Référentiels datas

Bases de données importées dans le Websig

- R1- l'organisation des bases de données dans Alteris
- R2 - Acquisitions des données OSM_expert
- R3 - OSM et les transports en commun

R1- l'organisation des bases de données dans Alteris

La base de données postgres du Websig a un mirroir d'échanges.

Comme cela GEMINI a la main sur les deux.

/var/www/alteris-geomatic/data/referentiels

home/debian/osm_data/referentiels/

Organisation de referentiels

Arborescence géographique claire calquée sur celle de l'INSEE et de l'IGN.

Voici l'ordre pyramidal des dossiers

Région (R93 pour la Région PACA)

Département

EPCI (code SIREN)

Commune

Partenaire de code a dit

C'est une excellente décision. Une arborescence structurée est la colonne vertébrale d'un SIG professionnel. En calquant ton stockage sur les codes officiels (INSEE et SIREN), tu garantis l'interopérabilité de tes données avec toutes les futures bases Open Data.

Voici comment nous allons configurer cette structure sur ton VPS pour qu'elle soit à la fois logique pour l'humain et efficace pour le code Node.js.

1. La Structure Physique des Dossiers

Nous allons utiliser les **Codes Insee** (Région, Département, Commune) et le **SIREN** (EPCI) pour nommer les dossiers. C'est plus fiable que les noms alphabétiques qui peuvent avoir des espaces ou des accents.

Exemple pour **Toulon** : `Région 93 (PACA)` > `Département 83 (Var)` > `EPCI 248300543 (TPM)` > `Commune 83137 (Toulon)`

2. Création de l'arborescence

Tu peux lancer cette commande en SSH pour préparer le terrain pour Toulon :

Bash

```
mkdir -p /var/www/alteris-geomatic/data/referentiels/R93/D83/E248300543/C83137
```

3. Organisation des fichiers GPKG

À l'intérieur de chaque dossier "Commune", nous stockerons les fichiers selon une nomenclature stricte :

- `C83137_contour.gpkg` : La limite administrative (IGN).
- `C83137_filaires.gpkg` : Tes extractions de voies spécifiques si besoin.
- `C83137_iris.gpkg` : Le découpage infra-communal de l'INSEE (très utile pour tes futurs calculs de densité).

4. Pourquoi cette structure est vitale pour ton code ?

En structurant ainsi, nous allons pouvoir automatiser les requêtes. Par exemple, si tu sélectionnes Toulon sur l'interface, le serveur saura instantanément où chercher : `path = "/data/referentiels/R" + reg_code + "/D" + dep_code + "/E" + siren + "/C" + insee + "/"`

5. Intégration dans PostgreSQL

Même si les fichiers sont rangés dans des dossiers, nous allons les importer dans des tables structurées de la même manière dans `alteris_db`. Je te suggère de créer une table maître pour les communes :

SQL

```
CREATE TABLE referentiel_communes (
    id SERIAL PRIMARY KEY,
    code_insee VARCHAR(5) UNIQUE,
    nom_commune TEXT,
    code_epci VARCHAR(9),
    code_dept VARCHAR(3),
    code_reg VARCHAR(2),
    geom geometry(MultiPolygon, 4326)
);
CREATE INDEX idx_communes_geom ON referentiel_communes USING GIST(geom);
```

Prochaine action pour toi :

1. **Transférer ton GPKG** de Toulon dans le dossier `/C83137/` que nous venons de créer.
2. **Me donner le nom exact du fichier** pour que je te génère la commande d'importation `ogr2ogr` qui remplira automatiquement les colonnes de code (Insee, EPCI, etc.).

Une fois cela fait, nous pourrons passer à l'étape supérieure : **croiser ce contour de Toulon avec ta matrice de "Profondeur du motif urbain"** (les buffers de 30m, 80m, 150m, 200m et 300m) pour les filaires N1 à N7.

On valide cette arborescence ? Si oui, dès que le fichier est en place, on passe à l'analyse spatiale par buffer.

R2 - Acquisitions des données OSM_expert

La méthodologie est normée avec Gemini

- Exporter le territoire d'études à partir de hotosm.org à partir d'un masque Geojson en WGS84
- retailler l'export car hotosm impose un tampon
- sortir deux GPKG - points et lignes dénommés par exemple pour Toulon
  - 83137-Toulon-points-OSM-alteris
  - 83137-Toulon-lignes-OSM-alteris

