Université Abdelmalek Essaadi
Faculté ses Sciences et techniques de Tanger
Département Génie Informatique

Big Data

Pipeline de projet Data avec Apache Airflow

Mini Projet

Auteurs:

ESSALHI Sara
KAISSI Houda

Encadré par: Pr. Yasyn EL YUSUFI

Contents

Objectif du Projet

Le projet consiste à développer une pipeline ETL (Extraction, Transformation et Chargement) pour l'ingestion, la transformation et le chargement de données provenant de multiples sources (API, bases de données, fichiers). Le pipeline alimente un entrepôt de données (Datawarehouse) permettant d'organiser les données pour faciliter l'analyse. Après la structuration des données dans l'entrepôt, un tableau de bord sera conçu pour visualiser les rapports, graphiques et indicateurs résultants de l'analyse.

Architecture de la Pipeline

2.1 Rôle d'Apache Airflow dans la Pipeline ETL

Apache Airflow joue un rôle central dans l'orchestration des différentes étapes de la pipeline ETL (Extract, Transform, Load). Il permet de :

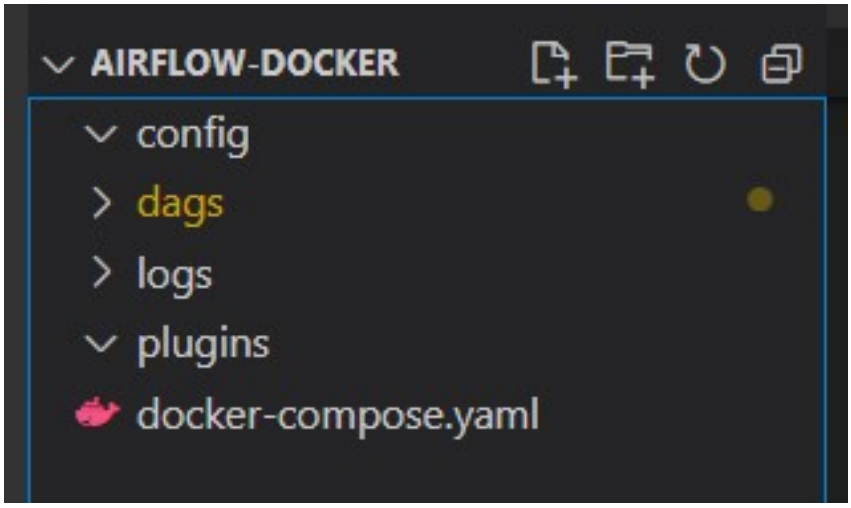
- **Orchestrer les tâches :** Airflow gère l'exécution de chaque tâche dans un ordre spécifique, en fonction des dépendances définies entre elles.
- **Automatiser l'exécution :** Le pipeline peut être exécuté automatiquement à intervalles réguliers, ce qui réduit la nécessité d'interventions manuelles et permet une exécution continue.
- **Gérer les erreurs et les alertes :** En cas d'échec d'une tâche, Airflow envoie des alertes et peut être configuré pour réessayer les tâches échouées.
- **Visualiser les pipelines :** Grâce à son interface web, Airflow permet de visualiser le graphique des tâches, de suivre leur exécution en temps réel, et d'inspecter les logs pour diagnostiquer les problèmes.

2.2 Structure d'un Projet Airflow

Lors de la création d'un projet avec Apache Airflow, une organisation de fichiers bien définie est nécessaire pour garantir la maintenance, l'évolutivité et la lisibilité du projet. Voici la structure typique d'un projet Airflow et le rôle de chaque fichier :

2.2.1 Explication des fichiers et répertoires principaux

- **dags :** Ce répertoire contient les fichiers définissant les workflows Airflow. Chaque fichier DAG (Directed Acyclic Graph) représente un pipeline ou un processus,



comme le fichier **etl-pipeline-dag.py** qui contient la définition de la pipeline ETL (c'est ce fichier qui décrit les différentes étapes d'extraction, de transformation et de chargement, avec leurs dépendances respectives).

- **etl-pipeline-dag.py :** Ce fichier est le cœur de votre pipeline ETL. Il définit :

  Les tâches à exécuter, comme l'extraction des données à partir des différentes sources, la transformation des données, et le chargement dans l'entrepôt de données. Les dépendances entre les tâches, en spécifiant l'ordre d'exécution. Le planning d'exécution de la pipeline, qui peut être définie avec un intervalle **(schedule-interval)** ou une exécution manuelle.

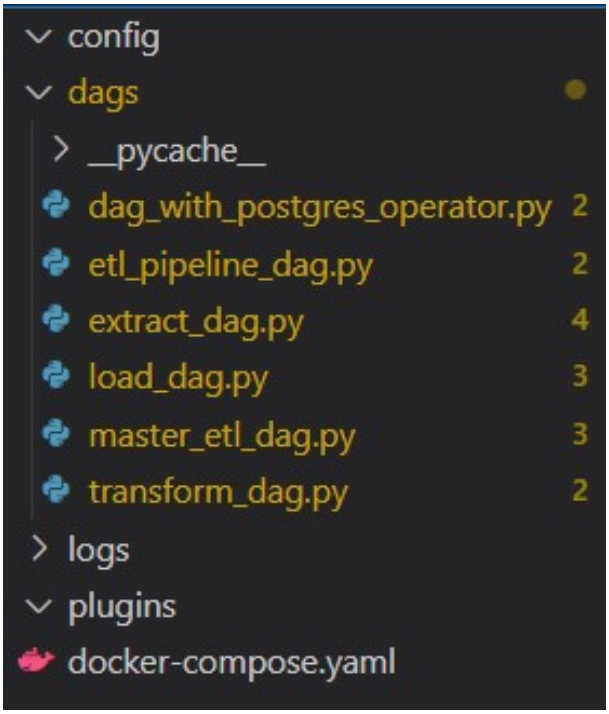
- **plugins :** Ce répertoire contient des extensions personnalisées telles que des opérateurs, des capteurs ou des hooks que vous pouvez créer pour répondre à des besoins spécifiques du projet.

  Par exemple, le fichier **custom-operators.py** pourrait définir un opérateur personnalisé si des tâches nécessitent des actions spécifiques non couvertes par les opérateurs de base d'Airflow.

- **logs :** Ce répertoire est utilisé pour stocker les logs d'exécution des tâches. Airflow génère des logs détaillant les étapes de chaque tâche, ce qui est utile pour le diagnostic en cas de problème.

