Rapport de Compétences

SAE : Collecte Automatisée de Données Web

Aimad Hamdaoui

BUT Sciences des Données

22 juin 2025

Table des matières

1 Contexte du Projet

Le projet "SAE - Collecte Automatisée de Données Web" a pour objectif de développer un script Python capable d'automatiser le processus de collecte, de stockage et d'interrogation de données scientifiques. Le script interagit avec une API pour récupérer des informations sur des articles, les stocke dans une base de données SQLite et propose une interface en ligne de commande pour effectuer des recherches.

2 Niveau 1 : Traiter des données structurées

2.1 Mesurer l'importance de maîtriser la structure des données

Le projet met en évidence la transition cruciale entre les données semi-structurées de l'API (JSON) et un schéma relationnel structuré (SQL). Cette transformation a nécessité une conception réfléchie pour garantir la cohérence et l'efficacité des recherches. La création de deux tables normalisées, `articles` et `authors`, en est la preuve.

```sql
CREATE TABLE IF NOT EXISTS articles (
    id INTEGER PRIMARY KEY,
    title TEXT,
    year INTEGER,
    n_citation INTEGER,
    doc_type TEXT,
    publisher TEXT,
    ...
);

CREATE TABLE IF NOT EXISTS authors (
    author_id INTEGER PRIMARY KEY AUTOINCREMENT,
    article_id INTEGER,
    name TEXT,
    org TEXT,
    FOREIGN KEY (article_id) REFERENCES articles(id)
);
```

Listing 1 – Définition SQL de la structure de la base de données.

2.2 Comprendre les structures algorithmiques de base

Le script est organisé autour d'une structure algorithmique procédurale claire dans la fonction `main`, qui orchestre chaque étape du traitement : récupération, connexion, création, insertion, et interaction.

3 Niveau 2 : Automatiser le traitement de données multidimensionnelles

3.1 Réaliser le rôle central de l'entrepôt de données

La base de données `articles.db` agit comme un mini-entrepôt de données. Elle centralise et structure les informations extraites, les rendant disponibles pour l'analyse via le menu interactif. Ce dernier est l'outil qui interroge l'entrepôt, comme le montre la fonction de recherche par auteur qui joint les deux tables.

```python
def search_by_author(conn, author_name):
    sql = """
    SELECT a.id, a.title, a.year
    FROM articles a
    JOIN authors au ON a.id = au.article_id
    WHERE au.name LIKE ?
    """
    cursor = conn.cursor()
    cursor.execute(sql, ('%' + author_name + '%',))
    # ... Affiche les resultats
```

Listing 2 – Exemple de requête joignant les tables pour la recherche.

3.2 Identifier et résoudre les problèmes d'intégration de sources

Le projet est un cas d'école d'un processus ETL (Extract, Transform, Load). La principale difficulté réside dans la transformation des données brutes de l'API (source hétérogène, Figure **??**) en un format propre et structuré pour la base de données. Le script gère les champs potentiellement manquants pour assurer la robustesse de l'insertion.

3.3 Comprendre la nécessité de documenter et tester

La robustesse du script est assurée par une gestion systématique des erreurs (`try...except`) lors des opérations critiques (connexion, requêtes, saisie). Le code est commenté et accompagné d'une documentation externe, démontrant une compréhension de l'importance de la maintenabilité.

4 Démonstration Visuelle du Script

L'enchaînement des figures ci-dessous illustre une session utilisateur complète, de l'exécution du script à l'obtention d'informations détaillées sur un article.

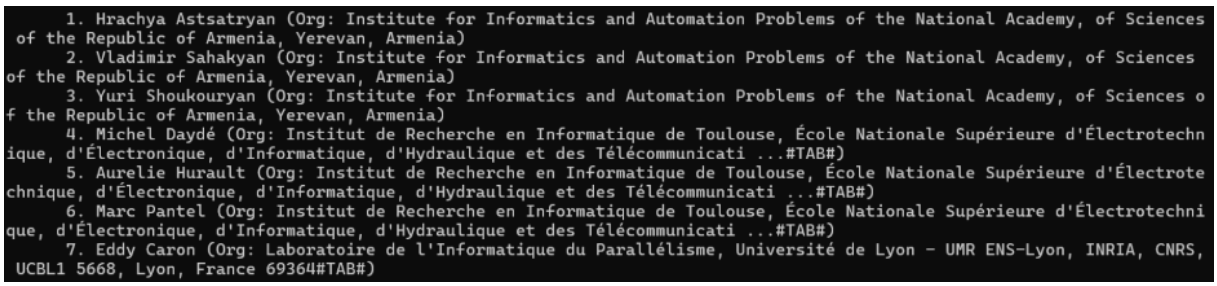

FIGURE 1 – Lancement du script : initialisation de la collecte de données depuis l'API et connexion à la base de données locale `articles.db`.

