Soutenance

POEI : Analytics Engineer

Mélanie Donne

Jems WE MAKE DATA INGENIOUS × DataScientest

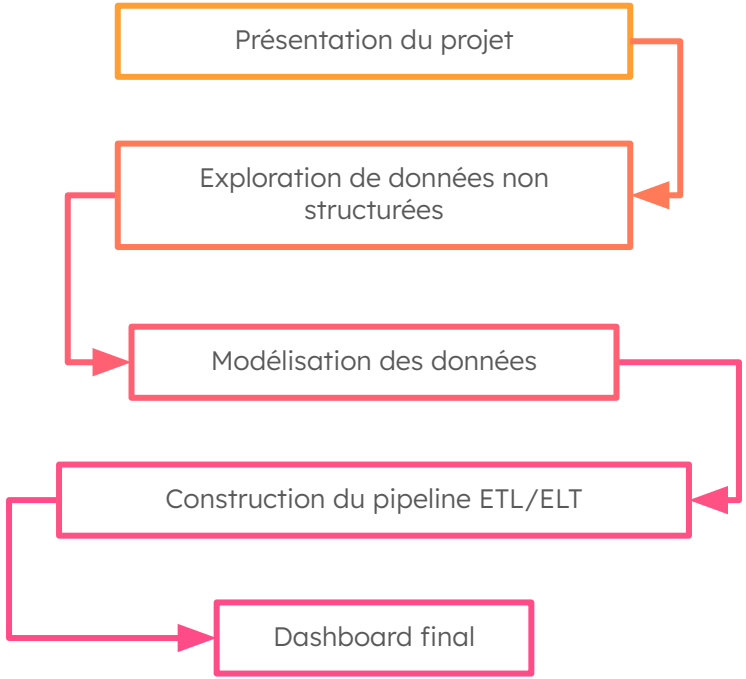
Qui suis-je ?

Basée sur la région nantaise depuis 2019

Titre RNCP Niv. 7 (Equi. Bac+5) : "Concepteur Développeur environnement Objet"

A travaillé en tant que dev et dans l'IA

Sommaire


Présentation du projet
Exploration de données non structurées
Modélisation des données
Construction du pipeline ETL/ELT
Dashboard final

Présentation du projet

Sujet : Analyse de l'immobilier à Paris en 2022

Objectif : Exploiter plusieurs sources de données pour analyser le prix de l'immobilier à Paris

Jeu de données « Demandes de valeurs foncières »

Liberté • Égalité • Fraternité
RÉPUBLIQUE FRANÇAISE

data.gouv.fr

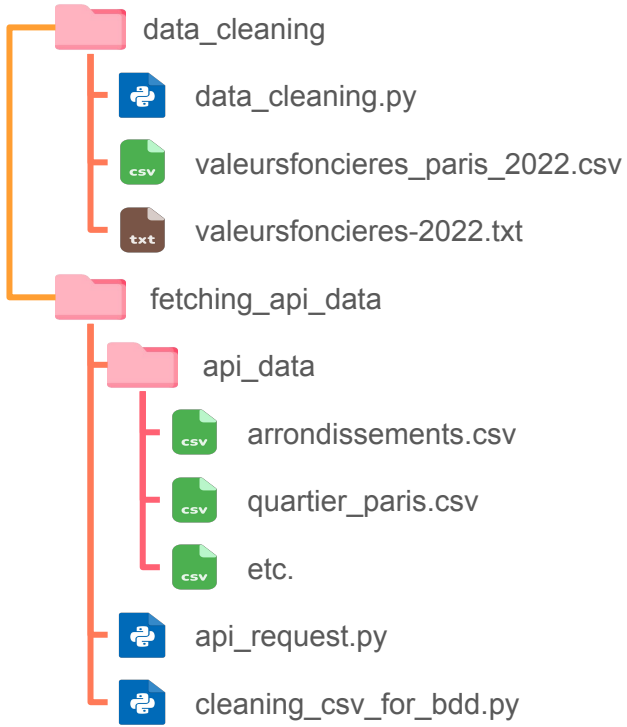
APIs

PARIS | DATA

Exploration de données non structurées

Objectifs :

