Rapport: Extraction des données Avito.ma

AZZIM Taha
INPT INE1 DATA ENGINEERING

9 Mars 2020

Contents

Objectif:

Le but de ce rapport est de résumer l'ensemble des étapes suivies pour extraire les données depuis le site internet Avito.ma et les stocker dans un fichier CSV. Ces données vont être utilisées par la suite dans notre prochain projet (Estimation des prix des maisons à Rabat).

1 Conception de la logique du site

La première page de la section *immobilier* > *offre* est comme suite :

Figure 1: Avito.ma

Le grattage de le première page donne des informations sur le prix et le lieu et parfois des autres détails qui sont inclus dans la zone de titre. Nous avons donc besion de plus d'informations qui sont plus structurées.
En cliquant sur une annonce on obtient la page suivante:

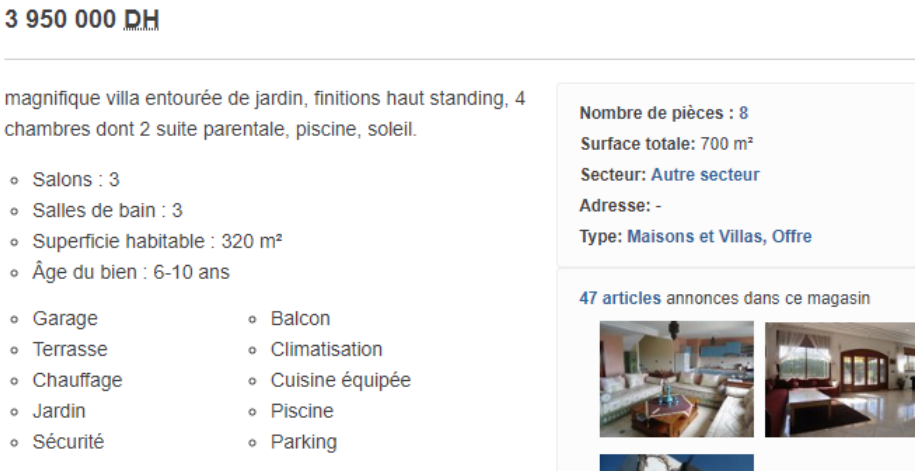

Figure 2: Détails d'annonce

Remarquons que cette page nous a offert plus d'inforamtions sur l'annonce ainsi qu'elles sont bien structurées. L'idée donc est de traiter annonce par annonce pour avoir accès aux données et les stockées en *DataFrame*

2 Grattage et code Python

Dans un premier temps on va tirer les données d'une seule annonce à partir de la première page, de toutes les annonces et finalement on va répéter le même processus pour plusieurs pages à l'aide d'une boucle.

Dans la Figure 2 on a 5 sections à traiter: Titre d'annonce, Prix, informations générales, informations détaillées et informations supplémentaires qu'on a encadrées dans la figure qui suit:

Figure 3: Sections

On va essayer par la suite de savoir la classe au quelle appartient chaque section puis réaliser un programme Python qui récupère ses données. Si on inspecte la page de l'annonce on peut facilement avoir la classe de chaque partie. Le tableaux suivant donne les résultats :

Section	classe
Titre	'page-header mbm'
Prix	'amount value'
Infos générales	'font-normal fs12 no-margin ln22'
Infos détaillés	none
Infos supplémentaires	'ul-flex-column'

La section Infos détaillés n'a pas de nom de classe ni d'autre identifiant ce qui va nous a poser un problème lors de la recherche. On va voir comment on va dépasser ce petit problème.
Commençons par faire appel à toute les bibliothèques qu'on va utiliser.

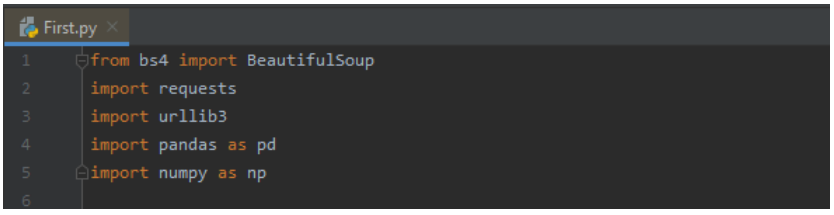

Figure 4: Bibliothèques

Par la suite on ne va plus afficher cette partie du code.

Maintenant on va récupérer le premier élément: Titre de l'annonce. Vu qu'on posséde du nom de la classe à laquelle il appartient, le code suivant nous a retourné le résultat souhaité.

```
7   def make_soup(url):
8       http = urllib3.PoolManager()
9       r = http.request("GET", url)
10      return BeautifulSoup(r.data,'html.parser')
11
12  soup = make_soup('https://www.avito.ma/fr/el_jadida/appartements/Appartement_de_60_m2_pres_marjane_35095825.htm')
13
14  item = soup.find_all(class_='page-header mbm')
15  print(item[0].text)
```

Figure 5: Code 1

```
Appartement de 60 m2 pres marjane
ANNONCE DE PROFESSIONNEL


Process finished with exit code 0
```

Figure 6: Résultat

De même pour le prix:

```
7   def make_soup(url):
8       http = urllib3.PoolManager()
9       r = http.request("GET", url)
10      return BeautifulSoup(r.data,'html.parser')
11
12  soup = make_soup('https://www.avito.ma/fr/el_jadida/appartements/Appartement_de_60_m2_pres_marjane_35095825.htm')
13
14  prix = soup.find_all(class_='amount value')
15  print(prix[0].text)
```

Figure 7: Code 2

```
C:\Users\toshiba\Anaconda3\envs\First.py\python.exe
340 000
```

Figure 8: Résultat

De même pour les informations générales:

```
def make_soup(url):
    http = urllib3.PoolManager()
    r = http.request("GET", url)
    return BeautifulSoup(r.data,'html.parser')

soup = make_soup('https://www.avito.ma/fr/el_jadida/appartements/Appartement_de_60_m2_pres_marjane_35095825.htm')

info_generales = soup.find_all(class_='font-normal fs12 no-margin ln22')
for x in info_generales:
    print(x.text)
```

Figure 9: Code 3

```
Nombre de pièces :
2

Surface totale:
60 m²

Prix / m²: 5666 DH/m²
Secteur:
Autre secteur

Adresse: -
Type:
Appartements, Offre


Process finished with exit code 0
```

Figure 10: Résultat

Pour les informations détaillées, cette section n'a pas de classe. Mais elle est enfant de la classe '*span10*', on va donc extraire le code de cette classe puis accéder aux informations détaillées.

```python
def make_soup(url):
    http = urllib3.PoolManager()
    r = http.request("GET", url)
    return BeautifulSoup(r.data,'html.parser')

soup = make_soup('https://www.avito.ma/fr/el_jadida/appartements/Appartement_de_60_m2_pres_marjane_35095825.htm')

info_detaille_class = soup.find_all(class_='span10')
info_detaille_list = info_detaille[0].find_all('li')

for x in info_detaille_list:
    print(x.text)
```

Figure 11: Code 4

```
C:\Users\toshiba\Anaconda3\envs\First.py\python.exe
 Salons : 1
 Salles de bain : 1
 Étage : 3
```

Figure 12: Résultat

Il reste à extraire les informations supplémentaires, l'annonce qu'on a içi ne présente aucune de ces informations (il ne s'agit pas donc d'un champ obligatoire), on va extraire celles d'une autre annonce juste pour voir si le code marche bien.

```python
def make_soup(url):
    http = urllib3.PoolManager()
    r = http.request("GET", url)
    return BeautifulSoup(r.data,'html.parser')

soup = make_soup('https://www.avito.ma/fr/hay_izdihar/appartements/Appartement_128m2_avec_Terrasse_Hay_Izdihar_3770

info_supp = soup.find_all(class_='ul-flex-column')

for x in info_supp:
    print(x.text)
```

Figure 13: Code 5

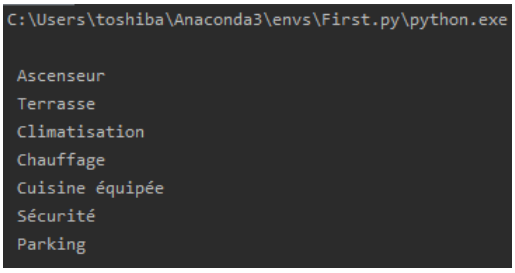

Figure 14: Résultat

On a bien pu récupérer toutes les données souhaitées, il reste donc de les stocker en *DataFrame*. Pour cela les résultats de chaque partie extraite seront stocker premièrement en une liste, puis chacune de ces listes serait la valeur d'une colonne du DataFrame. Le DataFrame semble être bizarre pour le moment, on va voir comment le bien structurer aprés.

```python
def make_soup(url):
    http = urllib3.PoolManager()
    r = http.request("GET", url)
    return BeautifulSoup(r.data,'html.parser')

soup = make_soup('https://www.avito.ma/fr/hay_izdihar/appartements/Appartement_128m2_avec_Terrasse_Hay_Izdihar_37706996.htm')

def get_title(soup):
    item = soup.find_all(class_='page-header mbm')
    return item[0]

def get_price(soup):
    prix = soup.find_all(class_='amount value')
    return prix[0]

def get_info_generale(soup):
    info_generale = soup.find_all(class_='font-normal fs12 no-margin ln22')
    info_generale_list = []
    for x in info_generale:
        info_generale_list.append(x.text)
    return info_generale_list

def get_info_detaille(soup):
    info_detaille_class = soup.find_all(class_='span10')
    info_detaille_list = info_detaille_class[0].find_all('li')
    info_detaille_list2 = []
    for x in info_detaille_list:
        info_detaille_list2.append(x.text)
    return info_detaille_list2

def get_info_supp(soup):
    info_supp = soup.find_all(class_='ul-flex-column')
    info_supp_list = []
    for x in info_supp:
        info_supp_list.append(x.text)
    return info_supp_list

Data = pd.DataFrame([[get_title(soup),get_price(soup),get_info_generale(soup),get_info_detaille(soup),get_info_supp(soup)]],
                    columns=['Titre','Prix', 'Info generales', 'Info Detailles', 'Info Supplimentaires'])
print(Data)
```

Figure 15: Code 6

On a donc bien réussit à extraire toutes les données relatives à l'annonce. Maintenant il ne faut qu'avoir les urls de toutes les annonces de la page pour les parcourir en leurs appliquant le code précedent. Remarquons que la classe des titres des annonces dans la première page est **fs14**.

