Formation VRT

Jean Pommier, jean.pommier@pi-geosolutions.fr

IdGeo, 1er mars 2024

Sources : https://github.com/pi-geosolutions/formation_VRT/tree/cpgeom202402

Ce cours a pour objectif de vous faire découvrir un format particulier de la librairie OGR, le format VRT. Ce format de donnée virtuelle permet de faire beaucoup de choses et dans bien des cas de se passer d'un ETL.

Préambule

La version PDF de ce cours est générée à partir du code en ligne sur github, à l'aide de l'outil en ligne de commande pandoc. La commande utilisée est `./make_pdf.sh`.

Le résultat est imparfait mais correct (à mon avis). Je vous prie de m'excuser pour les légers couacs de rendu.

A noter que le format PDF n'est guère adapté aux copier-coller, en particulier de code. C'est pourquoi j'ai choisi de présenter ce cours sous la forme d'un ensemble de fichiers Markdown, faciles à lire depuis la page github, mais aussi très pratiques pour copier-coller du code.

Déroulé proposé pour ce cours :

1. Tour de table :
    - vous faites comment, vous, l'intégration de données ?
        - Vous arrive-t-il de charger des fichiers en base de donnée ?
        - Quels outils utilisez-vous ?
        - Quelles difficultés rencontrez-vous ?
    - comment faciliter la vie de vos collègues non techniques ?
        - Rencontrez-vous des problèmes d'interopérabilité entre leurs outils logiciels et les vôtres ?
        - Que faites-vous si vous devez publier un ficher Excel ?
        - Et si le contenu de ce fichier devait être maintenu à jour dans une base de données ?
        - Comment assurez-vous la reproductibilité de vos process de publication ?
2. démos
3. Présentation rapide de la librairie GDAL/OGR et du format VRT
4. Exploration de fichiers VRT
5. Exercices accompagnés : génération et usage de fichiers VRT

GDAL/OGR

- https://gdal.org/
- Une librairie *opensource*, largement utilisée (sous le capot) par de nombreux outils y compris propriétaires
- Traitement de données géospatiales
- Essentiellement des supports entrées/sortie : un nombre incroyable de formats supportés
- Des outils en ligne de commande :
  - Support raster : GDAL
  - Support Vecteur : OGR

