Guide d'utilisation - Script DHIS2 Excel

Table des matières

Changements récents

Voici les modifications apportées récemment au dépôt (à prendre en compte lors de l'utilisation) :

- Ajout de `/.env.example` : modèle commenté contenant des placeholders pour toutes les variables de configuration. Copier ce fichier en `.env` et remplir localement (ne pas committer).
- Ajout de `/.gitignore` : ignore désormais `.env`, `output/`, `data/*.csv` et autres fichiers temporaires/IDE.
- Intégration des étapes de téléchargement directement dans `pivot_tracked_and_stage.py` : la logique de `download_tracked.py` et `download.py` a été fusionnée pour permettre au script de récupérer automatiquement les CSV si nécessaire.
- Intégration des étapes de téléchargement directement dans `pivot_tracked_and_stage.py` : la logique de `download_tracked.py` et `download.py` a été fusionnée pour permettre au script de récupérer automatiquement les CSV si nécessaire.
- Nouvelles options CLI pour contrôler le téléchargement : `--skip-download`, `--only-download`, et `--only-pivot`.
- Nouveau flag CLI `--apply-mapping` : applique les correspondances (par défaut `data/correspondance.csv` et `data/structure.xlsx`) sur CHAQUE onglet de l'Excel final généré. Options associées : `--mapping-correspondance`, `--mapping-structure`, `--mapping-col-a1`, `--mapping-col-a2`, `--mapping-col-b1`, `--mapping-col-b2`, `--mapping-log-file`, `--mapping-preview`.
- Ajout de vérifications préalables et messages d'erreur plus clairs pour les cas où les fichiers d'entrée sont manquants (évite les `FileNotFoundError`

non expliquées).
- Mise à jour de la documentation et masquage des tokens sensibles dans le README ; recommandations de sécurité ajoutées (révoquer les tokens exposés, utiliser `.env.example`).

Description du script

`pivot_tracked_and_stage.py` est un script Python qui transforme des données DHIS2 (système de gestion d'informations sanitaires) en un fichier Excel bien structuré.

Ce que fait le script :

Lit des données depuis des fichiers CSV exportés de DHIS2
Crée un fichier Excel avec plusieurs onglets organisés
Premier onglet : liste des entités suivies (personnes, patients, etc.)
Onglets suivants : un onglet par étape de programme (consultations, vaccinations, etc.)
Ajuste automatiquement la largeur des colonnes pour une meilleure lisibilité
Peut reprendre là où il s'est arrêté en cas d'interruption

Fichiers générés :

- `pivot_tracked_and_stage.xlsx` : le fichier Excel final avec tous vos données organisées

---

Prérequis

Avant de commencer, vous aurez besoin de :

1. **Un ordinateur avec :**
   - **Windows 10/11**, **Mac OS X 10.12+**, ou **Linux (Ubuntu, Debian, etc.)**
   - **Au moins 4 GB de RAM** (8 GB recommandé pour de gros fichiers)
   - **500 MB d'espace disque libre**

2. **Les fichiers CSV sources :**
   - `trackedEntityInstances.csv` : exporté depuis DHIS2 (entités suivies)
   - `data.csv` : exporté depuis DHIS2 (événements des programmes)

3. **Accès à DHIS2 :**
   - **URL de l'API DHIS2** (exemple : `https://dhis2.moh.gov.zm/hmis-events/api/29`)

- **Token d'authentification** (une longue chaîne de caractères fournie par votre administrateur DHIS2)
- **UID du programme** (identifiant unique du programme, exemple : `LlrP8fstjfM`)

Installation étape par étape

Windows

Étape 1 : Installer Python

1. **Télécharger Python**
   - Allez sur [https://www.python.org/downloads/%5D(https://www.python.org/downloads/)
   - Cliquez sur le bouton jaune **"Download Python 3.12.x"** (ou version plus récente)
2. **Installer Python**
   - Double-cliquez sur le fichier téléchargé (`python-3.12.x-amd64.exe`)
   - **TRÈS IMPORTANT** : Cochez la case **"Add Python to PATH"** en bas de la fenêtre
   - Cliquez sur **"Install Now"**
   - Attendez la fin de l'installation (2-5 minutes)
   - Cliquez sur **"Close"**
3. **Vérifier l'installation**
   - Appuyez sur les touches **Windows + R** en même temps
   - Tapez `cmd` et appuyez sur **Entrée**
   - Dans la fenêtre noire qui s'ouvre, tapez :
     `python --version`
   - Vous devriez voir quelque chose comme : `Python 3.12.1`
   - Si vous voyez un message d'erreur, recommencez l'étape 2 en cochant bien **"Add Python to PATH"**

