Projet BDDR

Binôme : Zeynep BALIKCI et Mariama MBAYE

Pour notre application, nous avons un socle minimal de requêtes:

1. Liste d'articles par thématique et sous-thématique.
2. Histogramme d'articles publiés par date, semaine, et mois.
3. Liste de thématiques
4. Nombre de publications par labo/institution.
5. Liste de journaux par nombre et type de publications.

Analyse des données

Après étude du jeux de données telechargé sur Kaggle, nous nous sommes rendus comptes que les données nécesaires à nos requêtes se trouvent dans le fichier metadata.csv, les dossiers /document_parses/pdf_json et /document_parses/pmc_json, et enfin le dossier /Kaggle/target_tables.

Pour les requêtes 1 et 3, nous avons récupérer ces données à partir du dossier /Kaggle/target_tables : le theme est le nom du dossier et les sous_themes le nom des fichiers csv se trouvant dans chaque dossier/theme

In [6]:
```
import pandas as pd
import os
import json

chemin_archive ="/users/2023/ds1/share/CORD-19"
chemin_tables=f'{chemin_archive}/Kaggle/target_tables'
elements = os.listdir(chemin_tables)
dossiers = [element for element in elements if os.path.isdir(os.path.join(chemin_tabl
for dossier in dossiers[1:-1]:
    theme=(dossier[2:].replace("_"," ")).upper()   # le theme est le nom du dossier
    print(f'{theme=}')
```

```
theme='POPULATION'
theme='RELEVANT FACTORS'
theme='PATIENT DESCRIPTIONS'
theme='MODELS AND OPEN QUESTIONS'
theme='MATERIALS'
theme='DIAGNOSTICS'
theme='THERAPEUTICS INTERVENTIONS AND CLINICAL STUDIES'
theme='RISK FACTORS'
```

Pour la requête 5 (type de publications), ces données se trouvent dans une colonne des csv dans les dossier-themes.

Mais la liste de journaux ? Et la date de publication ? Ces données se trouvent dans le fichier metadata.csv (colonnes publish_time et journal) :

In [8]:
```
DF=pd.read_csv(f'{chemin_archive}/metadata.csv')
DF.columns
```

Out[8]:
```
Index(['cord_uid', 'sha', 'source_x', 'title', 'doi', 'pmcid', 'pubmed_id',
       'license', 'abstract', 'publish_time', 'authors', 'journal', 'mag_id',
       'who_covidence_id', 'arxiv_id', 'pdf_json_files', 'pmc_json_files',
       'url', 's2_id'],
      dtype='object')
```

Finalement, pour la requête 4, les affiliations des auteurs ne se trouvent que dans les fichiers du dossier /document_parses/pdf_json. Nous avons bien vérifier qu'il n'y avait pas les affiliations dans les fichiers du dossier /document_parses/pmc_json:

In [11]:
```
chemin1 = f'{chemin_archive}/document_parses/pmc_json'
elements1 = os.listdir(chemin1)

files_load=0
no_affiliation_pmc=0
affiliation_pmc=0
for element in elements1:
    with open(f'{chemin1}/{element}', 'r') as f:
        data = json.load(f)
        files_load+=1
        a=data['metadata']['authors']
    if len(a)!=0:
        for i in range(len(a)):
            if len(data['metadata']['authors'][i]['affiliation'])!=0:
                affiliation_pmc+=1
            else:
                no_affiliation_pmc+=1
print(f'{files_load=}')
print(f'{affiliation_pmc=}')
print(f'{no_affiliation_pmc=}')
```

```
affiliation_pmc=2401600
no_affiliation_pmc=0
```

Est-ce qu'on peut récupérer les noms des auteurs qu'à partir du fichier metadata.csv ?

Nous avons essayer de vérifier s'il y avait le même nombre d'auteurs dans DF['authors'] et dans les fichiers .json du dossier /document_parses/pmc_json

Il y a DF['pmc_json_files'], donc on peut accéder au dossier directement à partir du fichier metadata.csv. Mais est-ce que ces fichiers de la colonne 'pmc_json_files' existe ?