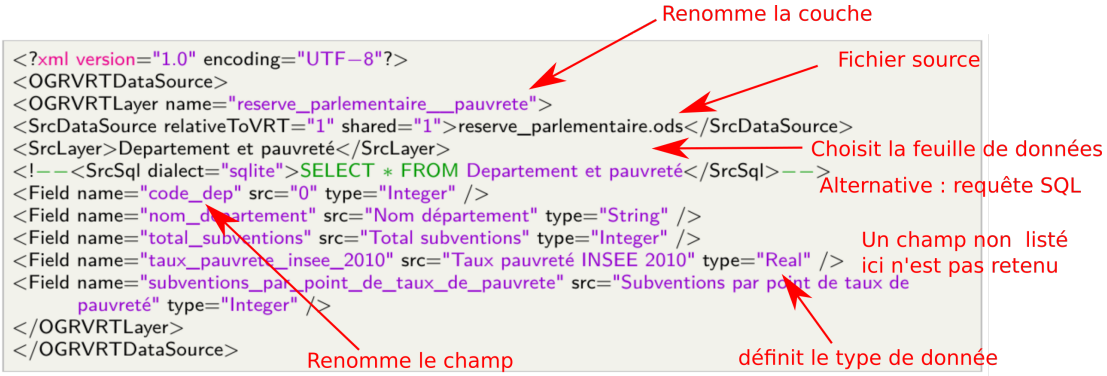
VRT

- Un format interne de GDAL/OGR
- Plus connu pour sa version raster, mais existe aussi pour des données vectorielles et tabulaires :
- https://gdal.org/drivers/vector/vrt.html
- OGR VRT, ça vous permet notamment de :
  - Faire l'économie d'un ETL
  - Autonomiser vos collègues amateurs d'Excel (et donc alléger votre charge de travail)
  - Augmenter la reproductibilité de vos flux de traitement de données
  - Automatiser vos flux de traitement de données
  - Frimer pendant une soirée (de géomaticiens)
- Supporté par QGIS
- Pas limité aux données tabulaires, loin de là. Mais les supporte
- Fichier XML, de configuration de la source de données. Permet de :
  - renommer des champs
  - changer le type des champs (entier, réel, date, texte)
  - ne conserver qu'un sous-ensemble des champs
  - choisir les champs définissant la géométrie (si présents)
  - filtrer le jeu de données via une requête SQL
  - découper sur une étendue via une requête SQL
  - reprojeter un jeu de données
  - fusionner plusieurs sources
  - charger des sources de données en ligne

Un exemple de fichier VRT simple

```
<?xml version="1.0" encoding="UTF-8"?>
<OGRVRTDataSource>
  <OGRVRTLayer name="reserve_parlementaire___pauvrete">
    <SrcDataSource relativeToVRT="1" shared="1">reserve_parlementaire.ods</SrcDataSource>
    <SrcLayer>Departement et pauvreté</SrcLayer>
    <!--<SrcSql dialect="sqlite">SELECT * FROM Departement et pauvreté</SrcSql>-->
    <Field name="code_dep" src="0" type="Integer" />
    <Field name="nom_departement" src="Nom département" type="String" />
    <Field name="total_subventions" src="Total subventions" type="Integer" />
    <Field name="taux_pauvrete_insee_2010" src="Taux pauvreté INSEE 2010" type="Real" />
    <Field name="subventions_par_point_de_taux_de_pauvrete" src="Subventions par point de taux de
        pauvreté" type="Integer" />
  </OGRVRTLayer>
</OGRVRTDataSource>
```

Exemple expliqué



- On définit la source de données
- On renomme les champs
- On publie le VRT (via QGIS ou ogr2ogr)

Conditions de fonctionnement

- Encodage UTF-8
- Pas de fusion de cellules
- Une vraie donnée tabulaire (pas de mise en forme)

Obtenir un VRT de départ

Pour créer votre fichier VRT, vous avez le choix. Vous pouvez :

- L'écrire à la main, grâce à la doc (vite pénible)
- Le générer via QGIS et l'extension Spreadsheet Layer : génère un fichier VRT à côté du fichier ouvert. Supporte XLS, XSLX, ODS, CSV
- Le générer en ligne de commande via https://github.com/jeanpommier/ogr2vrt_simple
- A partir de GDAL 3.6, un script ogr2vrt devrait être livré avec la librairie.

Quelques astuces avec OGR

ogrinfo : inspecter une donnée vecteur

Sans option, ogrinfo renvoie très peu d'infos. La liste des couches contenues par la source de données

J'aime souvent récupérer plus d'infos que ça, mais pas trop.

```
ogrinfo -so -al macouche.gpkg
```

marche bien

L'air de rien, ça vous permet de récupérer pas mal d'infos, sans avoir à ouvrir QGIS, charger la couche, ouvrir les propriétés etc

Exécuter des commandes sql

On peut aussi exécuter des commandes SQL sur une couche de données. Par exemple, un truc du genre :

```
ogrinfo -al -where "NAME10 LIKE '%NY'" tl_2013_us_uac10.shp
```

```
ogrinfo -al -sql "SELECT * FROM tl_2013_us_uac10 WHERE UACE10 = '16171'" tl_2013_us_uac10.shp
```

ogr2ogr : le couteau suisse

On peut presque tout faire avec ogr2ogr. Voir la doc, c'est important.

QQ commandes très fréquemment utilisées :

Afficher dans la console le contenu de la couche

Pratique pour explorer le contenu d'une donnée

```
ogr2ogr -f CSV /vsistdout/ monfichier.gpkg couche1
```

Publier une donnée en base (postgresql).

Pour alléger la ligne de commande, j'utilise des variables d'environnement pour déclarer les paramètres de connexion. Je ne les ai pas nommées au hasard. Pour que psql et ogr les reconnaissent, il faut suivre une convention. Cf. https://docs.postgresql.fr/15/libpq-envars.html.

