CAHIER DES CHARGES

Projet n°1 : ETL

Table des matières

Introduction

Le projet consiste à développer un outil de manipulation de données en Python, basé sur le processus Extract-Transform-Load (ETL).

Cet outil permettra d'extraire des données de diverses sources, de les transformer en fonction de besoins spécifiques, puis de les charger dans différents formats de sortie.

L'objectif est de concevoir un système modulaire, pilotable en ligne de commande de manière déclarative, et potentiellement intégrable dans d'autres applications.

La première livraison du projet se concentrera sur la lecture de données depuis différents types de sources, leur transformation selon des critères prédéfinis, et leur stockage dans divers formats.

Ce projet sera mené selon des méthodologies agiles, favorisant le travail collaboratif et itératif, avec un accent sur la spécification, le développement et les tests.

Analyse client

I. Analyse du Besoin du Client

1. Contexte :

Le client souhaite mettre en place un nouvel outil ETL (Extract, Transform, Load) répondant à des besoins spécifiques de gestion et manipulation de données provenant de diverses sources.
L'objectif est d'optimiser le processus d'acquisition, de transformation et de stockage des données.

2. Objectifs du Projet

2.1. Objectif Principal

Le client souhaite un outil ETL écrit en Python, flexible, modulaire et capable de traiter diverses sources de données, offrant une solution déclarative pour automatiser l'extraction, la transformation et le chargement de données.

2.2. Objectifs Spécifiques

Flexibilité et Modularité : L'outil doit permettre l'ajout de nouvelles fonctionnalités sans refonte majeure.

Pilotage Déclaratif : Le pilotage de l'outil doit être déclaratif, simplifiant la configuration et l'exécution des tâches.

Interface en Ligne de Commande : L'outil doit être utilisable en ligne de commande pour une intégration facile.

Intégration comme Module : L'outil peut être utilisé comme module dans d'autres applications, nécessitant une conception compatible ainsi qu'une documentation claire pour les développeurs tiers.

2.3. Objectifs Fonctionnels

Extraction de Données : Lire des données depuis diverses sources telles que fichiers textuels, JSON, XML, HTML, CSV, bases de données relationnelles et APIs, éventuellement simultanément.

Transformation de Données : Les fonctionnalités de transformation incluent le filtrage des données selon des critères définis, la correction des données manquantes ou aberrantes, et la réalisation de calculs complexes basés sur les attributs des jeux de données.

Chargement de Données : L'outil doit être en mesure de stocker les données transformées dans une base de données relationnelle, un fichier JSON ou XML, et potentiellement dans plusieurs formats simultanément.

Configuration Déclarative : Le client souhaite décrire le pipeline de traitement des données à l'aide d'un fichier YAML, offrant ainsi une configuration déclarative et lisible.

2.4. Objectifs de release v1.0

Lecture de données depuis différentes sources : L'outil sera capable d'extraire des données à partir de diverses sources telles que des fichiers textuels, des fichiers JSON, XML, HTML, des bases de données relationnelles et des API.

Transformation des données : Les données extraites pourront être transformées selon divers critères, notamment le filtrage, la correction des données manquantes ou aberrantes, les calculs, la normalisation des valeurs et l'ajout d'attributs.

Stockage des données dans différents formats : Une fois transformées, les données pourront être stockées dans différents formats de sortie, tels que des bases de données relationnelles, des fichiers JSON ou XML.

Choix des technologies :

1. Python

1.1. Description :

Python est un langage de programmation polyvalent utilisé pour le développement de l'outil ETL. Il offre une syntaxe claire, une grande flexibilité, et une vaste gamme de bibliothèques, le rendant adapté à la réalisation de projets variés.

1.2. Rôle dans le Projet :

Langage polyvalent : La polyvalence de Python permet d'implémenter efficacement toutes les phases du processus ETL, de l'extraction à la transformation et au chargement des données.

