AI Factory - Documentation Technique

AI Factory - Medical Fraud Detection Pipeline

Documentation Technique Complete

Version: 1.0.0 **Date:** Fevrier 2026 **Auteur:** AI Factory Team

---

Table des Matieres

---

1. Vue d'Ensemble

1.1 Objectif du Projet

AI Factory est un pipeline de donnees concu pour la detection de fraudes medicales. Il collecte, transforme et analyse des donnees publiques du systeme de sante americain (CMS/HHS) pour identifier des patterns suspects.

1.2 Cas d'Usage

- Detection de providers exclus (LEIE) facturant Medicare/Medicaid
- Analyse des paiements pharmaceutiques aux medecins (Open Payments)
- Identification de prescriptions anormales (Part D Prescribers)
- Correlation entre paiements et comportements de prescription

1.3 Stack Technologique

Composant	Technologie	Role
Orchestration	Apache Airflow 3.1.6	Scheduling et orchestration des DAGs
Stockage Object	AWS S3	Data Lake (Bronze layer)
Data Warehouse	Snowflake	Stockage et transformation
Transformation	dbt 1.9.0	Modelisation et documentation
Conteneurisation	Docker Compose	Environnement local
Infrastructure	Terraform	Provisioning AWS/Snowflake

---

2. Architecture

2.1 Architecture Medallion

Le projet suit l’architecture Medallion (Bronze/Silver/Gold) :

```
┌──────────────────────────────────────────────────────────────
│                         SOURCES EXTERNES
│
├────────────┬────────────┬────────────┬────────────┬──────────
│    CMS     │    HHS     │    LEIE    │   NPPES    │  Snowflake
│
│   Data     │   Open     │ Exclusion  │  Provider  │
Marketplace│
│            │  Payments  │    List    │  Registry  │
│
└────────────┴────────────┴────────────┴────────────┴──────────
       │            │            │            │            │
       │            │            │            │            │
       ▼            ▼            ▼            ▼            ▼
┌──────────────────────────────────────────────────────────────
│                         INGESTION (Airflow)
│
│  ┌──────────────────────────────────────────────────────────────┐
│
│  │  Download DAGs (8 sources) → Conversion Parquet → S3 Upload │
│
│  └──────────────────────────────────────────────────────────────┘
│
└──────────────────────────────────────────────────────────────
```

```
                                   │
                                   ▼
┌─────────────────────────────────────────────────────────────────────
│                         AWS S3 - DATA LAKE
│
│  ┌─────────────────────────────────────────────────────────────┐
│
│  │  s3://ai-factory-bckt/bronze/                               │
│
│  │    ├── leie/                                                │
│
│  │    ├── medicare_hospital_spending/                          │
│
│  │    ├── open_payments/                                       │
│
│  │    ├── provider_information/                                │
│
│  │    ├── longterm_care_hospital/                              │
│
│  │    ├── hospice/                                             │
│
│  │    ├── home_health_care/                                    │
│
│  │    └── medicare_part_d_prescribers/                         │
│
│  └─────────────────────────────────────────────────────────────┘
│
└─────────────────────────────────────────────────────────────────────
                                   │
                                   ▼
┌─────────────────────────────────────────────────────────────────────
│                         SNOWFLAKE
│
│  ┌───────────────────┐  ┌───────────────────┐  ┌─────────────────┐
│
│  │      BRONZE       │  │      SILVER       │  │      GOLD       │
│
│  │  (Raw Tables)     │→ │  (Cleaned/Typed)  │→ │  (Analytics)    │
│
│  │  10 tables        │  │  dbt models       │  │  Aggregations   │
│
│  └───────────────────┘  └───────────────────┘  └─────────────────┘
│
│
│
│  ┌───────────────────────────────────────────────────────────────┐
│
│  │  NPPES Marketplace (External)                                  │
│
│  │  AFFINE_NPPES_PROVIDER_DATA.REF_DW                            │
│
```

```
│  └────────────────────────────────────────────────────────────────┘
│
└──────────────────────────────────────────────────────────────────────
                                   │
                                   ▼
┌──────────────────────────────────────────────────────────────────────
│                         ANALYTICS / ML
│
│  ┌──────────────────────────────────────────────────────────────┐
│
│  │  Fraud Detection Models │ Dashboards │ Reporting            │
│
│  └──────────────────────────────────────────────────────────────┘
│
└──────────────────────────────────────────────────────────────────────
```