Figure 16: Code html

On récupére les *href* des annonces à l'aide du programme Pythton suivant:

```
def make_soup(url):
    http = urllib3.PoolManager()
    r = http.request("GET", url)
    return BeautifulSoup(r.data,'html.parser')

soup = make_soup('https://www.avito.ma/fr/maroc/immobilier-%C3%A0_vendre')

code = soup.find_all(lambda tag: tag.get('class') == ['fs14'])
for x in code:
    print(x.a['href'])
```

Figure 17: Code 7

Une partie du résultat:

```
https://www.avito.ma/fr/dar_bouazza/appartements/APPARTEMENT_HAUT_STANDING_en_Ve_à_DAR_BOUAAZA_39902760.htm
https://www.avito.ma/fr/dar_bouazza/appartements/Bel_appartement_ensoleille_à_Dar_Bouazza_39902756.htm
https://www.avito.ma/fr/ouled_tayeb/appartements/Appartement_de_65_m2_Ouled_Tayeb_35724281.htm
https://www.avito.ma/fr/centre_ville/appartements/39563023_شقق_محطة_بتطوان.htm
https://www.avito.ma/fr/sidi_rahal_chatai/appartements/39902735_تطل_على_شاطئ_سيدي_رحال_تسليم_فوري_2م70_غرف2_شقة.htm
https://www.avito.ma/fr/el_houda/magasins_et_commerces/39831569_سناك_مجهز_للبيع.htm
https://www.avito.ma/fr/ben_msick/appartements/Occasion_à_saisir_39620798.htm
https://www.avito.ma/fr/la_gironde/appartements/Appartement_spacieux_et_lumineux___La_Résistance_37673129.htm
https://www.avito.ma/fr/bouskoura/maisons_et_villas/Villa_de_400_m2_à_700_m2_à_Bouskoura_38682616.htm
https://www.avito.ma/fr/bouskoura/maisons_et_villas/Villa_à_Bouskoura_Projet_Tasnime__400_m2_à_700_m2__38682557.htm
https://www.avito.ma/fr/bouskoura/maisons_et_villas/Villas_à_Bouskoura_de_400_m2_à_700m2_38682511.htm
```

Figure 18: Résultat

A l'aide de tout ce qui précède on peut maintenant avoir les données de toute la page et les stocker en *DataFrame*.

```
urls = get_urls(soup)
soup = make_soup(urls[0])
Data = pd.DataFrame([[get_title(soup),get_price(soup),get_info_generale(soup),get_info_detaille(soup),get_info_supp(soup)]],
                    columns=['Titre','Prix', 'Info generales', 'Info Detailles', 'Info Supplimentaires'])
for url in urls:
    soup = make_soup(url)
    Data = pd.concat([Data,pd.DataFrame([[get_title(soup),get_price(soup),get_info_generale(soup),get_info_detaille(soup),get_info_supp(soup)]],
                    columns=['Titre','Prix', 'Info generales', 'Info Detailles', 'Info Supplimentaires'])])

print(Data)
Data.to_csv("C:\\Workplace\\Mini projet inpt\\data.csv")
```

Figure 19: Code 8

Pour une mellieure visualisation des données, on peut les voir en tant que fichier CSV.

A	B	C	D	E
Titre	Prix	Info generales	Info Detailles	Info Supplimentaires
Local comercialle de 100m2 a kenitraANNONCE DE PROFESSIONNEL		['Nombre de piÃ¨ces : \n-\n', 'Surface totale:\r	[]	['\n']
Appartements 2 et 3 chambres Salon Ã Ain SebaaANN	890 000	['Nombre de piÃ¨ces : \n5\n', 'Surface totale:\	['Salons : 1 ', 'Salles de bain : 2 ', 'Ã‰tage : 1	['\n Ascenseur \n Balcon \n Cuisine Ã©quipÃ©e \n
Terrain de 9995 mÂ²ANNONCE DE PROFESSIONNEL	5 500 000	['Surface totale:\n9995 mÂ²\n', 'Secteur:\nAu	['Zoning : Service public ', 'TitrÃ© ']	['\n TitrÃ© \n']
Maison a Casablanca	650 000	['Nombre de piÃ¨ces : \n6\n', 'Surface totale:\	['Salons : 2 ', 'Salles de bain : 3 ', 'Superficie l	['\n']
Appartement de 140 m2 aux PrincessesANNONCE DE	1 999 000	['Nombre de piÃ¨ces : \n5\n', 'Surface totale:\	['Salons : 2 ', 'Salles de bain : 2 ', 'Ã‰tage : 6	['\n Ascenseur \n Balcon \n Terrasse \n Cuisine Ã©
Magasin en Vente Ã RabatANNONCE DE PROFESSIONN	700 000	['Nombre de piÃ¨ces : \n-\n', 'Surface totale:\r	[]	['\n']
appartement de 90 m2 Les Jardins de WafaANNONCE	830 000	['Nombre de piÃ¨ces : \n2\n', 'Surface totale:\	['Salons : 1 ', 'Salles de bain : 1 ', 'Ã‰tage : 2	['\n Balcon \n Climatisation \n Cuisine Ã©quipÃ©e
Terrain zone villa trÃ¨s bien situÃ©ANNONCE DE PROF	2 148 000	['Surface totale:\n-\n', 'Secteur:\nAutre secte	[]	['\n']
Appartement de 73 mÂ² Ã CalifornieANNONCE DE PRO	949 000	['Nombre de piÃ¨ces : \n4\n', 'Surface totale:\	['Salons : 1 ', 'Salles de bain : 1 ', 'Ã‰tage : 1	['\n']
Appartement NEUF de 93 mÂ² Ã Bd Brahim RoudaniAN	1 500 000	['Nombre de piÃ¨ces : \n3\n', 'Surface totale:\	['Salons : 1 ', 'Salles de bain : 2 ', 'Ã‰tage : 3	['\n Ascenseur \n Balcon \n Climatisation \n Chauf
Appartement de 189 mÂ² Ã vendre Ã MaÃ¢rif Extensi	2 390 000	['Nombre de piÃ¨ces : \n4\n', 'Surface totale:\	['Salons : 1 ', 'Salles de bain : 2 ', 'Ã‰tage : 1	['\n Ascenseur \n Terrasse \n Cuisine Ã©quipÃ©e
Appartement de 160 mÂ² Ã BouskouraANNONCE DE PROFESSIONNEL		['Nombre de piÃ¨ces : \n5\n', 'Surface totale:\	['Salons : 1 ', 'Salles de bain : 2 ', 'Ã‰tage : 2	['\n']
Luxueuse villa de 1025 mÂ² Ã vendre Ã CalifornieANN	16 000 000	['Nombre de piÃ¨ces : \n8\n', 'Surface totale:\	['Salons : 1 ', 'Salles de bain : 5 ', 'Garage ', 'E	['\n Garage \n Balcon \n Terrasse \n Climatisation
Joli appartement de 3chambres FARANNONCE DE PRO	700 000	['Nombre de piÃ¨ces : \n3\n', 'Surface totale:\	[]	['\n']
Bureau en location Ã Casablanca	3 500	['Nombre de piÃ¨ces : \n1\n', 'Surface totale:\	['Ã‰tage : 5 ', 'Soupente : 1 mÂ², ' Frais de s	['\n Ascenseur \n CÃ©blage tÃ©lÃ©phonique \n Cli
Villa bien situÃ© a sidi masoudANNONCE DE PROFESSI	1 800 000	['Nombre de piÃ¨ces : \n4\n', 'Surface totale:\	[]	['\n']
Bel Appartement Ã KÃ©nitra MehdiaANNONCE DE PR	160 000	['Nombre de piÃ¨ces : \n3\n', 'Surface totale:\	['Salons : 1 ', 'Salles de bain : 1 ', 'Ã‰tage : 3	['\n Concierge \n SÃ©curitÃ© \n Parking \n']
Maison	700 000	['Nombre de piÃ¨ces : \n9\n', 'Surface totale:\	['Salles de bain : 5 ', 'Superficie habitable : 5	['\n']
bureau au cÅ“ur de bourgogneANNONCE DE PROFESSIO	1 200 000	['Nombre de piÃ¨ces : \n3\n', 'Surface totale:\	['Ã‰tage : 1 ', 'Soupente : 10 mÂ², ' Frais de	['\n Ascenseur \n AmenagÃ© \n CÃ¢blage tÃ©lÃ©p
ApparemmentANNONCE DE PROFESSIONNEL		['Nombre de piÃ¨ces : \n3\n', 'Surface totale:\	['Salons : 1 ', 'Salles de bain : 1 ', 'Ã‰tage : 2	['\n Balcon \n Cuisine Ã©quipÃ©e \n']
AppartementANNONCE DE PROFESSIONNEL	1 107 000	['Nombre de piÃ¨ces : \n2\n', 'Surface totale:\	['Salons : 1 ', 'Salles de bain : 2 ', 'Ã‰tage : 3	['\n Ascenseur \n Balcon \n Cuisine Ã©quipÃ©e \n
Maison et villa en Vente Ã Temara	1 750 000	['Nombre de piÃ¨ces : \n10+\n', 'Surface totali	['Salons : 2 ', 'Salles de bain : 2 ', 'Superficie l	['\n Garage \n Terrasse \n Climatisation \n Chauffa
Appartements de Standing de 58 Ã 110m2 bien situÃ	490 000	['Nombre de piÃ¨ces : \n3\n', 'Surface totale:\	['Salons : 1 ', 'Salles de bain : 2 ', 'Ã‚ge du bie	['\n Garage \n Ascenseur \n Balcon \n Terrasse \n
pharmacie titrÃ©e avec fond commerce		['Nombre de piÃ¨ces : \n-\n', 'Surface totale:\r	[]	['\n']

Figure 20: Données

Pas mal pour le moment.
La dernière étape consiste à refaire la même chose pour les autres pages. Le url de la 3^{eme} page est:

`https://www.avito.ma/fr/rabat/immobilier-%C3%A0_vendre?o=3`

Le '3' indique le numéro de la page. Il sera donc possible d'avoir les urls des pages souhaitées en changeant chaque fois ce numéro à l'aide d'une boucle **for**.

```
Data = pd.DataFrame(columns=['Titre','Prix', 'Info generales', 'Info Detailles', 'Info Supplimentaires'])

premiere_page = 2
derniere_page = 50

for page in range(premiere_page,derniere_page):
    soup = make_soup('https://www.avito.ma/fr/maroc/immobilier-%C3%A0_vendre?o='+str(page))
    urls = get_urls(soup)
    for url in urls:
        soup = make_soup(url)
        Data = pd.concat([Data,pd.DataFrame([[get_title(soup),get_price(soup),get_info_generale(soup),get_info_detaille(soup),
                                              get_info_supp(soup)]],
                         columns=['Titre','Prix', 'Info generales', 'Info Detailles', 'Info Supplimentaires'])])

print(Data)
Data.to_csv("C:\\Workplace\\Mini projet inpt\\data2.csv")
```

Figure 21: Code 9

On est arrivé à la fin du grattage. On va passer maintenant à l'étape suivante: Manipulation des données

3 Manipulation des données

On dispose maintenant d'un *DataFrame* de 5 colonnes (Titre, Prix, Informations générales, Informations détaillées, Informations supplémentaires). En générale les deux premières ne posent aucun probléme sauf qu'elles ont besoin de quelques modifications qu'on va effectuer à la fin. Commençons donc par la colonne Informations générales.

3.1 Informations générales

Cette colonne contient 5 champs:

- Nombre de pièces.
- Type (Villa, Maison, ...).
- Adresse.
- Secteur.
- Surface totale en m^2.

De ce fait on va définir 5 fonctions, chacune doit exraire un de ces champs.