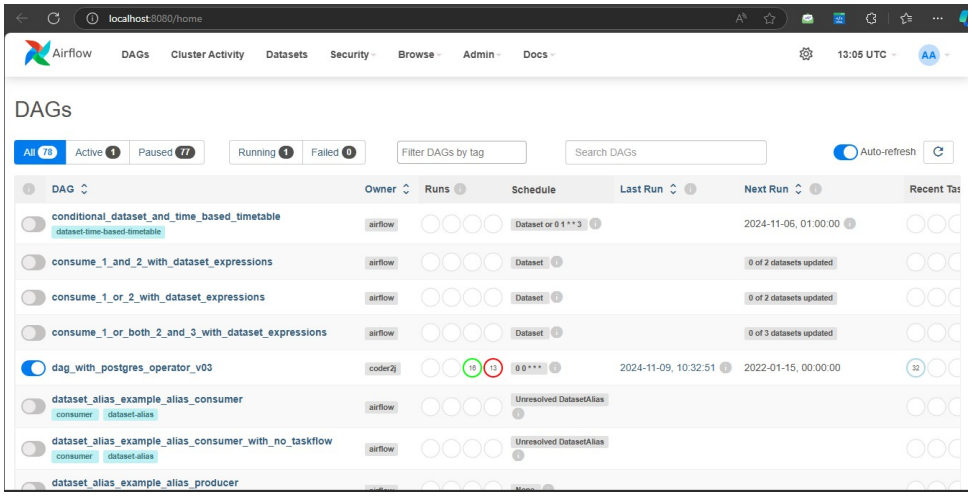
2.3 Détails de la Pipeline ETL dans Airflow

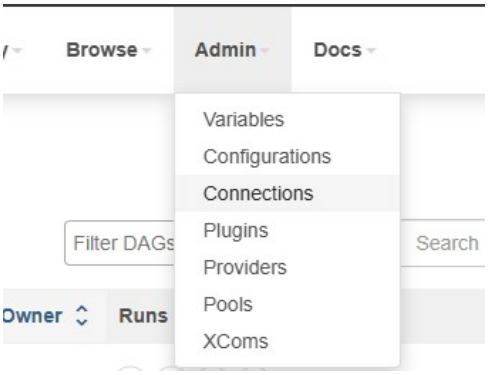
Dans cette approche, le processus ETL est divisé en quatre DAGs distincts, chacun étant responsable d'une partie spécifique du flux de travail. Cela permet une meilleure modularité et une gestion plus facile de chaque phase. Voici un aperçu détaillé de chaque DAG et de son rôle dans la pipeline globale.



2.3.1 Création de la Connexion avec MySQL pour Extraire les Données

Dans ce projet, une des étapes essentielles de la pipeline ETL est l'extraction des données depuis une base de données MySQL. Pour cela, Apache Airflow fournit un mécanisme simple de connexion avec MySQL via son interface de connexion et ses opérateurs. Voici comment vous pouvez configurer la connexion et l'extraction des données dans votre DAG Airflow.




Browse
Admin
Docs
Variables
Configurations
Connections
Plugins
Providers
Pools
XComs
Filter DAGs
Search
Owner
Runs

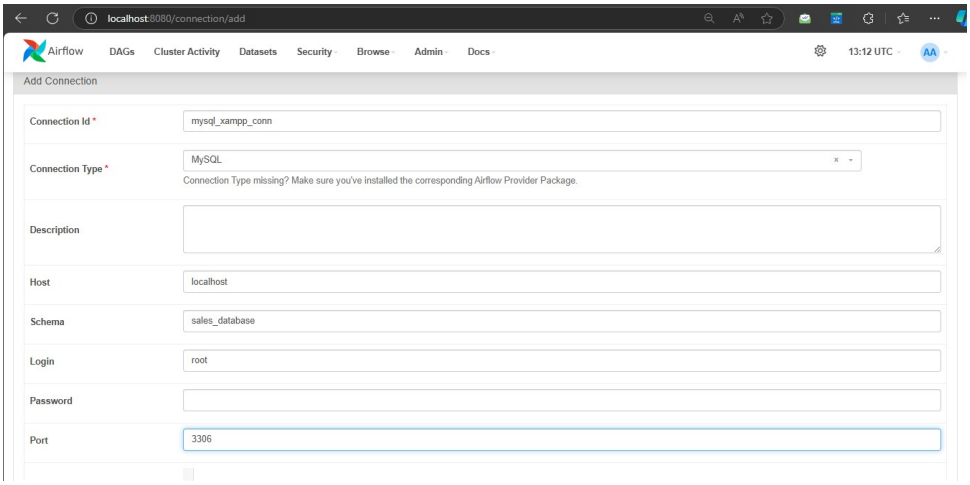

localhost:8080/connection/add
Airflow
DAGs
Cluster Activity
Datasets
Security
Browse
Admin
Docs
13:12 UTC
AA
Add Connection
Connection Id *
mysql_xampp_conn
Connection Type *
MySQL
Connection Type missing? Make sure you've installed the corresponding Airflow Provider Package.
Description
Host
localhost
Schema
sales_database
Login
root
Password
Port
3306

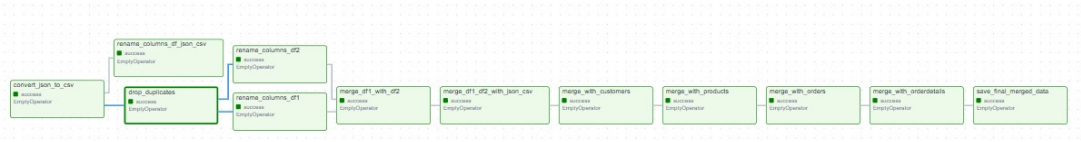
Table	Action	Lignes	Type	Interclassement	Taille	Perte
customers	Parcourir Structure Rechercher Insérer Vider Supprimer	122	InnoDB	utf8mb4_general_ci	64,0 kio	-
employees	Parcourir Structure Rechercher Insérer Vider Supprimer	23	InnoDB	utf8mb4_general_ci	48,0 kio	-
offices	Parcourir Structure Rechercher Insérer Vider Supprimer	7	InnoDB	utf8mb4_general_ci	16,0 kio	-
orderdetails	Parcourir Structure Rechercher Insérer Vider Supprimer	2 996	InnoDB	utf8mb4_general_ci	240,0 kio	-
orders	Parcourir Structure Rechercher Insérer Vider Supprimer	326	InnoDB	utf8mb4_general_ci	96,0 kio	-
payments	Parcourir Structure Rechercher Insérer Vider Supprimer	273	InnoDB	utf8mb4_general_ci	16,0 kio	-
productlines	Parcourir Structure Rechercher Insérer Vider Supprimer	7	InnoDB	utf8mb4_general_ci	16,0 kio	-
products	Parcourir Structure Rechercher Insérer Vider Supprimer	110	InnoDB	utf8mb4_general_ci	80,0 kio	-
8 tables	Somme	3 864	InnoDB	utf8mb4_general_ci	576,0 kio	0 o

