Projet : Pipeline de Données avec Dagster

Objectif

Ce projet met en place une pipeline de données ELT (Extract - Load - Transform) à l'aide de **Dagster**. Les données sont extraites depuis l'API **Deezer**, stockées dans une base **DuckDB**, transformées via **dbt**, puis visualisées avec **Streamlit**.

Stack Technique

- **Dagster** : Orchestrateur de pipelines (assets, jobs, partitions, schedules, sensors)
- **DuckDB** : Base de données analytique locale
- **dbt** : Transformation de données via SQL
- **Streamlit** : Visualisation web interactive
- **Plotly / Pandas** : Graphiques et analyses
- **Pytest** : Tests unitaires Python

Structure du projet

proj_data_final/

├── assets.py # Assets Dagster (fetch + save)

├── jobs.py # Définition des jobs

├── schedules.py # Schedules automatiques

├── sensors.py # Déclencheurs d'événements

├── definitions.py # Configuration globale Dagster

├── check_duckdb.py # Script de contrôle de la base DuckDB

├── dbt/ # Projet dbt

├── streamlit_app/ # Application de visualisation Streamlit

├── tests/ # Tests unitaires avec pytest

└── README.md

Installation et déploiement

1. Installation des dépendances

pip install -r requirements.txt

2. Lancer Dagster

dagster dev

3. Lancer un job partitionné

dagster job launch --job fetch_top_tracks_job --partition-key 2024-04-20

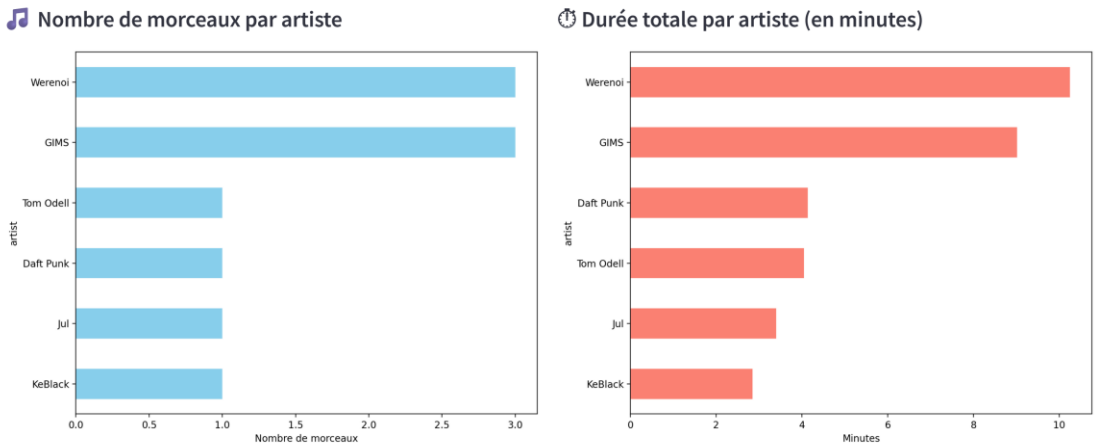
4. Lancer un job complet

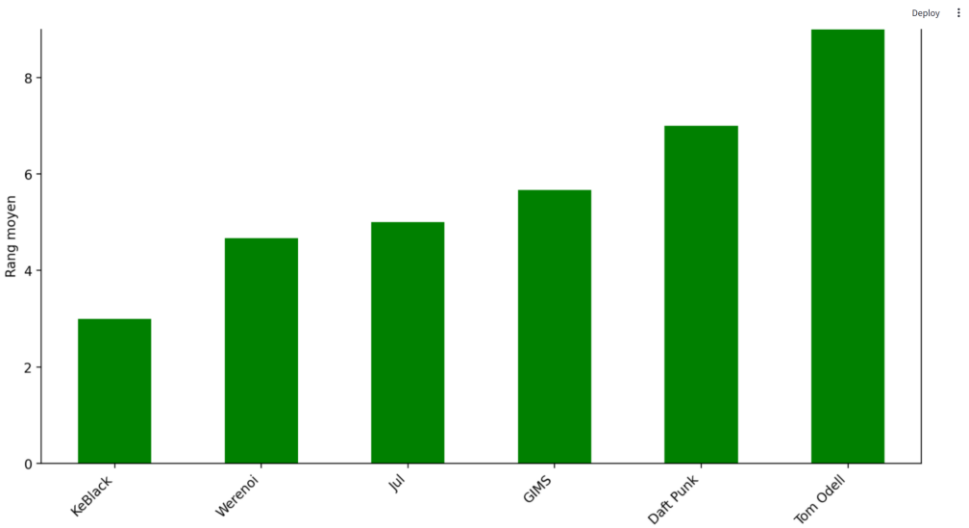
dagster job launch --job fetch_and_store_job