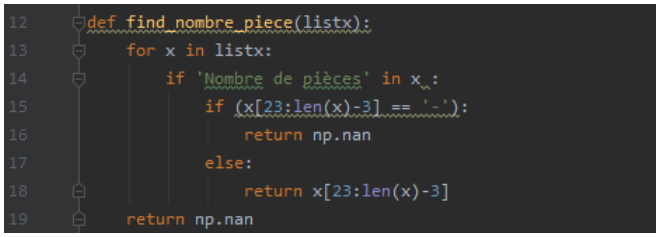

Figure 22: Nombre de pièces

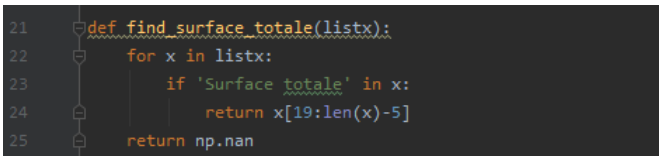

Figure 23: Surface totale

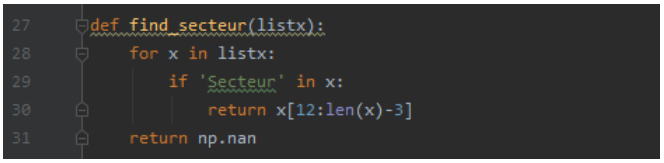

Figure 24: Secteur

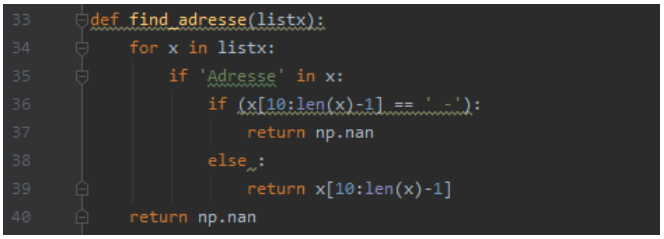

Figure 25: Adresse

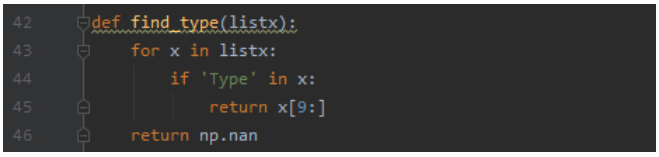

Figure 26: Type

On doit appliquer ces fonctions à la série data['Informations générales'] et puis la supprimer car elle semble être inutile par la suite.

Petit problème: Malgré que les lignes semblent être des listes, mais elles sont de classe $< str >$, on doit donc premièrement les retransformées en listes.

```
49  data['info generales'] = data['Info generales'].apply(str.split,args=(' '))
50
51  data['Nombre de pieces'] = data['Info generales'].apply(find_nombre_piece)
52  data['Type'] = data['Info generales'].apply(find_type)
53  data['Adresse'] = data['Info generales'].apply(find_adresse)
54  data['Secteur'] = data['Info generales'].apply(find_secteur)
55  data['Surface totale en m^2'] = data['Info generales'].apply(find_surface_totale)
56  del data['Info generales']
57  print(data.iloc[:,4:])
```

Figure 27: Code 13

```
C:\Users\toshiba\Anaconda3\envs\First.py\python.exe C:/Users/toshiba/PycharmProjects/
    Nombre de pieces               Type  ...         Secteur Surface totale en m^2
0                  1   Maisons et Villas  ...       Hay Nahda                   70
1                  2        Appartements  ...   Autre secteur                   90
2                NaN   Maisons et Villas  ...        Hay Riad                  300
3                  2        Appartements  ...       Bourgogne
4                  2        Appartements  ...   Autre secteur                   73
..               ...                 ...  ...             ...                  ...
695              NaN  Terrains et Fermes  ...  Toute la ville                 5900
696              NaN  Terrains et Fermes  ...  Toute la ville                  320
697                4        Appartements  ...         Souissi                  266
698                6   Maisons et Villas  ...   Autre secteur                  500
699                1        Appartements  ...   Autre secteur                   50
```

Figure 28: Résultat

On a terminé avec cette partie, passons à la colonne suivante.

3.2 Informations détaillées

Dans cette section, les champs ne sont pas obligatoires, il est possible d'avoir des lignes qui ne contient aucune information. On va se contenter de quelques informations qui se répétent souvent comme:

- Nombre des salons.
- Nombre des salle de bain.
- Superficie habitable.

- Âge du bien.
- Numéro d'étage.
- Frais de syndic / mois.

On va donc ,comme dans le paragraphe précedent, définir des fonctions pour extraire l'ensemble des champs.

```
60  def find_salon(listx):
61      for x in listx:
62          if 'Salons' in x:
63              return x[11:len(x)-2]
64      return np.nan
```

Figure 29: Nombre des salons

```
66  def find_salle_bain(listx):
67      for x in listx:
68          if 'Salles de bain' in x:
69              return x[20:len(x)-2]
70      return np.nan
```

Figure 30: Nombre des salle de bain

```
72  def find_Superficie_habitable(listx):
73      for x in listx:
74          if 'Superficie habitable' in x:
75              return x[26:len(x)-4]
76      return np.nan
```

Figure 31: Superficie habitable

```
78  def find_age_bien(listx):
79      for x in listx:
80          if 'Âge du bien' in x:
81              return x[17:len(x)-2]
82      return np.nan
```

Figure 32: Âge du bien

```
84  def find_etage(listx):
85      for x in listx:
86          if 'Étage' in x:
87              return x[10:len(x)-1]
88      return np.nan
```

Figure 33: Numéro d'étage

```python
def find_frais_syndic(listx):
    for x in listx:
        if 'Frais de syndic / mois' in x:
            return x[27:len(x)-4]
    return '0'
```

Figure 34: Frais de syndic / mois

Puis on les applique à la serie data['information detailles'].

```python
data['Info Detailles'] = data['Info Detailles'].apply(str).apply(str.split,args=(','))

data['Salons'] = data['Info Detailles'].apply(find_salon)
data['Salles de bain'] = data['Info Detailles'].apply(find_salle_bain)
data['Etage'] = data['Info Detailles'].apply(find_etage)
data['Age du bien'] = data['Info Detailles'].apply(find_age_bien)
data['Superficie habitable'] = data['Info Detailles'].apply(find_Superficie_habitable)
data['Frais de syndic / mois'] = data['Info Detailles'].apply(find_frais_syndic)
del data['Info Detailles']

print(data.iloc[:,9:])
```

Figure 35: Code 21

```
Run:  First

[1330 rows x 6 columns]
    Salons Salles de bain Etage         Age du bien Superficie habitable syndic
80       1             1     3           11-20 ans                   93      60
81     NaN           NaN   NaN                 NaN                  NaN     NaN
82     NaN           NaN   NaN                 NaN                  NaN     NaN
83       3             3   NaN  Neuf/Première main                  270     NaN
84       2             1     3  Neuf/Première main                   65      50
85       1             2     4             1-5 ans                   95    1000
86     NaN           NaN   NaN                 NaN                 1500     NaN
87     NaN           NaN   NaN                 NaN                  NaN     NaN
88       3             3     5           11-20 ans                  170     300
89     NaN           NaN   NaN                 NaN                  NaN     NaN

Process finished with exit code 0
```

Figure 36: Résultat

Fin de cette partie aussi.

3.3 Informations supplémentaires

Dans cette section, les annonces indiquent l'existence de quelques suppléments. On va se contenter de quelques-unes:

- Ascenseur.
- Balcon.
- Terrasse.
- Cuisine équipée.
- Loti.
- Jardin.
- Piscine.
- Concierge.
- Parking.
- Chauffage.
- Climatisation.
- Meublé.
- Sécurité.
- Garage.

On définit les fonctions avec lesquelles on va créer les colonnes de chaque champs.

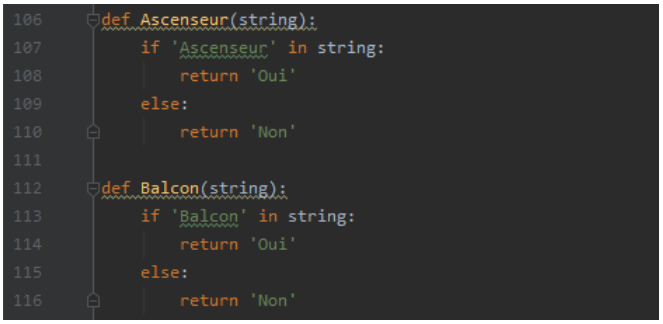

Figure 37: Ascenceur/Balcon

Le reste des fonctions est le même sauf qu'on doit changer les noms des champs. Pas la peine de les mentionner.

```
data['Info Supplimentaires'] = data['Info Supplimentaires'].apply(str)

data['Garage'] = data['Info Supplimentaires'].apply(Garage)
data['Securite'] = data['Info Supplimentaires'].apply(Securite)
data['Meuble'] = data['Info Supplimentaires'].apply(Meuble)
data['Climatisation'] = data['Info Supplimentaires'].apply(Climatisation)
data['Chauffage'] = data['Info Supplimentaires'].apply(Chauffage)
data['Parking'] = data['Info Supplimentaires'].apply(Parking)
data['Concierge'] = data['Info Supplimentaires'].apply(Concierge)
data['Piscine'] = data['Info Supplimentaires'].apply(Piscine)
data['Jardin'] = data['Info Supplimentaires'].apply(Jardin)
data['Loti'] = data['Info Supplimentaires'].apply(Loti)
data['Cuisine equipee'] = data['Info Supplimentaires'].apply(Cuisine_equipee)
data['Terrasse'] = data['Info Supplimentaires'].apply(Terrasse)
data['Ascenseur'] = data['Info Supplimentaires'].apply(Ascenseur)
data['Balcon'] = data['Info Supplimentaires'].apply(Balcon)

del data['Info Supplimentaires']

print(data.iloc[:,15:])
```

Figure 38: Code 23

```
C:\Users\toshiba\Anaconda3\envs\First.py\python.exe C:/Users/toshiba/PycharmProjects/MyyPro
   Securite Meuble Climatisation  ... Terrasse Ascenseur Balcon
30      Oui    Non           Oui  ...      Oui       Non    Oui
31      Non    Non           Non  ...      Non       Non    Non
32      Non    Non           Oui  ...      Oui       Oui    Oui
33      Non    Non           Non  ...      Non       Non    Oui
34      Non    Non           Non  ...      Non       Non    Non
35      Oui    Non           Non  ...      Oui       Non    Oui
36      Non    Non           Non  ...      Non       Non    Non
37      Oui    Non           Non  ...      Non       Non    Non
38      Non    Non           Non  ...      Non       Non    Non
39      Non    Non           Non  ...      Non       Non    Non
40      Oui    Non           Non  ...      Non       Non    Non
41      Oui    Non           Oui  ...      Non       Non    Oui
42      Non    Non           Non  ...      Non       Non    Oui
43      Oui    Non           Oui  ...      Non       Oui    Oui
44      Non    Non           Non  ...      Oui       Non    Non
45      Non    Non           Non  ...      Non       Non    Non
46      Non    Non           Non  ...      Non       Non    Oui
47      Non    Non           Non  ...      Non       Non    Non
48      Non    Non           Non  ...      Non       Non    Non
49      Non    Non           Non  ...      Non       Non    Non
```

Figure 39: Résultat

Les données sont bien sctructureés, il ne reste que résoudre des petits problémes qu'on va présnter dans le paragraphe suivant.