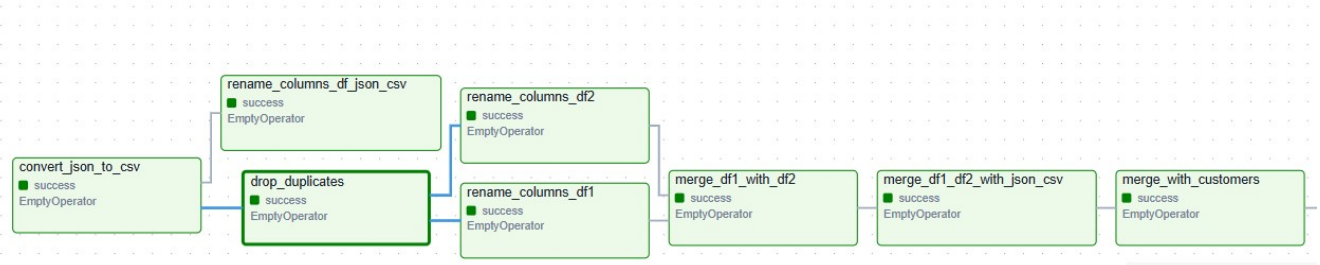
2.3.2 DAG d'Extraction (extract-dag)

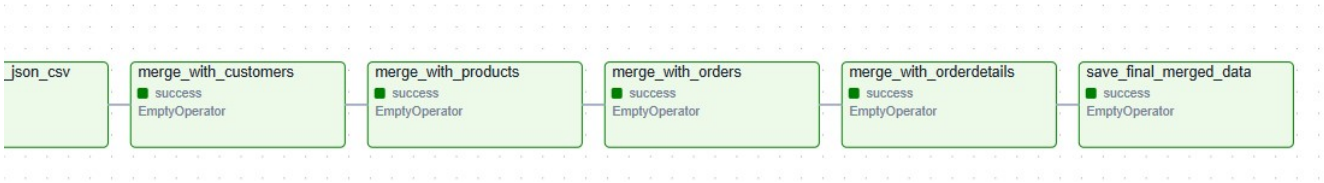
Le premier DAG est dédié à l'extraction des données depuis différentes sources (fichiers CSV, JSON, base de données MySQL, etc.). Ce DAG assure que toutes les données nécessaires sont récupérées et prêtes à être utilisées dans la phase de transformation.

- **Tâches :**
  1-Extraction des fichiers CSV (par exemple, **extract-csv1-task**, **extract-csv2-task**).
  2-Extraction des données JSON (**extract-json-task**).
  3-Extraction des données depuis la base de données MySQL locale (**extract-mysql-task**).

- **Fonctionnement :**
  1-Ce DAG est déclenché de manière indépendante ou à intervalle défini, selon la fréquence de l'extraction des données.
  2-Chaque tâche d'extraction est indépendante et peut échouer ou être réexécutée sans affecter les autres phases du pipeline.





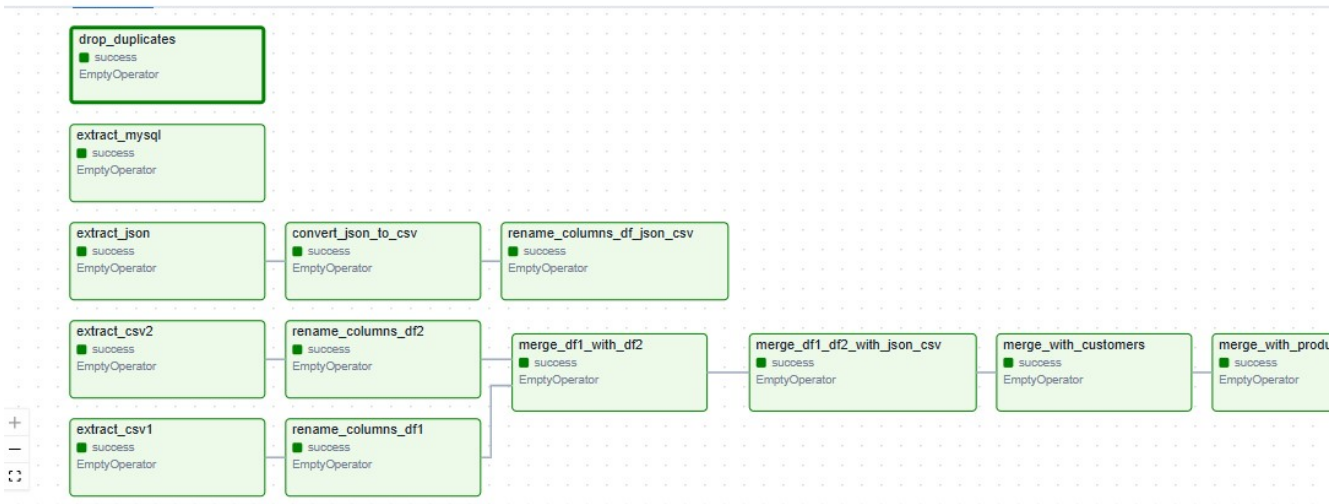


2.3.3 DAG de Transformation (transform-dag)

Une fois les données extraites, ce DAG prend en charge leur transformation. Cette phase comprend des étapes telles que la conversion de formats (par exemple, de JSON en CSV), le nettoyage des données (suppression des doublons) et l'harmonisation des colonnes avant leur fusion.

