URL HDFS utilisée dans le script dépend du répertoire dans lequel vous avez placé le fichier dans HDFS. Dans l'exemple ci-dessous, le fichier a été déposé directement à la racine :

```
log_file = "hdfs://hdfs-namenode:9000/web_server.log"
```

Si vous avez choisi de créer un répertoire spécifique (par exemple `/logs`), l'URL devra être adaptée en conséquence :

```
log_file = "hdfs://hdfs-namenode:9000/logs/web_server.log"
```

Il est donc indispensable de vérifier l'emplacement exact du fichier dans HDFS (en utilisant par exemple `hdfs dfs -ls /` dans le conteneur namenode).

b) Exécution du script

Entrer dans le conteneur Spark Master :

```
docker exec -it spark-master bash
```

Soumettre le script :

```
./bin/spark-submit /opt/spark-apps/analyze_logs.py
```

3 Ajout du parsing des logs

Ajouter au script la regex et la fonction de parsing :

```
import re

# Définir le pattern regex pour parser les logs
log_pattern = r'(\d+\.\d+\.\d+\.\d+) - - \[(.*?)\] "(.*?) (.*?) HTTP.*" (\d+) (\d+)'

# Fonction pour parser une ligne de log
def parse_log_line(line):
    match = re.match(log_pattern, line)
    if match:
        return (
            match.group(1),   # IP
```

```
            match.group(2),   # Timestamp
            match.group(3),   # HTTP Method
            match.group(4),   # URL
            int(match.group(5)), # HTTP Status
            int(match.group(6))  # Response Size
        )
    else:
        return None

# Parser les logs
parsed_logs_rdd = logs_rdd.map(parse_log_line).filter(lambda x: x is not None)

# Afficher 10 exemples
print("Exemple de logs parsés :")
for log in parsed_logs_rdd.take(10):
    print(log)
```

Re-exécuter le script.

4 Ajout des analyses

a) Nombre total de requêtes

```
total_requests = parsed_logs_rdd.count()
print(f"Nombre total de requêtes : {total_requests}")
```

b) Top 5 des URLs les plus demandées

```
top_urls = parsed_logs_rdd \
    .map(lambda x: (x[3], 1)) \
    .reduceByKey(lambda a, b: a + b) \
    .takeOrdered(5, key=lambda x: -x[1])

print("Les 5 URLs les plus demandées :")
for url, count in top_urls:
    print(f"{url}: {count}")
```

5 Aller plus loin

- Ajouter des instructions pour écrire les résultats sur HDFS au format texte.