3.4 Quelques petits problémes

Comme on est face à un *DataFrame* de plusieurs colonnes, il est plus judicieux de visualiser nos données en fichier CSV.

B	C	D	E	F	G	H	I	J	K	L	M
Titre	Prix	Nombre de pieces	Type	Adresse	Secteur	Surface total	Salons	Salles de bai	Etage	Age du bien	Superficie habitable
	1 100 000	1	Maisons et Villas		Hay Nahda	70					
	450 000	2	Appartements		Autre secteu	90	2	2	Rez de chaussÃ©e	Neuf/PremiÃ¨re main	
	5 400 000		Maisons et Villas		Hay Riad	300					
		2	Appartemen	Rue ighrem	Bourgogne		1	2			
	510 000	2	Appartements		Autre secteu	73	1	1	2	Neuf/PremiÃ¨re main	
		2	Appartements		Maarif						
			Terrains et F	hay riad pro	Hay Riad	500					
	600 000	2	Appartements		Autre secteu	90	1	2	2	Neuf/PremiÃ¨re main	
		1	Maisons et V	MAYSSANE I	Toute la ville	590					
		#NOM?	Appartements		Sidi Maarouf						
	1 300 000		Terrains et F	imswan rout	Autre secteu	6500					
	1 276 500	3	Appartemen	Boulevard la	Mers Sultan	111	1	2		Neuf/PremiÃ¨re main	
	21 000 000	5	Maisons et V	AIN DIAB	AÃ¯n Diab	1591	1	3		11-20 ans	680
	540 000	2	Appartements		Sidi Maarouf						
	1 300 000		Terrains et F	haute tagha	Autre secteu	1500					
	550 000	2	Appartemen	hay mohana	Toute la ville	70	1	2	5		70
			Magasins	Rue lieuten	Hay Mohamr	255					

Figure 40: Csv

Apparament on constate que les problémes qu'on a sont:

- La colonne *'Titre'* n'apparait pas dans le fichier.
- La colonne *'Prix'* contient des prix ilogiques (2, 20, ...).
- La colonne *'Nombre des piéces'* contient des valeur non numeriques.
- Les colonnes *'Adresse'* et *'Secteur'* contient des mots non lisibles. Cela est dû à l'existence des caractéres non reconnus (é, è, à, ...).
- Dans la colonne *'Etage'* on doit changer *Rez de chaussée* par la valeur 0 vu qu'on va exploiter ces donneés en Machine Learning.
- Les colonnes *'Prix' 'Nombre de pieces' 'Etage' 'Salons'* et *'Salles de bain'* doivent êtres converties en entier vu qu'elle sont de classe $< str >$.

Définissons quelques fonctions :

```python
def regler_entier(x):
    try:
        return int(x)
    except:
        return np.nan

def remove_space(string):
    try:
        return string.replace(" ","")
    except:
        return np.nan

def nan(string):
    if string == 'nan':
        return np.nan
    else:
        return string

def fix_rez(element):
    try:
        return int(element)
    except:
        if type(element)==str:
            return 0
        return np.nan
```

Figure 41: Code 24

Notes:

- Avant de convertir le prix il faut éliminer les espaces.
- la convertion des caractères de type (é, è, à, ...) sera faite par la fonction prédefinit *unidecode* de la bibliothèque *unidecode*.

```python
import unidecode

data['Age du bien'] = data['Age du bien'].apply(str).apply(unidecode.unidecode).apply(nan)
data['Adresse'] = data['Adresse'].apply(str).apply(unidecode.unidecode).apply(nan)
data['Secteur'] = data['Secteur'].apply(str).apply(unidecode.unidecode).apply(nan)
data['Titre'] = data['Titre'].apply(str.split,args=('\n')).apply(lambda x: x[1])
data['Nombre de pieces'] = data['Nombre de pieces'].apply(regler_entier)
data['Prix'] = data['Prix'].apply(remove_space).apply(regler_entier)
data['Etage'] = data['Etage'].apply(fix_rez)
```

Figure 42: Code 25

Vérification: Pour bien vérifier notre travail, on va compter le nombre des *NaN* dans chaque colonne avant et aprés les modifications pour être sûr qu'il n'y a pas de perte d'informations. On va donc exécuter cette ligne avant et aprés les modifications.

```
252    print(data.isna().sum())
```

Figure 43: Code 26

Prix	361
Nombre de pieces	437
Type	0
Adresse	519
Secteur	1
Surface totale en m^2	0
Salons	598
Salles de bain	539
Etage	842
Age du bien	786
Superficie habitable	842

(a) Avant

Prix	361
Nombre de pieces	454
Type	0
Adresse	519
Secteur	1
Surface totale en m^2	0
Salons	598
Salles de bain	539
Etage	842
Age du bien	786
Superficie habitable	842

(b) Aprés

Figure 44: Comparaison

On a perdu quelques donneés du champs *'Nombre de pieces'*, c'est parceque dans quelques annonces le nombre des pièces mentionnées est **'+10'**.
La fonction suivante va régler le probléme:

```
237  def fix_pieces(element):
238      try:
239          if element == '10+':
240              return 11
241          else:
242              return int(element)
243      except:
244          return np.nan
245
246  data['Nombre de pieces'] = data['Nombre de pieces'].apply(fix_pieces)
247  print(data['Nombre de pieces'].isna().sum())
```

Figure 45: Code 29

```
C:\Users\toshiba\Anaconda3\envs\First.py\python.exe C:/Users/toshiba/PycharmProjects,
441

Process finished with exit code 0
```

Figure 46: Résultat

Voyons qu'on a pu sauver 13 valeurs qui valent +10 et qu'on a les remplacer par 11.
Dernière chose à faire, c'est éliminé toutes les annonces dont le prix est inférieur à 150 000 DH. Car il ne s'agit clairement pas d'une maison situer à Rabat.

```
262    data = data[(data['Prix'] > 150000)]
263    data.to_csv("C:\\Workplace\\Mini projet inpt\\Données Avito.ma.csv")
```

Figure 47: Code 30

Voila les 20 premières lignes du résultat finale en CSV:

	Titre	Prix	Nombre de pieces	Type	Adresse	Secteur	Surface totale en m^2	Salons	Salles de bain	Etage	Age du bien
2	Maison	1100000	1	Maisons et Villas		Hay Nahda	70				
3	Appartement 2 FaÃ§ades Alliance mehdia Kenitra	450000	2	Appartements		Autre secteur	90	2	2	0	Neuf/Premiere
4	villa d'angle pr bureau habitation	5400000		Maisons et Villas		Hay Riad	300				
5	Appartement 73 m2 double voie El Haddada	510000	2	Appartements		Autre secteur	73	1	1	2	Neuf/Premiere
6	Appartement 2 faÃ§ades opposÃ©es 90 m2 El Haddada	600000	2	Appartements		Autre secteur	90	1	2	2	Neuf/Premiere
7	terrain 6500 m imswan route principal de swira	1300000		Terrains et Fermes	imswan rou	Autre secteur	6500				
8	Appartement Haut Standing de 57 m2 Ã 111 m2	1276500	3	Appartements	Boulevard l	Mers Sultan	111	1	2		Neuf/Premiere
9	SPLANDIDE VILLA AIN DIAB 1591 m2	21000000	5	Maisons et Villas	AIN DIAB	Ain Diab	1591	1	3		11-20 ans
10	Appartement en Ve Ã Casablanca	540000	2	Appartements		Sidi Maarouf					
11	haute taghazot terrain 1500 m pret de plage 1 km	1300000		Terrains et Fermes	haute tagha	Autre secteur	1500				
12	vend appt	550000	2	Appartements	hay mohana	Toute la ville	70	1	2	5	
13	bel Appartement neuf 68 m2	350000	2	Appartements		Toute la ville					
14	joli Appartement 58 m2 avec bon prix	290000	3	Appartements		Toute la ville					
15	Terrain et ferme en Vente Ã BÃ©ni Mellal	340000		Terrains et Fermes	Oulad ayad	Toute la ville	100				
16	Appartement 100 m la bille ville	440000	2	Appartements	la bielle vill	Centre Ville	100	1	1	1	1-5 ans
17	Ø˙Ù‚Ø© Ø–Ø˙Ù‡Ø˙Ø© Ø–Ø§Ù‡Ø²Ø©	320000	3	Appartements	1degetage j	M'Hamid	73	1	1	1	Neuf/Premiere
18	Terrain et ferme en Vente Ã BÃ©ni Mellal	340000		Terrains et Fermes	Oulad ayad	Toute la ville	100				
19	appartement	530000	2	Appartements	al Maghrib A	Al Maghrib Al Ar	100	1	2	1	6-10 ans
20	terrain takate sidi bibi 3000 m pret de plage 2 km	270000		Terrains et Fermes	takate sidi b	Autre secteur	3000				

Figure 48: Colonnes [1,11]

	Superficie habitable	Frais de syndic / mois	Garage	Securite	Meuble	Climatisation	Chauffage	Parking	Concierge	Piscine	Jardin	Loti	Cuisine equipee	Terrasse	Ascenseur	Balcon
2		0	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non
3		0	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non
4		0	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non
5		0	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non
6		0	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non
7		0	Non	Non	Non	Non	Non	Non	Non	Non	Non	Oui	Non	Non	Non	Non
8		0	Non	Oui	Non	Oui	Non	Oui	Non	Non	Non	Non	Oui	Oui	Oui	Oui
9	680	0	Oui	Oui	Non	Non	Non	Oui	Non	Oui	Oui	Non	Oui	Oui	Non	Oui
10		0	Non	Oui	Oui	Non	Non	Oui	Non	Non	Non	Non	Non	Oui	Oui	Oui
11		0	Non	Non	Non	Non	Non	Non	Non	Non	Non	Oui	Non	Non	Non	Non
12	70	120	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non
13		0	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non
14		0	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non
15		0	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non
16		0	Non	Oui	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non	Oui	Oui
17	70	100	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non
18		0	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non
19	83	20	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non	Oui	Non	Oui
20		0	Non	Non	Non	Non	Non	Non	Non	Non	Non	Oui	Non	Non	Non	Non

Figure 49: Colonnes [12,27]

Note: On a réexecuter le code pour un nombre de pages qui vaut 200, le fichier CSV contient 7045 lignes.

4 Visualisation des données:

4.0.1 Nombre des annonces et les valeurs nulles

le code :

```
print('Nombre des annonces récupérées: ',data['Titre'].count())


print('\nNombre des valeurs nulles pour chaque champ :\n')
print(data.iloc[:,2:12].isna().sum())
```

Figure 50: Code

Resultat :

```
Nombre des annonces récupérées:  4645

Nombre des valeurs nulles pour chaque champ :

Nombre de pieces         1260
Type                        0
Adresse                  1592
Secteur                     0
Surface totale en m^2     740
Salons                   2005
Salles de bain           1788
Etage                    2763
Age du bien              2687
Superficie habitable     2712
dtype: int64
```

Figure 51: Valeurs nulles par colonnes

Visualisons parsuite chaque colonne toute seule.