Visualisations avec Streamlit

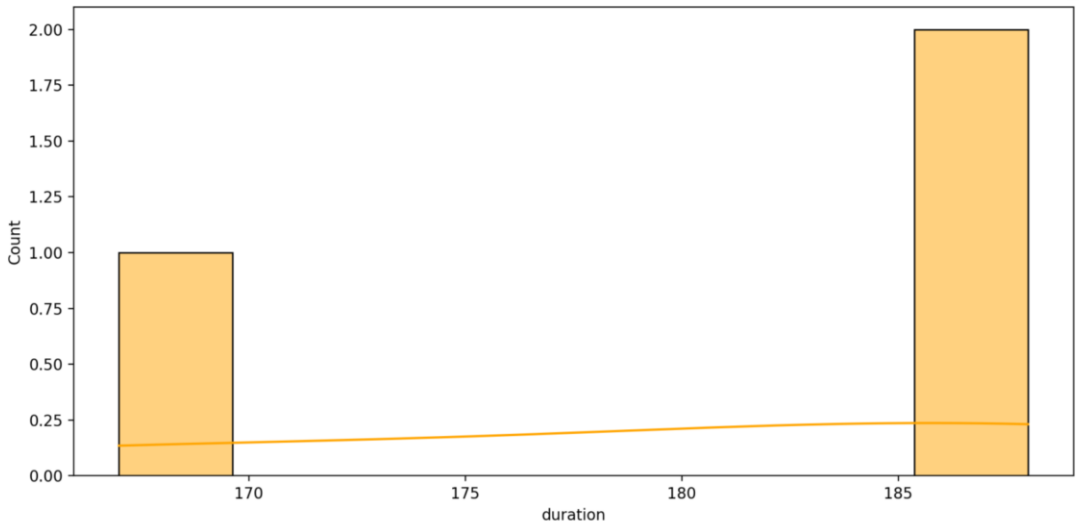
L'application Streamlit permet d'explorer les données extraites et transformées à partir de la base DuckDB :

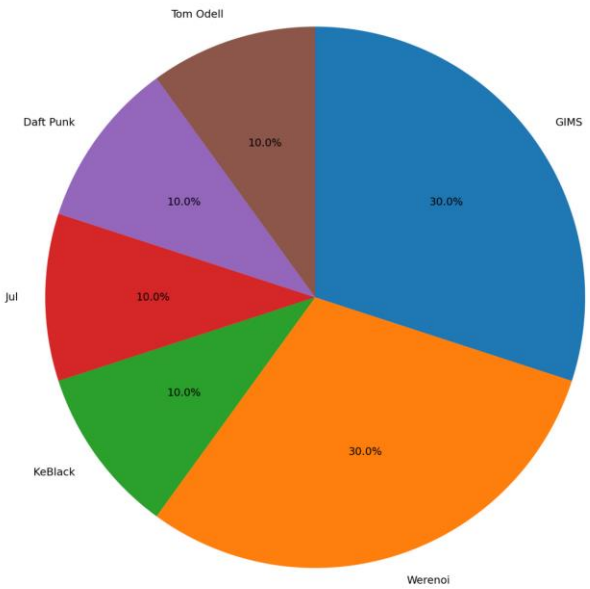
- **Top Tracks** : classements des morceaux du jour
- **Top Artists** : artistes les plus populaires
- **Tendances** : évolution des morceaux sur le temps (via top_tracks_partitioned)
- **Données nettoyées dbt** : affichage des modèles top_tracks_cleaned, etc.

