

E1 : Intégration et Analyse des Données Littéraires

Introduction

L'objectif était de collecter des données sur des auteurs, leurs œuvres, et les distinctions qu'ils ont reçues à partir de trois sources principales : une API externe, des pages web par scraping, et une base de données existante. Une fois les données centralisées, elles ont été structurées dans une base relationnelle et exposées via une API sécurisée développée en FastAPI.

Les défis principaux comprenaient la manipulation de données hétérogènes, la sécurisation de l'accès à la base de données, et l'implémentation de requêtes analytiques pertinentes. Ce rapport décrit en détail les étapes suivies, les technologies choisies, et les résultats obtenus.

Objectifs du projet

Ce projet avait plusieurs objectifs précis, qui peuvent être résumés comme suit :

1. **Collecte de données :**
 - Récupérer des informations sur les auteurs et leurs œuvres depuis des sources diversifiées : une API (Google Books), des pages Wikipédia (scraping), et une base de données existante (distinctions littéraires).
2. **Organisation des données :**
 - Centraliser ces données dans une base de données relationnelle MySQL, avec une structure robuste permettant des jointures efficaces entre les tables.
3. **Création d'une API :**
 - Développer une API REST avec FastAPI pour permettre un accès simple et structuré aux données.
4. **Analyse des données :**
 - Implémenter des requêtes SQL complexes pour extraire des informations analytiques, comme les catégories les plus fréquentes des livres ou les distinctions par auteur.
5. **Sécurisation des données et de l'API**

- Assurer la protection des informations sensibles comme les clés API et les identifiants de la base de données, ainsi que l'authentification de l'API finale.

Technologies choisies

Le choix des technologies a été guidé par leur pertinence et leur compatibilité avec les objectifs du projet :

- **Python** : Utilisé pour la collecte, le traitement, et la manipulation des données grâce à ses bibliothèques riches (requests, BeautifulSoup, pandas, SQLAlchemy).
- **FastAPI** : Framework rapide et léger pour créer une API RESTful, permettant une intégration simple avec Python et SQLAlchemy.
- **MySQL** : Système de gestion de base de données relationnelle pour stocker et organiser les données centralisées.
- **DBeaver** : Interface graphique utilisée pour gérer la base de données et exécuter des requêtes SQL.
- **dotenv** : Gestion des variables d'environnement pour sécuriser les informations sensibles comme les clés API.

Méthodologie et Étapes

1. Collecte des données

a) Extraction depuis l'API Google Books

Nous avons utilisé l'API Google Books pour collecter des informations sur les œuvres de dix auteurs célèbres. Les données extraites incluent les titres, les dates de publication, les descriptions, les catégories, et le nombre de pages. Une clé API a été utilisée pour authentifier les requêtes.

Les résultats ont été nettoyés et convertis en un DataFrame avant d'être exportés au format CSV (format date, colonnes nécessaire, Nan).

b) Scraping des données biographiques depuis Wikipédia

Pour enrichir nos données, nous avons récupéré les dates de naissance et de décès des auteurs depuis leurs pages Wikipédia respectives. Le scraping a été effectué avec BeautifulSoup.

Étapes principales :

1. Charger la page de chaque auteur.
2. Extraire les informations pertinentes (dates, biographie).
3. Nettoyer et formater les données.

