Préparation données pour SQL Server

Import des librairies

Entrée [1]:

```
import pandas as pd
import numpy as np
```

Import des données

Entrée [2]:

```
data2017=pd.read_csv("C:/Users/amand/Desktop/ProjetEcole/Data/DVF2017.csv", low_memory=Fals
data2017.head()
```

Out[2]:

	id_mutation	date_mutation	numero_disposition	nature_mutation	valeur_fonciere	adresse_nu
0	2017-1	2017-01-02	1	Vente	27000.0	
1	2017-2	2017-01-05	1	Vente	115000.0	
2	2017-3	2017-01-06	1	Vente	1.0	
3	2017-3	2017-01-06	1	Vente	1.0	
4	2017-3	2017-01-06	1	Vente	1.0	

5 rows × 40 columns

Entrée [3]:

```
data2018=pd.read_csv("C:/Users/amand/Desktop/ProjetEcole/Data/DVF2018.csv", low_memory=Fals
data2018.head()
```

Out[3]:

	id_mutation	date_mutation	numero_disposition	nature_mutation	valeur_fonciere	adresse_nu
0	2018-1	2018-01-03	1	Vente	109000.0	
1	2018-1	2018-01-03	1	Vente	109000.0	
2	2018-2	2018-01-04	1	Vente	239300.0	
3	2018-2	2018-01-04	1	Vente	239300.0	
4	2018-2	2018-01-04	1	Vente	239300.0	

5 rows × 40 columns

Entrée [4]:

```
data2019=pd.read_csv("C:/Users/amand/Desktop/ProjetEcole/Data/DVF2019.csv", low_memory=Fals
data2019.head()
```

Out[4]:

	id_mutation	date_mutation	numero_disposition	nature_mutation	valeur_fonciere	adresse_nu
0	2019-1	2019-01-11	1	Vente	84000.0	
1	2019-1	2019-01-11	1	Vente	84000.0	
2	2019-2	2019-02-08	1	Vente	210000.0	5
3	2019-2	2019-02-08	1	Vente	210000.0	5
4	2019-3	2019-04-04	1	Vente	36000.0	

5 rows × 40 columns

Description du dataset

- `id_mutation` : Identifiant de mutation (non stable, sert à grouper les lignes)
- `date_mutation` : Date de la mutation au format ISO-8601 (YYYY-MM-DD)
- `numero_disposition` : Numéro de disposition
- `valeur_fonciere` : Valeur foncière (séparateur décimal = point)
- `adresse_numero` : Numéro de l'adresse
- `adresse_suffixe` : Suffixe du numéro de l'adresse (B, T, Q)
- `adresse_code_voie` : Code FANTOIR de la voie (4 caractères)
- `adresse_nom_voie` : Nom de la voie de l'adresse

- `code_postal` : Code postal (5 caractères)
- `code_commune` : Code commune INSEE (5 caractères)
- `nom_commune` : Nom de la commune (accentué)
- `ancien_code_commune` : Ancien code commune INSEE (si différent lors de la mutation)
- `ancien_nom_commune` : Ancien nom de la commune (si différent lors de la mutation)
- `code_departement` : Code département INSEE (2 ou 3 caractères)
- `id_parcelle` : Identifiant de parcelle (14 caractères)
- `ancien_id_parcelle` : Ancien identifiant de parcelle (si différent lors de la mutation)
- `numero_volume` : Numéro de volume
- `lot_1_numero` : Numéro du lot 1
- `lot_1_surface_carrez` : Surface Carrez du lot 1
- `lot_2_numero` : Numéro du lot 2
- `lot_2_surface_carrez` : Surface Carrez du lot 2
- `lot_3_numero` : Numéro du lot 3
- `lot_3_surface_carrez` : Surface Carrez du lot 3
- `lot_4_numero` : Numéro du lot 4
- `lot_4_surface_carrez` : Surface Carrez du lot 4
- `lot_5_numero` : Numéro du lot 5
- `lot_5_surface_carrez` : Surface Carrez du lot 5
- `nombre_lots` : Nombre de lots
- `code_type_local` : Code de type de local
- `type_local` : Libellé du type de local
- `surface_reelle_bati` : Surface réelle du bâti
- `nombre_pieces_principales` : Nombre de pièces principales
- `code_nature_culture` : Code de nature de culture
- `nature_culture` : Libellé de nature de culture
- `code_nature_culture_speciale` : Code de nature de culture spéciale
- `nature_culture_speciale` : Libellé de nature de culture spéciale
- `surface_terrain` : Surface du terrain
- `longitude` : Longitude du centre de la parcelle concernée (WGS-84)
- `latitude` : Latitude du centre de la parcelle concernée (WGS-84)

Remarques :

- je vois que :
  - les datasets ont beaucoup de colonnes
  - certaines colonnes ont beaucoup de NaN
  - les `id_mutation` et `id_parcelle` ne sont pas uniques
  - il manque un `id_bien`
- je peux supposer qu'il faudrait
  - réduire le nombre de colonnes
  - avoir des id uniques
  - traiter les NaN

