Annexe

Installation des packages nécessaires

```
pip install requests pandas numpy matplotlib

Defaulting to user installation because normal site-packages is not
writeable
Requirement already satisfied: requests in c:\programdata\anaconda3\
lib\site-packages (2.31.0)
Requirement already satisfied: pandas in c:\programdata\anaconda3\lib\
site-packages (2.1.4)
Requirement already satisfied: numpy in c:\programdata\anaconda3\lib\
site-packages (1.26.4)
Requirement already satisfied: matplotlib in c:\programdata\anaconda3\
lib\site-packages (3.8.0)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\
programdata\anaconda3\lib\site-packages (from requests) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in c:\programdata\
anaconda3\lib\site-packages (from requests) (3.4)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\programdata\
anaconda3\lib\site-packages (from requests) (2.0.7)
Requirement already satisfied: certifi>=2017.4.17 in c:\programdata\
anaconda3\lib\site-packages (from requests) (2024.2.2)
Requirement already satisfied: python-dateutil>=2.8.2 in c:\
programdata\anaconda3\lib\site-packages (from pandas) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in c:\programdata\
anaconda3\lib\site-packages (from pandas) (2023.3.post1)
Requirement already satisfied: tzdata>=2022.1 in c:\programdata\
anaconda3\lib\site-packages (from pandas) (2023.3)
Requirement already satisfied: contourpy>=1.0.1 in c:\programdata\
anaconda3\lib\site-packages (from matplotlib) (1.2.0)
Requirement already satisfied: cycler>=0.10 in c:\programdata\
anaconda3\lib\site-packages (from matplotlib) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in c:\programdata\
anaconda3\lib\site-packages (from matplotlib) (4.25.0)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\programdata\
anaconda3\lib\site-packages (from matplotlib) (1.4.4)
Requirement already satisfied: packaging>=20.0 in c:\programdata\
anaconda3\lib\site-packages (from matplotlib) (23.1)
Requirement already satisfied: pillow>=6.2.0 in c:\programdata\
anaconda3\lib\site-packages (from matplotlib) (10.2.0)
Requirement already satisfied: pyparsing>=2.3.1 in c:\programdata\
anaconda3\lib\site-packages (from matplotlib) (3.0.9)
Requirement already satisfied: six>=1.5 in c:\programdata\anaconda3\
lib\site-packages (from python-dateutil>=2.8.2->pandas) (1.16.0)
Note: you may need to restart the kernel to use updated packages.
```

```
DEPRECATION: Loading egg at c:\programdata\anaconda3\lib\site-
packages\vboxapi-1.0-py3.11.egg is deprecated. pip 24.3 will enforce
this behaviour change. A possible replacement is to use pip for
package installation.. Discussion can be found at
https://github.com/pypa/pip/issues/12330
```

Récupération et nettoyage des données de l'API Scopus

```
import requests
import pandas as pd

def fetch_scopus_data(api_key, query, count=25):
    """Fetch data from the Scopus API."""
    url = 'https://api.elsevier.com/content/search/scopus'
    params = {
        'apiKey': api_key,
        'query': query,
        'count': count
    }

    try:
        # Effectuer la requête GET à l'API Scopus
        response = requests.get(url, params=params)
        response.raise_for_status()  # Vérifier s'il y a eu une erreur
dans la requête

        # Convertir la réponse JSON en dictionnaire
        data = response.json()

        # Vérifier si les données attendues sont présentes dans la
réponse
        if 'search-results' in data and 'entry' in data['search-
results']:
            print("La clé API est valide et fonctionne correctement.")
            print(f"Nombre de résultats obtenus : {len(data['search-
results']['entry'])}")
            return data['search-results']['entry']
        else:
            print("La structure de la réponse JSON ne contient pas les
clés attendues.")
            return None

    except requests.exceptions.HTTPError as http_err:
        print(f'Erreur HTTP {response.status_code}:
{response.reason}')
        print(response.text)
    except requests.exceptions.RequestException as req_err:
        print(f'Erreur de requête: {req_err}')
    except Exception as err:
```

```
        print(f'Erreur: {err}')

def parse_freetoread(value):
    """Transform the 'freetoread.value' column to readable values."""
    if isinstance(value, list):
        return ', '.join([item['$'] for item in value])
    return value

def clean_and_save_data(entries, filename):
    """Clean the data and save it to a CSV file."""
    if entries:
        # Convertir les entrées JSON en DataFrame Pandas
        df = pd.json_normalize(entries)

        # Nettoyer et organiser les données
        if 'freetoread.value' in df.columns:
            df['freetoread.value'] =
df['freetoread.value'].apply(parse_freetoread)

        # Définir les options d'affichage de Pandas pour afficher
toutes les lignes et colonnes
        pd.set_option('display.max_rows', None)
        pd.set_option('display.max_columns', None)
        pd.set_option('display.width', None)
        pd.set_option('display.max_colwidth', None)

        # Afficher le DataFrame nettoyé
        print(df)

        # Sauvegarder le DataFrame dans un fichier CSV
        df.to_csv(filename, index=False)
        print(f"Les données ont été nettoyées et sauvegardées dans le
fichier {filename}")

# Utiliser les fonctions pour récupérer, nettoyer et sauvegarder les
données
api_key = '9aebde1fa88b0b7325c7d8054dd3e754'
query = 'KEY(scopus)'
filename = 'api_scopus_data.csv'

entries = fetch_scopus_data(api_key, query)
clean_and_save_data(entries, filename)
```

```
La clé API est valide et fonctionne correctement.
Nombre de résultats obtenus : 25
    @_fa  \
0   true
1   true
2   true
3   true
```

```
4   true
5   true
6   true
7   true
8   true
9   true
10  true
11  true
12  true
13  true
14  true
15  true
16  true
17  true
18  true
19  true
20  true
21  true
22  true
23  true
24  true


link  \
0   [{'@_fa': 'true', '@ref': 'self', '@href':
'https://api.elsevier.com/content/abstract/scopus_id/85195598662'},
{'@_fa': 'true', '@ref': 'author-affiliation', '@href':
'https://api.elsevier.com/content/abstract/scopus_id/85195598662?
field=author,affiliation'}, {'@_fa': 'true', '@ref': 'scopus',
'@href': 'https://www.scopus.com/inward/record.uri?
partnerID=HzOxMe3b&scp=85195598662&origin=inward'}, {'@_fa': 'true',
'@ref': 'scopus-citedby', '@href':
'https://www.scopus.com/inward/citedby.uri?
partnerID=HzOxMe3b&scp=85195598662&origin=inward'}]
1   [{'@_fa': 'true', '@ref': 'self', '@href':
'https://api.elsevier.com/content/abstract/scopus_id/85195533502'},
{'@_fa': 'true', '@ref': 'author-affiliation', '@href':
'https://api.elsevier.com/content/abstract/scopus_id/85195533502?
field=author,affiliation'}, {'@_fa': 'true', '@ref': 'scopus',
'@href': 'https://www.scopus.com/inward/record.uri?
partnerID=HzOxMe3b&scp=85195533502&origin=inward'}, {'@_fa': 'true',
'@ref': 'scopus-citedby', '@href':
'https://www.scopus.com/inward/citedby.uri?
partnerID=HzOxMe3b&scp=85195533502&origin=inward'}]
2   [{'@_fa': 'true', '@ref': 'self', '@href':
'https://api.elsevier.com/content/abstract/scopus_id/85195438916'},
{'@_fa': 'true', '@ref': 'author-affiliation', '@href':
'https://api.elsevier.com/content/abstract/scopus_id/85195438916?
field=author,affiliation'}, {'@_fa': 'true', '@ref': 'scopus',
```