Étape 2 : Télécharger le script

1. **Créer un dossier de travail**
   - Ouvrez l'**Explorateur de fichiers** (icône de dossier dans la barre des tâches)
   - Allez dans **"Documents"**
   - Faites un clic droit → **Nouveau** → **Dossier**
   - Nommez-le `DHIS2_Script`
2. **Placer les fichiers**
   - Copiez tous les fichiers du projet dans le dossier `DHIS2_Script` :

- `pivot_tracked_and_stage.py`
- `.env`
- Le dossier `utils/` avec son contenu

Étape 3 : Installer les bibliothèques nécessaires

1. **Ouvrir l'invite de commandes dans votre dossier**
   - Ouvrez le dossier `DHIS2_Script` dans l'Explorateur
   - Maintenez la touche **Shift** enfoncée et faites un **clic droit** dans l'espace vide du dossier
   - Sélectionnez **"Ouvrir dans le Terminal"** ou **"Ouvrir une fenêtre PowerShell ici"**
2. **Installer les dépendances**
   - Dans la fenêtre qui s'ouvre, tapez cette commande et appuyez sur **Entrée** :
     `pip install pandas openpyxl requests python-dotenv`
   - Attendez quelques minutes (le téléchargement et l'installation peuvent prendre 2-5 minutes)
   - Vous devriez voir plusieurs lignes défiler avec "Successfully installed..."

Installation terminée pour Windows !

Mac

Étape 1 : Installer Python

1. **Vérifier si Python est déjà installé**
   - Ouvrez **Terminal** (Cherchez "Terminal" dans Spotlight avec **Cmd + Espace**)
   - Tapez :
     `python3 --version`
   - Si vous voyez `Python 3.8` ou plus récent, passez à l'Étape 2
   - Sinon, continuez ci-dessous
2. **Installer Python avec Homebrew (méthode recommandée)**
   - Dans le Terminal, installez d'abord Homebrew :
     `/bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/HEAD/`
   - Puis installez Python :
     `brew install python3`
   - Vérifiez l'installation :
     `python3 --version`

Étape 2 : Télécharger le script

1. **Créer un dossier de travail**
   - Ouvrez le **Finder**
   - Allez dans **Documents**
   - Créez un nouveau dossier appelé `DHIS2_Script`
2. **Placer les fichiers**
   - Copiez tous les fichiers du projet dans `DHIS2_Script` :
     - `pivot_tracked_and_stage.py`
     - `.env`
     - Le dossier `utils/` avec son contenu

Étape 3 : Installer les bibliothèques nécessaires

1. **Ouvrir Terminal dans votre dossier**
   - Ouvrez **Terminal**
   - Tapez `cd` (avec un espace après)
   - Glissez-déposez le dossier `DHIS2_Script` dans la fenêtre Terminal
   - Appuyez sur **Entrée**
2. **Installer les dépendances**

   ```
   pip3 install pandas openpyxl requests python-dotenv
   ```

   - Attendez la fin de l'installation (2-5 minutes)

Installation terminée pour Mac !

Linux

Étape 1 : Installer Python et pip Ubuntu/Debian :

```
sudo apt update
sudo apt install python3 python3-pip
```

Fedora/Red Hat :

```
sudo dnf install python3 python3-pip
```

Vérifiez l'installation :

```
python3 --version
pip3 --version
```

Étape 2 : Télécharger le script

```
mkdir -p ~/Documents/DHIS2_Script
cd ~/Documents/DHIS2_Script
# Copiez tous vos fichiers ici
```

Étape 3 : Installer les dépendances

```
pip3 install pandas openpyxl requests python-dotenv
```

Installation terminée pour Linux !