Feature Engineering & Data Analysis round 1

Concaténation des dataframes et mise en place d'un DF DVF unique

Entrée [5]:

```
dvf=pd.concat([data2017, data2018, data2019])
dvf.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 7453214 entries, 0 to 1017153
Data columns (total 40 columns):
 #   Column                          Dtype
---  ------                          -----
 0   id_mutation                     object
 1   date_mutation                   object
 2   numero_disposition              int64
 3   nature_mutation                 object
 4   valeur_fonciere                 float64
 5   adresse_numero                  float64
 6   adresse_suffixe                 object
 7   adresse_nom_voie                object
 8   adresse_code_voie               object
 9   code_postal                     float64
 10  code_commune                    object
 11  nom_commune                     object
 12  code_departement                object
 13  ancien_code_commune             float64
 14  ancien_nom_commune              object
 15  id_parcelle                     object
 16  ancien_id_parcelle              object
 17  numero_volume                   object
 18  lot1_numero                     object
 19  lot1_surface_carrez             float64
 20  lot2_numero                     object
 21  lot2_surface_carrez             float64
 22  lot3_numero                     object
 23  lot3_surface_carrez             float64
 24  lot4_numero                     float64
 25  lot4_surface_carrez             float64
 26  lot5_numero                     object
 27  lot5_surface_carrez             float64
 28  nombre_lots                     int64
 29  code_type_local                 float64
 30  type_local                      object
 31  surface_reelle_bati             float64
 32  nombre_pieces_principales       float64
 33  code_nature_culture             object
 34  nature_culture                  object
 35  code_nature_culture_speciale    object
 36  nature_culture_speciale         object
 37  surface_terrain                 float64
 38  longitude                       float64
 39  latitude                        float64
dtypes: float64(16), int64(2), object(22)
memory usage: 2.3+ GB
```

Entrée [6]:

```
dvf=dvf.drop(['numero_disposition',
              'code_postal',
              'adresse_numero',
              'adresse_suffixe',
              'adresse_code_voie',
              'code_commune',
              'ancien_code_commune',
              'ancien_nom_commune',
              'ancien_id_parcelle',
              'numero_volume',
              'lot1_numero',
              'lot1_surface_carrez',
              'lot2_numero',
              'lot2_surface_carrez',
              'lot3_numero',
              'lot3_surface_carrez',
              'lot4_numero',
              'lot4_surface_carrez',
              'lot5_numero',
              'lot5_surface_carrez',
              'code_nature_culture',
              'nature_culture',
              'code_nature_culture_speciale',
              'nature_culture_speciale'], axis=1)
dvf.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 7453214 entries, 0 to 1017153
Data columns (total 16 columns):
 #   Column                    Dtype
---  ------                    -----
 0   id_mutation               object
 1   date_mutation             object
 2   nature_mutation           object
 3   valeur_fonciere           float64
 4   adresse_nom_voie          object
 5   nom_commune               object
 6   code_departement          object
 7   id_parcelle               object
 8   nombre_lots               int64
 9   code_type_local           float64
 10  type_local                object
 11  surface_reelle_bati       float64
 12  nombre_pieces_principales float64
 13  surface_terrain           float64
 14  longitude                 float64
 15  latitude                  float64
dtypes: float64(7), int64(1), object(8)
memory usage: 966.7+ MB
```

Entrée [7]:

```
dvf=dvf.dropna(subset=['valeur_fonciere'])
```

Entrée [8]:

```
dvf= dvf.drop_duplicates(subset=['id_mutation'], keep='first')
```

Entrée [9]:

```
dvf= dvf.drop_duplicates(subset=['id_parcelle'], keep='last')
```

Entrée [10]:

```
dvf.head()
```

Out[10]:

	id_mutation	date_mutation	nature_mutation	valeur_fonciere	adresse_nom_voie	nom_comm
1	2017-2	2017-01-05	Vente	115000.0	LES VAVRES	Péro
2	2017-3	2017-01-06	Vente	1.0	LA POIPE	Saint-Cyr Mer
5	2017-4	2017-01-09	Vente	1.0	MONTGRIMOUX CENTRE	Fe
6	2017-5	2017-01-03	Vente	258000.0	IMP DES PINSONS	Saint-Denis E
10	2017-6	2017-01-05	Vente	175050.0	SAINT MICHEL	Val-Rever

Création d'id unique pour SQL Server

Entrée [11]:

```
dvf['id_bien']= dvf['id_parcelle'] +'-'+ dvf['code_departement']
```