'@href': 'https://www.scopus.com/inward/record.uri?
partnerID=HzOxMe3b&scp=85195438916&origin=inward'}, {'@_fa': 'true',
'@ref': 'scopus-citedby', '@href':
'https://www.scopus.com/inward/citedby.uri?
partnerID=HzOxMe3b&scp=85195438916&origin=inward'}]
3 [{'@_fa': 'true', '@ref': 'self', '@href':
'https://api.elsevier.com/content/abstract/scopus_id/85195353625'},
{'@_fa': 'true', '@ref': 'author-affiliation', '@href':
'https://api.elsevier.com/content/abstract/scopus_id/85195353625?
field=author,affiliation'}, {'@_fa': 'true', '@ref': 'scopus',
'@href': 'https://www.scopus.com/inward/record.uri?
partnerID=HzOxMe3b&scp=85195353625&origin=inward'}, {'@_fa': 'true',
'@ref': 'scopus-citedby', '@href':
'https://www.scopus.com/inward/citedby.uri?
partnerID=HzOxMe3b&scp=85195353625&origin=inward'}]
4 [{'@_fa': 'true', '@ref': 'self', '@href':
'https://api.elsevier.com/content/abstract/scopus_id/85195014060'},
{'@_fa': 'true', '@ref': 'author-affiliation', '@href':
'https://api.elsevier.com/content/abstract/scopus_id/85195014060?
field=author,affiliation'}, {'@_fa': 'true', '@ref': 'scopus',
'@href': 'https://www.scopus.com/inward/record.uri?
partnerID=HzOxMe3b&scp=85195014060&origin=inward'}, {'@_fa': 'true',
'@ref': 'scopus-citedby', '@href':
'https://www.scopus.com/inward/citedby.uri?
partnerID=HzOxMe3b&scp=85195014060&origin=inward'}]
5 [{'@_fa': 'true', '@ref': 'self', '@href':
'https://api.elsevier.com/content/abstract/scopus_id/85195009149'},
{'@_fa': 'true', '@ref': 'author-affiliation', '@href':
'https://api.elsevier.com/content/abstract/scopus_id/85195009149?
field=author,affiliation'}, {'@_fa': 'true', '@ref': 'scopus',
'@href': 'https://www.scopus.com/inward/record.uri?
partnerID=HzOxMe3b&scp=85195009149&origin=inward'}, {'@_fa': 'true',
'@ref': 'scopus-citedby', '@href':
'https://www.scopus.com/inward/citedby.uri?
partnerID=HzOxMe3b&scp=85195009149&origin=inward'}]
6 [{'@_fa': 'true', '@ref': 'self', '@href':
'https://api.elsevier.com/content/abstract/scopus_id/85194992309'},
{'@_fa': 'true', '@ref': 'author-affiliation', '@href':
'https://api.elsevier.com/content/abstract/scopus_id/85194992309?
field=author,affiliation'}, {'@_fa': 'true', '@ref': 'scopus',
'@href': 'https://www.scopus.com/inward/record.uri?
partnerID=HzOxMe3b&scp=85194992309&origin=inward'}, {'@_fa': 'true',
'@ref': 'scopus-citedby', '@href':
'https://www.scopus.com/inward/citedby.uri?
partnerID=HzOxMe3b&scp=85194992309&origin=inward'}]
7 [{'@_fa': 'true', '@ref': 'self', '@href':
'https://api.elsevier.com/content/abstract/scopus_id/85194835747'},
{'@_fa': 'true', '@ref': 'author-affiliation', '@href':
'https://api.elsevier.com/content/abstract/scopus_id/85194835747?

```
field=author,affiliation'}, {'@_fa': 'true', '@ref': 'scopus',
'@href': 'https://www.scopus.com/inward/record.uri?
partnerID=HzOxMe3b&scp=85194835747&origin=inward'}, {'@_fa': 'true',
'@ref': 'scopus-citedby', '@href':
'https://www.scopus.com/inward/citedby.uri?
partnerID=HzOxMe3b&scp=85194835747&origin=inward'}]
8   [{'@_fa': 'true', '@ref': 'self', '@href':
'https://api.elsevier.com/content/abstract/scopus_id/85194828622'},
{'@_fa': 'true', '@ref': 'author-affiliation', '@href':
'https://api.elsevier.com/content/abstract/scopus_id/85194828622?
field=author,affiliation'}, {'@_fa': 'true', '@ref': 'scopus',
'@href': 'https://www.scopus.com/inward/record.uri?
partnerID=HzOxMe3b&scp=85194828622&origin=inward'}, {'@_fa': 'true',
'@ref': 'scopus-citedby', '@href':
'https://www.scopus.com/inward/citedby.uri?
partnerID=HzOxMe3b&scp=85194828622&origin=inward'}]
9   [{'@_fa': 'true', '@ref': 'self', '@href':
'https://api.elsevier.com/content/abstract/scopus_id/85194036822'},
{'@_fa': 'true', '@ref': 'author-affiliation', '@href':
'https://api.elsevier.com/content/abstract/scopus_id/85194036822?
field=author,affiliation'}, {'@_fa': 'true', '@ref': 'scopus',
'@href': 'https://www.scopus.com/inward/record.uri?
partnerID=HzOxMe3b&scp=85194036822&origin=inward'}, {'@_fa': 'true',
'@ref': 'scopus-citedby', '@href':
'https://www.scopus.com/inward/citedby.uri?
partnerID=HzOxMe3b&scp=85194036822&origin=inward'}]
10  [{'@_fa': 'true', '@ref': 'self', '@href':
'https://api.elsevier.com/content/abstract/scopus_id/85193980287'},
{'@_fa': 'true', '@ref': 'author-affiliation', '@href':
'https://api.elsevier.com/content/abstract/scopus_id/85193980287?
field=author,affiliation'}, {'@_fa': 'true', '@ref': 'scopus',
'@href': 'https://www.scopus.com/inward/record.uri?
partnerID=HzOxMe3b&scp=85193980287&origin=inward'}, {'@_fa': 'true',
'@ref': 'scopus-citedby', '@href':
'https://www.scopus.com/inward/citedby.uri?
partnerID=HzOxMe3b&scp=85193980287&origin=inward'}]
11  [{'@_fa': 'true', '@ref': 'self', '@href':
'https://api.elsevier.com/content/abstract/scopus_id/85193819821'},
{'@_fa': 'true', '@ref': 'author-affiliation', '@href':
'https://api.elsevier.com/content/abstract/scopus_id/85193819821?
field=author,affiliation'}, {'@_fa': 'true', '@ref': 'scopus',
'@href': 'https://www.scopus.com/inward/record.uri?
partnerID=HzOxMe3b&scp=85193819821&origin=inward'}, {'@_fa': 'true',
'@ref': 'scopus-citedby', '@href':
'https://www.scopus.com/inward/citedby.uri?
partnerID=HzOxMe3b&scp=85193819821&origin=inward'}]
12  [{'@_fa': 'true', '@ref': 'self', '@href':
'https://api.elsevier.com/content/abstract/scopus_id/85192885989'},
{'@_fa': 'true', '@ref': 'author-affiliation', '@href':
```

'https://api.elsevier.com/content/abstract/scopus_id/85192885989?
field=author,affiliation'}, {'@_fa': 'true', '@ref': 'scopus',
'@href': 'https://www.scopus.com/inward/record.uri?
partnerID=HzOxMe3b&scp=85192885989&origin=inward'}, {'@_fa': 'true',
'@ref': 'scopus-citedby', '@href':
'https://www.scopus.com/inward/citedby.uri?
partnerID=HzOxMe3b&scp=85192885989&origin=inward'}]
13 [{'@_fa': 'true', '@ref': 'self', '@href':
'https://api.elsevier.com/content/abstract/scopus_id/85192058182'},
{'@_fa': 'true', '@ref': 'author-affiliation', '@href':
'https://api.elsevier.com/content/abstract/scopus_id/85192058182?
field=author,affiliation'}, {'@_fa': 'true', '@ref': 'scopus',
'@href': 'https://www.scopus.com/inward/record.uri?
partnerID=HzOxMe3b&scp=85192058182&origin=inward'}, {'@_fa': 'true',
'@ref': 'scopus-citedby', '@href':
'https://www.scopus.com/inward/citedby.uri?
partnerID=HzOxMe3b&scp=85192058182&origin=inward'}]
14 [{'@_fa': 'true', '@ref': 'self', '@href':
'https://api.elsevier.com/content/abstract/scopus_id/85191965533'},
{'@_fa': 'true', '@ref': 'author-affiliation', '@href':
'https://api.elsevier.com/content/abstract/scopus_id/85191965533?
field=author,affiliation'}, {'@_fa': 'true', '@ref': 'scopus',
'@href': 'https://www.scopus.com/inward/record.uri?
partnerID=HzOxMe3b&scp=85191965533&origin=inward'}, {'@_fa': 'true',
'@ref': 'scopus-citedby', '@href':
'https://www.scopus.com/inward/citedby.uri?
partnerID=HzOxMe3b&scp=85191965533&origin=inward'}]
15 [{'@_fa': 'true', '@ref': 'self', '@href':
'https://api.elsevier.com/content/abstract/scopus_id/85191834350'},
{'@_fa': 'true', '@ref': 'author-affiliation', '@href':
'https://api.elsevier.com/content/abstract/scopus_id/85191834350?
field=author,affiliation'}, {'@_fa': 'true', '@ref': 'scopus',
'@href': 'https://www.scopus.com/inward/record.uri?
partnerID=HzOxMe3b&scp=85191834350&origin=inward'}, {'@_fa': 'true',
'@ref': 'scopus-citedby', '@href':
'https://www.scopus.com/inward/citedby.uri?
partnerID=HzOxMe3b&scp=85191834350&origin=inward'}]
16 [{'@_fa': 'true', '@ref': 'self', '@href':
'https://api.elsevier.com/content/abstract/scopus_id/85191380743'},
{'@_fa': 'true', '@ref': 'author-affiliation', '@href':
'https://api.elsevier.com/content/abstract/scopus_id/85191380743?
field=author,affiliation'}, {'@_fa': 'true', '@ref': 'scopus',
'@href': 'https://www.scopus.com/inward/record.uri?
partnerID=HzOxMe3b&scp=85191380743&origin=inward'}, {'@_fa': 'true',
'@ref': 'scopus-citedby', '@href':
'https://www.scopus.com/inward/citedby.uri?
partnerID=HzOxMe3b&scp=85191380743&origin=inward'}]
17 [{'@_fa': 'true', '@ref': 'self', '@href':
'https://api.elsevier.com/content/abstract/scopus_id/85191351457'},