2.2 Flux de Donnees

```
1. master_dag (Orchestrateur)
   │
   ├── init_snowflake_environment
   │   └── Cree Warehouse, Database, Schemas, Stage S3
   │
   ├── Download DAGs (8 en parallele)
   │   ├── d0_leie_download
   │   ├── d1_medicare_hospital_spending_download
   │   ├── d2_open_payments_download
   │   ├── d3_provider_information_download
   │   ├── d4_longterm_care_hospital_download
   │   ├── d5_hospice_download
   │   ├── d6_home_health_care_download
   │   └── d7_medicare_part_d_prescribers_download
   │
   └── load_bronze_tables
       └── COPY INTO depuis S3 vers Snowflake Bronze
```

---

3. Infrastructure

3.1 Structure du Projet

```
ai_factory/
├── dags/                          # DAGs Airflow
│   ├── master_dag.py              # Orchestrateur principal
│   ├── init_snowflake_environment.py
│   ├── load_bronze_tables.py
│   └── d*_*_download.py           # 8 DAGs de telechargement
│
```

```
├── dbt/                           # Projet dbt
│   ├── dbt_project.yml            # Configuration dbt
│   ├── profiles.yml               # Connexion Snowflake
│   └── models/
│       ├── staging/               # Vues staging (14 modeles)
│       └── silver/                # Transformations (a venir)
│
├── snowflake/                     # Scripts SQL
│   ├── init_warehouse.sql
│   ├── init_schemas.sql
│   ├── init_s3_stage.sql
│   ├── keys/                      # Cle privee RSA
│   └── bronze/                    # Scripts Bronze
│       ├── create_tables.sql
│       ├── load_tables.sql
│       └── create_open_payments_*.sql
│
├── terraform/                     # Infrastructure as Code
│   ├── main.tf
│   ├── s3.tf
│   └── snowflake.tf
│
├── config/                        # Configuration Airflow
│   └── airflow.cfg
│
├── docker-compose.yaml            # Environnement Docker
├── Dockerfile
├── requirements.txt
└── .env                           # Variables d'environnement
```

3.2 Docker Compose

Services deployes :

Service	Port	Description
airflow-apiserver	8080	Interface web Airflow
airflow-scheduler	-	Planificateur de taches
airflow-worker	-	Executeur Celery
airflow-triggerer	-	Gestion des triggers
airflow-dag-processor	-	Traitement des DAGs
dbt-docs	8085	Documentation dbt
postgres	5432	Base Airflow
redis	6379	Broker Celery

3.3 Configuration

requirements.txt :

```
apache-airflow==3.1.6
apache-airflow-providers-amazon==8.1.0
apache-airflow-providers-http==4.11.0
apache-airflow-providers-snowflake==6.7.0
snowflake-connector-python>=2.8.0
dbt-snowflake==1.9.0
pandas==2.2.0
boto3==1.34.0
requests==2.32.3
```

Variables d'environnement (.env) :

```
AIRFLOW_UID=501
SNOWFLAKE_ACCOUNT=rdsnbdu-gbc86569
SNOWFLAKE_USER=FISHER
```

---

4. Sources de Donnees

4.1 Sources CMS/HHS (Bronze Layer)

Source	Description	Volume	Frequence
LEIE	Providers exclus de Medicare/Medicaid	~75K	Mensuel
Medicare Hospital Spending	Depenses hospitalieres par beneficiaire	~500K	Annuel
Open Payments General	Paiements pharma aux medecins	~15M	Annuel
Open Payments Research	Paiements pour recherche clinique	~1M	Annuel
Open Payments Ownership	Participations des medecins	~5K	Annuel
Provider Information	Infos nursing homes (ratings)	~15K	Mensuel
Long-Term Care Hospital	Hopitaux long sejour	~500	Annuel
Hospice	Etablissements de soins palliatifs	~5K	Annuel
Home Health Care	Agences de soins a domicile	~12K	Annuel
Medicare Part D Prescribers	Prescriptions Part D par medecin	~9M	Annuel

Total : ~25M+ lignes

4.2 Source NPPES (Snowflake Marketplace)

Table	Description	Volume
DIM_PROVIDER	Registre NPI (tous providers)	~8M
DIM_PROVIDER_ADDRESS	Adresses des providers	~16M
DIM_PROVIDER_TAXONOMY	Specialites des providers	~12M
REF_TAXONOMY_CODE	Reference des codes taxonomie	~1K