Configuration

Étape 1 : Comprendre le fichier `.env`

Le fichier `.env` contient toutes les configurations du script. Il centralise :

- l'accès à l'API DHIS2 (URL + token)
- l'UID du programme à télécharger
- les chemins des fichiers d'entrée/sortie et des caches locaux

Ouvrez `.env` avec un éditeur de texte simple (Bloc-notes, TextEdit, gedit, VS Code...).

Variables principales (extraites du `.env` utilisé dans ce dépôt)

```
# URL de base de l'API DHIS2
PIVOT_BASE_URL=https://dhis2.moh.gov.zm/hmis-events/api/29

# Token d'accès (Personal Access Token)
DOWNLOAD_TOKEN=d2pat_...    # NE PAS COMMITTER

# UID du programme à traiter
DOWNLOAD_PROGRAM=###########

# Fichiers CSV sources (chemins relatifs au repo)
TRACKED_OUTPUT=data/trackedEntityInstances.csv
PIVOT_INPUT=data/data.csv

# Fichier Excel final
MERGED_PIVOT_OUTPUT=output/pivot_tracked_and_stage.xlsx

# Fonction d'agrégation utilisée pour les pivots (pandas)
PIVOT_AGGFUNC=first

# Fichiers utilitaires / cache
```

```
PIVOT_MAPPING_FILE=utils/dataelement_mapping.json
PIVOT_STATE_FILE=utils/progress_state.json
```

Remarques :

- Les chemins `data/` et `output/` sont utilisés dans cet exemple ; adaptez-les si nécessaire.
- `DOWNLOAD_TOKEN` doit rester secret. Ne l'ajoutez jamais à un dépôt public.

Comment obtenir les valeurs

- Token : générez un Personal Access Token dans DHIS2 (Profil → Personal Access Tokens) ou demandez-le à l'administrateur.
- URL : l'URL de votre instance DHIS2 (ex. https://dhis2.example.org/hmis-events/api/29).
- Programme UID : copiez l'UID du programme depuis l'interface DHIS2.

Bonnes pratiques de sécurité

- Si votre `.env` contient déjà un token réel (comme dans l'exemple), il est fortement recommandé de :
  1. Révoquer/faire expirer ce token côté DHIS2.
  2. Remplacer la valeur dans `.env` par un placeholder (ex. `d2pat_XXXXX`) et stocker le vrai token de façon sécurisée (vault, variable CI/CD, ou fichier `.env` local non versionné).
- Ajoutez `.env` au `.gitignore` si ce n'est pas déjà fait.

Fichiers d'entrée et sortie (résumé)

- Entrées attendues (relatives au dépôt) :
  - `data/trackedEntityInstances.csv` (export Tracked Entity Instances)
  - `data/data.csv` (export des événements / program stages)
- Sortie :
  - `output/pivot_tracked_and_stage.xlsx`

Paramètres avancés

- `PIVOT_AGGFUNC` : fonction d'agrégation pandas appliquée lors des pivots (`first`, `last`, `sum`, `max`, `min`, etc.).
- `PIVOT_MAPPING_FILE` : cache JSON pour accélérer la traduction UID → displayName des data elements.
- `PIVOT_STATE_FILE` : état de progression pour permettre la reprise après interruption.

Étape suivante : sauvegarder

- Sauvegardez les modifications du `.env` localement.

- Ne poussez jamais un `.env` contenant des credentials dans un dépôt.

Utilisation du script

Méthode 1 : Utilisation simple (recommandée pour les débutants)

Cette méthode utilise toutes les configurations du fichier `.env`.

Sur Windows :

1. Ouvrez l'**Explorateur de fichiers** et allez dans votre dossier `DHIS2_Script`
2. Maintenez **Shift** et faites un **clic droit** dans le dossier
3. Sélectionnez **"Ouvrir dans le Terminal"**
4. Tapez cette commande et appuyez sur **Entrée** :
   `python pivot_tracked_and_stage.py`

Sur Mac/Linux :

1. Ouvrez **Terminal**
2. Naviguez vers votre dossier :
   `cd ~/Documents/DHIS2_Script`
3. Lancez le script :
   `python3 pivot_tracked_and_stage.py`

Méthode 2 : Configuration avancée et flags de contrôle

Ce script est configuré principalement via le fichier `.env` (ou variables d'environnement). Il n'expose pas d'arguments CLI pour chaque option DHIS2 — utilisez `.env` ou préfixez la commande avec des variables d'environnement si besoin.