{'@_fa': 'true', '@ref': 'author-affiliation', '@href':
'https://api.elsevier.com/content/abstract/scopus_id/85191351457?
field=author,affiliation'}, {'@_fa': 'true', '@ref': 'scopus',
'@href': 'https://www.scopus.com/inward/record.uri?
partnerID=HzOxMe3b&scp=85191351457&origin=inward'}, {'@_fa': 'true',
'@ref': 'scopus-citedby', '@href':
'https://www.scopus.com/inward/citedby.uri?
partnerID=HzOxMe3b&scp=85191351457&origin=inward'}]
18 [{'@_fa': 'true', '@ref': 'self', '@href':
'https://api.elsevier.com/content/abstract/scopus_id/85191067629'},
{'@_fa': 'true', '@ref': 'author-affiliation', '@href':
'https://api.elsevier.com/content/abstract/scopus_id/85191067629?
field=author,affiliation'}, {'@_fa': 'true', '@ref': 'scopus',
'@href': 'https://www.scopus.com/inward/record.uri?
partnerID=HzOxMe3b&scp=85191067629&origin=inward'}, {'@_fa': 'true',
'@ref': 'scopus-citedby', '@href':
'https://www.scopus.com/inward/citedby.uri?
partnerID=HzOxMe3b&scp=85191067629&origin=inward'}]
19 [{'@_fa': 'true', '@ref': 'self', '@href':
'https://api.elsevier.com/content/abstract/scopus_id/85190759010'},
{'@_fa': 'true', '@ref': 'author-affiliation', '@href':
'https://api.elsevier.com/content/abstract/scopus_id/85190759010?
field=author,affiliation'}, {'@_fa': 'true', '@ref': 'scopus',
'@href': 'https://www.scopus.com/inward/record.uri?
partnerID=HzOxMe3b&scp=85190759010&origin=inward'}, {'@_fa': 'true',
'@ref': 'scopus-citedby', '@href':
'https://www.scopus.com/inward/citedby.uri?
partnerID=HzOxMe3b&scp=85190759010&origin=inward'}]
20 [{'@_fa': 'true', '@ref': 'self', '@href':
'https://api.elsevier.com/content/abstract/scopus_id/85190699478'},
{'@_fa': 'true', '@ref': 'author-affiliation', '@href':
'https://api.elsevier.com/content/abstract/scopus_id/85190699478?
field=author,affiliation'}, {'@_fa': 'true', '@ref': 'scopus',
'@href': 'https://www.scopus.com/inward/record.uri?
partnerID=HzOxMe3b&scp=85190699478&origin=inward'}, {'@_fa': 'true',
'@ref': 'scopus-citedby', '@href':
'https://www.scopus.com/inward/citedby.uri?
partnerID=HzOxMe3b&scp=85190699478&origin=inward'}]
21 [{'@_fa': 'true', '@ref': 'self', '@href':
'https://api.elsevier.com/content/abstract/scopus_id/85190672606'},
{'@_fa': 'true', '@ref': 'author-affiliation', '@href':
'https://api.elsevier.com/content/abstract/scopus_id/85190672606?
field=author,affiliation'}, {'@_fa': 'true', '@ref': 'scopus',
'@href': 'https://www.scopus.com/inward/record.uri?
partnerID=HzOxMe3b&scp=85190672606&origin=inward'}, {'@_fa': 'true',
'@ref': 'scopus-citedby', '@href':
'https://www.scopus.com/inward/citedby.uri?
partnerID=HzOxMe3b&scp=85190672606&origin=inward'}]
22 [{'@_fa': 'true', '@ref': 'self', '@href':

```
'https://api.elsevier.com/content/abstract/scopus_id/85190542186'},
{'@_fa': 'true', '@ref': 'author-affiliation', '@href':
'https://api.elsevier.com/content/abstract/scopus_id/85190542186?
field=author,affiliation'}, {'@_fa': 'true', '@ref': 'scopus',
'@href': 'https://www.scopus.com/inward/record.uri?
partnerID=HzOxMe3b&scp=85190542186&origin=inward'}, {'@_fa': 'true',
'@ref': 'scopus-citedby', '@href':
'https://www.scopus.com/inward/citedby.uri?
partnerID=HzOxMe3b&scp=85190542186&origin=inward'}]
23  [{'@_fa': 'true', '@ref': 'self', '@href':
'https://api.elsevier.com/content/abstract/scopus_id/85190405818'},
{'@_fa': 'true', '@ref': 'author-affiliation', '@href':
'https://api.elsevier.com/content/abstract/scopus_id/85190405818?
field=author,affiliation'}, {'@_fa': 'true', '@ref': 'scopus',
'@href': 'https://www.scopus.com/inward/record.uri?
partnerID=HzOxMe3b&scp=85190405818&origin=inward'}, {'@_fa': 'true',
'@ref': 'scopus-citedby', '@href':
'https://www.scopus.com/inward/citedby.uri?
partnerID=HzOxMe3b&scp=85190405818&origin=inward'}]
24  [{'@_fa': 'true', '@ref': 'self', '@href':
'https://api.elsevier.com/content/abstract/scopus_id/85190390841'},
{'@_fa': 'true', '@ref': 'author-affiliation', '@href':
'https://api.elsevier.com/content/abstract/scopus_id/85190390841?
field=author,affiliation'}, {'@_fa': 'true', '@ref': 'scopus',
'@href': 'https://www.scopus.com/inward/record.uri?
partnerID=HzOxMe3b&scp=85190390841&origin=inward'}, {'@_fa': 'true',
'@ref': 'scopus-citedby', '@href':
'https://www.scopus.com/inward/citedby.uri?
partnerID=HzOxMe3b&scp=85190390841&origin=inward'}]

                                                    prism:url  \
0   https://api.elsevier.com/content/abstract/scopus_id/85195598662
1   https://api.elsevier.com/content/abstract/scopus_id/85195533502
2   https://api.elsevier.com/content/abstract/scopus_id/85195438916
3   https://api.elsevier.com/content/abstract/scopus_id/85195353625
4   https://api.elsevier.com/content/abstract/scopus_id/85195014060
5   https://api.elsevier.com/content/abstract/scopus_id/85195009149
6   https://api.elsevier.com/content/abstract/scopus_id/85194992309
7   https://api.elsevier.com/content/abstract/scopus_id/85194835747
8   https://api.elsevier.com/content/abstract/scopus_id/85194828622
9   https://api.elsevier.com/content/abstract/scopus_id/85194036822
10  https://api.elsevier.com/content/abstract/scopus_id/85193980287
11  https://api.elsevier.com/content/abstract/scopus_id/85193819821
12  https://api.elsevier.com/content/abstract/scopus_id/85192885989
13  https://api.elsevier.com/content/abstract/scopus_id/85192058182
14  https://api.elsevier.com/content/abstract/scopus_id/85191965533
15  https://api.elsevier.com/content/abstract/scopus_id/85191834350
16  https://api.elsevier.com/content/abstract/scopus_id/85191380743
17  https://api.elsevier.com/content/abstract/scopus_id/85191351457
```

```
18  https://api.elsevier.com/content/abstract/scopus_id/85191067629
19  https://api.elsevier.com/content/abstract/scopus_id/85190759010
20  https://api.elsevier.com/content/abstract/scopus_id/85190699478
21  https://api.elsevier.com/content/abstract/scopus_id/85190672606
22  https://api.elsevier.com/content/abstract/scopus_id/85190542186
23  https://api.elsevier.com/content/abstract/scopus_id/85190405818
24  https://api.elsevier.com/content/abstract/scopus_id/85190390841

             dc:identifier                   eid  \
0   SCOPUS_ID:85195598662  2-s2.0-85195598662
1   SCOPUS_ID:85195533502  2-s2.0-85195533502
2   SCOPUS_ID:85195438916  2-s2.0-85195438916
3   SCOPUS_ID:85195353625  2-s2.0-85195353625
4   SCOPUS_ID:85195014060  2-s2.0-85195014060
5   SCOPUS_ID:85195009149  2-s2.0-85195009149
6   SCOPUS_ID:85194992309  2-s2.0-85194992309
7   SCOPUS_ID:85194835747  2-s2.0-85194835747
8   SCOPUS_ID:85194828622  2-s2.0-85194828622
9   SCOPUS_ID:85194036822  2-s2.0-85194036822
10  SCOPUS_ID:85193980287  2-s2.0-85193980287
11  SCOPUS_ID:85193819821  2-s2.0-85193819821
12  SCOPUS_ID:85192885989  2-s2.0-85192885989
13  SCOPUS_ID:85192058182  2-s2.0-85192058182
14  SCOPUS_ID:85191965533  2-s2.0-85191965533
15  SCOPUS_ID:85191834350  2-s2.0-85191834350
16  SCOPUS_ID:85191380743  2-s2.0-85191380743
17  SCOPUS_ID:85191351457  2-s2.0-85191351457
18  SCOPUS_ID:85191067629  2-s2.0-85191067629
19  SCOPUS_ID:85190759010  2-s2.0-85190759010
20  SCOPUS_ID:85190699478  2-s2.0-85190699478
21  SCOPUS_ID:85190672606  2-s2.0-85190672606
22  SCOPUS_ID:85190542186  2-s2.0-85190542186
23  SCOPUS_ID:85190405818  2-s2.0-85190405818
24  SCOPUS_ID:85190390841  2-s2.0-85190390841


dc:title  \
0
Effect of zinc supplementation on glycemic biomarkers: an umbrella of
interventional meta-analyses
1
Barriers and facilitators to implementing workplace interventions to
promote mental health: qualitative evidence synthesis
2
Utility of thiol/disulphide homeostasis as a biomarker for acute
appendicitis: a systematic review and meta-analysis
3                                          The impact of
immunosuppression on the mortality and hospitalization of Monkeypox: a
systematic review and meta-analysis of the 2022 outbreak
```

4
75 years’ journey of malaria publications in English: what and where?

5
Sharper vision, steady hands: can robots improve subretinal drug delivery? Systematic review
6
Association of vitamin D receptor genetic polymorphisms with the risk of infertility: a systematic review and meta-analysis
7 Systematic review and meta-analysis of association between plasminogen activator inhibitor-1 4G/5G polymorphism and recurrent pregnancy loss: an update
8 Impact of frailty on mortality, hospitalization, cardiovascular events, and complications in patients with diabetes mellitus: a systematic review and meta-analysis
9
Use of platelet-rich plasma and platelet-rich fibrin in burn wound healing and skin grafting: a systematic review
10 mRNA markers for survival prediction in glioblastoma multiforme patients: a systematic review with bioinformatic analyses
11
Global prevalence of sexual dysfunction in cardiovascular patients: a systematic review and meta-analysis
12 Association of prothrombin time, thrombin time and activated partial thromboplastin time levels with preeclampsia: a systematic review and meta-analysis
13 A systematic review and meta-analysis of randomized controlled trials on the effectiveness of high-intensity laser therapy in the management of neck pain
14
What do we know about Aquafilling tissue filler? – A systematic review