4.1 Prix:

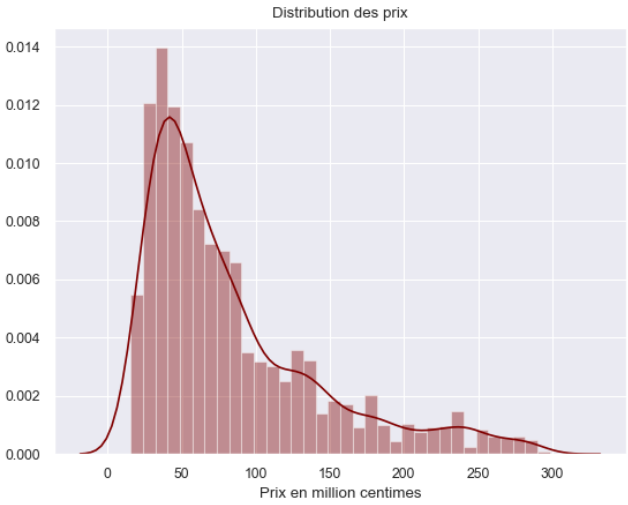

Figure 52: Distribution des prix

4.2 Surface totale:

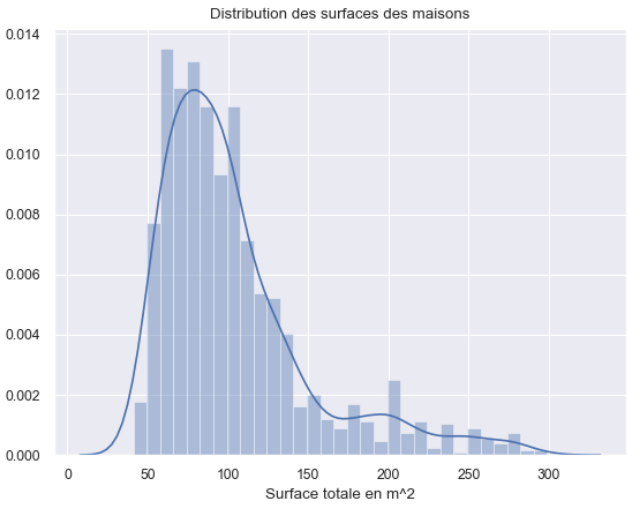

Figure 53: Distribution des surfaces totales

4.3 Nombre des pieces:

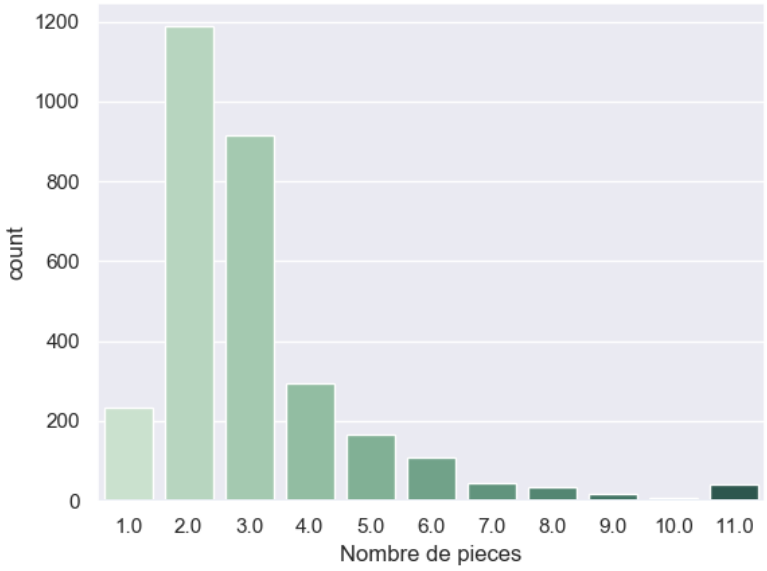

Figure 54: Nombre des pieces
en fonctions du nombre des maisons

4.4 Type:

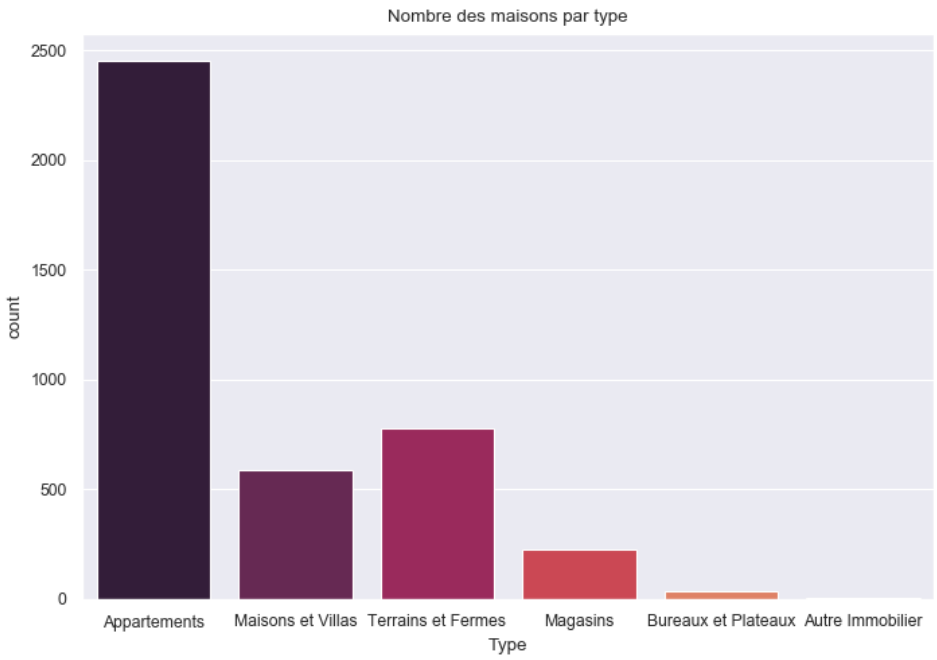

Figure 55: Nombre des maisons par type

4.5 Salons:

Executons aussi un *countplot* pour la colonne *Salon*.

```
sns.countplot(data["Salons"],palette="ch:2.5,-.2,dark=.3")
plt.title('distrubition par rapport aux salons')
```

Figure 56: Salons

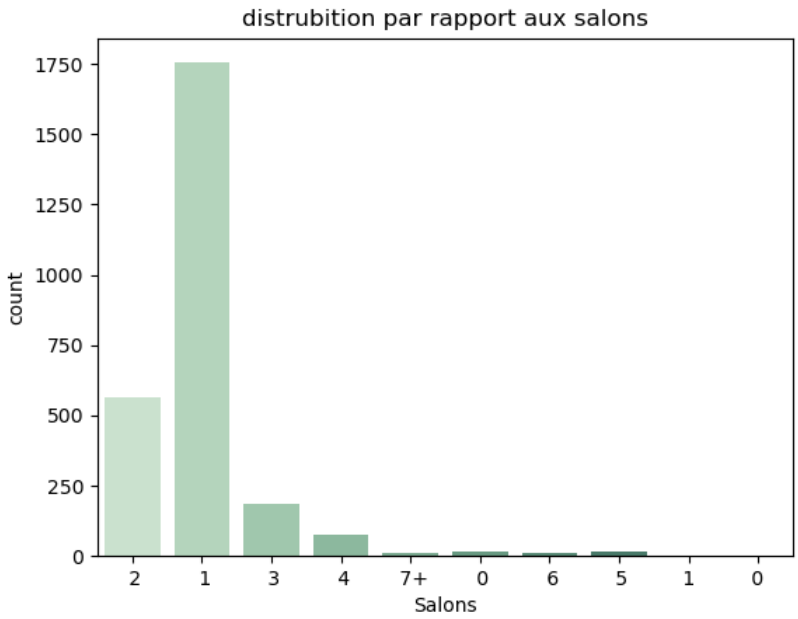

Figure 57: Salons

Problème:
La répétitions des élements de l'axe des abscisse est due au type des élements (*Classe str*). On va résoudre donc le problème par la méthode suivante:

```
def entier(nb):
    try:
        return int(nb)
    except:
        return np.nan

data['Salon'] = data['Salon'].apply(entier)
```

Figure 58: Salons

4.6 Salle de bain:

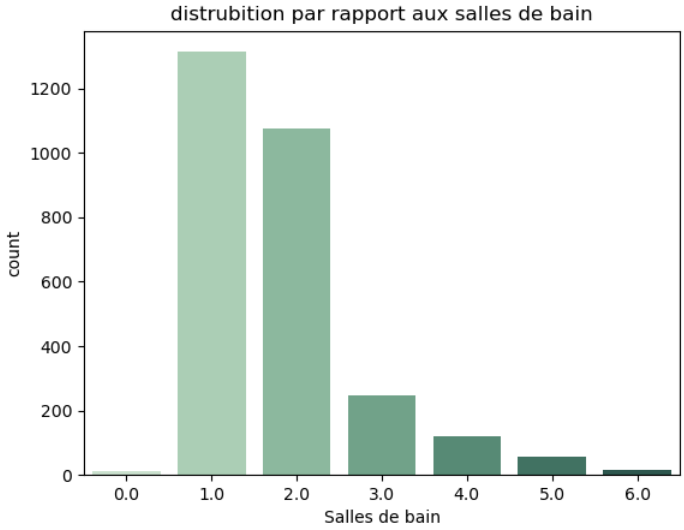

Figure 59: Salle de bain

4.7 Etage:

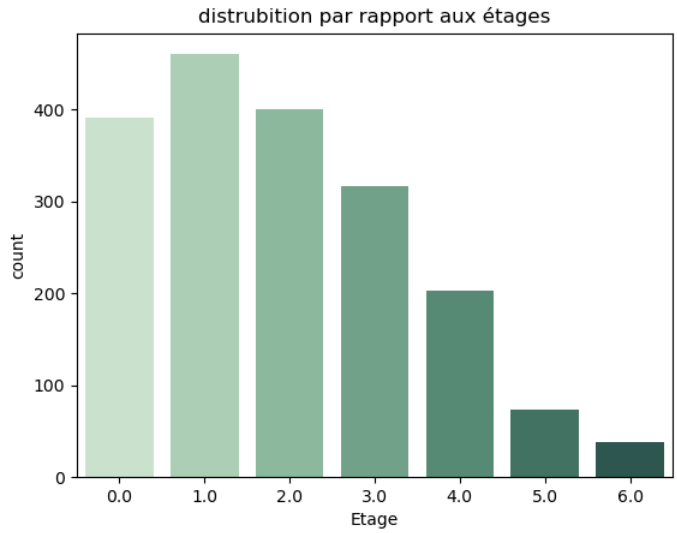

Figure 60: Etage

4.8 Age du bien:

Pour la visualisation de la colonne *age du bien* :