Contrôle du pipeline via flags (CLI)

- `--skip-download` : saute la phase de téléchargement et utilise les fichiers locaux (attendus aux chemins indiqués dans `.env`).
- `--only-download` : lance uniquement la phase de téléchargement (tracked + events) puis quitte.
- `--only-pivot` : saute la phase de téléchargement et exécute uniquement le pivot + génération Excel.

Exemples :

```
# Exécution complète (téléchargement si configuré puis pivot)
python3 pivot_tracked_and_stage.py

# Télécharger seulement (utile pour récupérer CSV sans générer l'Excel)
python3 pivot_tracked_and_stage.py --only-download
```

```
# Utiliser uniquement des fichiers locaux (ne pas télécharger)
python3 pivot_tracked_and_stage.py --skip-download

# Ne faire que le pivot (utile si vous avez déjà les CSV)
python3 pivot_tracked_and_stage.py --only-pivot
```

Options de mapping (nouveau) :

- `--apply-mapping` : après génération de l'Excel (`MERGED_PIVOT_OUTPUT`), applique les correspondances trouvées dans les fichiers de correspondance/structure à CHAQUE onglet de l'Excel. Utile pour convertir des libellés/textes en codes selon une table de correspondance.
- `--mapping-correspondance` : chemin vers le fichier de correspondance (par défaut `data/correspondance.csv`). Peut être `.csv` ou `.xlsx`.
- `--mapping-structure` : chemin vers le fichier de structure (par défaut `data/structure.xlsx`). Peut être `.csv` ou `.xlsx`.
- `--mapping-col-a1`, `--mapping-col-a2` : noms des colonnes dans le fichier de correspondance (par défaut `Option Codes` et `Option Details`).
- `--mapping-col-b1`, `--mapping-col-b2` : noms des colonnes dans le fichier de structure (par défaut `Option Codes` et `DataElement`). `--mapping-col-b2` indique la colonne cible (nom de la colonne à remplacer dans les onglets Excel).
- `--mapping-log-file` : fichier de log optionnel qui recevra la liste des remplacements appliqués.
- `--mapping-preview` : affiche un aperçu (quelques lignes) des onglets modifiés après application du mapping.

Exemple : définir une variable d'environnement temporairement et lancer le téléchargement

```
TRACKED_BASE_URL=https://dhis2.example.org/hmis-events/api/trackedEntityInstances.csv \
PIVOT_TOKEN=d2pat_XXXXX \
python3 pivot_tracked_and_stage.py --only-download
```

Variables `.env` lues par le script (principales)

- Tracked (download_tracked) : `TRACKED_BASE_URL`, `TRACKED_PROGRAM`, `TRACKED_PROGRAM_START_DATE`, `TRACKED_PROGRAM_END_DATE`, `TRACKED_OU_MODE`, `TRACKED_FORMAT`, `TRACKED_OUTPUT`
- Events (download) : `DOWNLOAD_BASE_URL`, `DOWNLOAD_ORG_UNIT`, `DOWNLOAD_PROGRAM`, `DOWNLOAD_START_DATE`, `DOWNLOAD_END_DATE`, `DOWNLOAD_OU_MODE`, `DOWNLOAD_SKIP_PAGING`, `DOWNLOAD_FORMAT`, `PIVOT_INPUT`
- Pivot / général : `PIVOT_BASE_URL`, `PIVOT_TOKEN` (utilisé pour les requêtes API et les téléchargements), `MERGED_PIVOT_OUTPUT`, `PIVOT_AGGFUNC`, `PIVOT_MAPPING_FILE`, `PIVOT_STATE_FILE`

Remarque : le token utilisé dans le script est `PIVOT_TOKEN` (il est réutilisé pour

les deux téléchargements et pour les appels API). Utilisez `.env.example` comme modèle pour remplir ces valeurs localement.

Mode strict (colonnes essentielles seulement)

Si vous voulez un fichier Excel simplifié avec uniquement les colonnes importantes :

```
python pivot_tracked_and_stage.py --strict
```

Ce mode garde uniquement :

- `trackedEntityInstance` (identifiant unique)
- Colonnes avec "serial_number" (numéros de série)
- `ID` (numéro de ligne)
- Colonnes avec "date" (dates)
- Colonnes avec "parent_consent" (consentements)

Exemples d'utilisation

Exemple 1 : Utilisation standard

```
python pivot_tracked_and_stage.py
```

Résultat : Crée `pivot_tracked_and_stage.xlsx` avec toutes les données.