4.3 Cles de Jointure

Le champ **NPI** (National Provider Identifier) permet de joindre : - LEIE.NPI - Open Payments.Covered_Recipient_NPI - Medicare Part D.PRSCRBR_NPI - Provider Information (via Federal Provider Number) - NPPES.NPI

---

5. Pipeline Airflow

5.1 master_dag.py

Orchestrateur principal qui coordonne l'ensemble du pipeline.

```
# Configuration
dag_id = 'master_dag'
schedule = None  # Declenchement manuel
catchup = False

# Flux
init_snowflake_environment
    │
    ▼
start_downloads (EmptyOperator)
    │
    ├── trigger d0_leie_download
    ├── trigger d1_medicare_hospital_spending_download
    ├── trigger d2_open_payments_download
    ├── trigger d3_provider_information_download
    ├── trigger d4_longterm_care_hospital_download
    ├── trigger d5_hospice_download
    ├── trigger d6_home_health_care_download
    └── trigger d7_medicare_part_d_prescribers_download
    │
    ▼
all_downloads_complete (EmptyOperator)
    │
    ▼
load_bronze_tables
    │
    ▼
pipeline_complete (EmptyOperator)
```

5.2 init_snowflake_environment.py

Initialise l’infrastructure Snowflake :

1. **Warehouse** : AI_FACTORY_WH (X-Small, auto-suspend 300s)
2. **Database** : AI_FACTORY_DB
3. **Schemas** : RAW_DATA, BRONZE, SILVER, GOLD
4. **File Format** : PARQUET_FORMAT
5. **Stage** : BRONZE_S3_STAGE (pointe vers s3://ai-factory-bckt/bronze/)

5.3 Download DAGs (d0 - d7)

Chaque DAG de telechargement suit le pattern :

```python
@task
def download_file():
    # Telecharge depuis l'API CMS/HHS
    response = requests.get(url)
    # Sauvegarde en local

@task
def convert_to_parquet():
    # Convertit CSV/ZIP en Parquet
    df = pd.read_csv(file)
    df.to_parquet(output, compression='snappy')

@task
def upload_to_s3():
    # Upload vers S3 bronze/
    s3_client.upload_file(file, bucket, key)

download >> convert >> upload
```

Gestion des gros fichiers (Open Payments) : - Traitement par chunks de 500K lignes - Evite les erreurs OOM - Fichiers Parquet multiples

5.4 load_bronze_tables.py

Charge les donnees S3 dans Snowflake Bronze.

Tables standard (7 tables) :

```sql
COPY INTO table_name
FROM @BRONZE_S3_STAGE/folder/
FILE_FORMAT = PARQUET_FORMAT
MATCH_BY_COLUMN_NAME = CASE_INSENSITIVE;
```

Tables Open Payments (3 tables) : Utilise `INFER_SCHEMA` pour detection automatique du schema :

```
def create_open_payments_table(table_name, file_path):
    hook = SnowflakeHook(snowflake_conn_id='snowflake_default')
    sql = f"""
    CREATE OR REPLACE TABLE {table_name}
    USING TEMPLATE (
        SELECT ARRAY_AGG(OBJECT_CONSTRUCT(*))
        FROM TABLE(INFER_SCHEMA(
            LOCATION => '@BRONZE_S3_STAGE/open_payments/{file_path}',
            FILE_FORMAT => 'PARQUET_FORMAT'
        ))
    );
    """
    hook.run(sql)
```

Note : `INFER_SCHEMA` n'est pas supporte par SnowflakeSqlApiOperator, d'ou l'utilisation de PythonOperator + SnowflakeHook.

---

6. Snowflake

6.1 Architecture

```
AI_FACTORY_DB
├── RAW_DATA (schema)       # Reserve pour donnees brutes futures
├── BRONZE (schema)         # Tables chargees depuis S3
│   ├── LEIE
│   ├── MEDICARE_HOSPITAL_SPENDING
│   ├── OPEN_PAYMENTS_GENERAL
│   ├── OPEN_PAYMENTS_RESEARCH
│   ├── OPEN_PAYMENTS_OWNERSHIP
│   ├── PROVIDER_INFORMATION
│   ├── LONGTERM_CARE_HOSPITAL
│   ├── HOSPICE
│   ├── HOME_HEALTH_CARE
│   └── MEDICARE_PART_D_PRESCRIBERS
├── SILVER (schema)         # Tables transformees (dbt)
└── GOLD (schema)           # Tables analytiques
```