In [12]:
```
len(elements1) == len(DF['pmc_json_files'].unique())-1  #le -1 c'est pour la valeur N
```

Out[12]:
```
True
```

In [13]:
```
no_pmc_files=[]
pmc_files_load=0
pmc_files=DF['pmc_json_files']
for file in pmc_files:
    if type(file)==str:     # car NaN est de type float
        try:
            with open(f'{chemin_archive}/{file}', 'r') as f:
                data = json.load(f)
                pmc_files_load+=1
        except:
            no_pmc_files.append(file)
print(f'{pmc_files_load=}')
print(f'{no_pmc_files=}')
```

```
pmc_files_load=315742
no_pmc_files=[]
```

Donc on a bien tous les fichiers du DF['pmc_json_files'] qui existe et sont uniques.

Nous avons fait pareil pour le DF['pdf_json_files'], mais pour certains articles il y en avait 2 ou même 3 fichiers pdf_json_files séparer par un ';'

Mais tous les fichiers existe, même si certains se repète, on accède à tous les fichiers .json du dossier /document_parses/pdf_json

Donc on peut accéder au fichier .json de chaque article directement à partir du fichier metadata.csv

Certains articles ont à la fois un fichier pmc_json_files et pdf_json_files.

In [ ]:
```
double=0
unique=0
for i in range(len(df)):
    if type(df['pdf_json_files'][i])==str and type(df['pmc_json_files'][i])==str:
        double+=1
    else :
        unique+=1
print(f'{double=}')
print(f'{unique=}')
```

On s'est donc demandé si il nous suffisait d'accéder au fichiers pdf_json_files à partir du fichier metadat.csv, pour optimiser le temps de peuplement, mais est-ce que les données dans pdf_json_files et les données dans pmc_json_files sont les mêmes ?

Mais nous avons rencontré un autre problème avec le fichier metadata.csv car il y a beaucoup de valeurs manquantes DOI et le titre des article ne sont pas unique :

In [14]:
```
print(len(DF))
print(len(DF['title'].unique()))
```

```
1056660
850367
```

Et finalement, nous nous sommes rendus comptes que certains auteurs avait 2 affiliation (laboratoire et institus), et que certains articles dans les sous_themes avait plusieurs 'Study Type'.

Peuplement des tables

Pour peupler les tables : theme, sous_themes, studytype, journal et affiliation nous avons choisi d'aller chercher directement les données sans les re-stocker dans un dataframe.

Cette étape a été faite avec le module psycopg2 avant qu'on décide de créer un autre dataframe pour peupler les autres tables. Donc on a décidé de garder notre script, même si on aurait pu faire le peuplement de ces tables avec le dataframe qu'on va créer.

Pour peupler les tables : articles, sous_themes_articles, studytype_articles, authors, articles_authors et affiliation_authors nous avons décider de reconstruire un autre dataframe à partir du fichier metadata.csv et en allant récupérer les données disponibles dans les fichiers .json pour chaque ligne de metadata.csv

Colonnes de metadata qu'on garde : Title, abstract, publish_time, authors, journal, url

Colonnes qu'on doit construire à partir des fichiers .json : Authors_pmc, Authors_pdf, Emails_pmc, Emails_pdf, Afilliation1, Country1, Affiliation2, Country2.