15
Does electrophysical agents work for cellulite treatment? a systematic review of clinical trials
16
Flank versus prone position in percutaneous nephrolithotomy: a meta-analysis of randomized controlled studies
17 Leisure-time and occupational physical activity and risk of cardiovascular disease incidence: a systematic-review and dose-response meta-analysis of prospective cohort studies
18
Protocol for a systematic review and meta-analysis on Janus kinase inhibitors in the management of vitiligo
19
Clinical and ex-vivo effect of LASERs on prevention of early-enamel caries: systematic review & meta-analyses
20 Analyzing global research trends and focal points in the

```
utilization of laser techniques for the treatment of urolithiasis from
1978 to 2022: visualization and bibliometric analysis
21
Between artificial intelligence and customer experience: a literature
review on the intersection
22                          Efficacy and safety of cangrelor as
compared to ticagrelor in patients with ST-elevated myocardial
infarction (STEMI): a systematic review and meta-analysis
23
Proportion of cancer cases and deaths attributable to potentially
modifiable risk factors in Peru
24
Acute spinal cord injury serum biomarkers in human and rat: a scoping
systematic review

                dc:creator  \
0             Daneshvar M.
1              Paterson C.
2              Krishnan N.
3                 Azzam A.
4                 Deora N.
5             Łajczak P.M.
6            Moradkhani A.
7             Maghsudlu M.
8                  Miao Z.
9              Manasyan A.
10                Azimi P.
11              Ziapour A.
12            Alemayehu E.
13  de la Barra Ortiz H.A.
14         Radziszewski M.
15              Longano C.
16                Zheng C.
17               Kazemi A.
18               Pranić S.
19         Abd El-Aal N.H.
20            Abushamma F.
21            Peruchini M.
22          Chakraborty S.
23  De La Cruz-Vargas J.A.
24                 Shool S.


prism:publicationName  \
0                                    Diabetology and Metabolic
Syndrome
1                                                    Systematic
Reviews
2                                       Pediatric Surgery
International
```

```
3                                                          Virology
Journal
4                                                           Malaria
Journal
5                                             Journal of Robotic
Surgery
6                                           BMC Pregnancy and
Childbirth
7                                                        Thrombosis
Journal
8                                      Diabetology and Metabolic
Syndrome
9                                    European Journal of Plastic
Surgery
10                                                                BMC
Cancer
11                                                       Systematic
Reviews
12                                          BMC Pregnancy and
Childbirth
13                                             Lasers in Medical
Science
14                                   European Journal of Plastic
Surgery
15                                             Lasers in Medical
Science
16
Urolithiasis
17  International Journal of Behavioral Nutrition and Physical
Activity
18                                                       Systematic
Reviews
19                                             Lasers in Medical
Science
20
Urolithiasis
21                                 Discover Artificial
Intelligence
22                                               Egyptian Heart
Journal
23                                                                BMC
Cancer
24                                          Spinal Cord Series and
Cases

   prism:eIssn prism:volume prism:issueIdentifier prism:pageRange  \
0     17585996           16                      1            None
1     20464053           13                      1            None
2     14379813           40                      1            None
3     1743422X           21                      1            None
```

```
4     14752875           23                       1             None
5     18632491           18                       1             None
6     14712393           24                       1             None
7     14779560           22                       1             None
8     17585996           16                       1             None
9     14350130           47                       1             None
10    14712407           24                       1             None
11    20464053           13                       1             None
12    14712393           24                       1             None
13    1435604X           39                       1             None
14    14350130           47                       1             None
15    1435604X           39                       1             None
16    21947236           52                       1             None
17    14795868           21                       1             None
18    20464053           13                       1             None
19    1435604X           39                       1             None
20    21947236           52                       1             None
21    27310809            4                       1             None
22    2090911X           76                       1             None
23    14712407           24                       1             None
24    20586124           10                       1             None

   prism:coverDate prism:coverDisplayDate                   prism:doi
\
0       2024-12-01          December 2024  10.1186/s13098-024-01366-0

1       2024-12-01          December 2024  10.1186/s13643-024-02569-2

2       2024-12-01          December 2024  10.1007/s00383-024-05728-7

3       2024-12-01          December 2024  10.1186/s12985-024-02392-0

4       2024-12-01          December 2024  10.1186/s12936-024-04992-1

5       2024-12-01          December 2024  10.1007/s11701-024-01991-x

6       2024-12-01          December 2024  10.1186/s12884-024-06590-0

7       2024-12-01          December 2024  10.1186/s12959-024-00612-9

8       2024-12-01          December 2024  10.1186/s13098-024-01352-6

9       2024-12-01          December 2024  10.1007/s00238-024-02190-5

10      2024-12-01          December 2024  10.1186/s12885-024-12345-z

11      2024-12-01          December 2024  10.1186/s13643-024-02525-0

12      2024-12-01          December 2024  10.1186/s12884-024-06543-7

13      2024-12-01          December 2024  10.1007/s10103-024-04069-0
```

```
14      2024-12-01          December 2024  10.1007/s00238-024-02197-y
15      2024-12-01          December 2024  10.1007/s10103-024-04068-1
16      2024-12-01          December 2024  10.1007/s00240-024-01557-4
17      2024-12-01          December 2024  10.1186/s12966-024-01593-8
18      2024-12-01          December 2024  10.1186/s13643-024-02522-3
19      2024-12-01          December 2024  10.1007/s10103-024-04049-4
20      2024-12-01          December 2024  10.1007/s00240-024-01568-1
21      2024-12-01          December 2024  10.1007/s44163-024-00105-8
22      2024-12-01          December 2024  10.1186/s43044-024-00480-8
23      2024-12-01          December 2024  10.1186/s12885-024-12219-4
24      2024-12-01          December 2024  10.1038/s41394-024-00636-3


   citedby-count  \
0              0
1              0
2              0
3              0
4              0
5              0
6              0
7              0
8              0
9              0
10             0
11             0
12             0
13             0
14             0
15             0
16             0
17             0
18             0
19             0
20             0
21             0
22             0
23             0
24             0
```

```
affiliation  \
0
[{'@_fa': 'true', 'affilname': 'Tehran University of Medical
Sciences', 'affiliation-city': 'Tehran', 'affiliation-country':
'Iran'}]
1
[{'@_fa': 'true', 'affilname': 'University of Stirling', 'affiliation-
city': 'Stirling', 'affiliation-country': 'United Kingdom'}]
2
[{'@_fa': 'true', 'affilname': 'All India Institute of Medical
Sciences, New Delhi', 'affiliation-city': 'New Delhi', 'affiliation-
country': 'India'}]
3
[{'@_fa': 'true', 'affilname': 'Faculty of Pharmacy', 'affiliation-
city': 'Helwan', 'affiliation-country': 'Egypt'}]
4   [{'@_fa': 'true', 'affilname': 'National Institute of Malaria
Research India', 'affiliation-city': 'New Delhi', 'affiliation-
country': 'India'}, {'@_fa': 'true', 'affilname': 'Kumaun University
India', 'affiliation-city': 'Nainital', 'affiliation-country':
'India'}]
5
[{'@_fa': 'true', 'affilname': 'Slaski Uniwersytet Medyczny w
Katowicach', 'affiliation-city': 'Katowice', 'affiliation-country':
'Poland'}]
6
[{'@_fa': 'true', 'affilname': 'Kurdistan University of Medical
Sciences', 'affiliation-city': 'Sanandaj', 'affiliation-country':
'Iran'}]
7
[{'@_fa': 'true', 'affilname': 'Tehran University of Medical
Sciences', 'affiliation-city': 'Tehran', 'affiliation-country':
'Iran'}]
8
[{'@_fa': 'true', 'affilname': 'Jinan Maternal and Child Health
Hospital', 'affiliation-city': 'Jinan', 'affiliation-country':
'China'}]
9
[{'@_fa': 'true', 'affilname': 'Keck School of Medicine of USC',
'affiliation-city': 'Los Angeles', 'affiliation-country': 'United
States'}]
10
[{'@_fa': 'true', 'affilname': 'SBUMS Neuroscience Research Center',
'affiliation-city': 'Tehran', 'affiliation-country': 'Iran'}]
11
[{'@_fa': 'true', 'affilname': 'Cardiovascular Research Center,
Kermanshah University of Medical Sciences', 'affiliation-city':
'Kermanshah', 'affiliation-country': 'Iran'}]
12
```

```
[{'@_fa': 'true', 'affilname': 'Wollo University', 'affiliation-city':
'Dessie', 'affiliation-country': 'Ethiopia'}]
13         [{'@_fa': 'true', 'affilname': 'Universidade Federal de São
Carlos', 'affiliation-city': 'Sao Carlos', 'affiliation-country':
'Brazil'}, {'@_fa': 'true', 'affilname': 'Universidad Andrés Bello',
'affiliation-city': 'Santiago', 'affiliation-country': 'Chile'}]
14
[{'@_fa': 'true', 'affilname': 'Medical University of Warsaw',
'affiliation-city': 'Warsaw', 'affiliation-country': 'Poland'}]
15
[{'@_fa': 'true', 'affilname': 'Universidade Brasil', 'affiliation-
city': 'Sao Paulo', 'affiliation-country': 'Brazil'}]
16
[{'@_fa': 'true', 'affilname': 'Chongqing Medical University',
'affiliation-city': 'Chongqing', 'affiliation-country': 'China'}]
17
[{'@_fa': 'true', 'affilname': 'Nutrition Research Center (SUMS)',
'affiliation-city': 'Shiraz', 'affiliation-country': 'Iran'}]
18                 [{'@_fa': 'true', 'affilname': 'School of Medicine,
University of Split', 'affiliation-city': 'Split', 'affiliation-
country': 'Croatia'}, {'@_fa': 'true', 'affilname': 'Cochrane
Croatia', 'affiliation-city': 'Split', 'affiliation-country':
'Croatia'}]
19
[{'@_fa': 'true', 'affilname': 'Mansoura University', 'affiliation-
city': 'Mansoura', 'affiliation-country': 'Egypt'}]
20
[{'@_fa': 'true', 'affilname': 'An-Najah National University',
'affiliation-city': 'Nablus', 'affiliation-country': 'Palestine'}]
21
[{'@_fa': 'true', 'affilname': 'Universidade Federal de Santa
Catarina', 'affiliation-city': 'Florianopolis', 'affiliation-country':
'Brazil'}]
22
[{'@_fa': 'true', 'affilname': 'RGKar Medical College', 'affiliation-
city': 'Saltlake CityKolkata', 'affiliation-country': 'India'}]
23
[{'@_fa': 'true', 'affilname': 'Universidad Ricardo Palma',
'affiliation-city': 'Lima', 'affiliation-country': 'Peru'}]
24   [{'@_fa': 'true', 'affilname': 'Loghman Hakim Educational
Hospital', 'affiliation-city': 'Tehran', 'affiliation-country':
'Iran'}, {'@_fa': 'true', 'affilname': 'Sina Trauma and Surgery
Research Center', 'affiliation-city': 'Tehran', 'affiliation-country':
'Iran'}]

   prism:aggregationType subtype subtypeDescription article-number  \
0                Journal      re             Review            124
1                Journal      ar            Article            152
2                Journal      re             Review            152
```

```
3                Journal      ar            Article            130
4                Journal      ar            Article            172
5                Journal      re             Review            235
6                Journal      ar            Article            398
7                Journal      re             Review             44
8                Journal      ar            Article            116
9                Journal      re             Review             59
10               Journal      re             Review            612
11               Journal      re             Review            136
12               Journal      ar            Article            354
13               Journal      re             Review            124
14               Journal      re             Review             49
15               Journal      re             Review            120
16               Journal      re             Review             70
17               Journal      re             Review             45
18               Journal      ar            Article            110
19               Journal      re             Review            107
20               Journal      ar            Article             67
21               Journal      ar            Article              4
22               Journal      re             Review             48
23               Journal      ar            Article            477
24               Journal      ar            Article             21

      source-id openaccess  openaccessFlag
freetoread.value  \
0   19700174930           1            True    all, publisherfullgold

1   21100237425           1            True    all, publisherfullgold

2         22163           0           False                       NaN

3        130035           1            True    all, publisherfullgold

4         22911           1            True    all, publisherfullgold

5    6400153172           1            True  all, publisherhybridgold

6         12550           1            True    all, publisherfullgold

7         20645           1            True    all, publisherfullgold

8   19700174930           1            True    all, publisherfullgold

9         21128           0           False                       NaN

10        28747           1            True    all, publisherfullgold

11  21100237425           1            True    all, publisherfullgold

12        12550           1            True    all, publisherfullgold
```

```
13        17065           0           False                          NaN
14        21128           0           False                          NaN
15        17065           0           False                          NaN
16  21100223310           0           False                          NaN
17        14875           1            True    all, publisherfullgold
18  21100237425           1            True    all, publisherfullgold
19        17065           1            True  all, publisherhybridgold
20  21100223310           0           False                          NaN
21  21101215114           1            True    all, publisherfullgold
22  21100199758           1            True    all, publisherfullgold
23        28747           1            True    all, publisherfullgold
24  21100904572           0           False                          NaN


                                freetoreadLabel.value pubmed-id
prism:issn
0          [{'$': 'All Open Access'}, {'$': 'Gold'}]       NaN
NaN
1          [{'$': 'All Open Access'}, {'$': 'Gold'}]  38849924
NaN
2                                                NaN  38847871
01790358
3          [{'$': 'All Open Access'}, {'$': 'Gold'}]  38840177
NaN
4          [{'$': 'All Open Access'}, {'$': 'Gold'}]  38825698
NaN
5   [{'$': 'All Open Access'}, {'$': 'Hybrid Gold'}]  38819533
18632483
6          [{'$': 'All Open Access'}, {'$': 'Gold'}]  38816754
NaN
7          [{'$': 'All Open Access'}, {'$': 'Gold'}]       NaN
NaN
8          [{'$': 'All Open Access'}, {'$': 'Gold'}]       NaN
NaN
9                                                NaN       NaN
0930343X
10         [{'$': 'All Open Access'}, {'$': 'Gold'}]  38773447
NaN
11         [{'$': 'All Open Access'}, {'$': 'Gold'}]  38769586
```

```
NaN
12         [{'$': 'All Open Access'}, {'$': 'Gold'}]  38741046
NaN
13                                                NaN  38709332
02688921
14                                                NaN       NaN
0930343X
15                                                NaN  38695965
02688921
16                                                NaN  38662047
21947228
17         [{'$': 'All Open Access'}, {'$': 'Gold'}]  38659024
NaN
18         [{'$': 'All Open Access'}, {'$': 'Gold'}]  38641831
NaN
19  [{'$': 'All Open Access'}, {'$': 'Hybrid Gold'}]  38635085
02688921
20                                                NaN  38630266
21947228
21         [{'$': 'All Open Access'}, {'$': 'Gold'}]       NaN
NaN
22         [{'$': 'All Open Access'}, {'$': 'Gold'}]       NaN
11102608
23         [{'$': 'All Open Access'}, {'$': 'Gold'}]  38622563
NaN
24                                                NaN  38615029
NaN
Les données ont été nettoyées et sauvegardées dans le fichier
api_scopus_data.csv
```