c) Intégration des distinctions littéraires

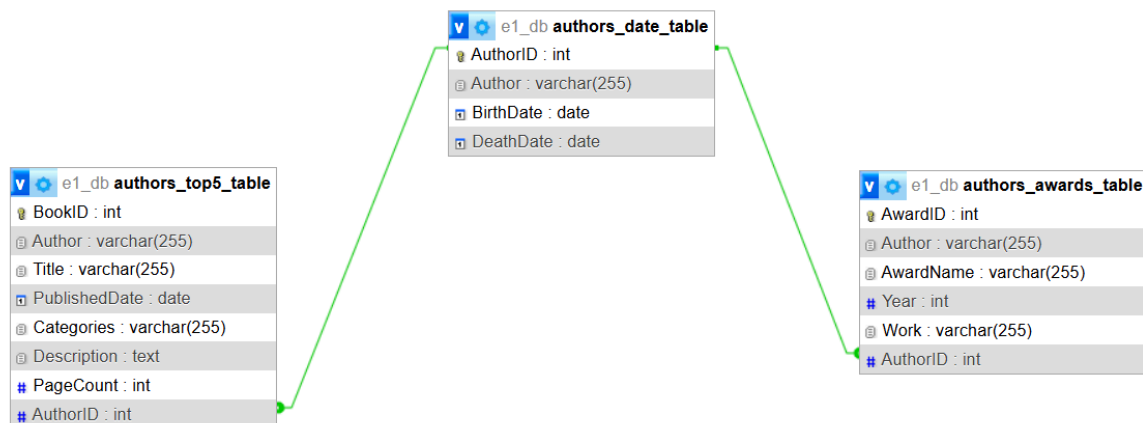
Une base de données existante a été utilisée pour collecter des informations sur les distinctions littéraires obtenues par les auteurs. Les données incluent le nom de l'auteur, le prix remporté, l'année, et l'œuvre primée. Cette base de données MySQL a été connectée à notre base finale.

2. Création de la base de données relationnelle

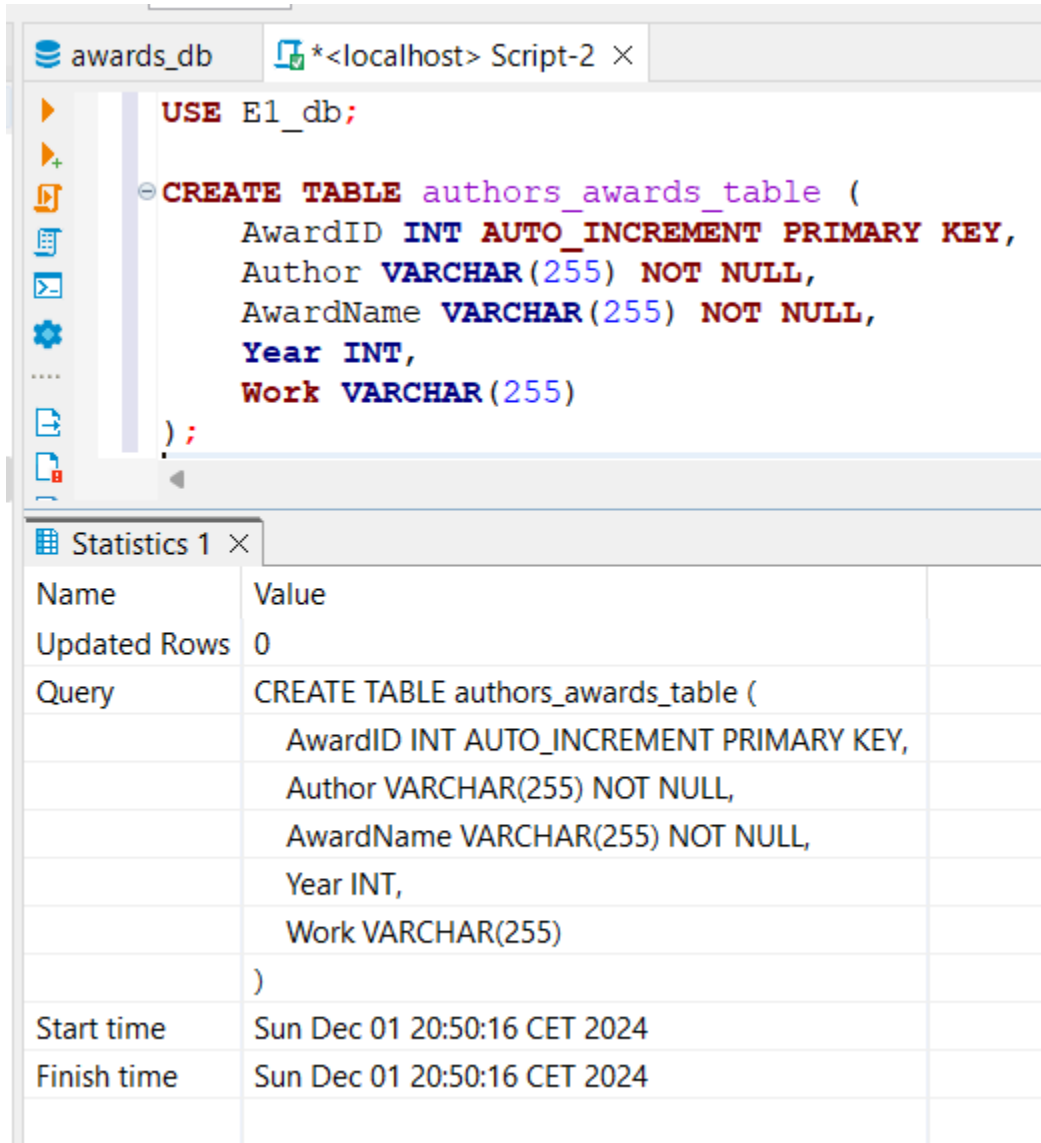
Les données collectées ont été organisées dans une base de données MySQL, composée de trois tables principales :