Export de la base de données OSM_expert

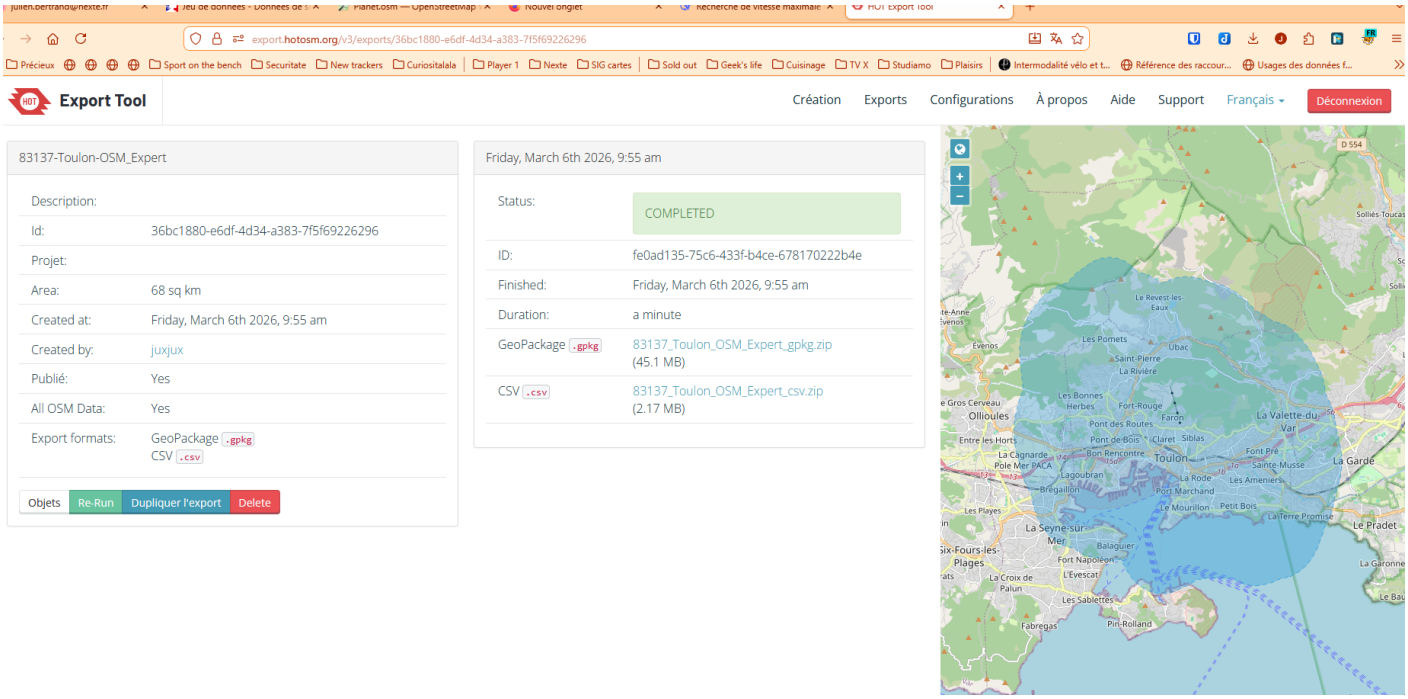
https://export.hotosm.org/v3/

1/ créer un fichier geojson du contour avec un SCR WGS84

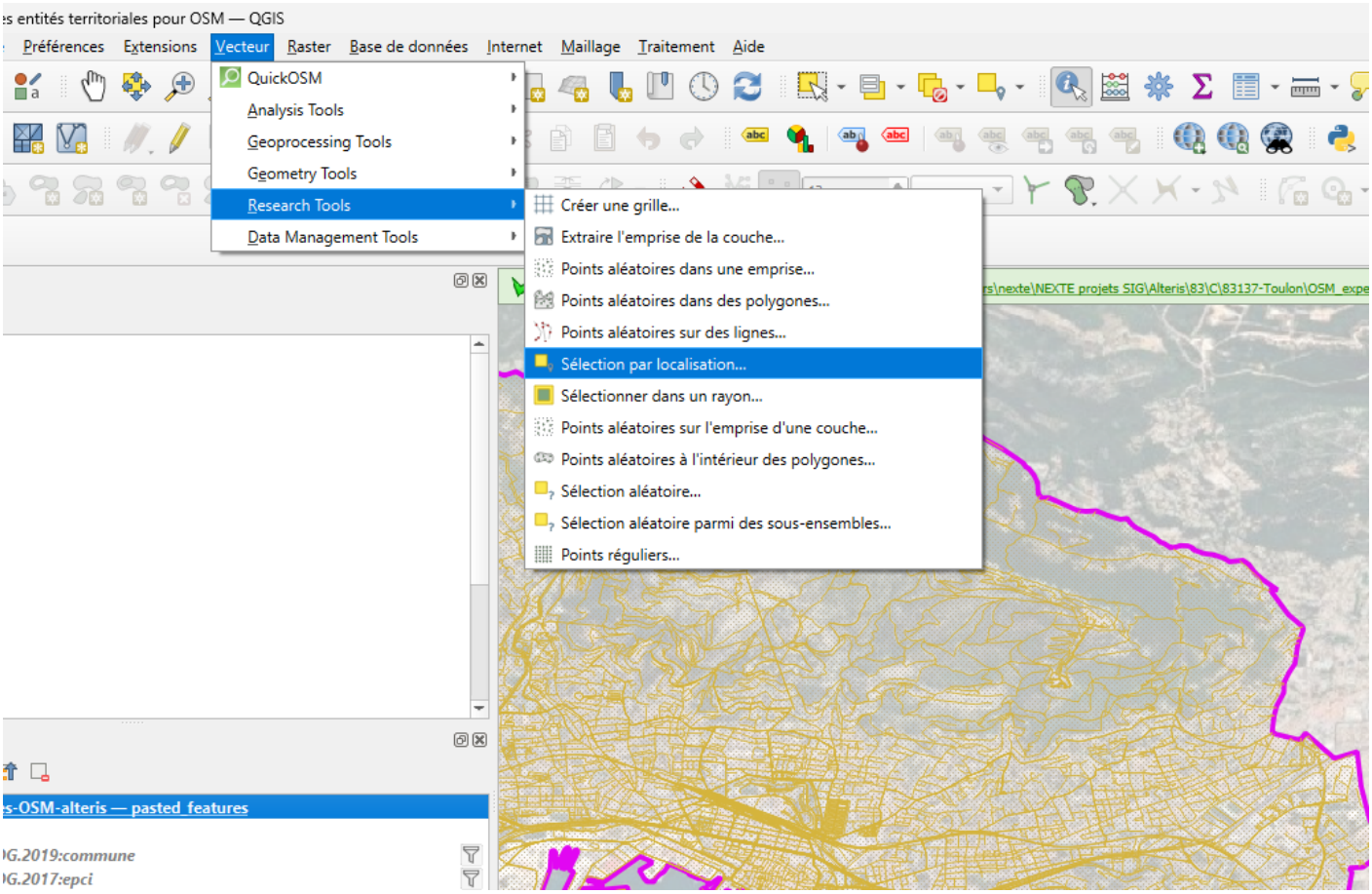
2/ téléverser le geojson dans HOTOSM

3/ récupérer les datas en GPKG et csv (comme prévu par l'interface).

4/ redécouper les lignes et les points par research tool - selection par localisation (couper supprime les tags)



Exemple - Toulon - 45 Mo - 3 minutes d'extraction



C'est la meilleure nouvelle de la journée ! Regardez la ligne `tags` dans votre tableau : son type est **`json`**.

Ce que vous voyiez dans QGIS (`highway: track, maxspeed: 50`) était juste la façon dont le logiciel "aplatissait" le JSON pour l'afficher à l'écran. Mais en réalité, à l'intérieur de PostgreSQL, vos données sont rangées dans un vrai format structuré.

C'est une avancée majeure pour nous, car PostgreSQL possède des "super-pouvoirs" natifs pour lire le JSON. Fini les expressions régulières compliquées (`substring`, `Regex`) ! Pour extraire la vitesse, il suffira d'écrire `tags->>'maxspeed'`. C'est propre, infiniment plus rapide, et impossible de se tromper.

R3 - OSM et les transports en commun

Les 3 niveaux du bus dans OpenStreetMap

1. Le TCSP Intégral (L'infrastructure physique séparée)

- **La réalité :** Une route construite *exclusivement* pour les bus (ou les trams), séparée du trafic automobile par un terre-plein ou des bordures. C'est le vrai BHNS.
- **Le tag OSM :** La rue entière a le tag principal `highway` = `busway`.
