Rapport ATL-Datamart – Projet de Datamart avec Beekeeper, Power BI et Airflow

Le projet ATL-Datamart, réalisé dans le cadre de l'atelier Architecture Décisionnelle (TRDE704), m'a permis de déployer une architecture complète de Big Data pour simuler un système décisionnel à l'échelle d'une entreprise du CAC40. Le but de ce projet était de manipuler des données, de les intégrer, de les transformer, et de les analyser à travers un processus de Data Engineering, tout en utilisant des outils professionnels comme Beekeeper Studio, Power BI et Apache Airflow.

Ce rapport retrace les différentes étapes du projet, en détaillant chaque TP, les outils utilisés, les choix faits, et les résultats obtenus. L'accent est mis sur la manière dont j'ai mis en place chaque étape, en utilisant des techniques et des outils adaptés à chaque problématique.

Membres du groupes :DANIOGO ABDOUL ,GNAMOU FAYE ,KOUAKOU REBECCA, HAMEDY CAMARA ,HOUNGA GRACE

Introduction

Le projet ATL-Datamart, réalisé dans le cadre de l'atelier Architecture Décisionnelle (TRDE704), m'a permis de déployer une architecture complète de Big Data pour simuler un système décisionnel à l'échelle d'une entreprise du CAC40. Le but de ce projet était de manipuler des données, de les intégrer, de les transformer, et de les analyser à travers un processus de Data Engineering, tout en utilisant des outils professionnels comme Beekeeper Studio, Power BI et Apache Airflow.

Ce rapport retrace les différentes étapes du projet, en détaillant chaque TP, les outils utilisés, les choix faits, et les résultats obtenus. L'accent est mis sur la manière dont j'ai mis en place chaque étape, en utilisant des techniques et des outils adaptés à chaque problématique.

TP 1 : Collecte des données vers un Datalake

Téléchargement des données

J'ai récupéré un fichier Parquet sur le site officiel de l'État de New York, contenant les données des courses de taxi pour le mois en question. Ce format de fichier est couramment utilisé dans les systèmes Big Data en raison de sa structure optimisée pour les données volumineuses.

Stockage dans Minio

Ensuite, j'ai utilisé Minio pour créer un bucket et y déposer le fichier Parquet. Minio permet de simuler un service S3 localement, ce qui facilite le stockage et la gestion des fichiers dans un environnement Docker.

Automatisation via script Python

J'ai complété le script grab_parquet.py en ajoutant les fonctions nécessaires pour télécharger et uploader le fichier vers le bucket Minio, automatisant ainsi la tâche de collecte des données.

Le TP 1 consistait à récupérer des données brutes de taxi jaunes à New York, actualisées chaque mois, et à les stocker dans un Datalake en utilisant Minio, un service Docker qui simule le stockage d'objets à la manière d'AWS S3.

TP 2 : Du Datalake vers un Data Warehouse

Le TP 2 avait pour objectif de récupérer les données stockées dans Minio et de les charger dans un Data Warehouse basé sur une base de données PostgreSQL. Ce processus d'intégration de données est crucial pour préparer les données brutes à l'analyse.

Création du Data Warehouse

J'ai créé une base de données PostgreSQL qui agit comme un Data Warehouse. J'ai utilisé Beekeeper Studio pour gérer les connexions à la base de données et pour faciliter la gestion des schémas et des tables.

Ingestion des données

J'ai écrit un script Python pour extraire les fichiers Parquet de Minio et les charger dans la base de données PostgreSQL. Ce script utilise Pandas pour lire les fichiers Parquet et SQLAlchemy pour insérer les données dans les tables de la base PostgreSQL.

Automatisation avec Python

Le script dump_to_sql.py a été modifié pour récupérer les fichiers Parquet directement depuis Minio et les insérer dans le Data Warehouse de manière automatisée. Cela permet de traiter de manière efficace les fichiers entrants sans intervention manuelle.

TP 3 : Création d'un Data Mart avec un modèle en flocon

Le TP 3 visait à structurer les données dans un modèle adapté à l'analyse en utilisant un Data Mart. Contrairement au Data Warehouse, le Data Mart est une version allégée et optimisée de ce dernier, qui contient uniquement les données nécessaires à un sous-ensemble d'utilisateurs métiers.

Mise en place de deux serveurs distincts

Pour ce TP, j'ai dû utiliser deux serveurs distincts pour le Data Warehouse et le Data Mart, conformément aux exigences du projet. Le Data Warehouse était hébergé sur le port 15432 et le Data Mart sur le port 15436. Cette configuration permet de séparer physiquement les deux environnements, garantissant ainsi des performances optimales et un meilleur contrôle des données.

Création du modèle en flocon

J'ai choisi le modèle en flocon pour organiser les données dans le Data Mart. Ce modèle est une extension du modèle en étoile, où les tables de dimension sont normalisées en sous-tables, ce qui réduit la redondance des données tout en optimisant les requêtes complexes.

Création des tables SQL

J'ai écrit un script SQL, create_tables.sql, pour créer les tables du modèle en flocon dans le Data Mart. Ce script définit les relations entre les tables et applique des contraintes d'intégrité pour garantir la cohérence des données. Le modèle inclut des tables de dimensions telles que dim_time, dim_vendor, et une table de faits fact_trips.

