DATA PROJECT

Pipeline ETL pour l'analyse de la traçabilité des chaînes d'approvisionnement

Réalisé par : Zineb HAOUDI

Encadré par : Mr. Vital GUINGUINNI

Sommaire

Introduction

Dans un contexte où la traçabilité et la transparence des chaînes d'approvisionnement constituent des enjeux stratégiques, les entreprises et organisations doivent pouvoir s'appuyer sur des données fiables pour suivre les flux de matières premières, contrôler la conformité des transactions et anticiper les risques liés à la durabilité. Or, les données disponibles proviennent souvent de sources hétérogènes et se présentent sous des formats complexes, comme les fichiers JSON hiérarchiques, difficiles à exploiter directement.

La problématique réside ainsi dans la difficulté à transformer ces données brutes, bien que riches en informations (acheteurs, fabricants, matières premières, certifications, taux de traçabilité…), en données utilisables pour l'analyse. Leur structure imbriquée, la redondance de certains champs et la présence d'informations sensibles constituent autant d'obstacles à une exploitation directe et fiable.

Pour répondre à ce défi, le travail entrepris a consisté à concevoir et mettre en œuvre un pipeline ETL (Extract, Transform, Load). Celui-ci a permis d'extraire les données JSON, de les transformer en un format tabulaire, de les anonymiser afin de préserver la confidentialité, puis de les charger dans une base relationnelle PostgreSQL structurée selon une modélisation en étoile. Enfin, les données ont été exploitées dans Power BI pour produire des tableaux de bord interactifs, offrant une vision claire et dynamique des volumes, des acteurs et des niveaux de traçabilité.

Les objectifs principaux de ce projet sont :

- Centraliser et fiabiliser les données issues du JSON pour obtenir une base homogène,
- Assurer la confidentialité des informations grâce à l'anonymisation,
- Optimiser l'analyse décisionnelle grâce à une modélisation en étoile,
- Mettre à disposition des visualisations interactives et pertinentes via Power BI,
- Contribuer ainsi à une meilleure compréhension et un meilleur suivi de la traçabilité dans la chaîne d'approvisionnement.

1 Source de données et préparation

1.1 Fichier source JSON

La source initiale est un fichier JSON exporté depuis une plateforme de traçabilité des achats, contenant la liste des purchase orders (commandes) avec une structure hiérarchique (nœuds imbriqués). On y trouve des informations sur les acheteurs, fabricants, traders, points de contact, matières premières (ex. PO, PKO, CNO), produits finis, ainsi que des attributs de traçabilité (moulins, plantations).

1.2 Conversion JSON → CSV tabulaire

La première étape du traitement a consisté à convertir le fichier source, fourni au format JSON, en un format tabulaire exploitable pour les analyses. En effet, le fichier initial présentait une structure hiérarchique complexe, composée de nœuds imbriqués et de listes, rendant son exploitation directe difficile.

Un script Python a été développé pour parcourir récursivement l'ensemble de la structure et identifier toutes les clés présentes, en respectant leur ordre d'apparition. Une fonction générique a ensuite permis d'extraire les valeurs correspondantes, qu'il s'agisse de données simples (texte, nombre, booléen) ou de collections sous forme de listes et d'objets. Les enregistrements ont ainsi été reconstitués ligne par ligne afin d'alimenter un DataFrame pandas représentant une version aplatie du fichier.

Afin d'assurer une véritable normalisation tabulaire, les colonnes contenant des listes ont été automatiquement éclatées, ce qui a permis d'obtenir une ligne distincte pour chaque valeur. Le jeu de données final a ensuite été exporté au format CSV, encodé en UTF-8 et séparé par des points-virgules.

1.3 Anonymisation des données sensibles

Après la conversion du JSON en format tabulaire, une étape essentielle a consisté à protéger les données sensibles avant toute exploitation ultérieure. En effet, le jeu de données contenait des informations nominatives ou commerciales (noms d'entreprises, points de contact, fabricants, traders, identifiants techniques tels que les rowId, GTIN ou

HS codes) qui ne pouvaient pas être utilisées directement pour des raisons de confidentialité.