Il suffit de les déclarer une fois tant qu'on ne change pas de console (sinon on doit recommencer)

Ajustez les valeurs si besoin

```
export PGPASSWORD=secret  # Remplacer par le vrai mdp bien sûr
# Et pour simplifier les paramètres de connexion, on peut faire pareil avec le reste :
export PGDATABASE=cpgeom
export PGUSER=cpgeom
export PGHOST=localhost
```

On aurait aussi pu écrire un fichier .pgpass comme documenté dans https://docs.postgresql.fr/10/libpq-pgpass.html.

```
ogr2ogr -progress -f PostgreSQL PG:"host='$PGHOST' user='$PGUSER' dbname='$PGDATABASE'" \
  -nln "roads" -nlt PROMOTE_TO_MULTI -lco OVERWRITE=YES \
  -lco SCHEMA=destschema monfichier.gpkg couche1
```

Ici, on a utilisé qq options en plus :

- `-nln` pour nommer la table
- `-lco OVERWRITE=YES` pour remplacer le contenu
- `-lco SCHEMA=destschema` pour désigner le schema de destination (et ne pas publier dans `public`)

Et puis comme j'ai dit : voir la doc

Démos

Dossier `demos`

Dézipper les deux archives zip avant usage.

Union de données géospatiales

On l'a dans union/

Objectif pédagogique

Démontrer comment le VRT permet de :

- filtrer sur une valeur attributaire
- rassembler plusieurs jeux de données, y compris geo
- rajouter du contenu

Objectif de la manip

Produire une carte des communes du Gers + Lot et Garonne, sans doublons.

Le pb m'avait été soumis par F., élève CQPGeom. Ils avaient récupéré des extractions BD topo par département. Sauf que à chaque fois, ils avaient en bonus les communes limitrophes. Après avoir importé les deux shp dans la BD, ils avaient donc des doublons. Voire des erreurs si contrainte d'unicité.

Ils avaient passé plusieurs jours à trouver comment éviter les doublons. Je leur ai trouvé une solution en 15 min en fin de cours.

En bonus, j'ai en commentaire dans le VRT l'ajout d'un champ texte avec concaténation, pour le Gers (hello).

On peut aussi rajouter le calcul de la surface totale par commune en hectares, pour montrer une opération geométrique

-> note : contrairement à admin express, cette donnée contient déjà la surface, ainsi que les codes postaux.

Déroulé

J'explique le pb. On charge chaque commune dans QGIS. On regarde le chevauchement. On peut ajouter la couche des départements si on veut.

On va voir la doc VRT, on regarde `OGRVRTUnionLayer`.
Je charge le VRT, on constate que c'est clean.

Je décommente hello et on regarde ça dans la table attributaire.

Achat de produit phyto (glyphosate) par code postal

Source de données :

https://www.data.gouv.fr/fr/datasets/achats-de-pesticides-par-code-postal/ https://www.data.gouv.fr/fr/datasets/codes
postaux-de-toulouse/

Données dans pesticides/

Objectif pédagogique

Démontrer comment le VRT permet de :

- générer un “geopackage virtuel” rassemblant des ressources variées
- assurer une reproductibilité. Le process est documenté et facile à dupliquer
- récupérer une ressource directement en ligne si on veut

Objectif de la manip

Faire une carte de la qté de glyphosate achetée par code postal sur les environs de Toulouse (on pourrait aussi utiliser les limites admin du Gers). On a des données sur plusieurs années donc on peut cartographier l'évolution

```
<OGRVRTLayer name="bnvd_occitanie_2021">
    <SrcDataSource relativeToVRT="1">BNVD_TRACABILITE_20221016_ACHAT_CP_SUBSTANCE_OCCITANIE_2021.c
    </SrcDataSource>
    <SrcSql dialect="sqlite">SELECT code_postal_acheteur, substance, SUM(CAST(quantite_substance a
</OGRVRTLayer>
```

On somme les données pour la substance glyphosate

Déroulé

Je charge les tables de bdnv.vrt. On fait une jointure sur les codes postaux avec 2021 et puis on fait une carte d'intensité d'achats, le rouge, c'est bien.

