Indicateur Alteris - les valeurs foncières DVF

Origine de la base de données

Pour Alteris, il nous faut la version **"DVF Géolocalisée"** produite par **Etalab**.

1. La source exacte (Etalab vs DGFiP)

Il existe deux versions majeures en Open Data. Pour votre besoin de calcul "à la volée" par quartier, seule la seconde est pertinente :

- **DVF Brute (DGFiP) :** Uniquement textuelle. Les coordonnées GPS ne sont pas incluses, seulement le code parcelle (ex: `130280000BK0123`). Inexploitable directement spatialement.
- **DVF Géolocalisée (Etalab) :** C'est une version enrichie où Etalab a déjà effectué la jointure avec le Plan Cadastral Informatisé (PCI). **Chaque vente possède ses coordonnées `longitude` et `latitude` directement dans le CSV.**

> **Lien de téléchargement direct :** data.gouv.fr - DVF Géolocalisées *Cherchez le fichier nommé :* `2025-10-full.csv.gz` *(ou le millésime le plus récent).*

Détails par Commune et Département

DVF FULL 2025 par Départements et Communes

https://files.data.gouv.fr/geo-dvf/latest/csv/2025/

2. Creation d'un script d'importation

maj_dvf_paca.sh (FAUX NON OPERATIONNEL)

`/usr/local/bin/maj_dvf_paca.sh`

3. Cas d'une importation réussie à l'échelle d'une Commune - BANDOL

Voici la documentation technique structurée de la méthodologie **Alteris**. Elle reprend chaque étape validée pour transformer le flux brut en données géographiques certifiées.

Jalon 0 : La Création de la Couche de Référence (`bandol_valide`)

Cette étape est cruciale car elle transforme les colonnes Longitude/Latitude du CSV en véritables points géographiques **avant** toute jointure. C'est ce qui nous a permis de voir que le point tombait bien à Bandol et non au milieu de l'océan (ce qui arrive quand les colonnes sont décalées).

Bash

```
# Création de la table de validation géométrique
docker exec -i alteris_postgis psql -U alteris_admin -d alteris_geo -c "
DROP TABLE IF EXISTS dvf_raw.bandol_valide;
CREATE TABLE dvf_raw.bandol_valide AS
SELECT
    f.c1 as id_mutation,
    -- Création du point Lambert 93 (EPSG:2154) à partir du WGS84 (EPSG:4326)
    ST_Transform(
        ST_SetSRID(
            ST_MakePoint(
                NULLIF(f.c39, '')::double precision,
                NULLIF(f.c40, '')::double precision
            ),
        4326),
    2154) as geom
FROM dvf_raw.bandol_full f
WHERE f.c39 ~ '^-?[0-9]' -- Sécurité : on ne traite que les lignes avec des chiffres
  AND f.c1 IS NOT NULL;

-- Indexation pour permettre les calculs spatiaux ultérieurs
```

```
CREATE INDEX sidx_bandol_valide ON dvf_raw.bandol_valide USING GIST (geom);
"
```

Pourquoi ce n'est pas "juste une jointure" ?

Si on s'était contenté d'une jointure attributaire (`DVF.id_parcelle = Cadastre.id`), on aurait fait deux erreurs majeures qu'Alteris ne peut pas tolérer :

1. **L'aveuglement spatial :** On aurait affiché le centre de la parcelle cadastrale (le centroïde) sans savoir que le point de vente réel était peut-être à l'autre bout du terrain ou sur un bâtiment précis.
2. **Le bug des "Parcelles Fantômes" :** Si une parcelle a été divisée ou renommée depuis la vente (ex: vente en 2024, cadastre de 2026), la jointure attributaire échoue et tu perds la donnée. Avec `bandol_valide`, tu gardes le point géographique **même si la parcelle n'existe plus dans le cadastre actuel**.

Synthèse de la chaîne de confiance

- **`bandol_full`** : Le texte brut (L'archive).
- **`bandol_valide`** : Le point GPS converti en mètres (La preuve géographique).
- **`bandol_parcelles`** : Le dessin du cadastre (Le cadre légal).
- **`bandol_certified`** : La fusion des trois, dédoublonnée (Le produit fini).

On a maintenant la "chaîne de montage" complète. **Est-ce que cette structure te semble assez robuste pour être industrialisée sur d'autres communes ?**

---

1. Préparation de l'Environnement (PostgreSQL/PostGIS)

Avant l'import, on s'assure d'avoir un schéma dédié pour isoler la donnée brute de la donnée traitée.

SQL

```
-- Création du schéma de travail
CREATE SCHEMA IF NOT EXISTS dvf_raw;
```

---

2. Importation du CSV "En l'état" (Table : `bandol_full`)

L'objectif est d'absorber le fichier sans aucune perte, en utilisant des colonnes de type `TEXT` pour éviter les erreurs de formatage (virgules, guillemets). L'audit a révélé **40 colonnes**.