1. **authors_date_table** : Contient les informations biographiques des auteurs.
2. **authors_top5_table** : Liste les œuvres principales des auteurs.
3. **authors_awards_table** : Répertoire les distinctions reçues.

Des relations ont été définies entre ces tables pour permettre des jointures efficaces. Voici le MCD (Modèle Conceptuel des Données) illustrant la structure relationnelle mise en place :



Exemple de création d'une table :



The screenshot shows a database management interface with two panes. The top pane displays a SQL script for creating a table named `authors_awards_table` in the `awards_db` database. The script uses the `USE` statement to select the database and the `CREATE TABLE` statement to define the table structure. The table has four columns: `AwardID` (an auto-incrementing primary key), `Author` (a VARCHAR of length 255, not null), `AwardName` (a VARCHAR of length 255, not null), `Year` (an integer), and `Work` (a VARCHAR of length 255). The bottom pane shows the execution statistics for the query, indicating that 0 rows were updated and the query was executed successfully on Sunday, December 1, 2024, at 20:50:16 CET.

```
USE E1_db;

CREATE TABLE authors_awards_table (
    AwardID INT AUTO_INCREMENT PRIMARY KEY,
    Author VARCHAR(255) NOT NULL,
    AwardName VARCHAR(255) NOT NULL,
    Year INT,
    Work VARCHAR(255)
);
```

Name	Value
Updated Rows	0
Query	CREATE TABLE authors_awards_table (AwardID INT AUTO_INCREMENT PRIMARY KEY, Author VARCHAR(255) NOT NULL, AwardName VARCHAR(255) NOT NULL, Year INT, Work VARCHAR(255))
Start time	Sun Dec 01 20:50:16 CET 2024
Finish time	Sun Dec 01 20:50:16 CET 2024

3. Développement de l'API

Une API REST a été créée à l'aide de FastAPI pour permettre l'accès aux données et la réalisation de requêtes analytiques.

- **Endpoint : Catégories des livres**
 - Retourne le nombre de livres par catégorie.

Requête SQL :

```
SELECT Categories, COUNT(BookID) AS NumberOfBooks
FROM authors_top5_table
GROUP BY Categories
ORDER BY NumberOfBooks DESC;
```

- **Endpoint : Distinctions des auteurs**

- Retourne les distinctions reçues par auteur, avec leurs dates de naissance.

Requête SQL :

```
SELECT ad.Author, ad.BirthDate, aa.AwardName, aa.Year, aa.Work
FROM authors_date_table ad
INNER JOIN authors_awards_table aa ON ad.AuthorID = aa.AuthorID
ORDER BY aa.Year DESC;
```

4. Sécurisation des données et de l'API

Un fichier `.env` a été utilisé pour stocker les informations sensibles :

- Clé API pour Google Books.
- URL de connexion à la base de données.
- Token d'authentification à l'API

Le fichier `.env` a été exclu du dépôt Git via `.gitignore` pour éviter toute fuite d'informations sensibles.