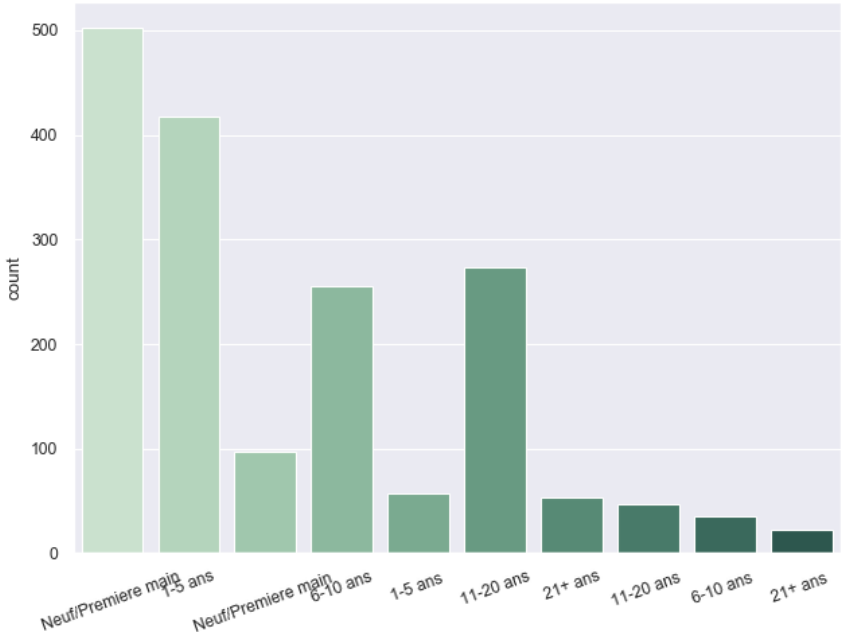

Figure 61: Age du bien

Problème: On rencontre une autre fois le même problème, on a choisi de le résoudre par le code suivant:

```python
def correction (str):
    try:
        if str[0]== '2':
            return '21+ans'
        elif str[0]=='N':
            return 'Neuf/Premiere main'
        elif str[0]== '6':
            return '6-10 ans'
        elif str[0]== '1' and str[1]=='1':
            return '11-20 ans'
        elif str[0]=='1' and str[1]!='1':
            return '1-5 ans'
    except:
        return np.nan
data["Age du bien"]=data["Age du bien"].apply(correction)
Age_du_bien = data["Age du bien"].value_counts().reset_index(name='age_du_bien')
g = sns.catplot(x="index", y="age_du_bien", data=Age_du_bien,
                height=6, kind="bar")
g.set_xlabels("Les types")
g.set_ylabels("counts")
plt.title('La visualisation d age du bien')
plt.show()
```

Figure 62: Age du bien

On obtient la classification suivante:

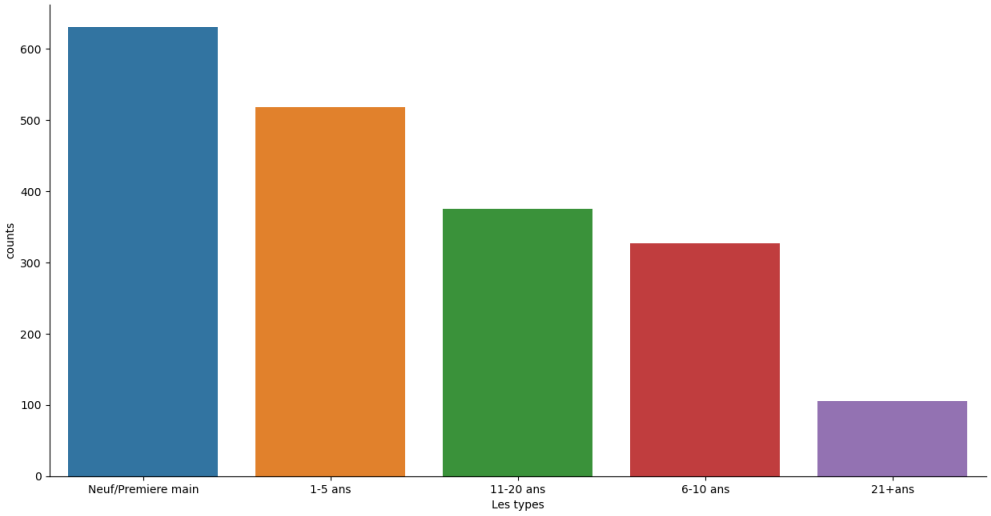

Figure 63: Age du bien

4.9 Superficie habitable:

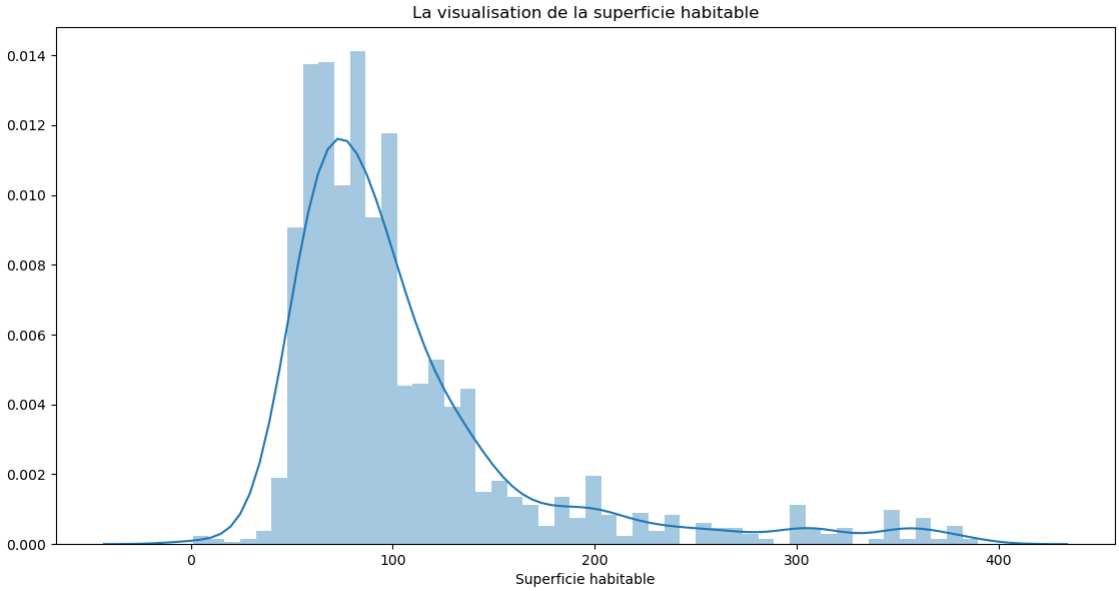

Figure 64: Distribution du superficie habitable

4.10 Description (Garage, sécurité, meublé, climatisation, chauffage, parking, concierge, piscine, loti, cuisine équipée, Terasse, Ascenseur, balcon):

On va tracer une *lmplot* de la surface totale en fonction du prix: deux regressions linéaires dans le cas de l'existence ou non d'un supplément.
Soir donc le programme:

```
for supplement in data.columns[13:]:
    plt.figure(figsize=(20,20))
    sns.lmplot(x = 'Surface totale en m^2',y = 'Prix',data=data,hue = supplement)
    plt.show()
```