Ensuite, je rajoute une jointure sur 2020 dans le VRT et charger cette nouvelle table. Jointure à nouveau. Puis clic droit sur la couche codes_postaux, styles, ajouter. Ca va me copier le style par défaut. J'en profite pour les nommer par année. Et je change l'attribut utilisé pour le style.

Ensuite, je décommente 2022. Je peux faire remarquer que là, je tape direct dans la ressource en ligne (utiliser ogr2vrt_simple pour choper l'adresse correcte). Et bis répétita.

Echantillons

Dossier `samples/`

Voilà quelques VRT échantillons qu'on pourra utiliser pour explorer.

On peut :

- les ouvrir dans l'éditeur de code pour voir sous le capot
- les charger dans QGIS
- les manipuler en ligne de commande :
  - lister les infos avec `ogrinfo -so -al`
  - afficher le contenu au format CSV avec `ogr2ogr -f CSV /vsistdout/ monfichier.vrt`

On pourra aussi regarder des VRTs sur https://github.com/geo2france/vrt.

Mise en place d'un flux de publication avec OGR VRT – exercices pratiques

Faute de mieux, on va jouer avec des données prises essentiellement sur data.gouv.fr.

Si vous avez des données à vous, n'hésitez pas à les mettre en jeu.

Exercices :

- [`Livrable attendu`] Exercice 1 : publication d'une donnée tabulaire simple et jointure
- Exercice 2 : filtrer le contenu à publier
- Exercice 3 : corriger un code directement dans le VRT
- [`Livrable attendu`] Exercice 4 : un fichier tabulaire avec coordonnées lat/lon
- Exercice 5 : prendre un fichier distant pour source
- Exercice 6 : union de jeux de données
- [`Livrable attendu`] Exercice 7 : on automatise
- Exercice 8 : faisons une 'appli' basique de crowdsourcing
- Exercice 9… ah, ben, y'en a plus

On peut regarder qq cas concrets dans lesquels le VRT m'a bien servi et fait gagner beaucoup de temps :

- https://github.com/pi-geosolutions/vrt2rdf
- un cas de réorganisation des données, pour le projet SAGUI : https://github.com/HydroMetGuyane-Hydro-Matters/sagui_backend/tree/main/data
- les VRT utilisés couramment par l'équipe SIG de la région des Hauts de France : https://github.com/geo2france/vrt/

Combiné à un peu de code python, on peut faire des miracles en termes de traitement de données.

Ah, et j'oubliais : remarquez dans la doc de vrt2rdf comment on peut même pointer vers une source WFS

Exercice en autonomie (déposer le résultat sur Digiforma):

On finira par un exercice en autonomie, pour l'évaluation digiforma : (sur la version PDF de ce document, ce lien ne marchera pas. Vous trouverez l'exercice en fin de document)

Exercice en autonomie

Nettoyage

Avant de quitter votre ordi, prenez le temps de stopper votre service cron si vous l'avez lancé durant l'exercice 7: dans une console WSL2, lancez la commande

```
sudo service cron stop
```

Exercice 1 : publication d'une donnée tabulaire simple et jointure

---

Livrable attendu Sur cet exercice, je vous demanderai de faire une copie écran de votre carte finale et de la déposer sur digiforma. Pas de panique, on va le dérouler ensemble, mais suivez bien, et n'hésitez pas à poser des questions.

Nommage attendu : nom_exo1_subventions.png

Sources de données

On va utiliser https://www.data.gouv.fr/fr/datasets/logements-vacants-du-parc-prive-par-anciennete-de-vacance-par-commune-et-par-epci/ et on prendra la donnée ***par commune au format xslx***.

Prérequis

QGIS, avec l'extension Spreadsheet Layer installée

Déroulé

Avec QGIS et l'extension Spreadsheet Layer, ouvrir la première feuille. Cela va générer un VRT correspondant. Le nom du fichier VRT est un peu long. Renommez le fichier et ouvrez-le dans un éditeur de code.