Exemple 2 : Fichiers personnalisés

```
python pivot_tracked_and_stage.py \
  --tracked-input mes_patients.csv \
  --stage-input mes_consultations.csv \
  --output rapport_janvier_2026.xlsx
```

Résultat : Crée `rapport_janvier_2026.xlsx` à partir de fichiers personnalisés.

Exemple 3 : Mode strict pour rapport simple

```
python pivot_tracked_and_stage.py --strict --output rapport_simplifie.xlsx
```

Résultat : Fichier Excel avec colonnes essentielles uniquement.

Exemple 4 : Changement de la fonction d'agrégation

```
python pivot_tracked_and_stage.py --aggfunc last
```

Résultat : Utilise la dernière valeur en cas de doublons (au lieu de la première).

Exemple 5 : Appliquer le mapping automatiquement après génération

```
# Génère l'Excel puis applique les mappings par défaut (data/correspondance.csv & data/stru
python3 pivot_tracked_and_stage.py --apply-mapping

# Spécifier des fichiers personnalisés et écrire un log
python3 pivot_tracked_and_stage.py --apply-mapping \
   --mapping-correspondance custom/corresp.csv \
   --mapping-structure custom/structure.xlsx \
   --mapping-log-file mapping_changes.txt --mapping-preview
```

Résolution des problèmes courants

Problème 1 : "python n'est pas reconnu..."

Symptôme : Message d'erreur "python n'est pas reconnu en tant que commande interne..."

Solution Windows :

1. Réinstallez Python en cochant **"Add Python to PATH"**
2. Ou utilisez `python3` au lieu de `python` dans les commandes

Solution Mac/Linux :

- Utilisez `python3` au lieu de `python`

Problème 2 : "ModuleNotFoundError: No module named 'pandas'"

Symptôme : Le script dit qu'il manque un module Python

Solution :

```
# Windows
pip install pandas openpyxl requests python-dotenv

# Mac/Linux
pip3 install pandas openpyxl requests python-dotenv
```

Problème 3 : "FileNotFoundError: [Errno 2] No such file or directory: 'trackedEntityInstances.csv'"

Symptôme : Le script ne trouve pas les fichiers CSV

Solution :

1. Vérifiez que vos fichiers CSV sont dans le même dossier que le script
2. Vérifiez les noms de fichiers dans le `.env` :
   `TRACKED_OUTPUT=trackedEntityInstances.csv`
   `PIVOT_INPUT=data.csv`

3. Si vos fichiers ont d'autres noms, changez-les dans `.env` ou renommez vos fichiers

Problème 4 : "requests.exceptions.HTTPError: 401 Client Error: Unauthorized"

Symptôme : Erreur d'authentification DHIS2

Solution :

1. Vérifiez que votre token dans `.env` est correct (ne jamais placer le token réel dans un dépôt) :
   ```
   # Exemple (placeholder) - ne pas committer la valeur réelle
   DOWNLOAD_TOKEN=d2pat_XXXXX
   ```
2. Vérifiez que votre token n'a pas expiré (demandez un nouveau token à l'administrateur)
3. Vérifiez l'URL de l'API :
   ```
   PIVOT_BASE_URL=https://votre-serveur-dhis2.com/api/29
   ```

Problème 5 : Le script s'arrête au milieu

Symptôme : Le script se ferme brusquement ou affiche une erreur

Solution :

1. **Ne vous inquiétez pas !** Le script peut reprendre où il s'est arrêté
2. Relancez simplement la même commande :
   `python pivot_tracked_and_stage.py`
3. Le script détectera le fichier de progression (`utils/progress_state.json`) et continuera

Problème 6 : Excel est corrompu ou ne s'ouvre pas

Symptôme : Le fichier Excel généré ne s'ouvre pas correctement

Solution :

1. Supprimez le fichier Excel partiellement créé :
   ```
   # Windows (dans l'invite de commandes)
   del pivot_tracked_and_stage.xlsx

   # Mac/Linux
   rm pivot_tracked_and_stage.xlsx
   ```
2. Supprimez le fichier de progression :
   ```
   # Windows
   del utils\progress_state.json

   # Mac/Linux
   rm utils/progress_state.json
   ```