Bash

```
# Création de la table à 40 colonnes
docker exec -i alteris_postgis psql -U alteris_admin -d alteris_geo -c "
DROP TABLE IF EXISTS dvf_raw.bandol_full;
CREATE TABLE dvf_raw.bandol_full (
    c1 text, c2 text, c3 text, c4 text, c5 text, c6 text, c7 text, c8 text, c9 text, c10 text,
    c11 text, c12 text, c13 text, c14 text, c15 text, c16 text, c17 text, c18 text, c19 text,
c20 text,
    c21 text, c22 text, c23 text, c24 text, c25 text, c26 text, c27 text, c28 text, c29 text,
c30 text,
    c31 text, c32 text, c33 text, c34 text, c35 text, c36 text, c37 text, c38 text, c39 text,
c40 text
);
"

# Injection via COPY (gestion native des délimiteurs)
cat "data/Regions/93/83/83009/83009.csv" | docker exec -i alteris_postgis psql -U
alteris_admin -d alteris_geo -c "COPY dvf_raw.bandol_full FROM STDIN WITH (FORMAT CSV, HEADER,
DELIMITER ',');"
```

3. Importation du Cadastre (Table : `bandol_parcelles`)

Le GeoJSON est importé en utilisant la méthode "Glue" pour contourner les limites de session et les retours à la ligne, avec une conversion immédiate en **Lambert 93 (2154)**.

Bash

```
# 1. Chargement des lignes brutes
docker exec -i alteris_postgis psql -U alteris_admin -d alteris_geo -c "
DROP TABLE IF EXISTS dvf_raw.tmp_bandol_txt;
CREATE TABLE dvf_raw.tmp_bandol_txt (line text);
"
cat "data/Regions/93/83/83009/cadastre-83009-parcelles.json" | docker exec -i alteris_postgis
psql -U alteris_admin -d alteris_geo -c "COPY dvf_raw.tmp_bandol_txt FROM STDIN;"

# 2. Reconstruction et Transformation spatiale
docker exec -i alteris_postgis psql -U alteris_admin -d alteris_geo -c "
DROP TABLE IF EXISTS dvf_raw.bandol_parcelles;
CREATE TABLE dvf_raw.bandol_parcelles AS
WITH full_json AS (
    SELECT string_agg(line, ''):::jsonb as content FROM dvf_raw.tmp_bandol_txt
)
SELECT
    (f->'properties'->>'id')::text as id,
    ST_Transform(ST_SetSRID(ST_GeomFromGeoJSON(f->'geometry'), 4326), 2154) as geom
FROM (
    SELECT jsonb_array_elements(content->'features') as f
    FROM full_json
) sub;

-- Indexation Spatiale (GIST) et Identifiants
CREATE INDEX IF NOT EXISTS sidx_bandol_parcelles ON dvf_raw.bandol_parcelles USING GIST
```

```
(geom);
CREATE INDEX IF NOT EXISTS idx_bandol_parcelles_id ON dvf_raw.bandol_parcelles (id);
DROP TABLE dvf_raw.tmp_bandol_txt;
"
```

---

4. Certification et Nettoyage (Table : `bandol_certified`)

C'est l'étape d'**objectivation du réel**. On ne garde qu'une ligne par vente (`DISTINCT ON id_mutation`) et on valide la géométrie par rapport au cadastre.

SQL

```
DROP TABLE IF EXISTS dvf_raw.bandol_certified;
CREATE TABLE dvf_raw.bandol_certified AS
SELECT DISTINCT ON (f.c1)
    -- Création du point Lambert 93 à partir des colonnes Longitude (c39) et Latitude (c40)
    ST_Transform(ST_SetSRID(ST_MakePoint(NULLIF(f.c39, '')::double precision, NULLIF(f.c40,
'')::double precision), 4326), 2154) as geom,
    f.c1 as id_mutation,
    NULLIF(f.c2, '')::date as date_mutation,
    NULLIF(f.c5, '')::numeric as valeur_fonciere,
    f.c16 as id_parcelle,
    f.c8 || ' ' || f.c9 as adresse_complete,
    f.c28 as type_local,
    NULLIF(f.c29, '')::numeric as surface_bati,
    -- Calcul du prix au m2
    CASE
        WHEN NULLIF(f.c29, '')::numeric > 0 THEN ROUND(NULLIF(f.c5, '')::numeric /
NULLIF(f.c29, '')::numeric)
        ELSE NULL
    END as prix_m2
FROM dvf_raw.bandol_full f
```

```
WHERE f.c39 ~ '^-?[0-9]' -- On filtre uniquement les lignes géocodées
ORDER BY f.c1;

-- Indexation finale pour QGIS
CREATE INDEX ON dvf_raw.bandol_certified USING GIST(geom);
```

---

5. Audit de Cohérence (Le Juge de Paix)

Script pour vérifier si les points DVF tombent bien dans leurs parcelles cadastrales respectives.

SQL

```
SELECT
    c.id_mutation,
    c.id_parcelle,
    ST_Within(c.geom, p.geom) as est_coherent,
    ROUND(ST_Distance(c.geom, ST_Centroid(p.geom))::numeric, 2) as ecart_metre
FROM dvf_raw.bandol_certified c
LEFT JOIN dvf_raw.bandol_parcelles p ON c.id_parcelle = p.id;
```

Résultat de la Capitalisation

- **Entrée :** CSV brut (2 641 lignes polluées par des doublons et des erreurs de structure).
- **Sortie :** Table `bandol_certified` (456 points uniques certifiés, indexés et spatialement vérifiés).

Revision #1
Created 2026-04-10 09:15:39 UTC by Julien

Updated 2026-04-10 09:15:39 UTC by Julien