6.2 Connexion

Methode d'authentification : Key Pair (RSA)

```
# Connexion Airflow
snowflake_conn_id = 'snowflake_default'
extra = {
    "account": "rdsnbdu-gbc86569",
    "warehouse": "AI_FACTORY_WH",
    "database": "AI_FACTORY_DB",
    "role": "ACCOUNTADMIN",
    "private_key_file": "/opt/airflow/snowflake/keys/rsa_key.p8"
}
```

6.3 Integration S3

```sql
-- Storage Integration (cree via Terraform)
CREATE STORAGE INTEGRATION S3_INTEGRATION
  TYPE = EXTERNAL_STAGE
  STORAGE_PROVIDER = 'S3'
  ENABLED = TRUE
  STORAGE_AWS_ROLE_ARN = 'arn:aws:iam::xxx:role/snowflake-s3-role'
  STORAGE_ALLOWED_LOCATIONS = ('s3://ai-factory-bckt/');

-- Stage
CREATE STAGE BRONZE_S3_STAGE
  STORAGE_INTEGRATION = S3_INTEGRATION
  URL = 's3://ai-factory-bckt/bronze/'
  FILE_FORMAT = PARQUET_FORMAT;
```

6.4 Scripts SQL

snowflake/init_warehouse.sql :

```sql
CREATE WAREHOUSE IF NOT EXISTS AI_FACTORY_WH
    WAREHOUSE_SIZE = 'X-SMALL'
    AUTO_SUSPEND = 300
    AUTO_RESUME = TRUE
    INITIALLY_SUSPENDED = TRUE;
```

snowflake/init_schemas.sql :

```sql
CREATE DATABASE IF NOT EXISTS AI_FACTORY_DB;
CREATE SCHEMA IF NOT EXISTS AI_FACTORY_DB.RAW_DATA;
CREATE SCHEMA IF NOT EXISTS AI_FACTORY_DB.BRONZE;
CREATE SCHEMA IF NOT EXISTS AI_FACTORY_DB.SILVER;
CREATE SCHEMA IF NOT EXISTS AI_FACTORY_DB.GOLD;
```

snowflake/bronze/create_tables.sql :

```sql
CREATE OR REPLACE TABLE LEIE (
    LASTNAME VARCHAR,
    FIRSTNAME VARCHAR,
    ...
    _LOAD_TIMESTAMP TIMESTAMP_NTZ DEFAULT CURRENT_TIMESTAMP(),
    _SOURCE_FILE VARCHAR
);
```

---

7. dbt - Transformation

7.1 Configuration

dbt_project.yml :

```
name: 'ai_factory'
version: '1.0.0'
config-version: 2
profile: 'ai_factory'

models:
  ai_factory:
    staging:
      +schema: staging
      +materialized: view
    silver:
      +schema: silver
      +materialized: table
```

profiles.yml :

```
ai_factory:
  target: prod
  outputs:
    prod:
      type: snowflake
      account: "{{ env_var('SNOWFLAKE_ACCOUNT') }}"
      user: "{{ env_var('SNOWFLAKE_USER') }}"
      private_key_path: /opt/airflow/snowflake/keys/rsa_key.p8
      role: ACCOUNTADMIN
      database: AI_FACTORY_DB
      warehouse: AI_FACTORY_WH
      schema: SILVER
      threads: 4
```

7.2 Modeles Staging

14 modeles staging (vues 1:1 sur les sources) :

Modele	Source
stg_leie	bronze.LEIE
stg_medicare_hospital_spending	bronze.MEDICARE_HOSPITAL_SPENDING
stg_open_payments_general	bronze.OPEN_PAYMENTS_GENERAL
stg_open_payments_research	bronze.OPEN_PAYMENTS_RESEARCH
stg_open_payments_ownership	bronze.OPEN_PAYMENTS_OWNERSHIP
stg_provider_information	bronze.PROVIDER_INFORMATION
stg_longterm_care_hospital	bronze.LONGTERM_CARE_HOSPITAL

Modele	Source
stg_hospice	bronze.HOSPICE
stg_home_health_care	bronze.HOME_HEALTH_CARE
stg_medicare_part_d_prescribers	bronze.MEDICARE_PART_D_PRESCRIBERS
stg_nppes_provider	nppes.DIM_PROVIDER
stg_nppes_provider_address	nppes.DIM_PROVIDER_ADDRESS
stg_nppes_provider_taxonomy	nppes.DIM_PROVIDER_TAXONOMY
stg_nppes_taxonomy_code	nppes.REF_TAXONOMY_CODE

Exemple de modele staging :

```
-- models/staging/stg_leie.sql
with source as (
    select * from {{ source('bronze', 'LEIE') }}
)

select
    *,
    current_timestamp() as _loaded_at
from source
```

7.3 Documentation

La documentation dbt est generee automatiquement et accessible sur http://localhost:8085.