- Préparation les données foncières
- Extraction des données via des requêtes API
- Nettoyage des résultats obtenus



data_cleaning.py

1. Le nettoyage des données

On détermine les codes postaux de Paris et on filtre les données afin de n'avoir que les données de cette ville

```
# Déterminer les codes postaux de Paris
paris_postal_codes = [75001, 75002, 75003, 75004, 75005, 75006,
75007, 75008, 75009, 75010, 75011, 75012, 75013, 75014, 75015, 75016,
75017, 75018, 75019, 75020]

# Filtrer les données pour n'inclure que les codes postaux de Paris
vf_paris2022 = vf2022[vf2022['Code postal'].isin(paris_postal_codes)]
```

On parcourt les colonnes avec des valeurs manquantes, on calcule ensuite le pourcentage de valeurs manquantes dans la colonnes et on supprime la colonne si le pourcentage est supérieur à 50%

```
# Parcourir les colonnes avec des valeurs manquantes
for column in columns_with_nan:
    # Calculer le pourcentage de valeurs manquantes dans la colonne
    percentage_missing = vf2022[column].isna().sum() * 100.0 /
vf2022.shape[0]
    # Supprimer la colonne si le pourcentage de valeurs manquantes
est supérieur à 50%
    if percentage_missing > 50:
        vf2022.drop(column, axis=1, inplace=True)
```

api_request.py

2. Extraction des données via des requêtes API

La fonction download_csv_from_url télécharge un fichier CSV à partir d'une URL spécifiée et le sauvegarde localement dans un répertoire appelé "api_data". Elle affiche un message de succès si le téléchargement est réussi et gère les erreurs en affichant un message explicatif si une exception se produit.

```python
def download_csv_from_url(url, destination):
    try:
        response = requests.get(url)
        if response.status_code == 200:
            api_data_dir = os.path.join(os.path.dirname(__file__), 'api_data')
            if not os.path.exists(api_data_dir):
                os.makedirs(api_data_dir)
            destination_path = os.path.join(api_data_dir, destination)
            with open(destination_path, 'wb') as f:
                f.write(response.content)
            print(f"Le fichier '{destination}' a été téléchargé avec succès.")
        else:
            print(f"Erreur lors du téléchargement du fichier '{destination}': {response.status_code}")
    except Exception as e:
        print(f"Une erreur s'est produite lors du téléchargement du fichier '{destination}': {str(e)}")
```

api_request.py

2. Extraction des données via des requêtes API

Ensuite on définit une liste contenant des tuples où chacun représente un url et son nom. On les télécharge et les enregistre dans le dossier "api_data".

```
# Les URL des différents fichiers CSV avec leurs destinations dans le dossier 'api_data'
files = [
("https://opendata.paris.fr/api/explore/v2.1/catalog/datasets/arrondissements/exports/csv?lang=fr&timezone=Europe%2FBerlin&use_labels=true&delimiter=%3B", "arrondissements.csv"),
("https://opendata.paris.fr/api/explore/v2.1/catalog/datasets/logement-encadrement-des-loyers/exports/csv?lang=fr&timezone=Europe%2FBerlin&use_labels=true&delimiter=%3B", "logement_encadrement_des_loyers.csv")
]

# Télécharger chaque fichier dans 'api_data'
for file in files:
    download_csv_from_url(file[0], file[1])
```

cleaning_csv_for_bdd.py

3. Nettoyage des résultats obtenus

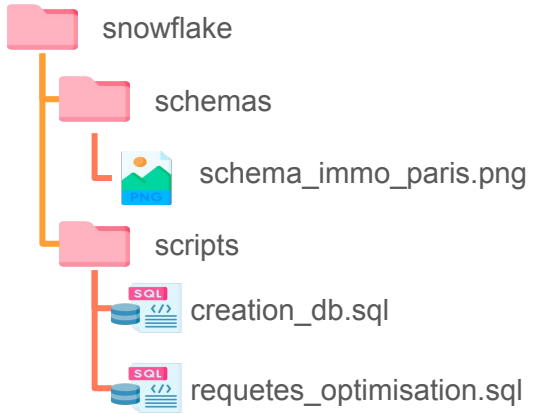
Ce code vérifie d'abord si le fichier arrondissements.csv existe. S'il existe, il charge le fichier CSV dans un DataFrame, renomme des colonnes, puis enregistre le DataFrame modifié dans le fichier CSV. Si le fichier n'est pas trouvé, il affiche un message approprié.

```
# Vérifier si le fichier des arrondissements existe
if os.path.isfile(arrondissements_csv_path):
    arrondissements_df = pd.read_csv(arrondissements_csv_path, delimiter=',')# Charger le fichier CSV des arrondissements

    arrondissements_df.rename(columns={ # Renommer les colonnes des arrondissements
        'n_sq_ar': 'IDENTIFIANT_SEQUENTIEL',
        'c_ar': 'NUMERO_ARRONDISSEMENT',
    }, inplace=True)

    arrondissements_df.to_csv(arrondissements_csv_path, sep=';', index=False)# Enregistrer le fichier CSV
    print("Les noms des en-têtes du fichier des arrondissements ont été modifiés avec succès.")
else:
    print("Le fichier 'arrondissements.csv' n'a pas été trouvé dans 'api_data'. Veuillez vérifier que le téléchargement a réussi.")
```