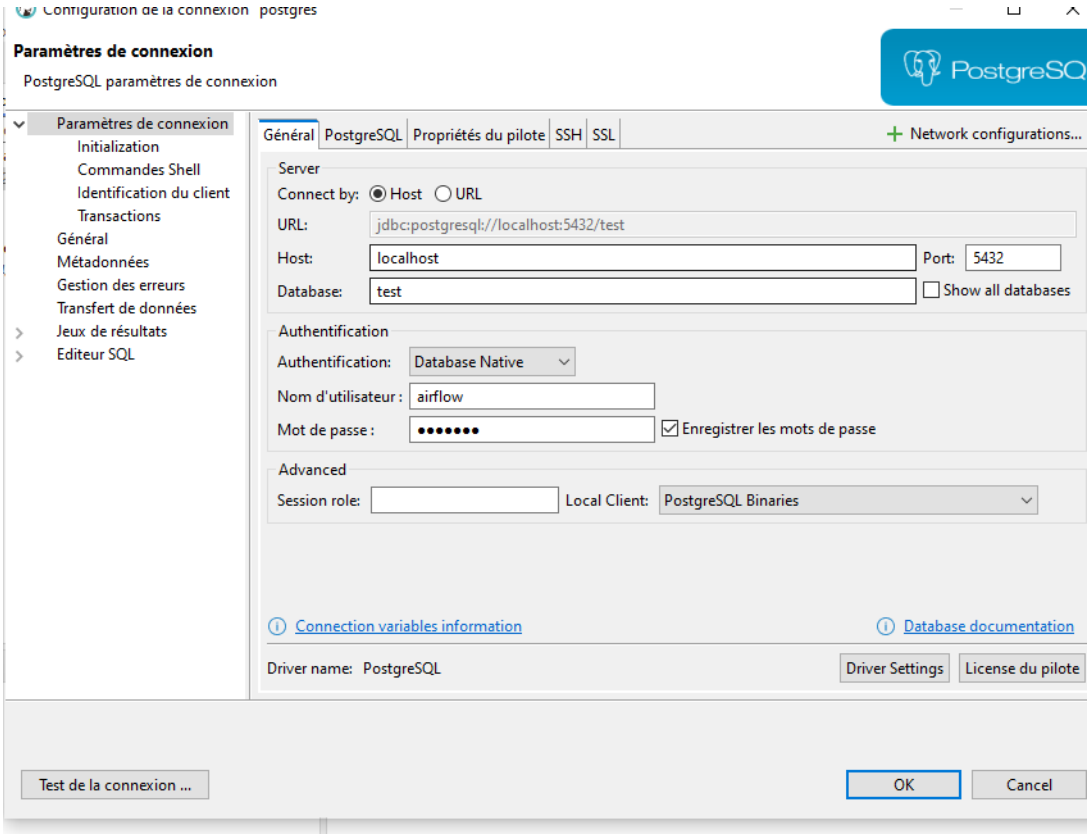
- **Tâches :**
  1. Conversion des données JSON en CSV (**convert-json-task**).
  2. Renommage des colonnes pour chaque dataset **(rename-columns-df1-task, rename-columns-df2-task, etc.)**.
  3. Fusion des différents datasets extraits **(merge-df1-df2-task, merge-with-customers-task**, etc.).

- **Fonctionnement :**
  1. Ce DAG dépend de l'extraction réussie des données, et les tâches de transformation ne commencent que lorsque l'extraction est terminée.
  2. Chaque tâche de transformation prend les données extraites comme entrée, applique les transformations nécessaires, et passe le résultat à la tâche suivante.



2.3.4 Connexion d'Airflow avec PostgreSQL pour Envoyer les Données

Une fois les données extraites et transformées, la dernière étape du processus ETL consiste à charger ces données dans une base de données PostgreSQL. Pour ce faire, Apache Airflow offre une manière simple de se connecter à PostgreSQL via son interface de connexion et l'utilisation d'opérateurs comme PostgresOperator. Voici les étapes pour configurer la connexion et charger les données dans PostgreSQL.



2.3.5 DAG de Chargement dans PostgreSQL (load_{dag})

Le troisième DAG est responsable du chargement des données transformées dans une base de données PostgreSQL. Les données sont organisées et chargées dans les tables

appropriées pour un traitement ultérieur.

- **Tâches :**
  1. Chargement des différentes tables dans PostgreSQL **(load-customers-task, load-products-task, load-orders-task**, etc.).
  2. Ajout des clés étrangères pour relier les tables entre elles **(add-foreign-keys-task)**.

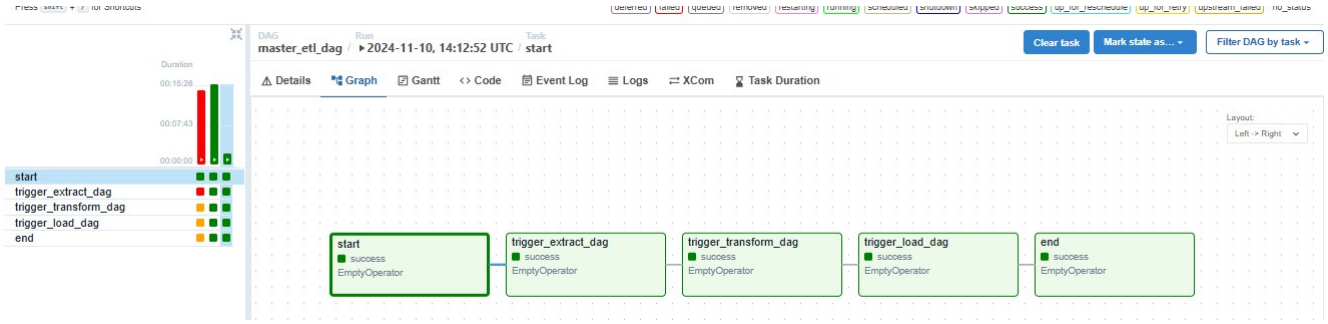
- **Fonctionnement :**
  1. Ce DAG dépend de la fin de la phase de transformation. Une fois les données prêtes, elles sont chargées dans la base de données cible.
  2. Le DAG de chargement peut également inclure des étapes pour vérifier l'intégrité des données avant leur insertion.