4 Insertion des données

Après avoir créé les tables, j'ai utilisé un autre script SQL, insert_data.sql, pour extraire les données du Data Warehouse et les insérer dans le Data Mart. Ce script exécute des requêtes SQL pour transférer les données de manière structurée et efficace entre les deux bases de données. Ce processus a permis de remplir les tables du Data Mart avec des données optimisées pour l'analyse.

TP 4 : Visualisation des données avec Power BI

Le TP 4 consistait à connecter un outil de visualisation de données au Data Mart pour permettre une analyse approfondie des données.

Utilisation de Power BI

Pour ce TP, j'ai choisi Power BI, un outil de visualisation de données, pour créer des dashboards interactifs. J'ai connecté Power BI à la base de données PostgreSQL du Data Mart en utilisant des requêtes SQL pour extraire les données nécessaires à l'analyse.

Création des visualisations

J'ai conçu plusieurs visualisations, notamment des graphiques et des cartes, pour explorer les tendances des courses de taxi, les revenus par quartier, et d'autres métriques pertinentes. Ces visualisations ont permis de découvrir des informations clés comme les périodes de forte activité, les zones géographiques à forte demande, et les revenus par heure ou jour.

TP 5 : Introduction à l'orchestration avec Airflow

Le TP 5 avait pour objectif de m'initier à l'orchestration des tâches à l'aide d'Apache Airflow, un outil qui permet d'automatiser et de planifier des pipelines de données.

Création d'un DAG pour Airflow

J'ai créé un DAG (Directed Acyclic Graph) dans Airflow pour automatiser le téléchargement du fichier Parquet depuis le site de l'État de New York et son stockage dans Minio. Ce DAG est exécuté selon un planning défini, permettant d'automatiser la collecte des données de manière régulière sans intervention manuelle.

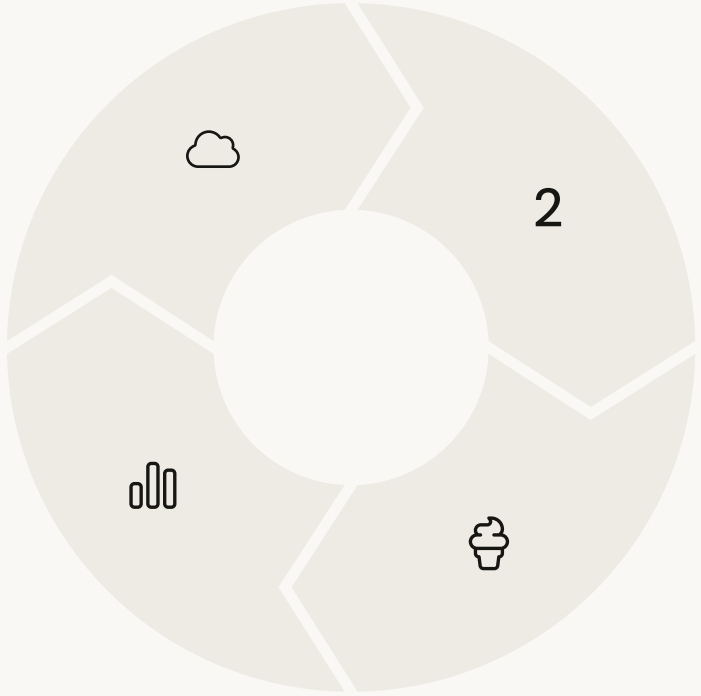
Automatisation des processus

Une fois le DAG compris, j'ai automatisé les processus des TP 2 et 3, afin de rendre l'intégration des données entre Minio, le Data Warehouse, et le Data Mart totalement autonome. Ce système d'orchestration garantit la mise à jour régulière des données et la transformation automatique des données brutes en données prêtes à l'analyse.

Architecture globale du projet

Collecte des données

Téléchargement des fichiers Parquet depuis le site officiel de New York



Stockage dans le Datalake

Utilisation de Minio comme service de stockage d'objets

Transformation et analyse

Création d'un Data Mart et visualisation avec Power BI

Intégration dans le Data Warehouse

Chargement des données dans PostgreSQL via scripts Python

Cette architecture complète permet de gérer efficacement le flux de données depuis leur collecte jusqu'à leur analyse, en passant par leur stockage et leur transformation. L'orchestration avec Airflow garantit l'automatisation et la régularité des processus.

Conclusion

Ce projet m'a permis d'acquérir des compétences pratiques dans la gestion de données dans une architecture Big Data, en utilisant des outils comme Beekeeper Studio, Power BI, et Apache Airflow. J'ai appris à mettre en place des processus d'intégration et de transformation des données, tout en garantissant une séparation logique des environnements (Data Warehouse et Data Mart) pour optimiser les performances. Ce projet m'a également permis de comprendre l'importance de l'orchestration des tâches pour automatiser et garantir la régularité des processus de collecte et de traitement des données.

Compétences acquises

Maîtrise des outils de gestion de données, conception d'architecture Big Data, automatisation des processus

Outils maîtrisés

Beekeeper Studio, Power BI, Apache Airflow, Minio, PostgreSQL

Perspectives

Application de ces connaissances dans des projets professionnels à grande échelle