Modélisation des données

Objectifs :

- Concevoir un schéma de base de données
- Créer une base de données
- Stocker les données dans une base de données relationnelle



1. Optimisation et Prétraitement des Données Géospatiales avec Mapshaper

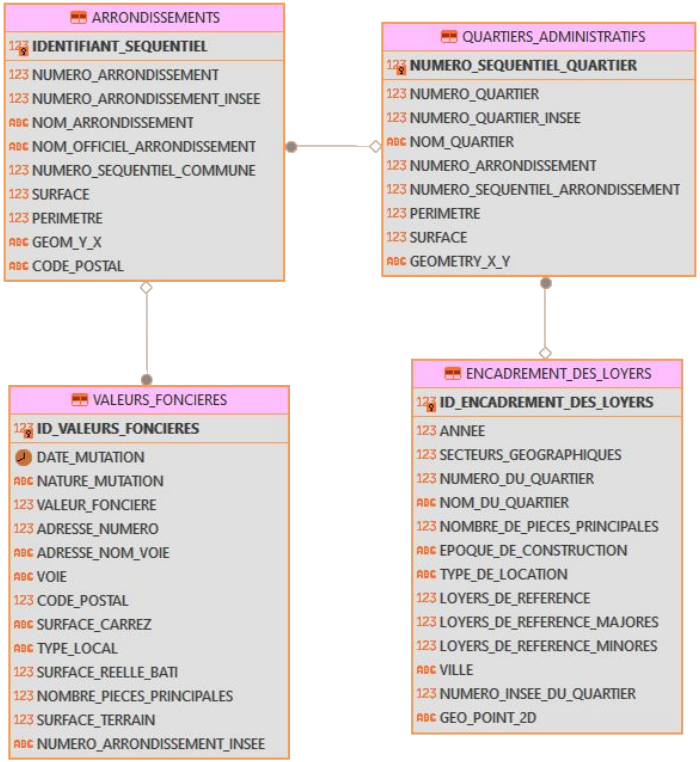
Il a été décidé d'utiliser Mapshaper pour simplifier les données géospatiales avant leur importation dans la base de données.


```
ARRONDISSEMENTS AVANT = {
    "IDENTIFIANT_SEQUENTIEL": "750000001",
    "NUMERO_ARRONDISSEMENT": "1",
    "NUMERO_ARRONDISSEMENT_INSEE": "75101",
    "NOM_ARRONDISSEMENT": "1er Ardt",
    "NOM_OFFICIEL_ARRONDISSEMENT": "Louvre",
    "N_SQ_CO": "750001537",
    "SURFACE": "1824612.86048666",
    "PERIMETRE": "6054.93686218",
    "GEOMETRY_X_Y": "48.862562701836005,
2.3364433620533878",
    "GEOMETRY": '{"coordinates": [[[2.328007329038849,
48.86991742140714], [2.329965588686571, 48.868514169174276],
[2.3303067953208765, 48.86835619167468], ...
[2.328007329038849, 48.86991742140714]]], "type":
"Polygon"}'}
```


Cela a permis de réduire la taille des fichiers, d'améliorer les performances et la qualité des données, tout en offrant une visualisation et une validation pratiques avant l'importation.


```
ARRONDISSEMENTS APRES = {
    "IDENTIFIANT_SEQUENTIEL_ARRONDISSEMENT": "750000001",
    "NUMERO_ARRONDISSEMENT": "1",
    "NUMERO_ARRONDISSEMENT_INSEE": "75101",
    "NOM_ARRONDISSEMENT": "1er Ardt",
    "NOM_OFFICIEL_ARRONDISSEMENT": "Louvre",
    "N_SQ_CO": "750001537",
    "VILLE": "PARIS",
    "SURFACE": "1824612.8604866600",
    "PERIMETRE": "6054.9368621800",
    "GEOMETRY_X_Y": '{"lon": 2.3364433620533878, "lat": 48.862562701836005}'}
```

2. Schéma de base de données



Ce schéma représente les différentes entités et de leurs relations dans une base de données pour la gestion des données immobilières et urbaines à Paris.