Elle inclut : - Graphe de lineage (DAG visuel) - Description de toutes les sources - Description de tous les modeles - Documentation des colonnes

Commandes :

```
# Generer la doc
dbt docs generate --profiles-dir .

# Servir la doc
dbt docs serve --port 8085 --host 0.0.0.0 --profiles-dir .
```

---

8. Deploiement

8.1 Prerequis

- Docker & Docker Compose
- AWS CLI configure
- Compte Snowflake avec Storage Integration

8.2 Installation

```
# 1. Cloner le repo
git clone https://github.com/Mo-oM-1/ai_factory.git
cd ai_factory

# 2. Configurer .env
echo "AIRFLOW_UID=$(id -u)" >> .env
echo "SNOWFLAKE_ACCOUNT=xxx" >> .env
echo "SNOWFLAKE_USER=xxx" >> .env

# 3. Generer la cle RSA Snowflake
openssl genrsa 2048 | openssl pkcs8 -topk8 -nocrypt -out
        snowflake/keys/rsa_key.p8
openssl rsa -in snowflake/keys/rsa_key.p8 -pubout -out
        snowflake/keys/rsa_key.pub

# 4. Configurer la cle publique dans Snowflake
# ALTER USER xxx SET RSA_PUBLIC_KEY='...';

# 5. Demarrer les services
docker compose build
docker compose up -d

# 6. Creer la connexion Airflow
docker exec ai_factory-airflow-worker-1 airflow connections add
        snowflake_default \
    --conn-type snowflake \
    --conn-login FISHER \
    --conn-extra '{"account": "xxx", "warehouse": "AI_FACTORY_WH",
        "database": "AI_FACTORY_DB", "role": "ACCOUNTADMIN",
        "private_key_file": "/opt/airflow/snowflake/keys/rsa_key.p8"}'
```

8.3 Lancement du Pipeline

```
# Via CLI
docker exec ai_factory-airflow-worker-1 airflow dags trigger
        master_dag

# Ou via l'interface web
# http://localhost:8080 → DAGs → master_dag → Trigger
```

8.4 URLs

Service	URL
Airflow UI	http://localhost:8080
dbt Docs	http://localhost:8085
Flower (Celery)	http://localhost:5555

9. Monitoring

9.1 Logs Airflow

```bash
# Voir les logs d'un DAG
docker exec ai_factory-airflow-worker-1 airflow tasks logs master_dag <task_id> <date>

# Logs du worker
docker logs ai_factory-airflow-worker-1 -f
```

9.2 Metriques Snowflake

```sql
-- Historique des requetes
SELECT * FROM SNOWFLAKE.ACCOUNT_USAGE.QUERY_HISTORY
WHERE DATABASE_NAME = 'AI_FACTORY_DB'
ORDER BY START_TIME DESC
LIMIT 100;

-- Credits consommes
SELECT * FROM SNOWFLAKE.ACCOUNT_USAGE.WAREHOUSE_METERING_HISTORY
WHERE WAREHOUSE_NAME = 'AI_FACTORY_WH'
ORDER BY START_TIME DESC;
```

9.3 Verification des Donnees

```sql
-- Comptage des tables Bronze
SELECT
    TABLE_NAME,
    ROW_COUNT
FROM AI_FACTORY_DB.INFORMATION_SCHEMA.TABLES
WHERE TABLE_SCHEMA = 'BRONZE'
ORDER BY TABLE_NAME;
```

Annexes

A. Glossaire

Terme	Definition
NPI	National Provider Identifier - Identifiant unique des providers
LEIE	List of Excluded Individuals/Entities
CMS	Centers for Medicare & Medicaid Services
HHS	Department of Health and Human Services
CCN	CMS Certification Number

Terme	Definition
Part D	Medicare Prescription Drug Benefit

B. Ressources

- Documentation Airflow
- Documentation Snowflake
- Documentation dbt
- CMS Data Portal
- Open Payments

C. Releases

Version	Date	Description
v1.0.0	Feb 2026	Bronze Layer Complete - Pipeline fonctionnel de bout en bout

Document genere automatiquement - AI Factory v1.0.0