Classes SQLAlchemy

Pourquoi utiliser SQLAlchemy dans notre API ?

Lorsque vous créez une application qui interagit avec une base de données, comme notre API de films, vous avez deux choix pour gérer les données :

1. **Exécuter des requêtes SQL directement**
    - Vous devez établir une connexion avec SQLite.
    - Vous écrivez des requêtes SQL brutes pour insérer, récupérer et modifier des données.
    - Vous devez gérer manuellement les types de données (convertir entre les formats SQLite et Python).
    - Il faut se protéger contre les attaques par injection SQL.
2. **Utiliser un ORM (Object-Relational Mapper) comme SQLAlchemy**
    - SQLAlchemy permet d'interagir avec la base de données en manipulant des objets Python au lieu d'écrire du SQL brut.
    - Il simplifie la gestion des requêtes tout en garantissant la sécurité contre les injections SQL.
    - Il convertit automatiquement les données entre Python et SQLite.
    - Il facilite la migration de la base de données si on change de moteur SQL (ex: passer de SQLite à PostgreSQL).

Dans notre projet, SQLAlchemy joue un rôle clé dans la couche "Database Classes". Il agit comme **un intermédiaire entre notre API (FastAPI) et la base de données (SQLite)**, en traduisant les requêtes API en opérations sur la base de données tout en maintenant un code propre et sécurisé.

Pour utiliser SQLAlchemy, nous devons préalablement l'installer dans notre environnement virtuel :

```
pip install sqlalchemy
```