Pour cela, un script Python basé sur la bibliothèque **Faker** a été mis en place. Son fonctionnement repose sur deux approches complémentaires :

- **Anonymisation des noms** : les colonnes contenant des informations relatives aux entreprises, aux contacts ou aux entités commerciales sont remplacées par des valeurs fictives mais réalistes. Par exemple, les acheteurs et traders sont remplacés par des noms d'entreprises générés, tandis que les points de contact sont substitués par des noms de personnes.
- **Anonymisation des identifiants** : les identifiants techniques (tels que rowId, GTIN ou HS code) sont remplacés par des codes factices construits selon une logique séquentielle (PO_000001, FAKE_GTIN_000001, etc.), garantissant à la fois l'anonymisation et la possibilité de retracer les correspondances.

Afin de conserver la cohérence et la traçabilité technique du jeu de données, deux fichiers de correspondance sont générés automatiquement :

- **Fichier de mapping des noms**, qui relie chaque valeur originale à sa version anonymisée,
- **Fichier de mapping des identifiants**, qui documente les remplacements opérés pour les identifiants transactionnels et produits.

Le traitement est réalisé par lots (chunks) pour permettre de gérer efficacement de gros volumes de données, et le résultat final est enregistré dans un fichier CSV anonymisé, utilisable sans risque dans les étapes suivantes du pipeline.

1.4 Création d'un échantillon de données

Étant donné la taille importante du fichier de données anonymisé, il a été nécessaire de générer un échantillon réduit afin de faciliter les tests et d'accélérer les différentes étapes

de développement du pipeline. Pour cela, un script Python a été conçu afin de sélectionner uniquement un nombre défini de lignes du fichier initial.

Le principe est simple : à partir du fichier CSV anonymisé, le script lit uniquement les **N** premières lignes (par exemple 100 000), puis les enregistre dans un nouveau fichier CSV dédié aux tests. Ce sous-ensemble conserve la même structure et les mêmes colonnes que la base complète, ce qui garantit sa compatibilité avec l'ensemble du processus ETL, tout en réduisant considérablement le temps de traitement.

L'échantillon ainsi créé constitue donc une base de travail pratique pour valider les transformations, les chargements dans PostgreSQL et les visualisations dans Power BI, avant d'appliquer le pipeline complet à l'intégralité des données.

2 Création de la base de données et table de staging

Après la génération de l'échantillon, les données ont été intégrées dans une base PostgreSQL afin de préparer la modélisation. Le script charge le fichier CSV et convertit les colonnes numériques (volumes, quantités, taux de traçabilité) pour garantir la cohérence des types.

À l'aide de **SQLAlchemy**, une connexion administrateur est utilisée pour recréer automatiquement la base *Projet*. Les connexions actives sont d'abord fermées, puis la base est supprimée et recréée en mode **AUTOCOMMIT**, ce qui assure un environnement propre à chaque exécution.

Enfin, une table de staging (*staging_orders*) est alimentée avec le contenu du CSV. Cette zone tampon centralise les données sources avant leur transformation en tables analytiques (dimensions et faits).

3 Modélisation en étoile (star schema)

3.1 Tables de dimensions

Une fois la table de staging alimentée, la base a été structurée selon une **modélisation en étoile** afin de préparer l'analyse décisionnelle. Pour ce faire, plusieurs tables de dimensions et une table centrale des faits ont été créées dans PostgreSQL via un script SQL exécuté par SQLAlchemy.