Exemple de relation :
arrondissements et quartiers_administratifs

Un arrondissement peut avoir plusieurs quartiers administratifs (1:N)

creation_db.sql

3. Création de la base de données

1. Création de la table "VALEURS_FONCIERES"

```
CREATE TABLE "ANALYSE_IMMO_PARIS_2022"."PUBLIC"."VALEURS_FONCIERES" ( DATE_MUTATION DATE , NATURE_MUTATION VARCHAR , VALEUR_FONCIERE
NUMBER(38, 1) , ADRESSE_NUMERO NUMBER(38, 1) , ADRESSE_NOM_VOIE VARCHAR , Voie VARCHAR , CODE_POSTAL NUMBER(38, 1) , COMMUNE VARCHAR ,
Code_departement NUMBER(38, 0) , Code_commune NUMBER(38, 0) , SURFACE_CARREZ VARCHAR , TYPE_LOCAL VARCHAR , SURFACE_REELLE_BATI NUMBER(38, 1)
, NOMBRE_PIECES_PRINCIPALES NUMBER(38, 1) , SURFACE_TERRAIN NUMBER(38, 1) );

CREATE TEMP FILE FORMAT "ANALYSE_IMMO_PARIS_2022"."PUBLIC"."temp_file_format_2024-04-29T09:53:02.537Z"
    TYPE=CSV
    SKIP_HEADER=1
    FIELD_DELIMITER=','
    TRIM_SPACE=TRUE
    FIELD_OPTIONALLY_ENCLOSED_BY='"'
    REPLACE_INVALID_CHARACTERS=TRUE
    DATE_FORMAT=AUTO
    TIME_FORMAT=AUTO
    TIMESTAMP_FORMAT=AUTO;

COPY INTO "ANALYSE_IMMO_PARIS_2022"."PUBLIC"."VALEURS_FONCIERES"
FROM (SELECT $1, $2, $3, $4, $5, $6, $7, $8, $9, $10, $11, $12, $13, $14, $15
    FROM '@"ANALYSE_IMMO_PARIS_2022"."PUBLIC"."__snowflake_temp_import_files__"')
FILES = ('2024-04-29T09:52:52.767Z/valeurs_foncieres_paris_2022.csv')
FILE_FORMAT = '"ANALYSE_IMMO_PARIS_2022"."PUBLIC"."temp_file_format_2024-04-29T09:53:02.537Z"'
ON_ERROR=ABORT_STATEMENT
```

2. Création d'un format de fichier temporaire spécifique pour les fichiers CSV

3. Chargement des données à partir d'un fichier CSV. Les données sont extraites du fichier CSV et chargées dans la table en utilisant le format de fichier temporaire spécifié. En cas d'erreur lors du chargement, l'opération est arrêtée pour éviter les problèmes de données.

requetes_optimisation.sql

4. Requêtes diverses pour optimisation

Optimiser sa base de données avant de créer notre nouveau schéma et cela permet de maximiser les performances, la fiabilité et de contribuer à maintenant l'intégrité des données

```sql
CREATE SEQUENCE encadrement_des_loyers_seq;

ALTER TABLE ENCADREMENT_DES_LOYERS
ADD COLUMN ID_ENCADREMENT_DES_LOYERS INT;

UPDATE ENCADREMENT_DES_LOYERS
SET ID_ENCADREMENT_DES_LOYERS = encadrement_des_loyers_seq.NEXTVAL;

ALTER TABLE ENCADREMENT_DES_LOYERS
ADD CONSTRAINT PK_ENCADREMENT_DES_LOYERS PRIMARY KEY
(ID_ENCADREMENT_DES_LOYERS);
```