Entrée [12]:

```
dvf.head()
```

Out[12]:

	id_mutation	date_mutation	nature_mutation	valeur_fonciere	adresse_nom_voie	nom_comm
1	2017-2	2017-01-05	Vente	115000.0	LES VAVRES	Péro
2	2017-3	2017-01-06	Vente	1.0	LA POIPE	Saint-Cyr Mer
5	2017-4	2017-01-09	Vente	1.0	MONTGRIMOUX CENTRE	Fe
6	2017-5	2017-01-03	Vente	258000.0	IMP DES PINSONS	Saint-Denis E
10	2017-6	2017-01-05	Vente	175050.0	SAINT MICHEL	Val-Rever

Entrée [13]:

```
dvf.describe()
```

Out[13]:

	valeur_fonciere	nombre_lots	code_type_local	surface_reelle_bati	nombre_pieces_princi
count	2.224142e+06	2.224142e+06	1.348953e+06	1.223008e+06	1.346962
mean	2.052168e+05	2.253525e-01	1.535372e+00	1.196465e+02	3.31552(
std	2.292359e+06	7.998093e-01	8.871148e-01	5.772342e+02	1.988447
min	1.000000e-02	0.000000e+00	1.000000e+00	1.000000e+00	0.000000
25%	4.166666e+04	0.000000e+00	1.000000e+00	6.400000e+01	2.000000
50%	1.200000e+05	0.000000e+00	1.000000e+00	8.800000e+01	4.000000
75%	2.190000e+05	0.000000e+00	2.000000e+00	1.150000e+02	5.000000
max	1.750000e+09	3.300000e+02	4.000000e+00	2.778140e+05	1.120000

Création des 3 DF Mutation, Cadastre et Bien

Entrée [14]:

```
dvfMutation=dvf.drop(['adresse_nom_voie',
                      'nom_commune',
                      'code_departement',
                      'nombre_lots',
                      'code_type_local',
                      'type_local',
                      'surface_reelle_bati',
                      'nombre_pieces_principales',
                      'surface_terrain',
                      'longitude',
                      'latitude'], axis= 1)
dvfMutation.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2224142 entries, 1 to 1017132
Data columns (total 6 columns):
 #   Column           Dtype
---  ------           -----
 0   id_mutation      object
 1   date_mutation    object
 2   nature_mutation  object
 3   valeur_fonciere  float64
 4   id_parcelle      object
 5   id_bien          object
dtypes: float64(1), object(5)
memory usage: 118.8+ MB
```

Entrée [15]:

```
dvfCadastre=dvf.drop(['id_mutation',
                      'date_mutation',
                      'nature_mutation',
                      'valeur_fonciere',
                      'code_type_local',
                      'type_local',
                      'surface_reelle_bati',
                      'nombre_pieces_principales',
                      'surface_terrain',
                      'id_bien'], axis=1)
dvfCadastre.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2224142 entries, 1 to 1017132
Data columns (total 7 columns):
 #   Column            Dtype  
---  ------            -----  
 0   adresse_nom_voie  object 
 1   nom_commune       object 
 2   code_departement  object 
 3   id_parcelle       object 
 4   nombre_lots       int64  
 5   longitude         float64
 6   latitude          float64
dtypes: float64(2), int64(1), object(4)
memory usage: 135.8+ MB
```

Entrée [16]:

```
dvfBien=dvf.drop(['id_parcelle',
                  'adresse_nom_voie',
                  'nom_commune',
                  'code_departement',
                  'nombre_lots',
                  'longitude',
                  'latitude',
                  'id_mutation',
                  'date_mutation',
                  'nature_mutation',
                  'valeur_fonciere'], axis=1)
dvfBien.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2224142 entries, 1 to 1017132
Data columns (total 6 columns):
 #   Column                    Dtype  
---  ------                    -----  
 0   code_type_local           float64
 1   type_local                object 
 2   surface_reelle_bati       float64
 3   nombre_pieces_principales  float64
 4   surface_terrain           float64
 5   id_bien                   object 
dtypes: float64(4), object(2)
memory usage: 118.8+ MB
```

Export en csv

Entrée [17]:

```
dvfMutation.to_csv('dvfMutation.csv', index=False)
```

Entrée [18]:

```
dvfCadastre.to_csv('dvfCadastre.csv', index=False)
```

Entrée [19]:

```
dvfBien.to_csv('dvfBien.csv', index=False)
```

Entrée [20]:

```
dvf.to_csv('dvf.csv', index=False)
```

Modele EA

Le modèle entité-association (EA) (le terme « entité-relation » est une traduction erronée largement répandue), ou diagramme entité-association ou (en anglais « entity-relationship diagram », abrégé en ERD), est un modèle de données ou diagramme pour des descriptions de haut niveau de modèles conceptuels de données. Il a été conçu par Peter Chen dans les années 1970 afin de fournir une notation unifiée pour représenter les informations gérées par les systèmes de gestion de bases de données de l'époque. Il fournit une description graphique pour représenter des modèles de données sous la forme de diagrammes contenant des entités et des associations. De tels modèles sont utilisés dans les phases amont de conception des systèmes informatiques.

Entrée [ ]:

```

```