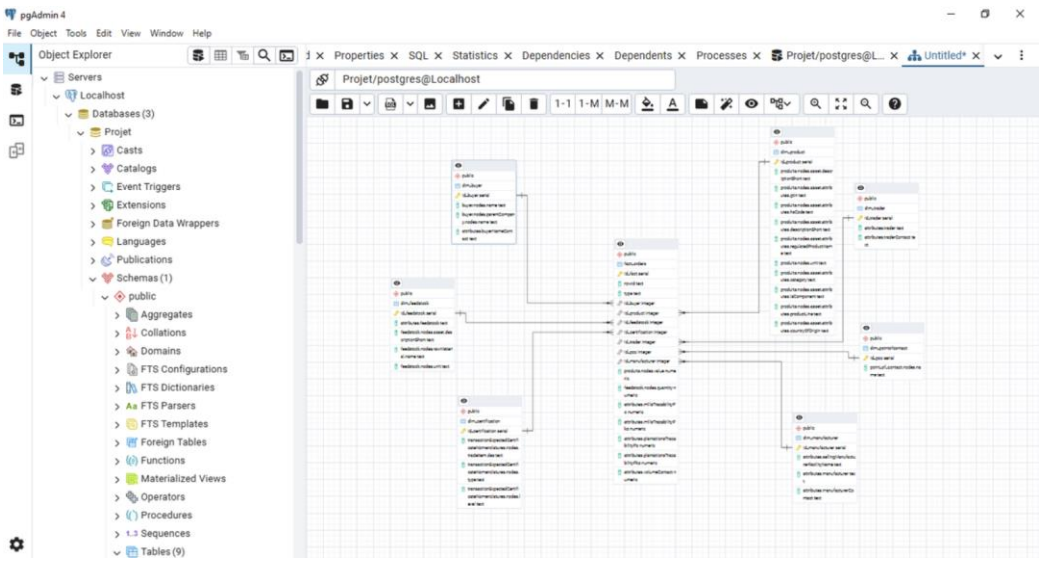
Les **tables de dimensions** stockent les informations descriptives associées aux différentes entités :

- *Dim_Buyer* pour les acheteurs et sociétés mères,
- *Dim_Product* pour les produits et leurs attributs (GTIN, HS code, unité, catégorie, pays d'origine…),
- *Dim_Feedstock* pour les matières premières et leurs unités,
- *Dim_Certification* pour les types et niveaux de certification,
- *Dim_Trader* pour les traders et leurs contacts,
- *Dim_PointOfContact* pour les points de contact,
- *Dim_Manufacturer* pour les fabricants et sites de production.

3.2 Table des faits (Fact_Orders)

La **table des faits** (*Fact_Orders*) constitue la table centrale du modèle. Elle contient les données quantitatives liées aux transactions (valeur des produits, quantités de matières premières, taux de traçabilité, volumes de contact) ainsi que des clés étrangères reliant chaque enregistrement aux dimensions correspondantes (acheteur, produit, feedstock, certification, etc.).

Cette structuration en étoile permet d'optimiser les jointures et de faciliter l'exploration des données sous différents axes d'analyse, tout en assurant la cohérence du modèle.



4 Alimentation des dimensions et de la table des faits

Après la création des différentes tables, l'étape suivante a consisté à alimenter les dimensions et la table des faits à partir de la table de staging.

Les dimensions ont été remplies à l'aide de requêtes `INSERT INTO … SELECT DISTINCT`, afin de ne conserver que les valeurs uniques pour chaque entité (acheteur, produit, feedstock, certification, trader, point de contact et fabricant).

La table des faits (*Fact_Orders*) a ensuite été complétée grâce à des `LEFT JOIN` avec les dimensions, ce qui permet d'associer chaque transaction à ses entités correspondantes tout en intégrant les mesures quantitatives (valeurs, quantités, taux de traçabilité, volumes).

Ce processus a permis de mettre en place un **schéma en étoile** complet, adapté à l'analyse décisionnelle et à la visualisation dans Power BI.