Figure 65: Code

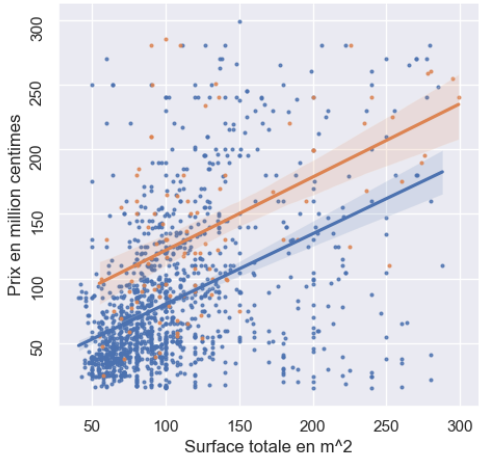

Figure 66: Garage

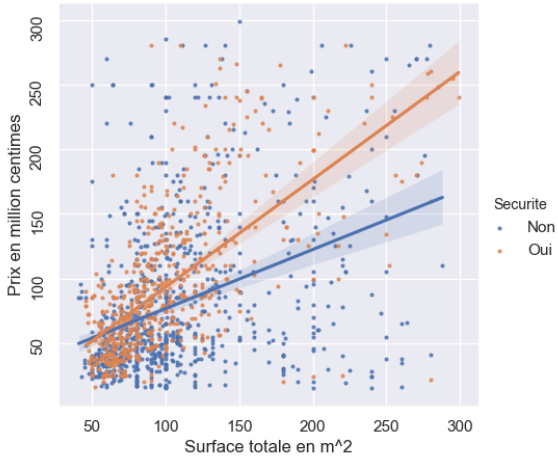

Figure 67: Sécurité

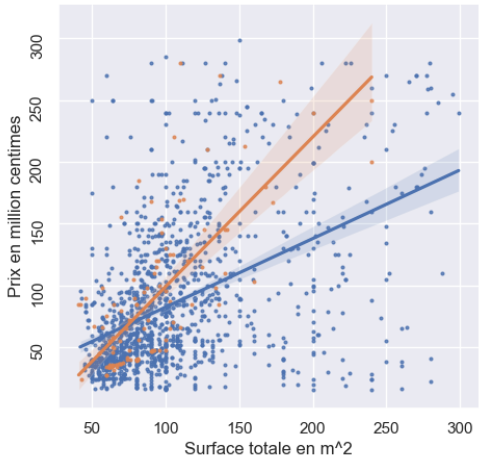

Figure 68: Meuble

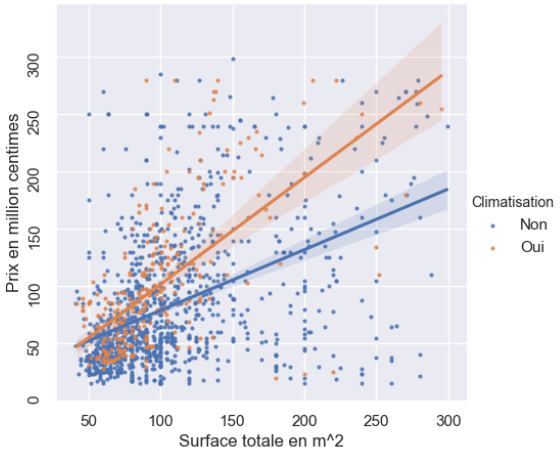

Figure 69: Climatisation

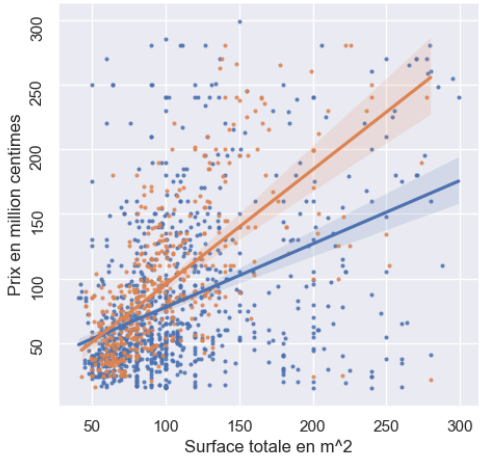

Figure 70: Chauffage

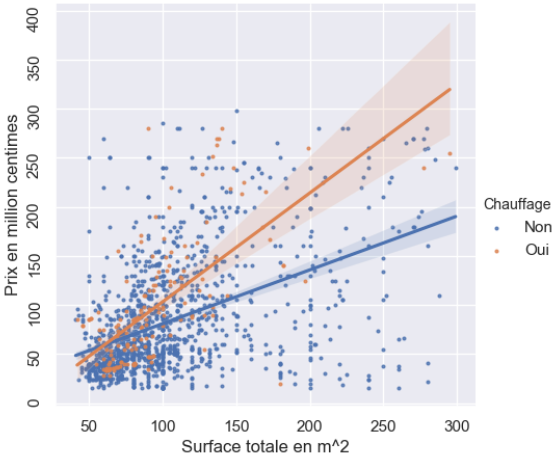

Figure 71: Parking

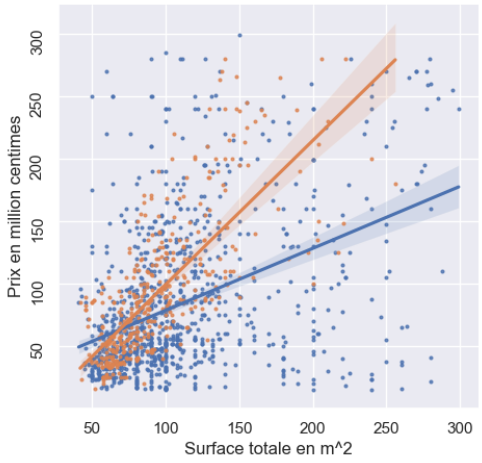

Figure 72: Concierge

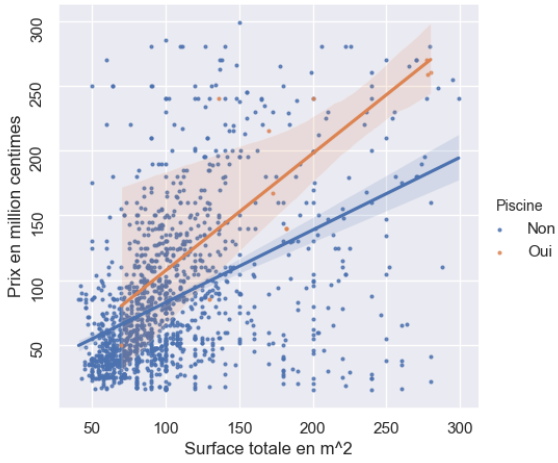

Figure 73: Piscine

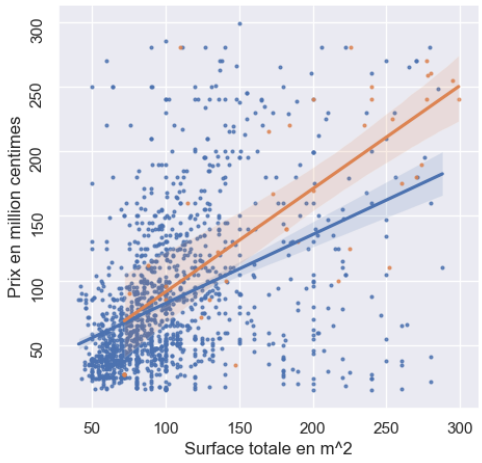

Figure 74: Jardin

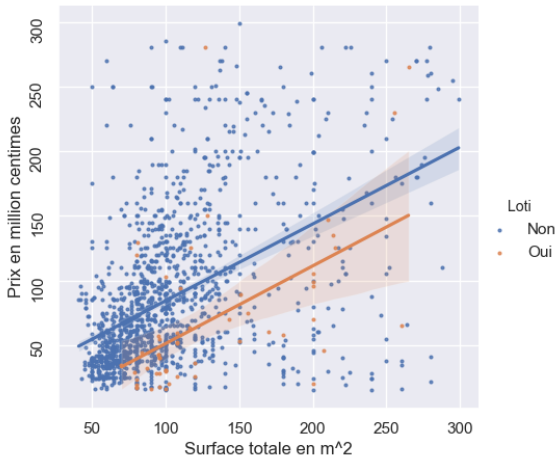

Figure 75: loti

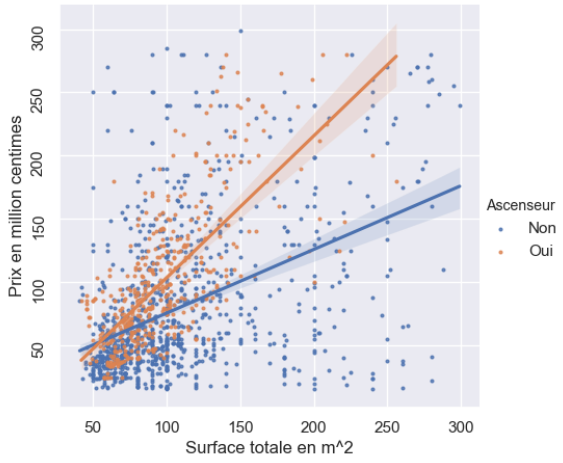

Figure 76: Ascenseur

Remarque: On observe une augmentation du prix si l'un des suppléments existe.

4.11 Visualisation et analyse plus profonde

4.11.1 Moyenne des prix et des surfaces pour chaque type

```
Appartement_surface_mean = data[(data['Type'] =='Appartements')]['Surface totale en m^2'].mean()
print('\nSurface totale moyenne des appartements: {:.2f} m²'.format(Appartement_surface_mean))

Villas_surface_mean = data[data['Type'] =='Maisons et Villas']['Surface totale en m^2'].mean()
print('Surface totale moyenne des maisons et villas: {:.2f} m² '.format(Villas_surface_mean))

Villas_suraface2_mean = data[data['Type'] =='Maisons et Villas']['Superficie habitable'].mean()
print('Surface habitable moyenne des maisons et villas : {:.2f} m²'.format(Villas_suraface2_mean))

Magasins_surface_mean = data[data['Type'] =='Magasins']['Surface totale en m^2'].mean()
print('Surface totale moyenne des magasins: {:.2f} m²'.format(Magasins_surface_mean))

Appartement_price_mean = data[(data['Type'] =='Appartements')]['Prix'].mean()
print('\nPrix moyen des appartements: {:.2f} millions centimes'.format(Appartement_price_mean))

Villas_price_mean = data[(data['Type'] =='Maisons et Villas')]['Prix'].mean()
print('Prix moyen des maisons et villas: {:.2f} millions centimes'.format(Villas_price_mean))

Magasins_price_mean = data[data['Type'] =='Magasins']['Prix'].mean()
print('Prix moyen des magasins: {:.2f} millions centimes'.format(Magasins_price_mean))
```

Figure 77: Code

```
Surface totale moyenne des appartements: 90.00 m²
Surface totale moyenne des maisons et villas: 260.52 m²
Surface habitable moyenne des maisons et villas : 159.20 m²
Surface totale moyenne des magasins: 1486.39 m²

Prix moyen des appartements: 73.30 millions centimes
Prix moyen des maisons et villas: 128.26 millions centimes
Prix moyen des magasins: 83.97 millions centimes
```

Figure 78: Résultat

Visualisation:

```
sns.barplot(x=['Appartements','Maisons et villas','Magasins'],
            y=[Appartement_price_mean,Magasins_price_mean,Villas_price_mean],
            palette='ch:2.5,-.2,dark=.3')
plt.title('Prix moyen de chaque type')
plt.show()


sns.barplot(x=['Appartements','Maisons et villas','Magasins'],
            y=[Appartement_surface_mean,Magasins_surface_mean,Villas_suraface_mean],
            palette='ch:2.5,-.2,dark=.3')
plt.title('Surface moyenne de chaque type')
plt.show()
```

Figure 79: Code

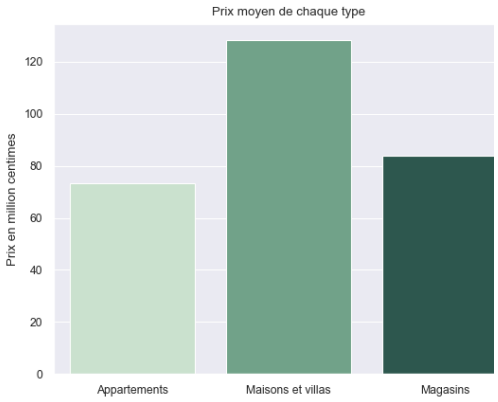

Figure 80: Prix moyen

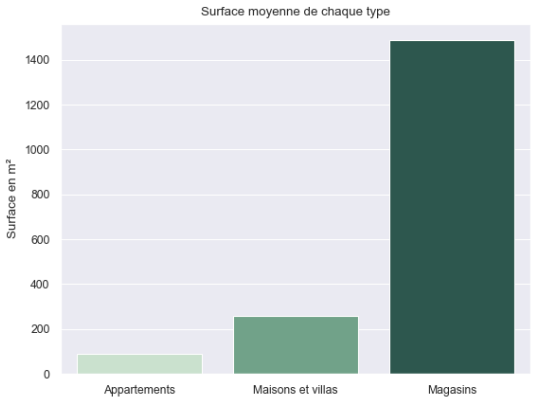

Figure 81: Surface moyenne

4.11.2 Critères de choix des appartements

Parmi les critères qui empèchent l'achat ou le vente d'une appartement et celle de l'étage où elle se situe, surtout au cas de l'indisponibilité de l'ascenseur. L'appartement devient donc indésirable et son prix subit une dimunition. C'est ce résultat qu'on va confirmer par la suite. **Le code:**

```
print("\nPourcentage des appartements dont l'étage est superieur à 2, mais qui n'ont pas d'ascenseur: {:.2f} %"
      .format(data[(data['Etage']>=2)&(data['Ascenseur']==0)]['Titre'].count()/data[(data['Etage']>=2)]['Titre'].count()*100))

print('Leurs prix moyen: {:.2f} M.C'.format(data[(data['Etage']>=2) & (data['Ascenseur']==0)]['Prix'].mean()))

print("\nPourcentage des appartements dont l'étage est superieur à 3, mais qui n'ont pas d'ascenseur: {:.2f} %"
      .format(data[(data['Etage']>=3)&(data['Ascenseur']==0)]['Titre'].count()/data[(data['Etage']>=3)]['Titre'].count()*100))

print('Leurs prix moyen: {:.2f} M.C'.format(data[(data['Etage']>=3) & (data['Ascenseur']==0)]['Prix'].mean()))

print("\nPourcentage des appartements dont l'étage est superieur à 4, mais qui n'ont pas d'ascenseur: {:.2f} %"
      .format(data[(data['Etage']>=4)&(data['Ascenseur']==0)]['Titre'].count()/data[(data['Etage']>=4)]['Titre'].count()*100))

print('Leurs prix moyen: {:.2f} M.C'.format(data[(data['Etage']>=4) & (data['Ascenseur']==0)]['Prix'].mean()))
```

Figure 82: Code

Resultat:

```
Pourcentage des appartements dont l'étage est superieur à 2, mais qui n'ont pas d'ascenseur: 55.27 %
Leurs prix moyen: 59.68 M.C

Pourcentage des appartements dont l'étage est superieur à 3, mais qui n'ont pas d'ascenseur: 49.67 %
Leurs prix moyen: 56.63 M.C

Pourcentage des appartements dont l'étage est superieur à 4, mais qui n'ont pas d'ascenseur: 37.00 %
Leurs prix moyen: 52.42 M.C
```

Figure 83: Résultat

Remarque:
Pour les appartements sans ascenseur, plus elles augmentent en étage plus le prix baisse.

