Documentation des Transformations Appliquées aux Données

1. Ingestion des Données Brutes

Le premier script consiste à ingérer les données provenant de différentes sources, à savoir :

- **Transactions** : Fichiers CSV contenant les informations des transactions.
- **Web Logs** : Fichiers texte contenant les logs des actions sur un site web.
- **Social Media** : Fichiers JSON contenant les informations des posts sur les réseaux sociaux.

2. Transformations Appliquées

Après l'ingestion des données, plusieurs transformations sont appliquées afin de nettoyer, structurer et enrichir les données avant de les stocker dans le Data Lake.

a) Transactions

Les données des transactions sont filtrées pour ne garder que celles dont l'**amount** (montant de la transaction) est supérieur à zéro. Cette transformation permet de se débarrasser des transactions invalides ou erronées.

```
transactions_cleaned = transactions.filter(transactions["amount"] > 0)
```

Objectif : Garder uniquement les transactions valides avec un montant positif.

b) Web Logs

Les logs bruts provenant des fichiers texte sont transformés en une structure plus lisible. Chaque ligne des logs est renommée sous la colonne `log_entry`. Ce changement permet de mieux structurer les logs pour un traitement ultérieur.

web_logs_parsed = web_logs.withColumnRenamed("value", "log_entry")

Objectif : Convertir les logs textuels bruts en une structure de données avec une colonne `log_entry`, facilitant ainsi l'analyse.

c) Social Media

Les données des réseaux sociaux sont filtrées pour ne garder que celles où l'**email** est présent. Cette transformation est nécessaire pour s'assurer que seules les entrées complètes sont retenues pour l'analyse, en éliminant les posts où l'email est manquant.

social_media_filtered = social_media.filter(social_media["email"].isNotNull())

Objectif : Filtrer les posts où l'email de l'utilisateur est manquant, garantissant la qualité des données pour l'analyse.

3. Stockage des Données Transformées

Après ces transformations, les données nettoyées et structurées sont ensuite sauvegardées dans les répertoires appropriés du Data Lake :

- **Transactions** : Sauvegardées au format Parquet pour permettre des lectures efficaces et performantes.
- **Web Logs** : Sauvegardés au format texte pour conserver leur structure initiale.
- **Social Media** : Sauvegardées au format JSON pour conserver la hiérarchie et la flexibilité des données.

Sauvegarde :

transactions_cleaned.write.mode("overwrite").parquet("./processed_data/transactions/")
web_logs_parsed.write.mode("overwrite").text("./processed_data/web_logs/")
social_media_filtered.write.mode("overwrite").json("./processed_data/social_media/")

4. Résumé des Transformations

- **Filtrage** : Les données sont filtrées pour supprimer les enregistrements invalides ou incomplets (ex. : transactions avec montant négatif, posts sans email).
- **Renommage** : Les colonnes sont renommées pour améliorer la lisibilité des données (ex. : `value` devient `log_entry` pour les logs).
- **Stockage** : Les données transformées sont sauvegardées dans différents formats adaptés aux types de données et aux besoins d'analyse (Parquet, JSON, Texte).

5. Bénéfices des Transformations

- **Qualité des Données** : Les transformations permettent de garantir la qualité des données en éliminant les erreurs, les doublons et les informations manquantes.
- **Efficacité** : En transformant et nettoyant les données avant de les stocker, nous facilitons l'analyse et le traitement ultérieur, en utilisant des formats optimisés comme Parquet pour les données structurées.
- **Flexibilité** : La transformation des logs et des données JSON permet de conserver leur hiérarchie tout en facilitant leur traitement avec des outils comme Apache Spark.