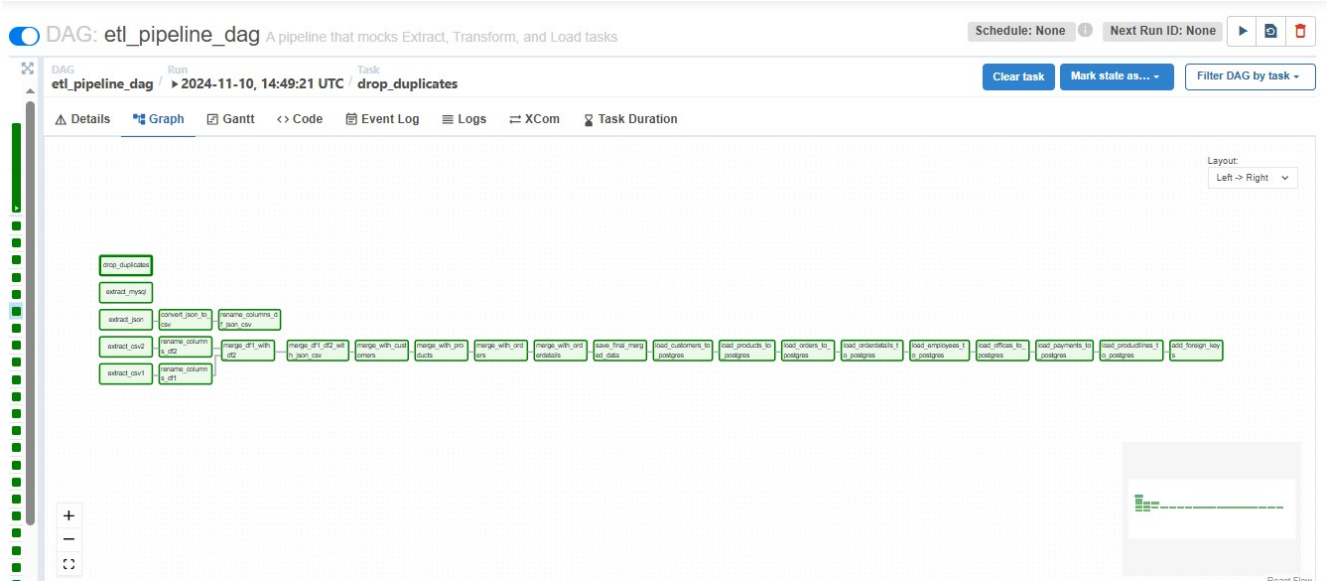
2.3.6 DAG Maître (master-etl-dag)

Le dernier DAG, **master-etl-dag**, orchestre l'exécution des trois autres DAGs. Il s'agit d'un DAG principal qui coordonne les autres DAGs pour garantir qu'ils s'exécutent dans le bon ordre et dans une séquence logique.

- **Tâches :** Exécution des DAGs d'extraction, de transformation et de chargement dans un ordre spécifique.

- **Fonctionnement :**
  Ce DAG peut être planifié pour démarrer manuellement ou de manière régulière.
  Il appelle chaque DAG dans l'ordre : extraction → transformation → chargement.
  Grâce à la hiérarchie des DAGs, **master-etl-dag** garantit que chaque phase de la pipeline est exécutée une fois la phase précédente terminée avec succès.