4.11.3 Classification selon l'équipement et la disponibilité

Cette fois, on a choisi de faire une visualisation qui classifie les données en trois catégories selon leurs *équipements* (Garage, jardin, sécurité...):
Le code:

```python
def convertion (string):
 try:
    if string== 'Oui' :
        return 1
    elif string == 'Non':
        return 0
 except:
     return np.nan
for i in data.columns[13:]:
    data[i]=data[i].apply(convertion)
data["somme"]=data.iloc[:,13:].sum(axis=1)
def type(nb):
 try:
    if nb > 9:
        return 'bien équipée'
    elif 4<nb<10:
        return 'moyen équipement'
    else:
        return 'mauvais équipement'
 except:
    return np.nan
data["somme"]=data["somme"].apply(type)
print(data["somme"].head(5))
equipement = data["somme"].value_counts().reset_index(name="type_equipement")
sns.catplot(x="index",y="type_equipement",data=equipement,kind="bar")
plt.show()
```

Figure 84: Code

Graphe

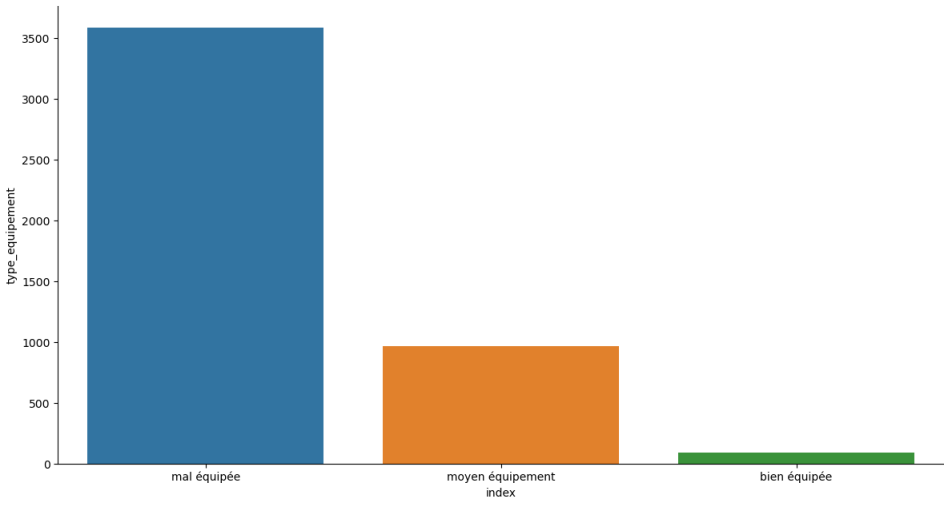

Figure 85: Type d'équipement

4.11.4 Estimation du prix d'un mètre carré

Dans la fin de cette partie on va récupérer une estimation du prix d'un m^2 d'un appartement. Pour se faire on va premièrement fixer un intervalle de la surface totale, soit l'intervalle $[60, 80]$

```
data = data[data['Type']=='Appartements']

print('\nDescription de la colonne prix: ')
data1 = data[(data['Surface totale en m^2']<81) & (data['Surface totale en m^2']>59)]
print(data1['Prix'].describe())
```

Figure 86: Code

Resultat:

```
Description de la colonne prix:
count    684.000000
mean      54.014892
std       26.135337
min       16.000000
25%       37.000000
50%       48.450000
75%       66.000000
max      250.000000
Name: Prix, dtype: float64
```

Figure 87: Résultat

684 annonces semble être suffisante pour une telle estimation.
Le prix d'un mètre carré diffère d'un secteur à l'autre. On va donc diviser les appartements selon le prix en 4 classes:

Classe	Prix en million centimes
Première classe	$\geq$100
Deuxiéme classe	[60,100[
Troisiéme classe	[40,60[
Dernière classe	<40

A l'aide du code suivant, on obtient la colonne *Classe* qui indique la classe de l'appartement.

```
data1.loc[data1['Prix'] >= 100,'Classe'] = 'Première classe'
data1.loc[(data1['Prix'] < 100) & (data['Prix'] >=60),'Classe'] = 'Deuxiéme classe'
data1.loc[(data1['Prix'] < 60) & (data['Prix'] >=40),'Classe'] = 'Troisiéme classe'
data1.loc[data1['Prix'] < 40,'Classe'] = 'Dernière classe'
```

Figure 88: Code

Vérifiant qu'on a un nombre suffisant d'annances pour une estimation plus précise.

```
First_price = data1[data1['Classe'] == 'Première classe']['Prix']
Second_price = data1[data1['Classe'] == 'Deuxiéme classe']['Prix']
Third_price = data1[data1['Classe'] == 'Troisiéme classe']['Prix']
Last_price = data1[data1['Classe'] == 'Dernière classe']['Prix']

print('\nNombre des appartements de la classe :')
print('Première  :', First_price.count())
print('Deuxiéme  :', Second_price.count())
print('Troisiéme :', Third_price.count())
print('Dernière  :', Last_price.count())
```

Figure 89: Code

```
Nombre des appartements de la classe :
Première  : 35
Deuxiéme  : 196
Troisiéme : 239
Dernière  : 214
```

Figure 90: Résultat

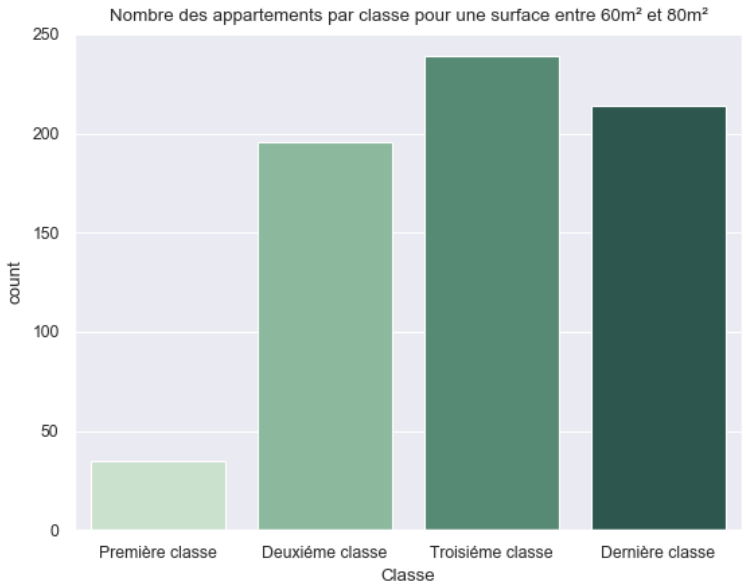

Figure 91: Graphe

Pas mal sauf pour la prémiere classe, mais cela reste comme même suffisant.

Il faut parsuite verifier que l'écart entre les differentes surfaces moyennes des 4 classes n'est pas assez large. Puis on récupére le prix moyen du m^2 pour chaque classe.

```
First_surface = data1[data1['Classe'] == 'Première classe']['Surface totale en m^2']
Second_surface = data1[data1['Classe'] == 'Deuxiéme classe']['Surface totale en m^2']
Third_surface = data1[data1['Classe'] == 'Troisiéme classe']['Surface totale en m^2']
Last_surface = data1[data1['Classe'] == 'Dernière classe']['Surface totale en m^2']

print('\nSurface moyenne des appartements :')
print('Première classe:  {:.2f} m²'.format(First_surface.mean()))
print('Deuxiéme classe:  {:.2f} m²'.format(Second_surface.mean()))
print('Troisiéme classe: {:.2f} m²'.format(Third_surface.mean()))
print('Derinière classe: {:.2f} m²'.format(Last_surface.mean()))

print("\nPrix moyen d'un mètre carré :")
print('Première classe:  {:.2f} DH'.format((First_price/First_surface).mean()*10000))
print('Deuxiéme classe:  {:.2f} DH'.format((Second_price/Second_surface).mean()*10000))
print('Troisiéme classe: {:.2f} DH'.format((Third_price/Third_surface).mean()*10000))
print('Derinière classe: {:.2f} DH'.format((Last_price/Last_surface).mean()*10000))
```

Figure 92: Résultat

```
Surface moyenne des appartements :
Première classe:  74.23 m²
Deuxiéme classe:  71.81 m²
Troisiéme classe: 68.97 m²
Derinière classe: 67.11 m²

Prix moyen d'un mètre carré :
Première classe:  17747.41 DH
Deuxiéme classe:  10141.81 DH
Troisiéme classe: 7077.24 DH
Derinière classe: 4671.93 DH
```

Figure 93: Résultat

Conclution:
On obtient le résultat si dessous

Classe	Prix
Première classe	17747.41 DH/m^2
Deuxiéme classe	10141.81 DH/m^2
Troisiéme classe	7077.24 DH/m^2
Dernière classe	4671.93 DH/m^2

Voila le résultat si l'on change l'intervalle de la surface totale à [80,100].

```
Nombre des appartements de la classe :
Première  : 91
Deuxiéme  : 193
Troisiéme : 139
Dernière  : 51

Surface moyenne des appartements :
Première classe:  91.42 m²
Deuxiéme classe:  90.73 m²
Troisiéme classe: 90.93 m²
Derinière classe: 89.75 m²

Prix moyen d'un mètre carré :
Première classe:  14143.63 DH
Deuxiéme classe:  8605.09 DH
Troisiéme classe: 5387.31 DH
Derinière classe: 3730.93 DH
```

Figure 94: Résultat

La question qu'on peut poser c'est: lequelle des prix doit-on choisir? Evidemment celui qui vient des appartements situés en centre ville. Pour cela, on va voir pour chaque classe les secteurs qui ont contribués le plus dans l'estimation.

```
print('\nPremière classe')
print(data1[data1['Classe'] == 'Première classe'].groupby('Secteur').count().sort_values(by='Titre',ascending=False)['Titre'].head(4))

print('\nDeuxiéme classe')
print(data1[data1['Classe'] == 'Deuxiéme classe'].groupby('Secteur').count().sort_values(by='Titre',ascending=False)['Titre'].head(4))

print('\nTroisiéme classe')
print(data1[data1['Classe'] == 'Troisiéme classe'].groupby('Secteur').count().sort_values(by='Titre',ascending=False)['Titre'].head(4))

print('\nDerinière classe')
print(data1[data1['Classe'] == 'Derinière classe'].groupby('Secteur').count().sort_values(by='Titre',ascending=False)['Titre'].head(4))
```

Figure 95: Code

```
Première classe                Troisiéme classe
Secteur                        Secteur
Toute la ville    7            Toute la ville         36
Agdal             6            Autre secteur          31
Maarif            5            Bir Rami                6
Palmeraie         5            Al Maghrib Al Arabi     4
Name: Titre, dtype: int64      Name: Titre, dtype: int64

Deuxiéme classe                Derinière classe
Secteur                        Secteur
Toute la ville    28           Autre secteur     20
Autre secteur     17           Toute la ville     7
Ain Sebaa         10           Ben Souda          3
Sidi Maarouf      10           Boustane           2
Name: Titre, dtype: int64      Name: Titre, dtype: int64
```

Figure 96: Résultat

Ce sont les appartements de la première classe qui donnent une estimation plus précise du prix, vu que la plupart d'entre eux sont situés dans des secteurs en centre ville.
Une petite recherche Google confirmera le résultat.

Figure 97: Résultat