- **Dans Alteris :** C'est capté automatiquement par notre SQL comme un **F_T3**.

2. Le Couloir Réservé (L'aménagement sur voirie classique)

- **La réalité :** Une grande avenue (`primary` ou `secondary`) où la mairie a décidé de peindre une voie de bus au sol. Les voitures roulent à côté.
- **Le tag OSM :** La rue reste `highway=primary`, mais on lui ajoute un tag spécifique d'aménagement : `busway` = `lane` (ou `opposite_lane`) OU `lanes:bus` = `1`.
- **Dans Alteris :** Notre SQL traque spécifiquement la présence de ces mots-clés (`tag_busway IS NOT NULL`). S'il voit qu'une voie est physiquement retirée aux voitures pour la donner aux bus, il considère que l'emprise est structurante et passe la rue en **F_T3**.

3. Le Bus en Circulation Banalisée (Le trafic mixte - 90% du réseau)

- **La réalité :** Le bus Mistral roule au milieu des voitures sur une rue normale, s'arrête à un poteau, et repart.
- **Le tag OSM :** Sur le tronçon de rue lui-même, **il n'y a strictement aucun tag lié au bus**. La rue reste une simple `residential` ou `tertiary`.
- **Comment OSM sait que le bus passe par là ?** OSM utilise un outil invisible appelé une **"Relation"**. C'est un groupe logique (la "Ligne 3") qui dit : *Le bus passe par la rue A, puis la rue B, puis la rue C*.
- **Dans Alteris :** Comme nous interrogeons uniquement les caractéristiques *physiques* de la voirie (les tags de la ligne), **notre code SQL est totalement aveugle aux lignes de bus en circulation banalisée**. Une rue résidentielle avec un bus restera sagement classée en F_L1 ou F_L2.