5 Intégration sur GitHub et orchestration CI/CD

L'ensemble des éléments du projet a été centralisé sur **GitHub** sur ce lien : https://github.com/Zineb-HAOUDI/Pipeline-ETL-APIGraphQL-PostgreSQL-PowerBi ,

incluant le code Python, le notebook Jupyter, le fichier des dépendances (requirements.txt) et la documentation (README.md).
Les fichiers sensibles, tels que le .env contenant le mot de passe PostgreSQL, ainsi que les fichiers volumineux (CSV, JSON), ont été exclus grâce à l'utilisation d'un fichier .gitignore, ce qui permet de maintenir un dépôt clair et sécurisé, sans exposition de données confidentielles.

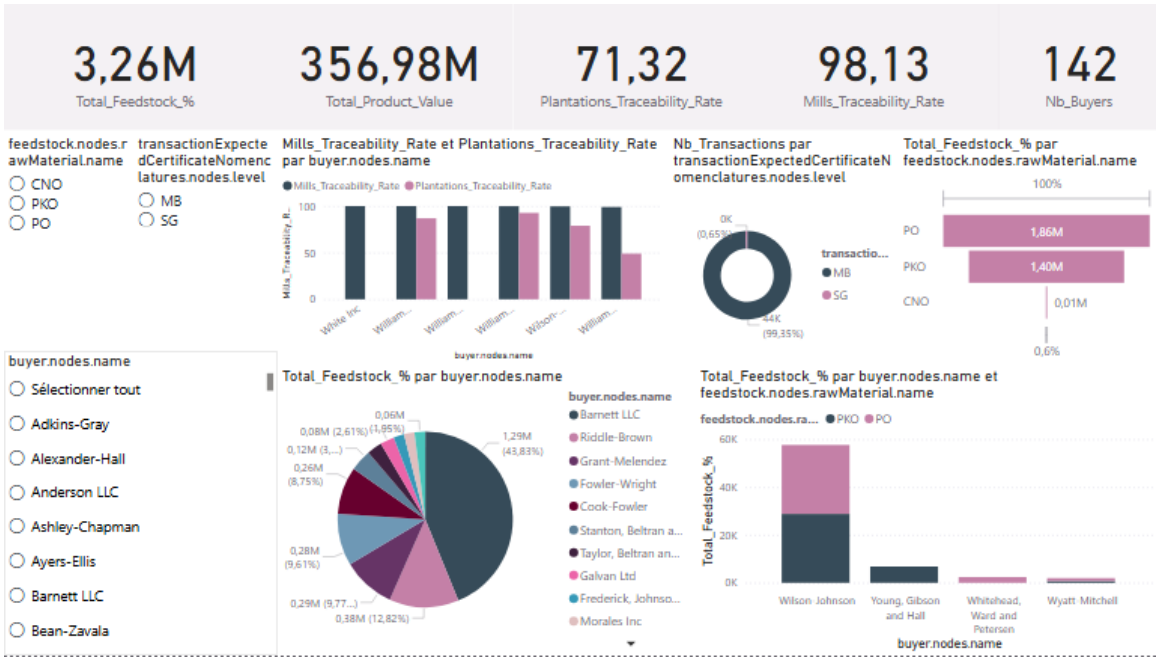
GitHub a également été utilisé pour assurer la traçabilité du code grâce au système de versionnage (git add, git commit, git push) et pour offrir des possibilités de collaboration par l'ajout de contributeurs disposant de droits spécifiques.

En complément, une orchestration **CI/CD** a été mise en place via GitHub Actions. Un fichier etl.yml a été créé dans le dossier .github/workflows, définissant un pipeline automatisé. À chaque push sur la branche principale, GitHub déclenche ce workflow qui : installe Python, récupère les dépendances listées dans requirements.txt, puis exécute automatiquement le script ETL.

Cette automatisation garantit la **reproductibilité des traitements**, réduit les erreurs manuelles et renforce la traçabilité des mises à jour du projet.