Large écosystème : L'écosystème Python offre un accès à de nombreuses bibliothèques spécialisées, ce qui facilite l'intégration d'outils et de fonctionnalités complémentaires.

Facilité de lecture et de maintenance : La syntaxe claire de Python favorise la lisibilité du code, facilitant ainsi la maintenance et l'évolution du projet.

2. Pandas (Librairie de Data Analyses en Python)

2.1. Description :

Pandas est une bibliothèque Python puissante pour la manipulation et l'analyse des données. Elle offre des structures de données flexibles et performantes, notamment les DataFrame, qui simplifient le traitement, la transformation et l'analyse de données tabulaires.

2.2. Rôle dans le Projet :

Manipulation efficace des données : Pandas fournit des fonctionnalités avancées pour manipuler les données, y compris le filtrage, la transformation, et l'agrégation. Les opérations sur les DataFrame permettent une gestion aisée des données tabulaires.

Traitement des données provenant de différentes sources : Pandas facilite l'importation et la manipulation de données provenant de diverses sources telles que des fichiers CSV, des bases de données, des feuilles Excel, etc. Cela renforce la capacité du projet ETL à gérer des sources de données variées.

Intégration harmonieuse dans le flux de travail Python du projet ETL : En tant que bibliothèque de data analysis populaire, Pandas s'intègre de manière transparente dans l'écosystème Python, ce qui simplifie son utilisation dans le cadre du projet ETL. Les fonctionnalités de Pandas sont adaptées aux tâches de manipulation de données complexes.

3. SQLAlchemy :

3.1. Description :

SQLAlchemy est une bibliothèque SQL en Python qui fournit une abstraction puissante pour interagir avec des bases de données relationnelles. Elle simplifie la gestion des connexions, l'exécution de requêtes SQL, et offre une approche flexible pour la modélisation des données.

3.2. Rôle dans le Projet :

Interfaçage avec des bases de données : SQLAlchemy offre une interface Python pour interagir avec des bases de données relationnelles, permettant ainsi au projet ETL de traiter efficacement les données stockées dans ces systèmes.

Abstraction des requêtes SQL : L'abstraction fournie par SQLAlchemy simplifie la rédaction et l'exécution de requêtes SQL, facilitant ainsi les opérations de transformation et de chargement des données.

Flexibilité dans la modélisation des données : SQLAlchemy offre des options flexibles pour modéliser les données, adaptées aux besoins spécifiques du projet ETL.

4. Lxml :

4.1. Description :

Lxml est une bibliothèque Python dédiée au traitement de données XML et HTML. Elle est utilisée de manière efficace pour extraire et manipuler des données à partir de ces formats, améliorant ainsi la performance du processus d'extraction des données.

4.2. Rôle dans le Projet :

Traitement efficace des données XML/HTML : L'utilisation de Lxml permet une manipulation efficace des données au format XML et HTML, essentielle pour l'extraction de données à partir de ces sources.

Extraction de données structurées : Lxml offre des outils puissants pour extraire des données structurées à partir de documents XML et HTML, alignant ainsi le projet ETL sur les besoins de traitement des données.

5. XML :

5.1. Description :

Le module XML de Python est inclus dans la bibliothèque standard. Il offre des fonctionnalités intégrées pour travailler avec des données au format XML.

5.2. Rôle dans le Projet :

Manipulation des données XML : Le module XML de Python fournit des fonctionnalités intégrées pour travailler avec des données au format XML, simplifiant ainsi l'extraction et la transformation de telles données dans le projet ETL.

Utilisation transparente avec d'autres bibliothèques : Étant une partie de la bibliothèque standard, le module XML s'intègre de manière transparente avec d'autres bibliothèques Python, offrant une expérience cohérente dans le traitement des données XML.