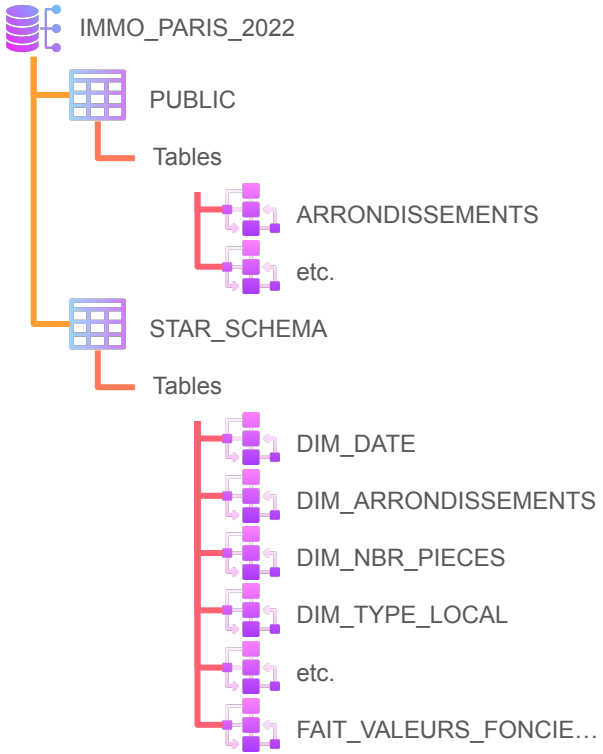
Création d'une colonne ID auto-incrémentée

Création d'une séquence pour générer des valeurs séquentielles, ajoute une colonne à une table existante et met à jour cette colonne avec les valeurs générées par la séquence. Enfin, définition de la colonne ajoutée comme clé primaire si nécessaire.

Construction du pipeline ETL/ELT

Objectifs :

- Dénormaliser (schéma en constellation) et charger les données dans le système cible
- Transformer la donnée



creation_constellation_schema.sql

1. Dénormalisation

Création de la table de dimension "DIM_ARRONDISSEMENTS"

```
CREATE TABLE ANALYSE_IMMO_PARIS_2022.SCHEMA.DIM_ARRONDISSEMENTS AS
SELECT * FROM ANALYSE_IMMO_PARIS_2022.PUBLIC.ARRONDISSEMENTS;
```

Création d'une table de faits qui concentre les mesures principales et de ses tables de dimension qui fournissent du contexte pour ces mesures

Création de la table de faits "FAIT_VALEURS_FONCIERES"

```
CREATE TABLE ANALYSE_IMMO_PARIS_2022.SCHEMA.FAIT_VALEURS_FONCIERES (
    ID_VALEURS_FONCIERES_PARIS INT PRIMARY KEY,
    ... -- autres colonnes
    FOREIGN KEY (...) REFERENCES DIM_ARRONDISSEMENTS(IDENTIFIANT_SEQUENTIEL_ARRONDISSEMENT)
    ... -- autres clés étrangères
);
```

creation_constellation_schema.sql

2. Chargement dans le système cycle

Insertion des données de la source dans la table de faits

```
INSERT INTO ANALYSE_IMMO_PARIS_2022.SCHEMA.FAIT_VALEURS_FONCIERES, --etc
SELECT ID_VALEURS_FONCIERES_PARIS, --etc
ANALYSE_IMMO_PARIS_2022.PUBLIC.VALEURS_FONCIERES_PARIS;
```

Mise à jour de la table "Fait_Valeurs_Foncieres_Paris" en remplaçant la valeur de la colonne "IDENTIFIANT_SEQUENTIEL_ARRONDISSEMENT" par la valeur correspondante de la table "DIM_ARRONDISSEMENTS", lorsque les numéros d'arrondissement correspondent entre les deux tables.

```
UPDATE FAIT_VALEURS_FONCIERES
SET FAIT_VALEURS_FONCIERES.IDENTIFIANT_SEQUENTIEL_ARRONDISSEMENT = DIM_ARRONDISSEMENTS.IDENTIFIANT_SEQUENTIEL_ARRONDISSEMENT
FROM DIM_ARRONDISSEMENTS
WHERE FAIT_VALEURS_FONCIERES.NUMERO_ARRONDISSEMENT = DIM_ARRONDISSEMENTS.NUMERO_ARRONDISSEMENT;
```

