Rapport TP Data EPSI

Papa Samba Thiam

Cylia Haddouche

Introduction

Ce rapport décrit le pipeline de traitement de données pour simuler, traiter, et analyser des transactions financières. Ce pipeline intègre plusieurs technologies clés, notamment Apache Kafka pour le streaming de données, MinIO pour le stockage de données, et Apache Spark pour le traitement en temps réel.

Description des Scripts

1. capteur.py

Objectifs :

- Simuler la génération de transactions financières variées.
- Publier ces transactions sous forme de messages JSON dans un sujet Kafka spécifique pour leur traitement ultérieur.

Méthodologie :

- Utilisation de la bibliothèque `uuid` pour générer des identifiants uniques.
- Les types de transactions et les méthodes de paiement sont sélectionnés aléatoirement.
- Les montants des transactions sont générés dans une plage de 10 à 1000 USD.
- Les messages sont envoyés au topic Kafka 'transaction' où ils seront consommés par un autre service pour traitement.

2. consumer.py

Objectifs :

- Consommer les messages du sujet Kafka 'transaction'.

- Convertir le montant des transactions de USD à EUR.
- Ajuster les dates des transactions au fuseau horaire de Paris.

Méthodologie :

- Utilisation de `pytz` et `datetime` pour le traitement des zones horaires.
- Filtrage des transactions pour s'assurer que seules les données valides sont traitées (pas d'erreurs, adresse existante).
- Affichage des résultats du traitement pour vérification et analyse.

3. main.py

Objectifs :

- Configurer et se connecter à une instance MinIO pour le stockage des données.
- Stocker les données transformées sous format Parquet dans MinIO.

Méthodologie :

- Création d'un 'bucket' dans MinIO s'il n'existe pas déjà.
- Utilisation de `pandas` pour créer un DataFrame représentant les données.
- Conversion du DataFrame en format Parquet et stockage dans MinIO via `pyarrow`.

4. readparquet.py

Objectifs :

- Lire et afficher les données depuis des fichiers Parquet pour vérification et analyse.

Méthodologie :

- Lecture des fichiers Parquet stockés potentiellement sur MinIO en utilisant Spark.
- Affichage des données pour permettre un contrôle de qualité et une inspection rapide.

5. readstream.py

Objectifs :

- Lire les données en streaming de Kafka, les traiter et les stocker en continu dans des fichiers Parquet dans MinIO.

Méthodologie :

- Configuration de Spark pour lire les données en streaming du sujet Kafka 'transaction'.
- Traitement des données en temps réel et stockage en format Parquet pour une analyse ultérieure optimisée.

Conclusion

Les scripts développés forment un pipeline complet qui simule des transactions, les traite via streaming, et les stocke pour analyse ultérieure.