Récupération et parsing des données de DOI depuis l'API Elsevier

```python
import requests
import pandas as pd
import xml.etree.ElementTree as ET

# Liste de DOI à récupérer depuis l'API Elsevier
dois = [
    '10.1016/j.cplett.2020.137481',
    '10.1016/j.joule.2020.11.010',
    '10.1016/j.jacc.2020.11.012'
]

# Clé API Elsevier
api_key = '9aebde1fa88b0b7325c7d8054dd3e754'

# Fonction pour récupérer et parser les données d'un DOI spécifique
depuis l'API Elsevier
```

```
def get_data_from_doi(doi):
    url = f'https://api.elsevier.com/content/article/doi/{doi}'
    headers = {'X-ELS-APIKey': api_key, 'Accept': 'application/xml'}

    try:
        response = requests.get(url, headers=headers)
        response.raise_for_status()

        print(f'Statut pour DOI {doi}: {response.status_code}')

        root = ET.fromstring(response.content)
        return parse_xml(root)

    except requests.exceptions.HTTPError as err:
        print(f'Erreur HTTP lors de la récupération du DOI {doi}:
{err}')
        return None

    except ET.ParseError as e:
        print(f'Erreur de parsing XML pour le DOI {doi}: {e}')
        return None

    except Exception as err:
        print(f'Erreur lors de la récupération du DOI {doi}: {err}')
        return None

# Fonction pour parser les données XML et extraire les informations
pertinentes
def parse_xml(root):
    namespaces = {
        'dtd': 'http://www.elsevier.com/xml/svapi/article/dtd',
        'dc': 'http://purl.org/dc/elements/1.1/',
        'prism': 'http://prismstandard.org/namespaces/basic/2.0/',
        'xocs': 'http://www.elsevier.com/xml/xocs/dtd'
    }

    data = {
        'doi': root.findtext('.//xocs:doi', namespaces=namespaces),
        'title': root.findtext('.//dc:title', namespaces=namespaces),
        'creator': root.findtext('.//dc:creator',
namespaces=namespaces),
        'publicationName': root.findtext('.//prism:publicationName',
namespaces=namespaces),
        'volume': root.findtext('.//prism:volume',
namespaces=namespaces),
        'issue': root.findtext('.//prism:issueIdentifier',
namespaces=namespaces),
        'pageRange': root.findtext('.//prism:pageRange',
namespaces=namespaces),
```

```
        'coverDate': root.findtext('.//prism:coverDate',
namespaces=namespaces),
        'citedby_count': root.findtext('.//xocs:citedby-count',
namespaces=namespaces)
    }

    link_elements = root.findall('.//xocs:link',
namespaces=namespaces)
    for link in link_elements:
        if link.get('rel') == 'scidir':
            data['doiLink'] = link.get('href')
            break

    return data

# Liste pour stocker les données récupérées depuis les DOI
data_list = []

try:
    # Parcourir la liste des DOI et récupérer les données pour chaque
DOI
    for doi in dois:
        data = get_data_from_doi(doi)
        if data:
            data_list.append(data)

    if data_list:
        # Créer un DataFrame à partir des données récupérées
        df_xml = pd.DataFrame(data_list)

        # Convertir le DataFrame en JSON
        df_xml_json = df_xml.to_json(orient='records')

        # Afficher le JSON
        print("\nDataFrame JSON à partir des DOI de l'API Elsevier:")
        print(df_xml_json)

        # Sauvegarder le JSON dans un fichier
        with open('scopus_data_from_dois.json', 'w', encoding='utf-8')
as f:
            f.write(df_xml_json)
    else:
        print("Aucune donnée valide n'a été récupérée depuis les
DOI.")

except requests.exceptions.HTTPError as err:
    print(f'Erreur HTTP lors de la requête à l\'API Elsevier: {err}')
except Exception as err:
    print(f'Erreur: {err}')
```

```
Statut pour DOI 10.1016/j.cplett.2020.137481: 200
Statut pour DOI 10.1016/j.joule.2020.11.010: 200
Statut pour DOI 10.1016/j.jacc.2020.11.012: 200

DataFrame JSON à partir des DOI de l'API Elsevier:
[{"doi":"10.1016\/j.cplett.2020.137481","title":"Effect of Ni2+ ions
concentration on the local crystal field of Zn1-\n               \n
x\n               Ni\n                  x\n               Te
nanocrystals ","creator":"Silva, Alessandra
S.","publicationName":"Chemical Physics
Letters","volume":"750","issue":null,"pageRange":"137481","coverDate":
"2020-07-31","citedby_count":null},{"doi":"10.1016\/
j.joule.2020.11.010","title":"Design and Manufacture of 3D-Printed
Batteries ","creator":"Lyu,
Zhiyang","publicationName":"Joule","volume":"5","issue":"1","pageRange
":"89-114","coverDate":"2021-01-20","citedby_count":null},
{"doi":"10.1016\/j.jacc.2020.11.012","title":"2021 ACC\/AHA Key Data
Elements and Definitions for Heart Failure A Report of the American
College of Cardiology\/American Heart Association Task Force on
Clinical Data Standards (Writing Committee to Develop Clinical Data
Standards for Heart Failure)","creator":"Bozkurt,
Biykem","publicationName":"Journal of the American College of
Cardiology","volume":"77","issue":"16","pageRange":"2053-
2150","coverDate":"2021-04-27","citedby_count":null}]
```

Affichage des colonnes du DataFrame

```
df.columns
```

```
Index(['@_fa', 'link', 'prism:url', 'dc:identifier', 'eid',
'dc:title',
       'dc:creator', 'prism:publicationName', 'prism:eIssn',
'prism:volume',
       'prism:issueIdentifier', 'prism:pageRange', 'prism:coverDate',
       'prism:coverDisplayDate', 'prism:doi', 'citedby-count',
'affiliation',
       'prism:aggregationType', 'subtype', 'subtypeDescription',
       'article-number', 'source-id', 'openaccess', 'openaccessFlag',
       'freetoread.value', 'freetoreadLabel.value', 'prism:issn',
'pubmed-id',
       'coverYear'],
      dtype='object')
```