creation_constellation_schema.sql

3. Transformation

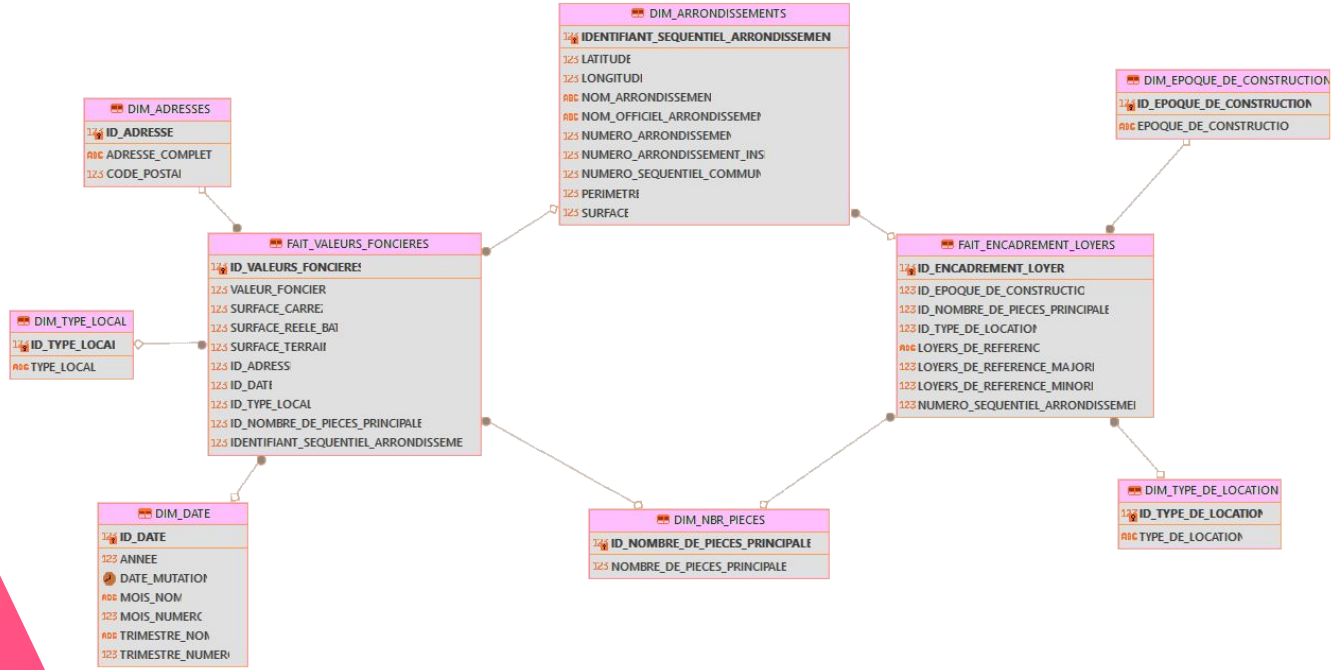
```sql
ALTER TABLE ANALYSE_IMMO_PARIS_2022.SCHEMA.DIM_ARRONDISSEMENTS
ADD COLUMN LONGITUDE DECIMAL(10, 8),
ADD COLUMN LATITUDE DECIMAL(10, 8);

UPDATE ANALYSE_IMMO_PARIS_2022.SCHEMA.DIM_ARRONDISSEMENTS
SET
    GEOM_Y_X = REPLACE(REPLACE(REPLACE(GEOM_Y_X, '{"lon":', ''),
'"lat":', ''), '}', '');
```

Ajout de colonnes pour stocker les coordonnées géographiques des arrondissements, puis mise à jour des valeurs existantes dans une colonne spécifique pour extraire et stocker ces informations dans les nouvelles colonnes.

```sql
DELETE FROM
ANALYSE_IMMO_PARIS_2022.SCHEMA.FAIT_ENCADREMENT_DES_LOYERS
WHERE ANNEE <> 2022;

ALTER TABLE
ANALYSE_IMMO_PARIS_2022.SCHEMA.FAIT_ENCADREMENT_DES_LOYERS
DROP COLUMN ANNEE;
```

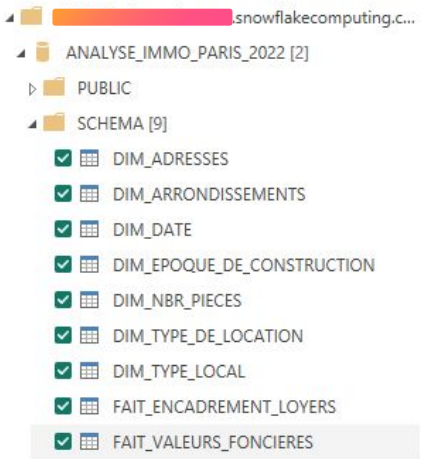
Suppression des données de l'année autre que 2022 de la table et supprime ensuite la colonne "ANNEE" de cette même table.

Schéma en étoile



Dashboard final

Connexion entre Power Bi et Snowflake afin de récupérer nos données



Choix des tables et importation

Dernières modifications si nécessaire sur les tables


On peut enfin aller voir le dashboard !