> Une alternative en mode *ligne de commande* : vous pouvez aussi utiliser le script ogr2vrt_simple.
>
> *Pour la suite, vous aurez toujours le choix : utiliser le script et votre clavier, ou l'interface graphique et la souris.*

Retouchez un peu le VRT si nécessaire pour le rendre plus compatible avec les contraintes d'une base de données (pas de caractères accentués, pas d'espace), puis le publier en base. Là aussi, vous avez le choix : le publier via QGIS et son database manager (lent) ou bien utiliser ogr2ogr

On va aussi publier en BD les contours des communes dans le Gers (prendre dans demos/union), pour pouvoir faire une jointure. On n'a pas encore vu les serveurs carto, donc on fera le rendu dans QGIS. Faites-nous une jolie carte mettant en évidence les communes les mieux dotés en logements sociaux.

Faites une copie d'écran de la carte obtenue, pour le livrable de cet exercice.

Exercice suivant

Exercice 2 : filtrer le contenu à publier

On va publier une carte montrant la proportion occupée par la production agricole bio par rapport à la surface totale, pour chaque commune du Gers, en 2020.

Sources de données

Téléchargez le fichier https://www.data.gouv.fr/fr/datasets/surfaces-cheptels-et-nombre-doperateurs-bio-a-la-commune/.

Déroulé

ogr2vrt_simple

On va installer le script ogr2vrt_simple.

Windows ne nous facilite pas trop la vie pour le code python, surtout avec les bindings GDAL. C'est pourquoi **on va travailler dans linux WSL2 (debian)**.

- Ouvrez une console debian
- vérifier si gdal/ogr est installé : `ogrinfo --version`
- si vous avez une erreur, c'est que non, donc on va l'installer :
  - on met à jour la liste des paquets : `sudo apt update`
  - on installe GDAL/OGR : `sudo apt install gdal-bin libgdal-dev`
- on va vérifier votre version de python, on veut > 3.8 : `python3 --version`
- on crée un environnement virtuel, afin de cloisonner nos installations de paquets python :
  - `python3 -m venv ogr2vrt_venv`
  - `source ogr2vrt_venv/bin/activate`
  - `(ogr2vrt_venv)` devrait apparaitre à gauche du prompt
- on installe ogr2vrt_simple : suivre les instructions d'installation de https://github.com/jeanpommier/ogr2vrt_simple

Générer le VRT

Utilisez la même commande pour générer votre VRT pour la donnée bio. Puis éditez-le dans un éditeur de texte.

- Première étape : on filtre les entrées pour ne garder que les données sur le Gers (32). On utilisera une source SrcSql pour cela
- Deuxième étape : ne garde que les colonnes liées à 2020
- On vérifie bien le résultat
- Et puis on publie en base
- puis jointure avec les limites administratives du Gers (on pourra prendre la couche geo dans ../demos/union/communes3247.vrt)
- et affichage carto

Exercice suivant

Exercice 3 : corriger un code directement dans le VRT

Un exercice simple, avec correction de données mal formattées, directement via le VRT.

On va corriger le code de département dans le VRT directement.

Sources de données

On va prendre https://www.data.gouv.fr/fr/datasets/boursiers-par-departement/, le fichier CSV.

Déroulé

Générer le VRT avec l'outil de votre choix. Examinez les valeurs (via QGIS, ou en ligne de commande avec ogr2ogr). Notamment le code de département. Ca vous rappelle qq chose ?

On pourrait corriger ça dans PostgreSQL avec, par exemple, la commande LPAD. C'est assez facile. Mais ça serait plus satisfaisant de le faire dès le VRT. Malheureusement, la commande LPAD ne marchera pas. En effet, dans VRT, c'est une syntaxe sqlite qui est utilisée (et encore, pas complète, ou alors pas récente). Le mieux que j'aie trouvé, c'est la syntaxe suivante : `printf("%02d", "numero_departement") AS code_dep`

Vérifiez que ça rectifie bien le code de département.