6. Requests :

6.1. Description :

Requests est une bibliothèque Python qui permet d'envoyer des requêtes HTTP de manière simple et efficace. Elle facilite l'interaction avec des services web et des API en permettant d'effectuer des requêtes GET, POST, PUT, DELETE et en gérant les cookies, les en-têtes HTTP, les données de formulaire, les fichiers téléchargés, etc.

6.2. Rôle dans le Projet :

Les appels API peuvent être nécessaires pour récupérer des données à partir de services web ou d'applications tierces. Le rôle de la bibliothèque Requests dans le projet est donc de faciliter ces appels API.

Gestion de projet :

I. Spécifications :

Durée : 5 semaines
Taille de l'Équipe : 4 personnes
Évaluation du temps de travail : 5 semaines

II. Déroulement du projet :

- Constitution de l'équipe
- Lecture et compréhension du brief
- Recherches sur le fonctionnement d'un ETL
- Définitions de tâches préliminaires au développement (mise en place et veille)
- Création d'un environnement de travail sur Trello
- Mise en place des documents de support/livrables (cahier des charges, documentation)
- Définition de l'architecture & fonctionnement du projet
- Choix des outils
- Découpage en tâches
- Écriture du code
- Documentation

III. Liste fonctionnelle :

Configuration :
- Création d'un fichier YAML pour décrire le pipeline de traitement des données.

Extraction de données :
- Lecture de données à partir de fichiers textuels (CSV, TXT, etc.).
- Extraction de données depuis des fichiers JSON, XML, et HTML.
- Connexion à des bases de données relationnelles (MySQL, PostgreSQL, etc.) pour extraire des données.
- Intégration avec des API pour récupérer des données distantes.
- Possibilité de lire des données depuis plusieurs sources simultanément.

Transformation de données :
- Filtrage des données selon des critères spécifiques (valeurs, conditions, etc.).
- Correction des données manquantes ou aberrantes. Calculs basés sur des attributs des jeux de données (somme, moyenne, etc.).
- Normalisation des valeurs pour uniformiser les formats.
- Ajout d'attributs calculés ou dérivés à partir des données existantes.

Chargement de données :
- Stockage des données transformées dans une base de données relationnelle (SQLite, etc.).
- Sauvegarde des données transformées dans des fichiers JSON ou XML.
- Possibilité de choisir le format de sortie en fonction des besoins spécifiques.
- Gestion de plusieurs formats de sortie simultanément si nécessaire.

IV. Outils :

1. Trello (Gestion des tâches)

1.1. Description :

Trello est une plateforme de gestion de projet qui utilise des tableaux, des listes et des cartes pour organiser les tâches et collaborer en équipe. Chaque carte représente une tâche, et vous pouvez les organiser dans des listes selon leur statut (à faire, en cours, terminé).

1.2. Rôle dans le Projet :

Gestion efficace des tâches et des étapes du projet.
Suivi visuel des avancements individuels et de l'équipe.
Facilitation de la collaboration et de la communication entre les membres de l'équipe.

2. Github (Partage de code en mise en commun)

1.3. Description :

Github est une plateforme de développement collaboratif qui permet aux équipes de partager et de collaborer sur des projets logiciels. Il offre des fonctionnalités telles que le contrôle de version avec Git, le suivi des problèmes, la gestion des branches, et la fusion de code.

1.4. Rôle dans le projet :

Hébergement centralisé du code source du projet.
Suivi détaillé des modifications apportées au code via les commits.

3. Google Doc :

1.5. Description :

Google Docs est un service de traitement de texte en ligne qui permet la création et la modification de documents collaboratifs en temps réel. Il offre des fonctionnalités telles que le partage de documents, les commentaires en direct et la gestion des versions.

1.6. Rôle dans le projet :

Google Docs sera utilisé pour la collaboration sur la documentation du projet, y compris la rédaction du cahier de spécifications, la documentation des décisions prises lors des réunions, et la rédaction de tout autre document nécessaire au projet. En permettant à plusieurs membres de l'équipe de travailler simultanément sur les documents, Google Docs facilite la coordination et la communication, tout en assurant la cohérence et la traçabilité des informations.

