# Data Sanitizer (POO) — Notice d’utilisation

Dernière mise à jour : 2025‑08‑26

## 1. Objet du projet

Data Sanitizer fournit une chaîne fiable pour importer, normaliser et écrire des historiques de prix d’actions dans PostgreSQL. La refonte POO vise : - **stabilité/fiabilité** des données (flux idempotent, upserts, vues de compatibilité); - **intégration simple** par d’autres projets (ex. *MarketAdvisor*) via une API CLI et des conventions de schéma stables.

## 2. Prérequis

* **Python** 3.12+
* **PostgreSQL** 14+ (local ou distant)
* Accès réseau vers *Yahoo Finance* (utilisé par yfinance).

## 3. Installation

python -m venv .venv  
source .venv/bin/activate  
pip install -e .

## 4. Configuration

Créer un fichier **.env** à la racine :

DATABASE\_URL=postgresql://pea\_user:pea\_password@127.0.0.1:5432/pea\_db  
DS\_PRICE\_READ\_VIEW=v\_prices\_compat  
DS\_PRICE\_WRITE\_TABLE=equity\_prices  
DS\_PRICE\_DATE\_COL=price\_date  
LOG\_LEVEL=INFO  
REQUEST\_PAUSE\_S=0.6  
YF\_TIMEOUT\_S=10

Le chargement de .env est **automatique** (aucun export requis).

### 4.1 Objets de base attendus côté DB

* Table **equities** : univers des titres (incluant isin, symbol, is\_active, is\_valid, etc.).
* Table **equity\_prices** : historique des prix (clé : (isin, symbol, price\_date)).
* Vue **v\_prices\_compat** : vue de lecture (compatibilité).

*Exemple SQL idempotent pour la vue :*

DROP VIEW IF EXISTS public.v\_prices\_compat;  
CREATE VIEW public.v\_prices\_compat (  
 isin, symbol, price\_date, open\_price, close\_price, high\_price, low\_price, volume, adj\_close  
) AS  
SELECT isin, symbol, price\_date, open\_price, close\_price, high\_price, low\_price, volume, adj\_close  
FROM public.equity\_prices;

## 5. Utilisation — CLI

Le binaire **data-sanitizer** charge .env automatiquement et expose une **commande unique**.

### 5.1 Vérifier la connexion DB

data-sanitizer --check-db

Affiche un diagnostic ✅/❌ (hôte, base, user).

### 5.2 Mettre à jour les prix

# Lecture seule (aucun write)  
data-sanitizer --dry-run --since 2024-01-01 -o SAN -o AIR.PA  
  
# Écriture effective (par défaut), alias explicite possible  
data-sanitizer --write --limit 10 --sleep 0.3

**Options :** - --since YYYY-MM-DD : date de départ (sinon reprend à la dernière date connue). - --limit N : nombre max de titres à traiter. - --only / -o : filtrer par ISIN/SYMBOL (répétable). - --sleep S : pause entre tickers (throttle Yahoo si besoin). - --dry-run : ne rien écrire en base. - --write : alias explicite pour écrire (optionnel, l’écriture est le mode par défaut si --dry-run est absent).

### 5.3 Journalisation & progression

Le service affiche des lignes de progression :

[1/10] AIR.PA (ISIN …) … résolution du ticker  
[1/10] AIR.PA (ISIN …) … téléchargement (AIR.PA, since=2024-01-01)  
[1/10] AIR.PA (ISIN …) → ✅ écriture : 254 barres (total=6200, 1y=220)

Ajuster la verbosité via LOG\_LEVEL dans .env.

## 6. Intégration projet tiers (ex. MarketAdvisor)

Deux approches :

### 6.1 Intégration SQL directe (recommandée)

Lire depuis **v\_prices\_compat** (contrat stable) :

SELECT \* FROM v\_prices\_compat WHERE symbol = 'AIR.PA' AND price\_date >= '2024-01-01';

Avantages : couplage minimal, pas de dépendance Python, compatibilité ascendante.

### 6.2 Intégration Python (en dépendance)

Dans le projet client :

# pyproject.toml  
[project]  
dependencies = [  
 "data-sanitizer @ git+ssh://git@github.com/mnorodine/data\_sanitizer\_poo.git#egg=data-sanitizer",  
]

Et appel en sous-process ou module :

# sous-process  
python -m venv .venv && source .venv/bin/activate  
pip install "data-sanitizer @ git+ssh://git@github.com/mnorodine/data\_sanitizer\_poo.git#egg=data-sanitizer"  
DATA\_SANITIZER\_OPTS="--write --since 2025-01-01" data-sanitizer

Pour un couplage plus fort, exposer ultérieurement un mini-API Python (ports) est possible.

## 7. Modèle de données (résumé)

**equity\_prices** - isin (text), symbol (text), price\_date (date) - open\_price, high\_price, low\_price, close\_price, adj\_close (numeric) - volume (bigint) - Index : (isin, price\_date), (symbol, price\_date); clé unique (isin, symbol, price\_date).

**equities** - isin (PK), symbol, is\_active, is\_valid, etc. (attributs de suivi et qualité)

**v\_prices\_compat** - Projection lecture‐seule identique à equity\_prices pour compatibilité.

## 8. Bonnes pratiques d’exploitation

* Lancer d’abord en --dry-run sur un échantillon (--limit 5).
* Programmer un job périodique (ex. cron / Actions) avec --sleep 0.3 pour limiter le throttling Yahoo.
* Surveiller la volumétrie : index présents, vacuum/analyse réguliers.

## 9. Dépannage

* **Pas de logs / très lent** : ajouter --limit 5 --sleep 0.3. Yahoo peut limiter.
* **Connexion DB KO** : vérifier DATABASE\_URL dans .env ; tester data-sanitizer --check-db.
* **Aucun prix** : certains tickers Yahoo peuvent être déslistés ou renommés. Vérifier la résolution de ticker et le mapping equities.

## 10. CI & Qualité

* **GitHub Actions** : lint (ruff) + tests (pytest).
* **Ruff** : respect PEP8, imports, etc.
* **Tests** : smoke tests fournis, à compléter (mocks provider/DB) selon besoins.

## 11. Roadmap courte

* Option --progress bar (barre compacte)
* Endpoint Python (service) minimal pour intégration directe MarketAdvisor
* Stubs/implémentations pour recompute\_counts() et update\_bounds() selon usage réel

© 2025 Data Sanitizer POO — Tous droits réservés.