Vous savez déjà publier et faire une jointure, donc peut-être pas besoin de le refaire ici.

Exercice suivant

Exercice 4 : un fichier tabulaire avec coordonnées lat/lon

Livrable attendu Sur cet exercice, je vous demanderai de déposer sur digiforma votre fichier vrt. Pas de panique, on va le dérouler ensemble, mais suivez bien, et n'hésitez pas à poser des questions.

Nommage attendu : nom_exo4_musees_latlon.vrt

Sources de données

On prendra https://www.data.gouv.fr/fr/datasets/liste-et-localisation-des-musees-de-france/, version 2022 (fichier xslx).

Déroulé

L'idée est d'afficher les emplacements précis (points) des musées. Nous avons de la chance, ils nous fournissent les coordonnées (lat, lon).

Configurez votre VRT pour en faire un fichier géospatial, utilisant les coordonnées fournies.

La doc VRT vous donnera les infos nécessaires : https://gdal.org/drivers/vector/vrt.html#example-odbc-point-layer

Pensez à déposer le fichier vrt, pour le livrable de cet exercice.

Exercice suivant

Exercice 5 : prendre un fichier distant pour source

Sources de données

Reprenons notre donnée de l'exercice 2. Faire évoluer la définition de la source pour utiliser directement une URL.

Déroulé

Indice : on utilisera `vsicurl` (https://gdal.org/user/virtual_file_systems.html)

A noter que si besoin, on peut même chainer avec un dézippage (`viszip`, `vsigz`).

On pourra s'inspirer de l'article
https://static.geotribu.fr/articles/2021/2021-09-07_traiter_fichiers_adresse_gdal_csv_vrt/.

Info : dans certains cas, l'extension du fichier n'est pas fournie par le service web. VRT n'est alors pas capable de détecter le type de fichier et ça ne marche plus. Dans le cas bien particulier d'un CSV (ne marche pas pour d'autres extensions), on peut expliciter le format par la syntaxe suivante : `<SrcDataSource>CSV:/vsicurl/https://....</SrcDataSource>`.
Noter le `CSV:` devant `vsicurl`

Votre donnée doit pouvoir s'afficher dans QGIS.

Vous pouvez aussi recourir à ogr2vrt_simple, qui doit pouvoir générer une définition fonctionnelle de source en mode vsicurl.

Exercice suivant

Exercice 6 : union de jeux de données

Pour celui là, j'ai récupéré des données ailleurs que sur data.gouv.fr. J'ai un peu triché, on aurait pu les récupérer déjà ensemble. Mais ce n'est pas toujours le cas.

Données dispo dans le dossier union. Vous pouvez les télécharger via les liens suivants :

- EN17-1999.xls
- EN17-2009.xls

Exercice suivant

Exercice 7 : on automatise

Livrable attendu Sur cet exercice, je vous demanderai de faire une copie écran de votre tâche cron.

Nommage attendu : nom_exo7_crontab.png

Il y a plein de façons d'automatiser un flux de publication. Nous allons rester sur un cas de figure assez simple : actualisation régulière d'une donnée via une tâche planifiée (crontab) sur un serveur linux.

Source de donnée

Nous utiliserons la donnée de l'exercice 5.

Déroulé

Mise en route de cron

A défaut d'accéder à un serveur, on va activer temporairement cron sur votre instance WSL2. Dans une console linux :

```
# On regarde si cron tourne (en principe non)
sudo service cron status
# S'il n'est en effet pas activé, on le démarre
sudo service cron start
```

Vous trouverez plein de docs sur crontab sur le net. Par exemple : https://www.linuxtricks.fr/wiki/cron-et-crontab-le-planificateur-de-taches.

Utilisation de cron

On va commencer par une tâche bêbête : créer un fichier et changer sa date de modif. Comme ça on saura si cron marche

`crontab -e` pour éditer le ficher de config cron.

Et on ajoute la ligne

```
* * * * * touch ~/cron-marche.txt
```

Sauvegardez.