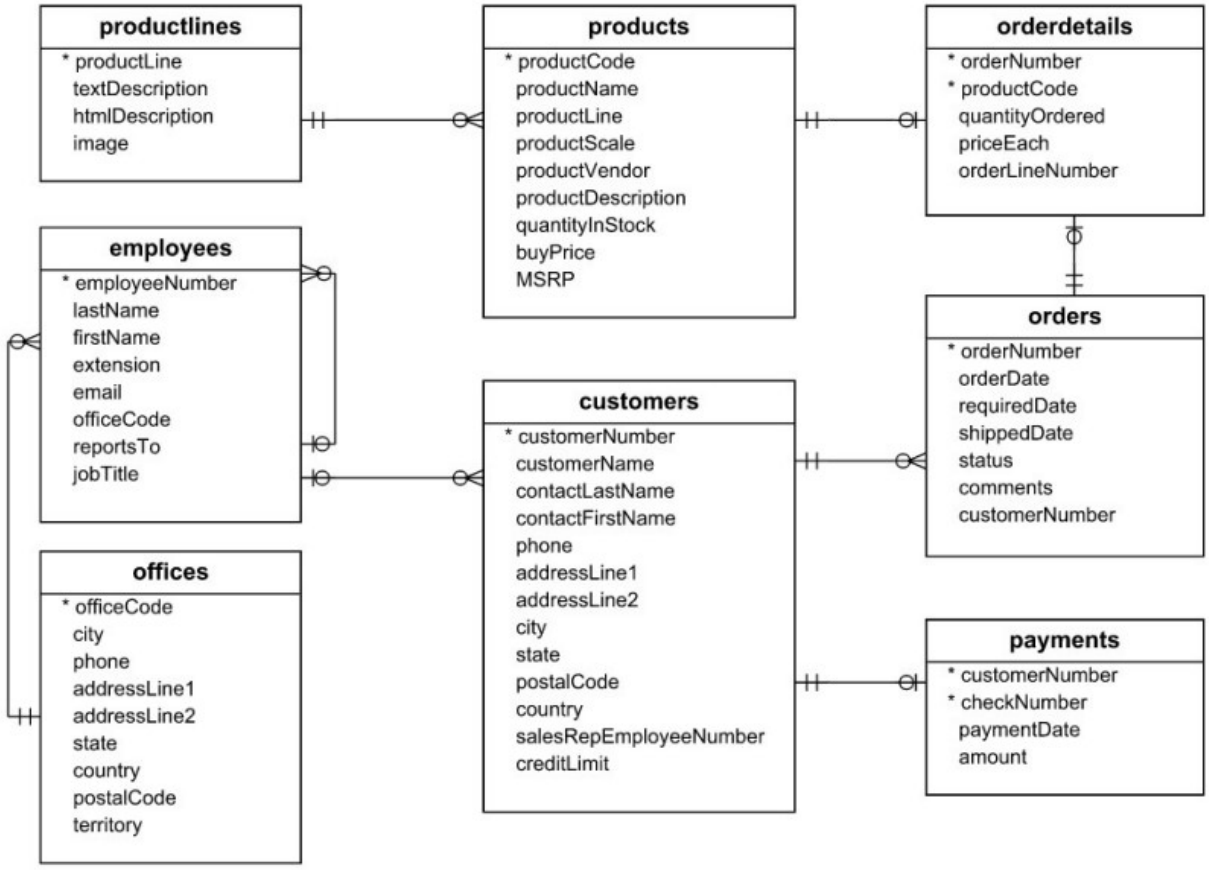


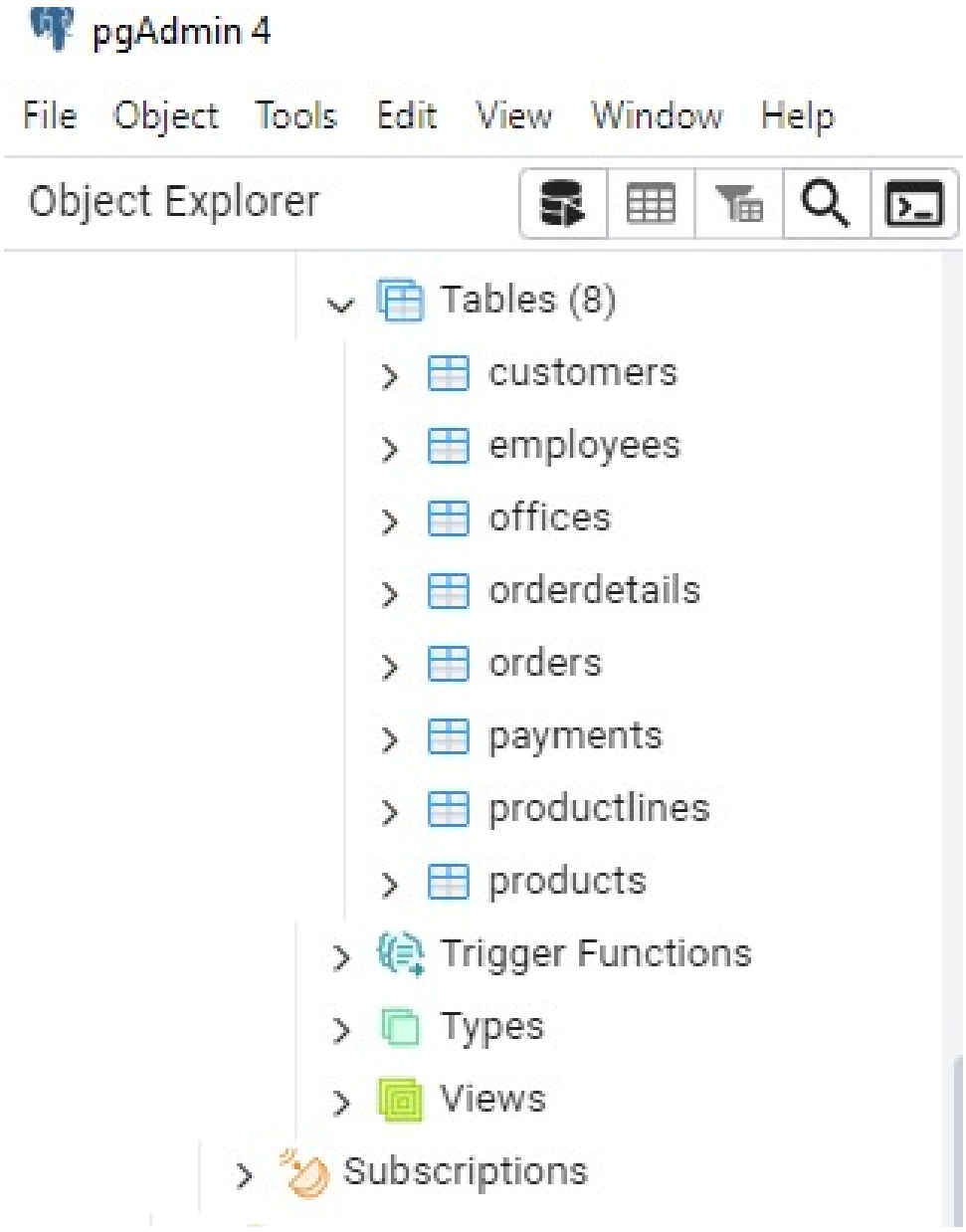


2.3.7 Structure de la Base de Données PostgreSQL

À la fin de la phase de chargement, les données transformées seront intégrées dans PostgreSQL, et plusieurs tables interconnectées permettront d'effectuer des analyses et des rapports. Ces données pourront être interrogées par des outils de BI, comme Tableau ou Power BI, pour générer des visualisations, des graphiques et des rapports qui aideront à la prise de décisions stratégiques.

Le chargement des données dans PostgreSQL crée un environnement centralisé où toutes les données provenant de différentes sources sont structurées et prêtes à être utilisées pour des analyses futures. Ce processus garantit une gestion efficace des données à grande échelle, tout en assurant leur qualité et leur intégrité.


productlines
* productLine
textDescription
htmlDescription
image
products
* productCode
productName
productLine
productScale
productVendor
productDescription
quantityInStock
buyPrice
MSRP
orderdetails
* orderNumber
* productCode
quantityOrdered
priceEach
orderLineNumber
employees
* employeeNumber
lastName
firstName
extension
email
officeCode
reportsTo
jobTitle
orders
* orderNumber
orderDate
requiredDate
shippedDate
status
comments
customerNumber
customers
* customerNumber
customerName
contactLastName
contactFirstName
phone
addressLine1
addressLine2
city
state
postalCode
country
salesRepEmployeeNumber
creditLimit
offices
* officeCode
city
phone
addressLine1
addressLine2
state
country
postalCode
territory
payments
* customerNumber
* checkNumber
paymentDate
amount


pgAdmin 4
File Object Tools Edit View Window Help
Object Explorer
Tables (8)
customers
employees
offices
orderdetails
orders
payments
productlines
products
Trigger Functions
Types
Views
Subscriptions

Visualisation des Graphes

Nous allons présenter la visualisation des données avec Streamlit, une plateforme permettant de créer des tableaux de bord interactifs. Cela permettra de visualiser les indicateurs clés, graphiques et rapports générés à partir des données traitées, offrant ainsi une interface simple et dynamique pour l'analyse des tendances et la prise de décision.