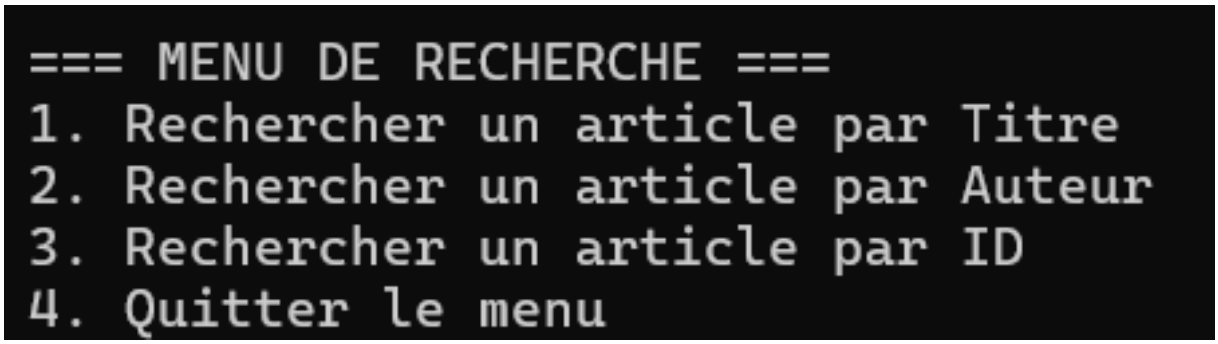

FIGURE 2 – Interaction avec le menu : l'utilisateur choisit la recherche par auteur (choix 2) et saisit le mot-clé "Caron".

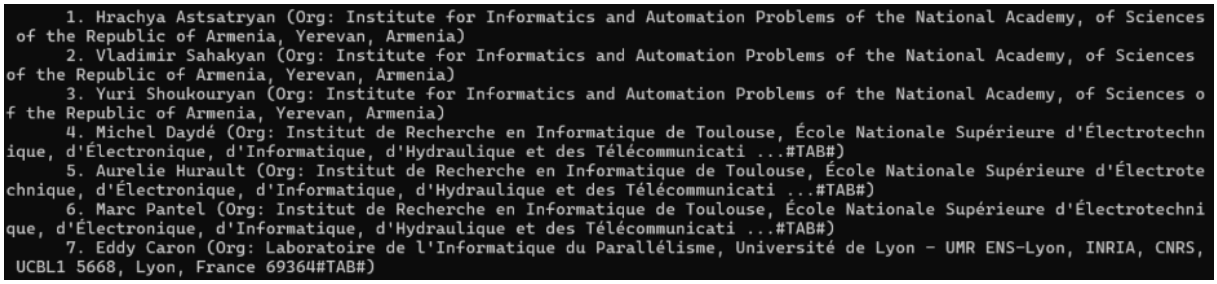

FIGURE 3 – Affichage des résultats : le script liste les articles correspondants, puis, après une nouvelle recherche par ID, affiche les détails complets d'un article, y compris la liste de ses auteurs.

5 Conclusion

Le projet de collecte automatisée a permis de mettre en pratique l'ensemble du cycle de vie de la donnée. Il valide solidement les compétences du **Niveau 1** (manipulation de structures, syntaxe) et démontre une maîtrise approfondie des concepts du **Niveau 2**.

Comme l'illustrent les captures d'écran, le projet a abouti à un outil fonctionnel qui automatise un flux de données complexe, depuis une source web hétérogène jusqu'à un entrepôt de données local et interrogeable. Cela prouve la capacité à concevoir et réaliser une solution logicielle complète, robuste et documentée pour un problème concret de traitement de données.