3. Relancez le script depuis le début

Problème 7 : "Permission denied"

Symptôme : Erreur de permission lors de la création du fichier Excel

Solution :

1. Fermez le fichier Excel s'il est ouvert dans Excel/LibreOffice
2. Vérifiez que vous avez les droits d'écriture dans le dossier
3. Sur Linux/Mac, utilisez :
   ```
   chmod +x pivot_tracked_and_stage.py
   ```

Problème 8 : Mémoire insuffisante

Symptôme : Le script est très lent ou affiche "MemoryError"

Solution :

1. Fermez les autres applications pour libérer de la RAM
2. Traitez vos données par période plus petite (changez les dates dans `.env`)
3. Utilisez le mode `--strict` pour réduire le nombre de colonnes

Questions fréquentes (FAQ)

Q1 : Combien de temps prend le script ?

R : Cela dépend de la taille de vos données :

- Petit fichier (< 1000 lignes) : 10-30 secondes
- Fichier moyen (1000-10000 lignes) : 1-5 minutes
- Gros fichier (> 10000 lignes) : 5-30 minutes

Q2 : Puis-je utiliser le script plusieurs fois ?

R : Oui ! Vous pouvez l'utiliser autant de fois que vous voulez. Changez simplement les fichiers d'entrée ou le nom du fichier de sortie.

Q3 : Le fichier Excel sera-t-il écrasé à chaque fois ?

R : Si un fichier Excel avec le même nom existe déjà, le script le complétera intelligemment au lieu de recommencer à zéro.

Q4 : Mes données sont-elles envoyées sur Internet ?

R : Le script communique uniquement avec votre serveur DHIS2 (l'URL que vous avez configurée) pour récupérer les noms des éléments de données. Aucune donnée n'est envoyée ailleurs.

Q5 : Puis-je modifier le script ?

R : Oui, si vous connaissez Python ! Le script est open-source. Mais faites une copie de sauvegarde avant de modifier.

Q6 : Comment avoir de l'aide supplémentaire ?

R :

1. Relisez ce README
2. Vérifiez la section "Résolution des problèmes"
3. Contactez votre administrateur DHIS2
4. Cherchez sur Google l'erreur exacte que vous voyez

Q7 : Le script fonctionne-t-il hors ligne ?

R : Partiellement. Une fois que les métadonnées (noms des éléments) sont en cache (`utils/dataelement_mapping.json`), le script peut fonctionner hors ligne pour les données déjà téléchargées. Mais la première exécution nécessite Internet pour accéder à l'API DHIS2.

Q8 : Que contient le dossier `utils/` ?

R :

- `dataelement_mapping.json` : Cache des noms d'éléments de données (pour éviter de redemander à DHIS2)
- `progress_state.json` : Fichier de progression (créé automatiquement, supprimé à la fin)

Q9 : Puis-je automatiser le script ?

R : Oui ! Sur Windows, créez un fichier `.bat` :

```
@echo off
cd C:\Users\VotreNom\Documents\DHIS2_Script
python pivot_tracked_and_stage.py
pause
```

Sur Mac/Linux, créez un fichier `.sh` :

```
#!/bin/bash
cd ~/Documents/DHIS2_Script
python3 pivot_tracked_and_stage.py
```

Puis planifiez-le avec le Planificateur de tâches (Windows) ou cron (Mac/Linux).

Q10 : Quelles sont les différences entre les versions de Python ?

R : Le script nécessite **Python 3.8 ou plus récent**. Python 2.x ne fonctionnera pas.

Comprendre le fichier Excel généré

Structure du fichier :

Le fichier Excel contient plusieurs onglets :

Onglet 1 : "TrackedEntities"

- Contient une ligne par entité suivie (patient, bénéficiaire, etc.)
- Colonnes :
  - `trackedEntityInstance` : Identifiant unique
  - `serial_number` : Numéro de série ou d'enregistrement
  - `ID` : Numéro de ligne (1, 2, 3, ...)
  - Autres attributs (nom, âge, localisation, etc.)