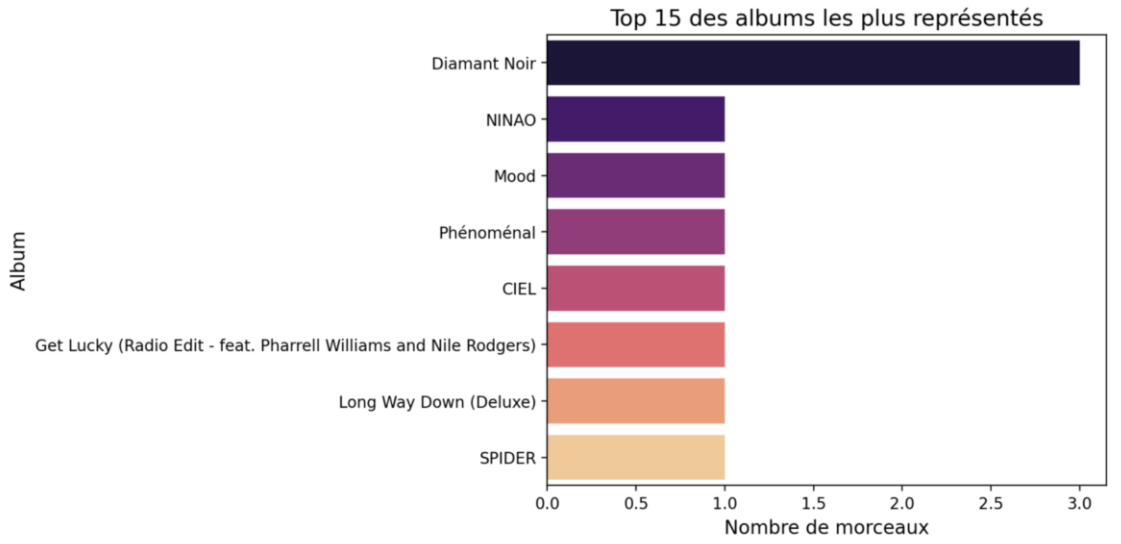


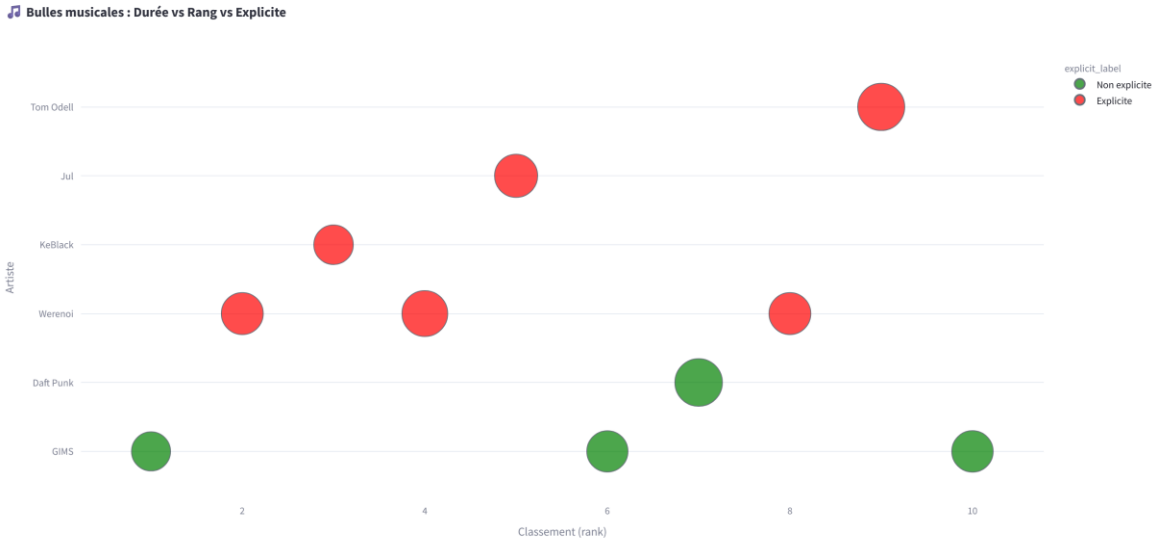


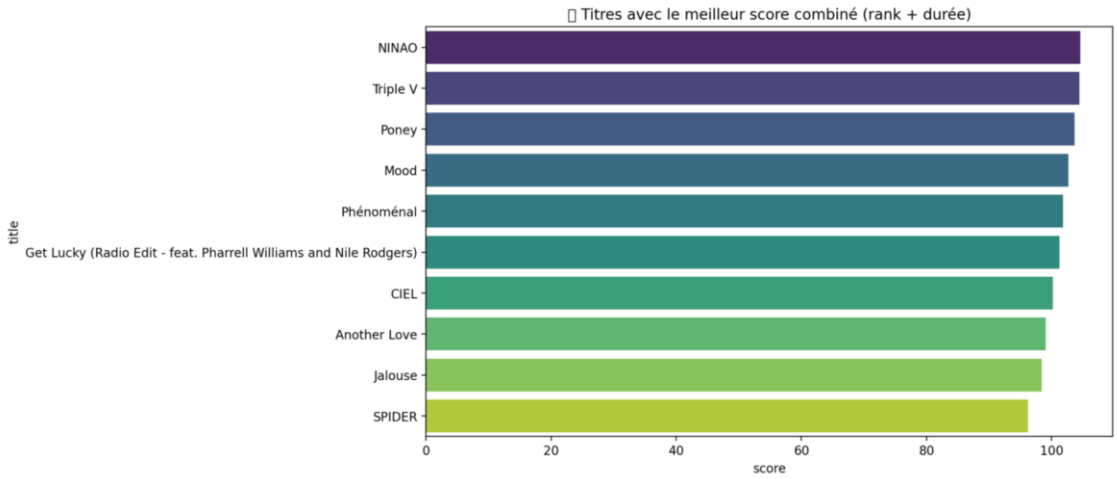
⏳ Durée des morceaux de GIMS






Top 15 des albums les plus représentés
Diamant Noir
NINAO
Mood
Phénoménal
CIEL
Get Lucky (Radio Edit - feat. Pharrell Williams and Nile Rodgers)
Long Way Down (Deluxe)
SPIDER
Album
0.0
0.5
1.0
1.5
2.0
2.5
3.0
Nombre de morceaux


Bulles musicales : Durée vs Rang vs Explicite
explicit_label
Non explicite
Explicite
Tom Odell
Jul
KeBlack
Werenoi
Daft Punk
GIMS
Artiste
2
4
6
8
10
Classement (rank)


Titres avec le meilleur score combiné (rank + durée)
NINAO
Triple V
Poney
Mood
Phénoménal
Get Lucky (Radio Edit - feat. Pharrell Williams and Nile Rodgers)
CIEL
Another Love
Jalouse
SPIDER
title
0
20
40
60
80
100
score

Lancer Streamlit :

streamlit run streamlit_app/app.py

Tests

Lancer les tests unitaires :

pytest tests/

Les tests couvrent la structure des DataFrames et la validité des types de données.

Monitoring

Des logs détaillés sont intégrés dans les assets via context.log.info() pour suivre :

- L'appel API et les erreurs potentielles
- L'écriture dans la base DuckDB
- Le passage des données entre assets

Exemple :

```
@asset(partitions_def=daily_partitions)
def fetch_top_tracks(context):
    date = context.partition_key
    context.log.info(f Fetching top tracks for {date}...")
```

Choix de conception

- **Dagster** permet de chaîner les étapes d'un pipeline en assets unitaires avec une orchestration flexible
- **Partitionnement journalier** : suivi temporel des données musicales
- **DuckDB** : base légère et performante, idéale pour l'analytique
- **dbt** : permet des transformations SQL claires, documentées et testables
- **Streamlit** : front-end rapide pour afficher et interpréter les résultats transformés

Auteur

Fadoua Belmokhtar
Master Ingénierie Data — SUPINFO Lyon