Affichage des premières lignes du DataFrame

```
df.head()
```

```
   @_fa                                                  link  \
0  True  [{'@_fa': 'true', '@ref': 'self', '@href': 'ht...
```

```
1  True  [{'@_fa': 'true', '@ref': 'self', '@href': 'ht...
2  True  [{'@_fa': 'true', '@ref': 'self', '@href': 'ht...
3  True  [{'@_fa': 'true', '@ref': 'self', '@href': 'ht...
4  True  [{'@_fa': 'true', '@ref': 'self', '@href': 'ht...

                                              prism:url
dc:identifier  \
0  https://api.elsevier.com/content/abstract/scop...
SCOPUS_ID:85196156602
1  https://api.elsevier.com/content/abstract/scop...
SCOPUS_ID:85196115487
2  https://api.elsevier.com/content/abstract/scop...
SCOPUS_ID:85196086633
3  https://api.elsevier.com/content/abstract/scop...
SCOPUS_ID:85195598662
4  https://api.elsevier.com/content/abstract/scop...
SCOPUS_ID:85195533502

                   eid
dc:title  \
0  2-s2.0-85196156602  Efficacy and safety of omega-3 fatty acids
sup...
1  2-s2.0-85196115487  A systematic review on the efficacy of
adjunct...
2  2-s2.0-85196086633  Influence of elastomeric and steel ligatures
o...
3  2-s2.0-85195598662  Effect of zinc supplementation on glycemic
bio...
4  2-s2.0-85195533502  Barriers and facilitators to implementing
work...

     dc:creator                prism:publicationName prism:eIssn
prism:volume  \
0     Bafkar N.                        BMC Psychiatry    1471244X
24
1    Montano N.                 Neurosurgical Review    14372320
47
2    Hussain U.             Progress in Orthodontics    21961042
25
3  Daneshvar M.  Diabetology and Metabolic Syndrome    17585996
16
4   Paterson C.                   Systematic Reviews    20464053
13

   ...  subtypeDescription  article-number    source-id openaccess  \
0  ...             Article             455        14260          1
1  ...              Review             276        22097          0
2  ...              Review              24        91796          1
3  ...              Review             124  19700174930          1
4  ...             Article             152  21100237425          1
```

```
  openaccessFlag         freetoread.value  \
0           True  all, publisherfullgold
1          False                     NaN
2           True  all, publisherfullgold
3           True  all, publisherfullgold
4           True  all, publisherfullgold

                         freetoreadLabel.value prism:issn   pubmed-id
coverYear
0  [{'$': 'All Open Access'}, {'$': 'Gold'}]        NaN         NaN
2024
1                                          NaN   03445607  38884812.0
2024
2  [{'$': 'All Open Access'}, {'$': 'Gold'}]   17237785  38880839.0
2024
3  [{'$': 'All Open Access'}, {'$': 'Gold'}]        NaN         NaN
2024
4  [{'$': 'All Open Access'}, {'$': 'Gold'}]        NaN  38849924.0
2024

[5 rows x 29 columns]
```

Informations sur le DataFrame

```
df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 25 entries, 0 to 24
Data columns (total 29 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   @_fa                    25 non-null     bool
 1   link                    25 non-null     object
 2   prism:url               25 non-null     object
 3   dc:identifier           25 non-null     object
 4   eid                     25 non-null     object
 5   dc:title                25 non-null     object
 6   dc:creator              25 non-null     object
 7   prism:publicationName   25 non-null     object
 8   prism:eIssn             25 non-null     object
 9   prism:volume            25 non-null     int64
 10  prism:issueIdentifier   25 non-null     int64
 11  prism:pageRange         0 non-null      float64
 12  prism:coverDate         25 non-null     object
 13  prism:coverDisplayDate  25 non-null     object
 14  prism:doi               25 non-null     object
 15  citedby-count           25 non-null     int32
 16  affiliation             25 non-null     object
 17  prism:aggregationType   25 non-null     object
```

```
 18  subtype                  25 non-null     object
 19  subtypeDescription       25 non-null     object
 20  article-number           25 non-null     int64
 21  source-id                25 non-null     int64
 22  openaccess               25 non-null     int64
 23  openaccessFlag           25 non-null     bool
 24  freetoread.value         17 non-null     object
 25  freetoreadLabel.value    17 non-null     object
 26  prism:issn               11 non-null     object
 27  pubmed-id                18 non-null     float64
 28  coverYear                25 non-null     int32
dtypes: bool(2), float64(2), int32(2), int64(5), object(18)
memory usage: 5.3+ KB
```

Analyse des données de citations par année

```
import pandas as pd
import matplotlib.pyplot as plt

def analyze_data(df):
    """Analyze the data and provide insights."""
    if df is not None:
        # Calculer le nombre total de citations
        if 'citedby-count' in df.columns:
            df['citedby-count'] = pd.to_numeric(df['citedby-count'],
errors='coerce').fillna(0).astype(int)
            total_citations = df['citedby-count'].sum()
            print(f"\nNombre total de citations pour toutes les
publications: {total_citations}")

        # Répartition des citations par année de publication
        if 'prism:coverDate' in df.columns:
            df['coverYear'] = pd.to_datetime(df['prism:coverDate'],
errors='coerce').dt.year
            citations_per_year = df.groupby('coverYear')['citedby-
count'].sum()

            # Affichage des citations par année de publication
détaillé
            print("\nCitations par année de publication :")
            for year, citations in citations_per_year.items():
                print(f"Année {year} : {citations} citations")

# Charger le DataFrame sauvegardé depuis le fichier CSV
df = pd.read_csv('api_scopus_data.csv')

# Analyser les données
analyze_data(df)
```

```
Nombre total de citations pour toutes les publications: 17

Citations par année de publication :
Année 2024 : 17 citations
Année 2025 : 0 citations
```

Calcul du nombre total de publications dans le dataset

```python
def total_publications(df):
    """Calculates the total number of publications."""
    if df is not None:
        total = len(df)
        print(f"\nNombre total de publications dans le dataset :
{total}")
    else:
        print("Aucune donnée à analyser.")

# Appel de la fonction
total_publications(df)
```

```
Nombre total de publications dans le dataset : 25
```

Calcul du nombre total de citations pour toutes les publications

```python
def total_citations(df):
    """Calculates the total number of citations."""
    if 'citedby-count' in df.columns:
        total_citations = df['citedby-count'].sum()
        print(f"\nNombre total de citations pour toutes les
publications : {total_citations}")
    else:
        print("La colonne 'citedby-count' n'est pas présente dans le
DataFrame.")

# Appel de la fonction
total_citations(df)
```

```
Nombre total de citations pour toutes les publications : 17
```

Calcul de la moyenne des citations par publication

```python
def average_citations(df):
    """Calculates the average citations per publication."""
    if 'citedby-count' in df.columns:
        average_citations = df['citedby-count'].mean()
```

```
        print(f"\nCitations moyennes par publication :
{average_citations:.2f}")
    else:
        print("La colonne 'citedby-count' n'est pas présente dans le
DataFrame.")

# Appel de la fonction
average_citations(df)
```

```
Citations moyennes par publication : 0.68
```

Identification des publications avec le plus de citations

```
def publications_most_citations(df, top_n=5):
    """Identifie les publications avec le plus de citations."""
    if 'citedby-count' in df.columns:
        top_publications = df.nlargest(top_n, 'citedby-count')
[['dc:title', 'citedby-count']]
        print(f"\nPublications avec le plus de citations (Top {top_n})
:")
        print(top_publications)
    else:
        print("La colonne 'citedby-count' n'est pas présente dans le
DataFrame.")

# Charger le DataFrame sauvegardé depuis le fichier CSV
df = pd.read_csv('api_scopus_data.csv')

# Appel de la fonction
publications_most_citations(df)
```

```
Publications avec le plus de citations (Top 5) :

dc:title  \
16
Beyond playing 20 questions with nature: Integrative experiment design
in the social and behavioral sciences
20
A practical guide to adopting Bayesian analyses in clinical research

0
Influence of phytoplankton, bacteria and viruses on nutrient supply in
tropical waters
1   Safety and efficacy of low-power pure-cut hot snare polypectomy
for small nonpedunculated colorectal polyps compared with conventional
resection methods: A propensity score matching analysis
2
Relationship between Self-regulated Learning with Academic Buoyancy: A
```

```
Case Study among Malaysia FELDA Secondary School Students

    citedby-count
16             16
20              1
0               0
1               0
2               0
```

Calcul de la corrélation entre le nombre de citations et les années de publication

```
def correlation_citations_annees(df):
    """Calcule la corrélation entre le nombre de citations et les
années de publication."""
    if 'citedby-count' in df.columns and ('prism:coverDate' in
df.columns or 'prism:coverDisplayDate' in df.columns):
        if 'prism:coverDate' in df.columns:
            df['coverYear'] = pd.to_datetime(df['prism:coverDate'],
errors='coerce').dt.year
        elif 'prism:coverDisplayDate' in df.columns:
            df['coverYear'] =
pd.to_datetime(df['prism:coverDisplayDate'], errors='coerce').dt.year

        correlation = df[['coverYear', 'citedby-
count']].corr().iloc[0, 1]
        print(f"\nCorrélation entre le nombre de citations et les
années de publication : {correlation:.2f}")
    else:
        print("Les colonnes nécessaires ('citedby-count' et
'prism:coverDate' ou 'prism:coverDisplayDate') ne sont pas présentes
dans le DataFrame.")

# Appel de la fonction pour calculer la corrélation entre le nombre de
citations et les années de publication
correlation_citations_annees(df)
```

```
Corrélation entre le nombre de citations et les années de
publication : -0.24
```

Analyse de la répartition des publications en accès libre

```
def publications_acces_libre(df):
    """Analyse la répartition des publications en accès libre."""
    if 'openaccessFlag' in df.columns:
        publications_open_access = df['openaccessFlag'].value_counts()
        print("\nRépartition des publications par statut d'accès libre
:")
```

```
        print(publications_open_access)
    else:
        print("La colonne 'openaccessFlag' n'est pas présente dans le
DataFrame.")

# Appel de la fonction pour analyser la répartition des publications
en accès libre
publications_acces_libre(df)
```

```
Répartition des publications par statut d'accès libre :
openaccessFlag
True     19
False     6
Name: count, dtype: int64
```