Résultats

1. Analyse des données

- **Catégories des livres :**
 - La catégorie la plus fréquente parmi les œuvres collectées est **Fiction**, suivie de **Literary Collections**. Ces données ont permis de mieux comprendre la répartition des œuvres des auteurs étudiés.
- **Distinctions littéraires :**
 - Une jointure SQL entre les tables a révélé les récompenses associées à chaque auteur. Les résultats montrent que des auteurs comme **George**

Orwell et **Romain Gary** ont été récompensés pour des œuvres qui continuent d'avoir un impact significatif dans la littérature mondiale.

2. Fonctionnalités de l'API

- L'API FastAPI permet d'accéder efficacement aux données via des endpoints spécifiques. Par exemple :
 - **Liste des livres par catégorie** : Un endpoint retourne les catégories des livres et le nombre d'ouvrages dans chaque catégorie.
 - **Auteurs et leurs distinctions** : Un autre endpoint expose les récompenses reçues par chaque auteur, avec des informations biographiques.
- Les endpoints développés offrent une interaction directe avec les données stockées dans la base MySQL.

Les requêtes SQL exécutées dans le cadre de l'API ont été testées avec succès dans le code, et les résultats des endpoints sont retournés au format JSON, facilitant leur exploitation dans d'autres applications.

awards_db * <localhost> Script-2 ×

```

USE E1_db;

CREATE TABLE authors_awards_table (
    AwardID INT AUTO_INCREMENT PRIMARY KEY,
    Author VARCHAR(255) NOT NULL,
    AwardName VARCHAR(255) NOT NULL,
    Year INT,
    Work VARCHAR(255)
);

```

Statistics 1 ×

Name	Value
Updated Rows	0
Query	CREATE TABLE authors_awards_table (AwardID INT AUTO_INCREMENT PRIMARY KEY, Author VARCHAR(255) NOT NULL, AwardName VARCHAR(255) NOT NULL, Year INT, Work VARCHAR(255))
Start time	Sun Dec 01 20:50:16 CET 2024
Finish time	Sun Dec 01 20:50:16 CET 2024

localhost - localhost:3306

- Bases de données
 - E1_db
 - Tables
 - authors_awards_table 16K
 - authors_date_table 16K
 - authors_top5_table 16K

```
SELECT ad.Author, ad.BirthDate, aa.AwardName, aa.Year, aa.Work
FROM authors_date_table ad
INNER JOIN authors_awards_table aa ON ad.AuthorID = aa.AuthorID
ORDER BY aa.Year DESC;SSS
```

authors_date_table(+) 1 ×

SELECT ad.Author, ad.BirthDate, aa.AwardName |  Entrez une expression SQL pour filtrer les résultats (util


	ABC Author	BirthDate	ABC AwardName	123 Year	ABC Work
1	George Orwell	1903-06-25	Prometheus Hall of Fame	1984	1984
2	Romain Gary	1914-05-21	Prix Goncourt	1956	The Roots of H
3	Edgar Allan Poe	1809-01-19	Edgar Award (posthumous)	1947	The Raven
4	Stefan Zweig	1881-11-28	Goethe Prize	1930	The World of '
5	Joseph Kessel	1898-02-10	Prix Goncourt	1927	L'Équipage
6	Jack London	1876-01-12	Pulitzer Prize	1903	The Call of the
7	Leo Tolstoy	1828-09-09	Nobel Peace Prize Nomine	1901	War and Peace
8	Jules Verne	1828-02-08	Legion of Honor	1870	Twenty Thousa
9	Fyodor Dostoev	1821-11-11	Russian Academy of Scien	1866	Crime and Pur
10	Victor Hugo	1802-02-26	Académie française Hono	1841	Les Misérables

FastAPI 0.1.0 OAS 3.1

/openapi.json

Authorize 

default

GET	/	Read Root	⌵
GET	/books/categories/	Get Books By Category	⌵ 
GET	/authors/awards/	Get Authors With Awards	⌵ 