Colonnes qu'on doit construire à partir des fichiers sous_themes.csv : sous_themes et Studytype

Cette dataframe sera converti en fichier .csv pour que ça soit plus pratique et ne pas créer un dataframe à chaque lancement du peuplement

In [17]:
```
Authors_pmc=[]
```

```
Authors_metadata=[]

for k in range(500):
    auteurs_metadata=[]
    auteurs_pmc=[]
    ligne=DF['authors'][k]
    file=DF['pmc_json_files'][k]
    if type(ligne)==str:
        try :
            for author in ligne.split(';'):
                if author.startswith(' '):
                    a=author[1:]
                    Authors_metadata.append(a)
                    auteurs_metadata.append(a)
                else:
                    Authors_metadata.append(author)
                    auteurs_metadata.append(author)
        except:
            Authors_metadata.append(ligne)
            auteurs_metadata.append(ligne)
    if type(file)==str:
        with open(f'{chemin_archive}/{file}','r') as f:
            data=json.load(f)
            L=data['metadata']['authors']
            if len(L)!=0:
                for i in range(len(L)):
                    name=L[i]['last']+', '+L[i]['first']
                    Authors_pmc.append(name)
                    auteurs_pmc.append(name)
    if len(auteurs_pmc)!=len(auteurs_metadata):
        print(ligne)
        print(file)
        if type(file)==str:
            with open(f'{chemin_archive}/{file}','r') as f:
                data=json.load(f)
                L=data['metadata']['authors']
                for i in range(len(L)):
                    print(L[i]['last']+', '+L[i]['first'])
        break
```

```
Froissart, Remy; Roze, Denis; Uzest, Marilyne; Galibert, Lionel; Blanc, Stephane; Mic
halakis, Yannis
document_parses/pmc_json/PMC1054884.xml.json
Froissart, Remy
Roze, Denis
Uzest, Marilyne
Galibert, Lionel
Blanc, Stephane
Michalakis, Yannis
Hull, Roger
```

In [20]:

```
for k in range(20):
    auteurs_pdf=[]
    auteurs_pmc=[]
    file_pmc=DF['pmc_json_files'][k]
    file_pdf=DF['pdf_json_files'][k]
    if type(file_pmc)==str:
        with open(f'{chemin_archive}/{file_pmc}','r') as f:
            data=json.load(f)
            L=data['metadata']['authors']
            if len(L)!=0:
                for i in range(len(L)):
                    name=L[i]['last']+', '+L[i]['first']
                    auteurs_pmc.append(name)
    if type(file_pdf)==str:
        if ';' in file_pdf:
```

```
            liste_file=file_pdf.split(';')
            for fil in liste_file:
                if fil.startswith(' '):
                    fi=fil[1:]
                else:
                    fi=fil
                try:
                    with open(f'{chemin_archive}/{fi}', 'r') as f:
                        data = json.load(f)
                        L=data['metadata']['authors']
                        if len(L)!=0:
                            for i in range(len(L)):
                                name=L[i]['last']+', '+L[i]['first']
                                auteurs_pdf.append(name)
                except:
                    print(fi)
        else:
            try:
                with open(f'{chemin_archive}/{file_pdf}', 'r') as f:
                    data = json.load(f)
                    L=data['metadata']['authors']
                    if len(L)!=0:
                        for i in range(len(L)):
                            name=L[i]['last']+', '+L[i]['first']
                            auteurs_pdf.append(name)
            except:
                print(file_pdf)

    if len(auteurs_pmc)!=len(auteurs_pdf):
        print(file_pmc)
        print(auteurs_pmc)
        print(file_pdf)
        print(auteurs_pdf)
        print(k)
```

```
document_parses/pmc_json/PMC59574.xml.json
['Fagan, Karen', 'McMurtry, Ivan', 'Rodman, David']
document_parses/pdf_json/348055649b6b8cf2b9a376498df9bf41f7123605.json
[]
3
document_parses/pmc_json/PMC306617.xml.json
['Ploubidou, Aspasia', 'Moreau, Violaine', 'Ashman, Keith', 'Reckmann, Inge', 'Gonzál
ez, Cayetano', 'Way, Michael']
document_parses/pdf_json/44102e3e69e70ad2a73e753133283334ce1f8736.json
['Ploubidou, Aspasia', 'Moreau, Violaine', 'Ashman, Keith', 'Reckmann, Inge', 'Gonza,
Cayetano', 'Lez, Â', 'Way, Michael']
13
document_parses/pmc_json/PMC468896.xml.json
['Verheij, Joanne', 'Groeneveld, AB', 'Beishuizen, Albertus', 'Lingen, Arthur', 'Simo
ons-Smit, Alberdina', 'van Schijndel, Rob']
document_parses/pdf_json/6a8ac55ea2a1fbd99deb683e24dd986e55e707b3.json
['Verheij, Joanne', 'Ab, Johan', 'Groeneveld, ', 'Beishuizen, Albertus', 'Van Lingen,
Arthur', 'Simoons-Smit, Alberdina', 'Jm, Rob', 'Van Schijndel, Strack']
16
```