6 Visualisation et analyse sous Power BI

L'intégration dans Power BI s'est faite par une connexion directe à la base PostgreSQL, permettant d'importer l'ensemble des tables issues du modèle en étoile. Le tableau de bord a été conçu sur une page unique, organisée de manière claire : un bandeau de KPI en haut pour mettre en avant les indicateurs clés, des graphiques de comparaison et de répartition au centre, et des filtres interactifs (slicers) placés à gauche pour affiner l'analyse par acheteur, produit, feedstock ou certification. Les visuels utilisés incluent des cartes KPI, des graphiques en barres, camemberts et donuts, un TreeMap, un entonnoir pour représenter des séquences, ainsi que des tableaux récapitulatifs. Le tout a été construit en suivant les bonnes pratiques de datavisualisation, avec une palette de couleurs cohérente, des titres courts et explicites, et un alignement harmonieux des éléments afin de garantir une lecture fluide et efficace.


3,26M
Total_Feedstock_%
356,98M
Total_Product_Value
71,32
Plantations_Traceability_Rate
98,13
Mills_Traceability_Rate
142
Nb_Buyers
feedstock.nodes.rawMaterial.name
CNO
PKO
PO
transactionExpectedCertificateNomenclatures.nodes.level
MB
SG
Mills_Traceability_Rate et Plantations_Traceability_Rate par buyer.nodes.name
Mills_Traceability_Rate
Plantations_Traceability_Rate
100
50
0
White Inc
William...
William...
William...
Wilson-...
William...
buyer.nodes.name
Nb_Transactions par transactionExpectedCertificateNomenclatures.nodes.level
0K
(0,65%)
44K
(99,35%)
transactio...
MB
SG
Total_Feedstock_% par feedstock.nodes.rawMaterial.name
100%
PO
1,86M
PKO
1,40M
CNO
0,01M
0,6%
buyer.nodes.name
Sélectionner tout
Adkins-Gray
Alexander-Hall
Anderson LLC
Ashley-Chapman
Ayers-Ellis
Barnett LLC
Bean-Zavala
Total_Feedstock_% par buyer.nodes.name
0,06M
(1,95%)
0,08M (2,61%)
0,12M (3,...)
0,26M
(8,75%)
0,28M
(9,61%)
0,29M (9,77...)
0,38M (12,82%)
1,29M
(43,83%)
buyer.nodes.name
Barnett LLC
Riddle-Brown
Grant-Melendez
Fowler-Wright
Cook-Fowler
Stanton, Beltran a...
Taylor, Beltran an...
Galvan Ltd
Frederick, Johnso...
Morales Inc
Total_Feedstock_% par buyer.nodes.name et feedstock.nodes.rawMaterial.name
feedstock.nodes.ra...
PKO
PO
60K
40K
20K
0K
Total_Feedstock_%
Wilson Johnson
Young, Gibson and Hall
Whitehead, Ward and Petersen
Wyatt Mitchell
buyer.nodes.name

Conclusion

Le pipeline ETL conçu permet de transformer efficacement une source JSON complexe en un modèle relationnel clair et optimisé, avant de le rendre directement exploitable dans des tableaux de bord interactifs sous Power BI. Ce processus assure non seulement une structuration fiable des données (via l'anonymisation et la modélisation en étoile), mais aussi une valorisation opérationnelle grâce à des visualisations adaptées aux besoins des parties prenantes.

Au-delà de l'analyse des volumes, de la traçabilité et des répartitions, cette approche crée une véritable chaîne de valeur des données, allant de la collecte brute jusqu'à la restitution visuelle et décisionnelle. Elle offre ainsi un socle robuste pour évaluer les performances, identifier des tendances et anticiper des risques.

Les prochaines étapes visent à renforcer encore cette architecture en :

- Ajoutant une dimension temporelle pour suivre l'évolution des indicateurs dans le temps,
- et améliorant les processus de qualité et de gouvernance des données, afin d'assurer leur cohérence, leur fiabilité et leur conformité aux standards.

Ce projet constitue ainsi une première brique vers une solution évolutive et pérenne, capable de s'adapter aux besoins croissants en matière de traçabilité, de durabilité et de pilotage stratégique des chaînes d'approvisionnement.