V. Organisation des réunions :

Nous avons mis en place a planning de réunions qui varie en fonction des semaines.

1. En semaine à l'École multimédia, nous réalisions des réunions tous les jours de la semaine à 9h.

2. En ce qui concerne les semaines en entreprise, nous avons réalisé des réunions 2 fois par semaine, le lundi et le jeudi.

3. Nous avons fait une réunion finale a fin de tout remettre en commun le lundi 15 avril à 15h30

VI. Évaluation du temps de travail :

Tâches	Temps de travail
Mise en place du groupe	20 min
Échange sur le projet (mise en commun de la compréhension)	1h
Estimation de la faisabilité du projet	30 min
Mise en place d'outils de collaboration	15 min
Mise en place d'outils de partage de code	15 min
Création d'un fichier YAML	1h
Lecture de données à partir de fichiers textuels (CSV, TXT, etc.).	2h
Extraction de données depuis des fichiers JSON, XML, et HTML.	2 jours
Connexion à des bases de données relationnelles (MySQL, PostgreSQL, etc.) pour extraire des données.	2 jours
Intégration avec des API pour récupérer des données distantes.	1 jours
Possibilité de lire des données depuis plusieurs sources simultanément.	1 jour
Filtrage des données selon des critères spécifiques (valeurs, conditions, etc.).	2h
Correction des données manquantes ou aberrantes	2 jours
Normalisation des valeurs pour uniformiser les formats.	1 jour
Ajout d'attributs calculés ou dérivés à partir des données existantes.	1 jour
Stockage des données transformées dans une base de données relationnelle (SQLite, etc.).	1 jour
Sauvegarde des données transformées dans des fichiers JSON ou XML.	5h
Possibilité de choisir le format de sortie en fonction des besoins spécifiques.	30 min
Gestion de plusieurs formats de sortie simultanément si nécessaire.	30 min
Read me	1h
Test	2 jours
Documentation	2h

VII. Recettage :

Tâches	Status (opérationnelle / non opérationnelle)
Lecture de données à partir de fichiers textuels (CSV, TXT, etc.).	Opérationnelle
Extraction de données depuis des fichiers JSON, XML, et HTML.	Opérationnelle
Connexion à des bases de données relationnelles (MySQL, PostgreSQL, etc.) pour extraire des données.	Non opérationnelle
Intégration avec des API pour récupérer des données distantes.	Opérationnelle
Possibilité de lire des données depuis plusieurs sources simultanément.	Non opérationnelle
Filtrage des données selon des critères spécifiques (valeurs, conditions, etc.).	Opérationnelle
Correction des données aberrantes	Non opérationnelle
Correction des données manquantes	Opérationnelle
Normalisation des valeurs pour uniformiser les formats.	Opérationnelle
Ajout d'attributs calculés ou dérivés à partir des données existantes.	Opérationnelle
Stockage des données transformées dans une base de données relationnelle (SQLite, etc.).	Non opérationnelle
Sauvegarde des données transformées dans des fichiers JSON ou XML.	Opérationnelle
Possibilité de choisir le format de sortie en fonction des besoins spécifiques.	Opérationnelle
Gestion de plusieurs formats de sortie simultanément si nécessaire.	Opérationnelle

VIII. Améliorations pour une version ultérieure :

Axes d'améliorations :

- Pouvoir accueillir plusieurs fichiers en entrée
- Mise en relation d'une base de donnée et traitement de fichier SQL

Axes d'évolutions :

- Choix du dossier de sortie (ou serveur)
- Mise en place d'une interface front (éviter la console)
- 2 exemples de format de nettoyage pour montrer le fonctionnement, à étendre sur tous les formats
- Optimisation du fichier job

Annexes :

Adresse GitHub :

https://github.com/Miguel-Novoa/Projet_ETL_EM_Big_DATA

Trello :