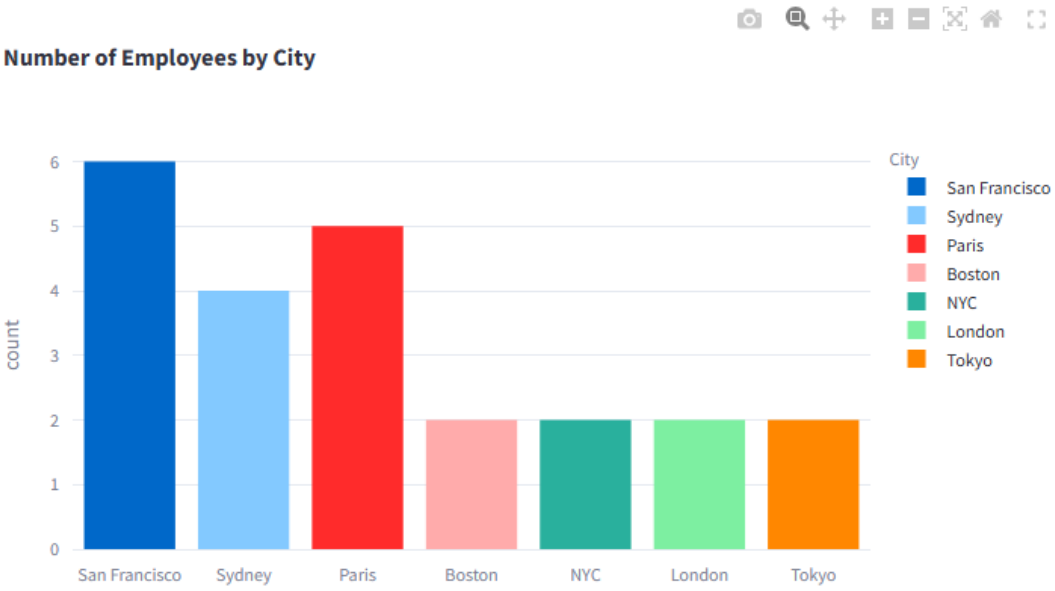


Figure 3.1

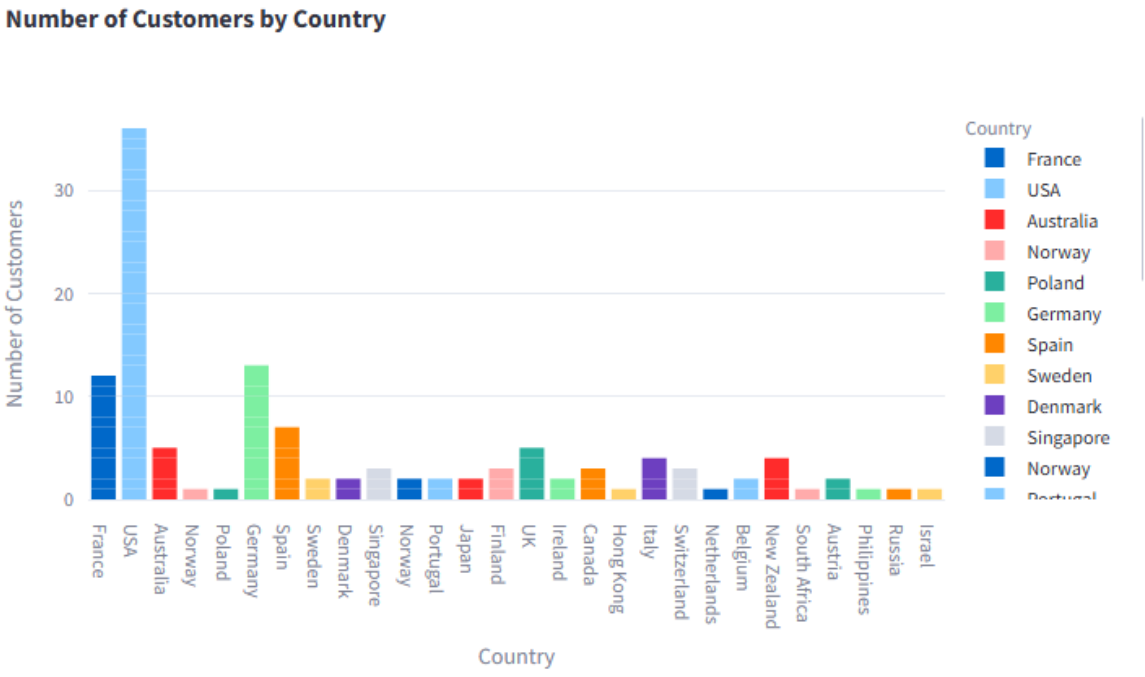


Figure 3.2

Figure 3.3

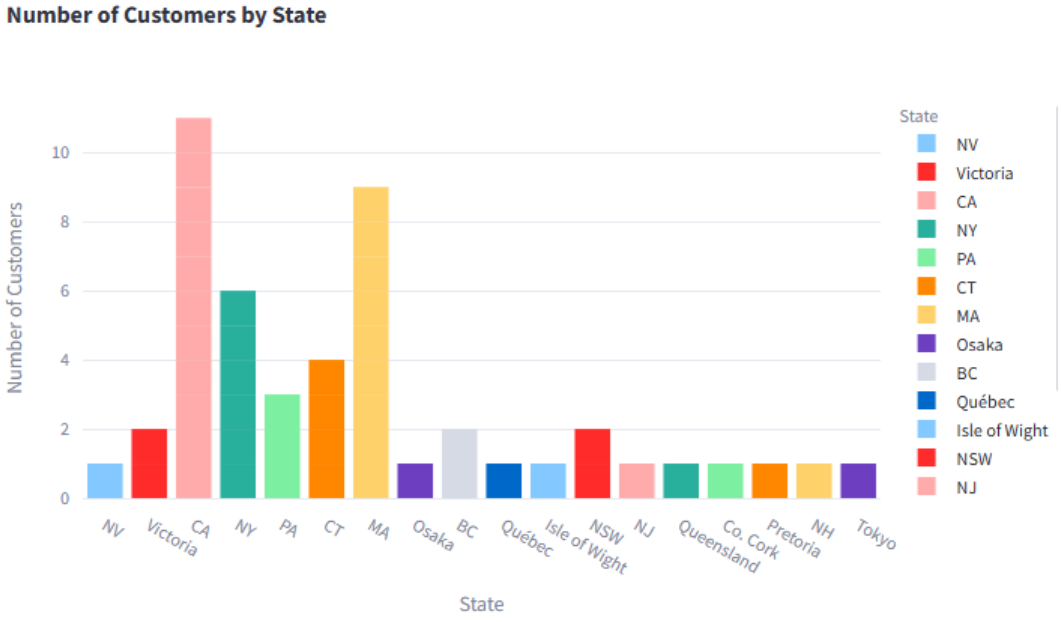


Figure 3.4

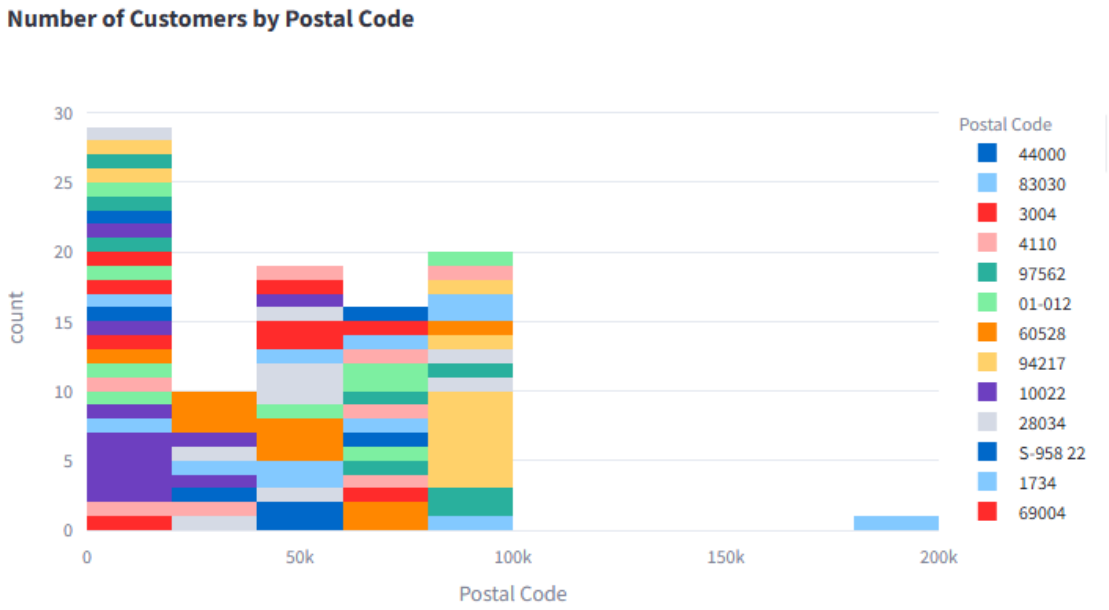


Figure 3.5

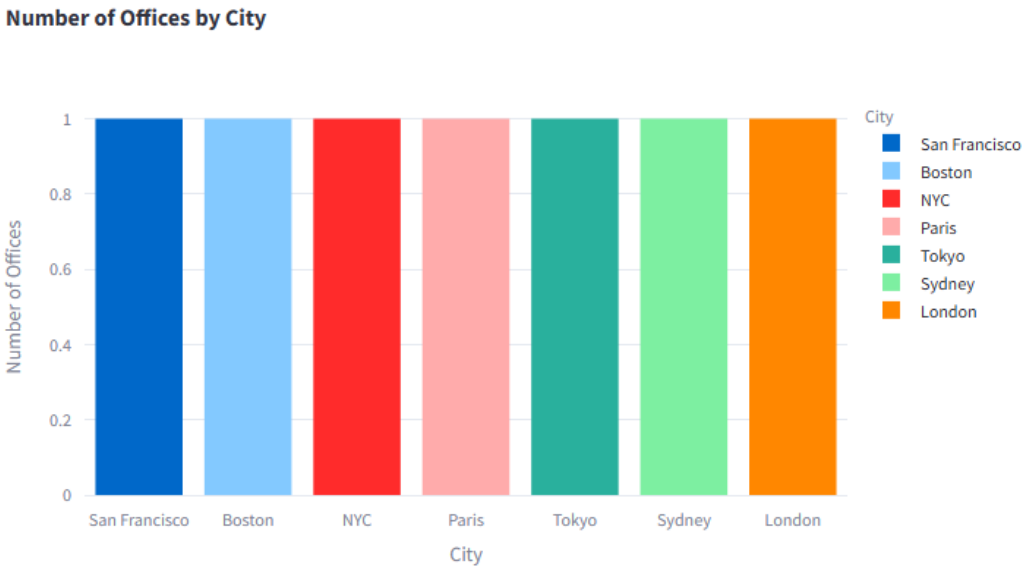


Figure 3.6

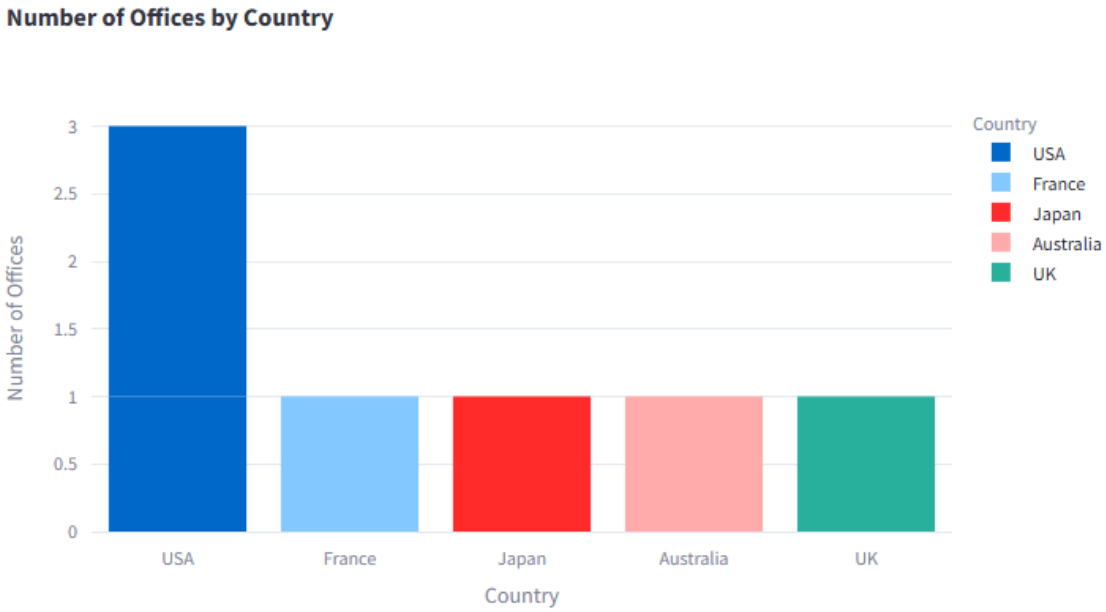

Number of Offices by Country
Number of Offices
3
2.5
2
1.5
1
0.5
0
USA
France
Japan
Australia
UK
Country
Country
USA
France
Japan
Australia
UK

Figure 3.7

Conclusion

Ce projet de création d'une pipeline ETL avec Apache Airflow permet d'automatiser l'intégration des données depuis plusieurs sources (fichiers CSV, JSON, bases de données) vers un entrepôt de données centralisé, tout en garantissant la qualité et la structure des données via des étapes de transformation. L'architecture modulaire du projet, divisée en quatre DAGs (Extraction, Transformation, Chargement et un DAG Maître), offre flexibilité, évolutivité et facilité de gestion.

Grâce à Airflow, la gestion des dépendances, le monitoring en temps réel et les alertes de défaillance assurent une gestion optimale des erreurs et une exécution fiable. Le tableau de bord final fournit aux utilisateurs des rapports et des visualisations basés sur les données de l'entrepôt, facilitant ainsi la prise de décisions stratégiques.

En résumé, cette solution complète garantit un processus de données automatisé et scalable, tout en offrant des outils robustes pour la surveillance et l'analyse, permettant ainsi d'optimiser la gestion des données et la prise de décision basée sur celles-ci.