Analyse de la répartition des publications par année

```
import pandas as pd
import matplotlib.pyplot as plt

def analyze_data(df):
    """Analyze the data and provide insights."""
    if df is not None:
        # Ajouter une colonne pour l'année de publication
        if 'prism:coverDate' in df.columns:
            df['coverYear'] = pd.to_datetime(df['prism:coverDate'],
errors='coerce').dt.year

        # Répartition des publications par année
        if 'coverYear' in df.columns:
            publications_per_year = df.groupby('coverYear')
['dc:title'].count().reset_index(name='Publications')
            print("\nRépartition des publications par année :")
            print(publications_per_year)

# Charger le DataFrame sauvegardé depuis le fichier CSV
df = pd.read_csv('api_scopus_data.csv')

# Analyser les données
analyze_data(df)
```

```
Répartition des publications par année :
   coverYear  Publications
0       2024            25
```

Analyse de la répartition des publications par source

```
def publications_par_source(df):
    """Analyse la répartition des publications par source."""
    if 'prism:publicationName' in df.columns:
        publications_by_source =
df['prism:publicationName'].value_counts().head(10)
        print("\nRépartition des publications par source (Top 10) :")
        print(publications_by_source)
    else:
        print("La colonne 'prism:publicationName' n'est pas présente
dans le DataFrame.")

# Appel de la fonction pour analyser la répartition des publications
par source
publications_par_source(df)
```

```
Répartition des publications par source (Top 10) :
prism:publicationName
Journal of Advanced Research in Applied Sciences and Engineering
Technology    3
Dados
3
International Journal of Religion and Spirituality in Society
2
Journal of Environmental Sciences (China)
1
Behavioral and Brain Sciences
1
Substance Abuse: Treatment, Prevention, and Policy
1
Sports Medicine - Open
1
Experimental Hematology and Oncology
1
Journal of Clinical and Translational Science
1
Chinese Journal of Tissue Engineering Research
1
Name: count, dtype: int64
```

Analyse de la présence des identifiants PubMed dans les publications

```
def publications_pubmed_id(df):
    """Analyse la présence des identifiants PubMed dans les
publications."""
    if 'pubmed-id' in df.columns:
```

```
        publications_with_pubmed_id = df['pubmed-id'].notna().sum()
        total_publications = len(df)
        print(f"\nNombre de publications avec identifiant PubMed : 
{publications_with_pubmed_id} sur {total_publications} publications au 
total.")
    else:
        print("La colonne 'pubmed-id' n'est pas présente dans le 
DataFrame.")

# Appel de la fonction pour analyser la présence des identifiants 
PubMed dans les publications
publications_pubmed_id(df)
```

```
Nombre de publications avec identifiant PubMed : 3 sur 25 publications 
au total.
```

Analyse de la répartition des publications par type de sous-catégorie

```
def publications_par_sous_categorie(df):
    """Analyse la répartition des publications par type de sous-
catégorie."""
    if 'subtypeDescription' in df.columns:
        publications_by_subtype = 
df['subtypeDescription'].value_counts().head(10)
        print("\nRépartition des publications par type de sous-
catégorie (Top 10) :")
        print(publications_by_subtype)
    else:
        print("La colonne 'subtypeDescription' n'est pas présente dans 
le DataFrame.")

# Appel de la fonction pour analyser la répartition des publications 
par type de sous-catégorie
publications_par_sous_categorie(df)
```

```
Répartition des publications par type de sous-catégorie (Top 10) :
subtypeDescription
Article    20
Review      4
Book        1
Name: count, dtype: int64
```

Analyse de la répartition des publications par type de source (aggregation type)

```
def publications_par_type_source(df):
    """Analyse la répartition des publications par type de source
(aggregation type)."""
    if 'prism:aggregationType' in df.columns:
        publications_by_aggregation_type =
df['prism:aggregationType'].value_counts()
        print("\nRépartition des publications par type de source
(Aggregation Type) :")
        print(publications_by_aggregation_type)
    else:
        print("La colonne 'prism:aggregationType' n'est pas présente
dans le DataFrame.")

# Appel de la fonction pour analyser la répartition des publications
par type de source
publications_par_type_source(df)
```

```
Répartition des publications par type de source (Aggregation Type) :
prism:aggregationType
Journal    24
Book        1
Name: count, dtype: int64
```

Analyse de la répartition des publications par ISSN

```
def publications_par_issn(df):
    """Analyse la répartition des publications par ISSN."""
    if 'prism:issn' in df.columns:
        publications_by_issn =
df['prism:issn'].value_counts().head(10)
        print("\nRépartition des publications par ISSN (Top 10) :")
        print(publications_by_issn)
    else:
        print("La colonne 'prism:issn' n'est pas présente dans le
DataFrame.")

# Appel de la fonction pour analyser la répartition des publications
par ISSN
publications_par_issn(df)
```

```
Répartition des publications par ISSN (Top 10) :
prism:issn
02688921    3
0930343X    2
```

```
21947228    2
01790358    1
18632483    1
11102608    1
Name: count, dtype: int64
```

Analyse de la répartition des publications par type de volume

```
def publications_par_volume(df):
    """Analyse la répartition des publications par type de volume."""
    if 'prism:volume' in df.columns:
        publications_by_volume =
df['prism:volume'].value_counts().head(10)
        print("\nRépartition des publications par type de volume (Top
10) :")
        print(publications_by_volume)
    else:
        print("La colonne 'prism:volume' n'est pas présente dans le
DataFrame.")

# Appel de la fonction pour analyser la répartition des publications
par type de volume
publications_par_volume(df)
```

```
Répartition des publications par type de volume (Top 10) :
prism:volume
68.0    3
19.0    2
43.0    2
27.0    1
10.0    1
13.0    1
8.0     1
28.0    1
77.0    1
14.0    1
Name: count, dtype: int64
```

Analyse de la répartition des publications par numéro d'article

```
def publications_par_article_number(df):
    """Analyse la répartition des publications par numéro
d'article."""
    if 'article-number' in df.columns:
        publications_by_article_number = df['article-
```

```
number'].value_counts().head(10)
        print("\nRépartition des publications par numéro d'article
(Top 10) :")
        print(publications_by_article_number)
    else:
        print("La colonne 'article-number' n'est pas présente dans le
DataFrame.")

# Appel de la fonction pour analyser la répartition des publications
par numéro d'article
publications_par_article_number(df)
```

```
Répartition des publications par numéro d'article (Top 10) :
article-number
e378          1
011005        1
e20220116     1
e20220091     1
e20220167     1
12            1
e42           1
e33           1
103587        1
e3            1
Name: count, dtype: int64
```

Analyse de la répartition des publications par type de sous-type

```
def publications_par_subtype(df):
    """Analyse la répartition des publications par type de sous-
type."""
    if 'subtype' in df.columns:
        publications_by_subtype =
df['subtype'].value_counts().head(10)
        print("\nRépartition des publications par type de sous-type
(Top 10) :")
        print(publications_by_subtype)
    else:
        print("La colonne 'subtype' n'est pas présente dans le
DataFrame.")

# Appel de la fonction pour analyser la répartition des publications
par type de sous-type
publications_par_subtype(df)
```

```
Répartition des publications par type de sous-type (Top 10) :
```

```
subtype
ar    20
re     4
bk     1
Name: count, dtype: int64
```

Analyse de la répartition des publications par type de description de sous-type

```
def publications_par_subtype_description(df):
    """Analyse la répartition des publications par type de description
de sous-type."""
    if 'subtypeDescription' in df.columns:
        publications_by_subtype_desc =
df['subtypeDescription'].value_counts().head(10)
        print("\nRépartition des publications par type de description
de sous-type (Top 10) :")
        print(publications_by_subtype_desc)
    else:
        print("La colonne 'subtypeDescription' n'est pas présente dans
le DataFrame.")

# Appel de la fonction pour analyser la répartition des publications
par type de description de sous-type
publications_par_subtype_description(df)
```

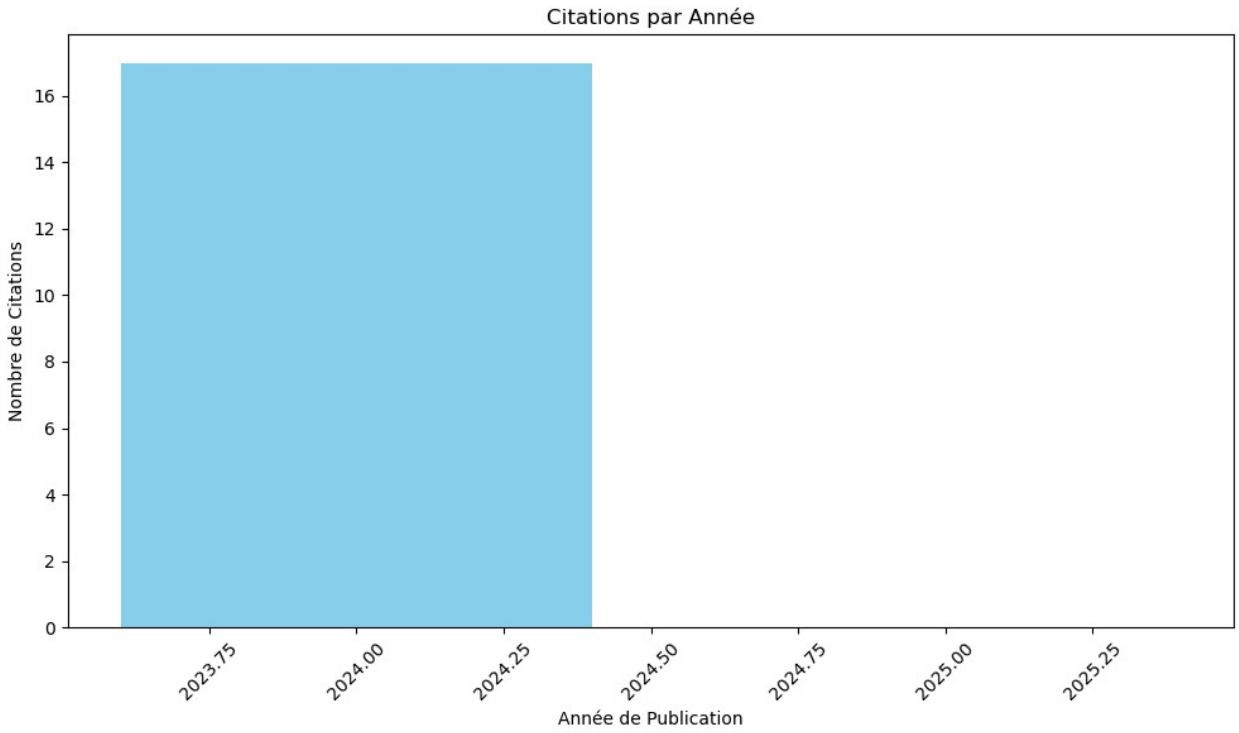
```
Répartition des publications par type de description de sous-type (Top
10) :
subtypeDescription
Review     14
Article    11
Name: count, dtype: int64
```