Attendons maintenant une minute, et le fichier devrait avoir été créé dans ~/cron-marche.txt : `ls -la ~/` devrait vous le montrer.

Attendez encore une minute ou deux, recommencez la commande ls, la date de modif du fichier doit avoir changé. Ca marche.

Publication/mise à jour automatique de la donnée

Vous avez noté la commande utilisée pour la publication de l'exercice 5 ? Allez, on l'automatise.

A savoir que cron ne sait pas tout. Pas tout ce qu'on sait. Il n'a pas accès à nos variables d'environnement. Le mot de passe postgresql par exemple. Et ne sait pas toujours où trouver les fichiers exécutables, en dehors de la base. On va donc lui faciliter la tâche :

1. on localise le chemin complet vers la commande ogr2ogr : `which ogr2ogr`. On utilisera ce chemin pour l'appeler dans la ligne cron
2. on va faire simple, on va utiliser le mot de passe et les autres valeurs en clair dans la commande. Ajustez votre commande ogr2ogr en fonction. Dans la "vie réelle", on procéderait différemment, par exemple avec un fichier `.pgpass`.
3. allez, zou, `crontab -l` et programmez une publi de votre fichier toutes les 5 minutes (on pensera à supprimer cette tâche en fin de cours, afin de ne pas republier cette donnée éternellement).

Faites un `crontab -l` et faites-en une copie d'écran, pour le livrable de cet exercice.

Cela vous donne-t-il des idées de tâches de votre quotidien (pro) que vous pourriez automatiser ainsi ? On peut prendre un instant pour en discuter.

Exercice suivant

Exercice 8 : faisons une 'appli' basique de crowdsourcing

Nous allons faire cet exercice ensemble.

Google Sheet est capable d'exposer son contenu sur le web au format CSV : Fichier->Partager->Publier sur le web.
Il faut bien faire attention à

- définir un lien pour le document complet, pas juste la feuille
- choisir un format CSV
- activer la publication

Source de donnée

Nous allons utiliser une feuille Google Sheet pour collecter des observations d'ours sur l'Ariège.

J'ai créé une feuille, sur laquelle vous pouvez saisir vos observations de plantigrades :

https://docs.google.com/spreadsheets/d/1S5FwbLntADv9ztYlmUrHw83PFOyWCycM5WD8308ttBo/edit?usp=sharing

Déroulé

- Créer le fichier VRT qui permet de publier cette donnée. Pour obtenir le lien correct, on va dans Fichier -> Partager-> Publier sur le web : https://docs.google.com/spreadsheets/d/e/2PACX-1vRRhuM4Y4JVH-f_ggR8EiHG8cEkqHR5hfLMzursUWTj130ffZEkRY9o8uEwUe63Nr8v-F5pFHqJRYo_/pub?output=csv
- La publier en BD
- La joindre avec la couche des communes, filtrée sur l'Ariège
- Faire une jolie carte montrant combien d'observations ont eu lieu par commune.

Avec une tâche cron, on peut mettre cette donnée à jour toutes les qq minutes, et avoir une carte quasi temps-réel pour suivre une collecte en mode collaboratif.

Lien suivant

Nettoyage

Avant de quitter votre ordi, prenez le temps de stopper votre service cron si vous l'avez lancé durant l'exercice 7: dans une console WSL2, lancez la commande

```
sudo service cron stop
```

Lien suivant

Exercice en autonomie : Base nationale consolidée des lieux de covoiturage

---

Livrable attendu Sur cet exercice, je vous demanderai de déposer sur digiforma votre fichier vrt.

Nommage attendu : nom_auto_bnlc.vrt

Sources de données

https://transport.data.gouv.fr/datasets/base-nationale-des-lieux-de-covoiturage/ prendre le fichier bnlc.csv

Déroulé

Vous allez générer un fichier VRT qui remplira les conditions suivantes:

- types de champs corrects
- le résultat doit être un fichier géospatial
- pas de téléchargement, on utilise l'URL (vsicurl)

Pensez à déposer le fichier vrt, pour le livrable de cet exercice.