Onglets suivants : Un par "Program Stage"

- Exemples : "Consultation initiale", "Suivi mensuel", "Vaccination", etc.
- Chaque onglet contient les événements de cette étape du programme
- Colonnes :
  - `enrollment` : Identifiant de l'inscription au programme
  - `ID` : Numéro de ligne
  - Éléments de données spécifiques à cette étape

Fonctionnalités Excel :

Colonnes auto-ajustées : La largeur s'adapte automatiquement au contenu
Onglets vides supprimés : Si une étape n'a pas de données, elle n'apparaît pas
Format Excel standard : Compatible avec Microsoft Excel, LibreOffice, Google Sheets

---

Sécurité et confidentialité

IMPORTANT - Protégez vos données !

1. **Ne partagez JAMAIS votre fichier `.env`** : Il contient votre token d'accès
2. **Protégez vos fichiers CSV et Excel** : Ils contiennent des données sensibles
3. **Utilisez des mots de passe** : Protégez votre ordinateur par mot de passe
4. **Sauvegardes** : Faites des copies de sauvegarde régulières de vos données

Où sont stockées les données ?

- Tout est stocké **localement** sur votre ordinateur
- Aucune donnée n'est envoyée à des services tiers
- La seule communication est avec votre serveur DHIS2 (pour récupérer les métadonnées)

Aide et support

Ressources utiles :

- **Documentation Python** : [https://docs.python.org/fr/3/%5D(https://docs.python.org/fr/3/)
- **Documentation Pandas** : [https://pandas.pydata.org/docs/%5D(https://pandas.pydata.org/docs/)
- **DHIS2 Documentation** : [https://docs.dhis2.org/%5D(https://docs.dhis2.org/)

En cas de problème :

1. Relisez ce README attentivement
2. Consultez la section "Résolution des problèmes"
3. Vérifiez que tous les fichiers sont au bon endroit
4. Vérifiez votre configuration `.env`
5. Contactez votre administrateur DHIS2 pour les questions d'accès

---

Informations techniques

Dépendances Python :

- **pandas** (1.3.0) : Manipulation de données et création de tableaux croisés
- **openpyxl** (3.0.0) : Lecture/écriture de fichiers Excel
- **requests** (2.25.0) : Communication avec l'API DHIS2
- **python-dotenv** (0.19.0) : Chargement des variables d'environnement depuis `.env`

Compatibilité :

- **Python** : 3.8, 3.9, 3.10, 3.11, 3.12
- **Systèmes d'exploitation** : Windows 10/11, macOS 10.12+, Linux (Ubuntu 18.04+, Debian 10+, Fedora 30+)
- **Excel** : Compatible avec Microsoft Excel 2010+, LibreOffice Calc 6+, Google Sheets

---

Félicitations !

Vous êtes maintenant prêt à utiliser le script `pivot_tracked_and_stage.py` !

Résumé rapide :

```
# 1. Installez Python
# 2. Installez les dépendances
pip install pandas openpyxl requests python-dotenv

# 3. Configurez le fichier .env
# 4. Placez vos fichiers CSV dans le dossier
# 5. Lancez le script
python pivot_tracked_and_stage.py

# 6. Récupérez votre fichier Excel !
# → pivot_tracked_and_stage.xlsx
```

Notes de version

Version actuelle : 1.0

Fonctionnalités :

- Pivot des entités suivies
- Création d'onglets par Program Stage
- Auto-ajustement des colonnes
- Reprise après interruption
- Mode strict pour colonnes essentielles
- Barre de progression
- Cache des métadonnées

Licence

Ce script est fourni tel quel, sans garantie. Utilisez-le à vos propres risques.

Date de création du README : Janvier 2026
Dernière mise à jour : Janvier 2026

Astuce finale : Créez un raccourci sur votre bureau pour lancer rapidement le script !

Sur **Windows**, créez un fichier `Lancer_Script.bat` :

```
@echo off
cd C:\Users\VotreNom\Documents\DHIS2_Script
python pivot_tracked_and_stage.py
pause
```

Sur **Mac**, créez un fichier `Lancer_Script.command` :

```
#!/bin/bash
cd ~/Documents/DHIS2_Script
python3 pivot_tracked_and_stage.py
read -p "Appuyez sur Entrée pour fermer..."
```

Puis rendez-le exécutable : `chmod +x Lancer_Script.command`