In [22]:

```
for k in range(100):
    auteurs_pdf=[]
    file_pdf=DF['pdf_json_files'][k]
    if type(file_pdf)==str:
        if ';' in file_pdf:
            liste_file=file_pdf.split(';')
            for fil in liste_file:
                auteurs=[]
                if fil.startswith(' '):
                    try:
                        print(fil[1:])
```

```
                        with open(f'{chemin_archive}/{fil[1:]}', 'r') as f:

                            data = json.load(f)
                            L=data['metadata']['authors']
                            if len(L)!=0:
                                for i in range(len(L)):
                                    name=L[i]['last']+', '+L[i]['first']
                                    auteurs.append(name)

                    except:
                        print(fil,'e')

                else:
                    try:
                        print(fil)
                        with open(f'{chemin_archive}/{fil}', 'r') as f:
                            data = json.load(f)
                            L=data['metadata']['authors']
                            if len(L)!=0:
                                for i in range(len(L)):
                                    name=L[i]['last']+', '+L[i]['first']
                                    auteurs.append(name)

                    except:
                        print(fil,'e')
                auteurs_pdf.append(auteurs)
    for j in auteurs_pdf:
        print(k)
        print(j)
```

```
document_parses/pdf_json/4eb6e165ee705e2ae2a24ed2d4e67da42831ff4a.json
document_parses/pdf_json/d4f0247db5e916c20eae3f6d772e8572eb828236.json
75
['Malanoski, Anthony', 'Lin, Baochuan', 'Wang, Zheng', 'Schnur, Joel', 'Stenger, Davi
d']
75
[]
document_parses/pdf_json/ad9ac0ac5e7da097253fd545b56e2b15ee9de34f.json
document_parses/pdf_json/daee7f7d31f4bf1c0ef883bcd6c124b6e94cbee7.json
76
['Mccrate, Nina', 'Varner, Mychel', 'Kim, Kenneth', 'Nagan, Maria', 'Uuu, [', 'Yaria
n, M', 'Marszalek, E', 'Sochacka, A', 'Malkiewicz, R', 'Guenther, A', 'Miskiewicz,
P']
76
['Mccrate, Nina', 'Varner, Mychel', 'Kim, Kenneth', 'Nagan, Maria']
document_parses/pdf_json/52566dccb4bd8044edc87b1a0aa268320a6ea3d4.json
document_parses/pdf_json/8b39433dd865c0f71c7b2f333e1f506b73d722f1.json
77
['Kutyavin, Igor', 'Milesi, Dave', 'Belousov, Yevgeniy', 'Podyminogin, Mikhail', 'Vor
obiev, Alexei', 'Gorn, Vladimir', 'Lukhtanov, Eugeny', 'Vermeulen, Nicolaas', 'Mahone
y, Walt']
77
[]
```

In [29]:

```
files_pdf=[]
for k in range(len(DF)):
    file_pdf=DF['pdf_json_files'][k]
    if type(file_pdf)==str:
        if ';' in file_pdf:
            liste_file=file_pdf.split(';')
            for fil in liste_file:
                if fil.startswith(' '):
                    files_pdf.append(fil[1:])
                else:
                    files_pdf.append(fil)
        else:
```

```
            files_pdf.append(file_pdf)
print(len(files_pdf))
print(len(set(files_pdf)))
```

```
401270
401214
```

In [ ]: