

Document Technique : Projet FakeSchoolData

Projet personnel réalisé par : Aymane RAMI

2025

Table des matières

Objectif du projet	3
Technologies utilisées	3
Structure du dépôt	3
Etape réalisées	4
1. Génération de données simulées (script Python generate_data.py).....	4
2. Création et alimentation de la base Snowflake	4
3. Transformation des données avec dbt.....	4
4. Analyse avec script Python analyze_results.py	5
5. Automatisation avec GitHub Actions	5
Suivi des prochaines étapes	6

Objectif du projet

Réaliser une pipeline de données simulées pour une fausse école, en utilisant :

- Un script Python pour générer des fichiers CSV d'étudiants, cours et résultats.
- Une base de données Snowflake pour stocker ces données.
- Des transformations de données avec dbt.
- Une analyse de résultats (statistiques et graphiques) via un autre script Python.
- L'automatisation avec GitHub Actions.

Technologies utilisées

- Python 3.10
- Bibliothèques : pandas, matplotlib, snowflake-connector-python, faker
- Base de données : Snowflake
- Outil de transformation : dbt
- CI/CD : GitHub Actions

Structure du dépôt

FakeSchoolData/

```
├── .github/
│   └── workflows/
│       └── run_analysis.yml
├── analyze_results.py
├── average_grades_chart.png
├── courses.csv
├── generate_data.py
├── log/
└── results.csv
```

```
├── students.csv
├── venv/
├── .gitignore
├── fakeschool_dbt/
│   └── models/
│       ├── average_grades.py
│       ├── schema.yml
│       └── top_students.py
```

Etape réalisées

1. Génération de données simulées (script Python generate_data.py)

- Utilisation de Faker pour générer :
 - 100 étudiants avec ID, nom, prénom.
 - 10 cours avec ID et nom aléatoire.
 - Des résultats (notes entre 0 et 20) aléatoires pour chaque étudiant dans plusieurs cours.
- Sauvegarde dans trois fichiers CSV : students.csv, courses.csv, results.csv.

2. Création et alimentation de la base Snowflake

- Connexion à Snowflake avec Python (snowflake-connector-python).
- Création du schéma RAW et des tables STUDENTS, COURSES, RESULTS.
- Chargement des CSV vers Snowflake avec une STAGE, puis COPY INTO pour insérer dans les tables.

3. Transformation des données avec dbt

- Initialisation d'un projet dbt nommé fakeschool_dbt.
- Configuration de la connexion à Snowflake dans profiles.yml.
- Création de modèles dans le dossier fakeschool_dbt/models/ :

- `average_grades.py` : moyenne des notes par cours.
 - `top_students.py` : top 5 étudiants avec meilleures moyennes.
 - `schema.yml` : documentation et validation de structure.
- Compilation et exécution de `dbt run` pour créer des tables ou vues transformées.

4. Analyse avec script Python `analyze_results.py`

- Connexion à Snowflake.
- Requête SQL pour charger les données enrichies.
- Calculs statistiques :
 - Moyenne, médiane, écart-type des notes par cours.
 - Nombre d'étudiants par cours.
 - Notes min/max par étudiant.
 - Top 5 étudiants par moyenne.
- Visualisations avec matplotlib :
 - Histogramme de distribution des notes.
 - Barres de répartition par tranches (0-5, 6-10, etc.).
- Génération d'un graphique `average_grades_chart.png`.

5. Automatisation avec GitHub Actions

- Workflow `.github/workflows/run_analysis.yml` configuré pour :
 - Exécuter automatiquement `analyze_results.py` à chaque push sur la branche `main`.
 - Lancer le script tous les jours à 8h UTC via une planification `cron`.
- Configuration incluant :
 - Checkout du dépôt.
 - Installation de Python et des dépendances.
 - Passage sécurisé de la variable `SNOWFLAKE_PASSWORD` via GitHub Secrets.
 - Exécution du script dans le bon répertoire.

- Les graphiques générés sont sauvegardés et uploadés comme artifacts dans GitHub Actions pour consultation.

Suivi des prochaines étapes

- Ajouter l'export de résultats complémentaires (ex. fichiers CSV) si besoin.
- Étendre dbt avec des tests ou documentation automatique.
- Possibilité d'envoyer un rapport par mail automatiquement.
- Améliorer la visualisation et le reporting.