Documentation Technique : Choix des Technologies et Solutions Cloud

1. Introduction

Ce document présente les choix technologiques et les solutions d'infrastructure que nous avons utilisés pour la mise en place d'un Data Lake pour une entreprise de commerce en ligne. L'objectif est de centraliser et de traiter les données provenant de différentes sources (transactions clients, logs des serveurs web, données des médias sociaux, et flux en temps réel des campagnes publicitaires). Cette infrastructure permet de collecter, transformer, stocker et analyser ces données afin de faciliter la prise de décisions commerciales éclairées.

2. Choix de l'Infrastructure de Stockage

File System Local :

Dans le cadre de ce projet, nous avons choisi d'utiliser un File System local comme solution de stockage pour les données brutes et transformées. Contrairement à une solution cloud comme Amazon S3 ou HDFS, cette approche utilise simplement un système de fichiers local sur la machine exécutant le projet.

Raisons du choix :

- Simplicité : Un système local est plus simple à configurer et à maintenir, surtout pour un projet de petite à moyenne envergure. Il n’entraîne pas de configurations complexes ni de coûts additionnels liés aux services cloud.

- Pas de besoin d’évolutivité massive : Le volume de données traitées dans ce projet est modéré et ne nécessite pas l'évolutivité d'un système distribué comme HDFS ou S3.

- Contrôle total sur les données : Le stockage local offre un contrôle direct sur les fichiers et leur gestion sans dépendre des services externes.

Solution de stockage utilisée :

- Format des fichiers : Les données brutes sont stockées en format JSON, et les données transformées en Parquet, un format optimisé pour l'analyse.

Structure des dossiers :

- Raw Data : `data_lake/raw_data/` contient les fichiers de données brutes provenant des différentes sources.
- Processed Data : `data_lake/processed_data/` contient les données transformées et nettoyées prêtes pour l'analyse.

3. Choix du Traitement des Données

Apache Spark

Le traitement des données est effectué à l'aide de Apache Spark, un framework de traitement distribué. Nous avons utilisé Spark en mode local (standalone), sans déployer de cluster distribué, mais il conserve sa capacité à traiter de grandes quantités de données de manière efficace.

Raisons du choix :

- Performance : Spark est extrêmement performant pour le traitement de grandes quantités de données en mémoire, ce qui le rend idéal pour ce projet où l'analyse des transactions et des logs doit être rapide.

- Compatibilité avec divers formats : Spark prend en charge plusieurs formats de données (JSON, CSV, Parquet), ce qui est nécessaire pour traiter les différentes sources de données (fichiers JSON des médias sociaux, logs, etc.).

- Facilité d'utilisation : Spark permet un traitement en batch et en streaming. Pour ce projet, nous avons utilisé Spark en mode batch pour le traitement des données brutes (transactions, logs, etc.).

Utilisation de Spark :

- Lecture des données : Les données sont lues directement depuis les fichiers JSON/ dans le dossier `raw_data`.

- Transformation des données : Des étapes de nettoyage et de filtrage sont effectuées avant de stocker les résultats dans le dossier `processed_data`.

- Format de sortie : Les résultats sont sauvegardés en Parquet, un format de fichier colonne très efficace pour les opérations analytiques.

4. Choix des Solutions Cloud

Choix de ne pas utiliser une solution Cloud :

Bien que des solutions Cloud comme AWS, Google Cloud ou Azure offrent des avantages en termes de stockage, de scalabilité et de gestion, nous avons choisi de ne pas utiliser de solution Cloud pour ce projet.

Raisons du choix :

- Coût et Complexité : Le coût associé à l'utilisation de services Cloud (notamment pour le stockage S3 ou le déploiement de clusters Hadoop sur un service Cloud) peut être élevé, surtout lorsque le projet n'a pas besoin d'une infrastructure évolutive.

- Simplicité d'implémentation : Le projet étant réalisé sur une machine locale, il nous a semblé plus simple d'utiliser un système de fichiers local pour la gestion des données sans la nécessité d'une solution Cloud complexe.

- Pas de besoins d'évolutivité massive : Les volumes de données traités dans ce projet sont relativement modestes et ne nécessitent pas une architecture Cloud hautement scalable comme celle que l'on pourrait déployer sur AWS ou Google Cloud.

Choix d'une infrastructure locale

Le choix de travailler sur une machine locale nous permet d'avoir un contrôle total sur l'infrastructure, d'éviter des coûts récurrents pour des services Cloud et de faciliter la gestion des données et des traitements.

5. Processus de Déploiement

Le déploiement de l'infrastructure se fait en deux étapes principales :

1. Préparation de l'environnement local :

- Création d'un environnement virtuel Python (venv).
- Installation des dépendances nécessaires avec `pip install pyspark`.
- Mise en place des dossiers pour stocker les données brutes et traitées.

2. Traitement des données avec Apache Spark :

- Chargement des données depuis le système local.
- Application des transformations nécessaires (nettoyage, filtrage).
- Sauvegarde des résultats sous le format Parquet dans le dossier `processed_data`.

Script d'automatisation

Nous avons développé un script Python pour automatiser le traitement des données. Ce script réalise les étapes suivantes :

- Lecture des fichiers JSON bruts depuis le dossier `raw_data`.
- Nettoyage et transformation des données.
- Sauvegarde des résultats sous le format Parquet dans le dossier `processed_data`.

6. Conclusion

L'infrastructure choisie pour ce projet repose sur un système de fichiers local pour le stockage des données et sur Apache Spark pour le traitement des données