Analyse et visualisation des citations par année

```
import pandas as pd
import matplotlib.pyplot as plt

# Charger le DataFrame sauvegardé depuis le fichier CSV
df = pd.read_csv('api_scopus_data.csv')

def citations_per_year(df):
    """Analyse les citations par année."""
    if 'citedby-count' in df.columns and 'prism:coverDate' in
df.columns:
        # Convertir prism:coverDate en année
        df['coverYear'] = pd.to_datetime(df['prism:coverDate'],
errors='coerce').dt.year
```

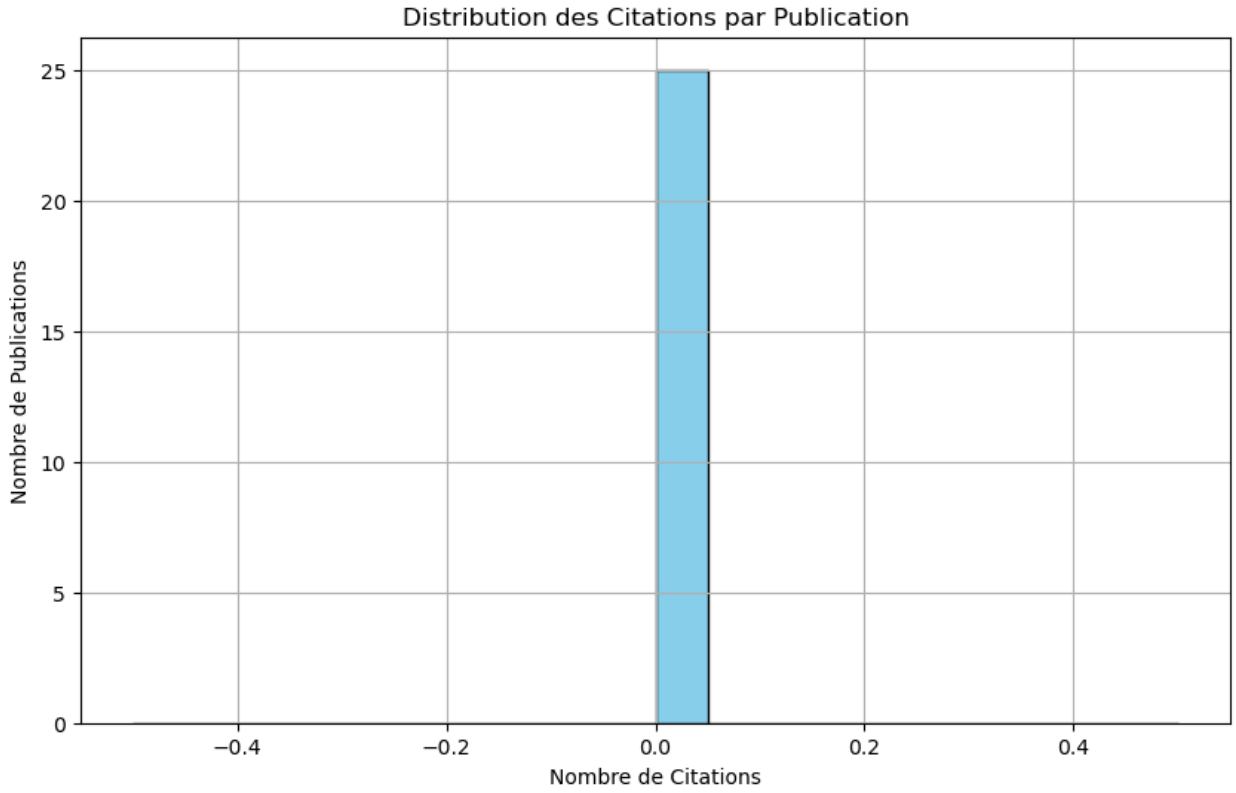
```
        citations_by_year = df.groupby('coverYear')['citedby-
count'].sum()
        print("\nCitations par année :")
        print(citations_by_year)
        return citations_by_year  # Retourne les citations par année
    else:
        print("Les colonnes nécessaires ('citedby-count' et
'prism:coverDate') ne sont pas présentes dans le DataFrame.")
        return None

# Appel de la fonction pour obtenir les citations par année
citations_by_year = citations_per_year(df)

# Vérification si les données ont été correctement chargées
if citations_by_year is not None:
    # Visualisation des citations par année
    plt.figure(figsize=(10, 6))
    plt.bar(citations_by_year.index, citations_by_year.values,
color='skyblue')
    plt.xlabel('Année de Publication')
    plt.ylabel('Nombre de Citations')
    plt.title('Citations par Année')
    plt.xticks(rotation=45)
    plt.tight_layout()
    plt.show()
else:
    print("Impossible de visualiser les données car les citations par
année n'ont pas été calculées correctement.")
```

```
Citations par année :
coverYear
2024    17
2025     0
Name: citedby-count, dtype: int64
```



Visualisation de la distribution des citations par publication

```
import matplotlib.pyplot as plt

def citations_distribution(df):
    """Visualizes the distribution of citations."""
    if 'citedby-count' in df.columns:
        plt.figure(figsize=(10, 6))
        plt.hist(df['citedby-count'], bins=20, color='skyblue',
edgecolor='black')
        plt.title('Distribution des Citations par Publication')
        plt.xlabel('Nombre de Citations')
        plt.ylabel('Nombre de Publications')
        plt.grid(True)
        plt.show()
    else:
        print("La colonne 'citedby-count' n'est pas présente dans le
DataFrame.")

# Appel de la fonction
citations_distribution(df)
```

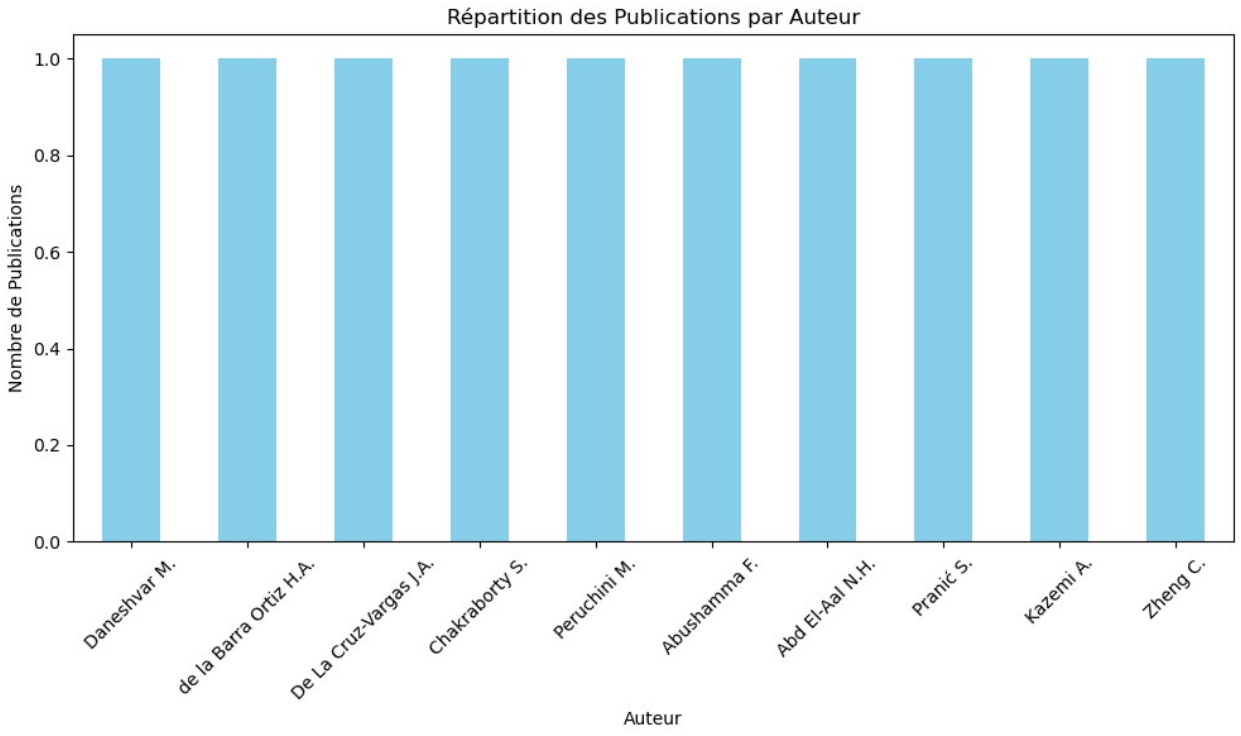


Analyse et visualisation de la répartition des publications par auteur

```
def publications_per_author(df):
    """Analyse la répartition des publications par auteur."""
    if 'dc:creator' in df.columns:
        publications_by_author = df['dc:creator'].value_counts()

        print("\nRépartition des publications par auteur :")
        print(publications_by_author.head(10))  # Afficher les 10
premiers auteurs par nombre de publications

        # Visualisation des publications par auteur (10 premiers
auteurs)
        plt.figure(figsize=(10, 6))
        publications_by_author.head(10).plot(kind='bar',
color='skyblue')
        plt.xlabel('Auteur')
        plt.ylabel('Nombre de Publications')
        plt.title('Répartition des Publications par Auteur')
        plt.xticks(rotation=45)
        plt.tight_layout()
        plt.show()
    else:
        print("La colonne 'dc:creator' n'est pas présente dans le
```

```
DataFrame.")

# Appel de la fonction pour analyser la répartition des publications
par auteur
publications_per_author(df)
```

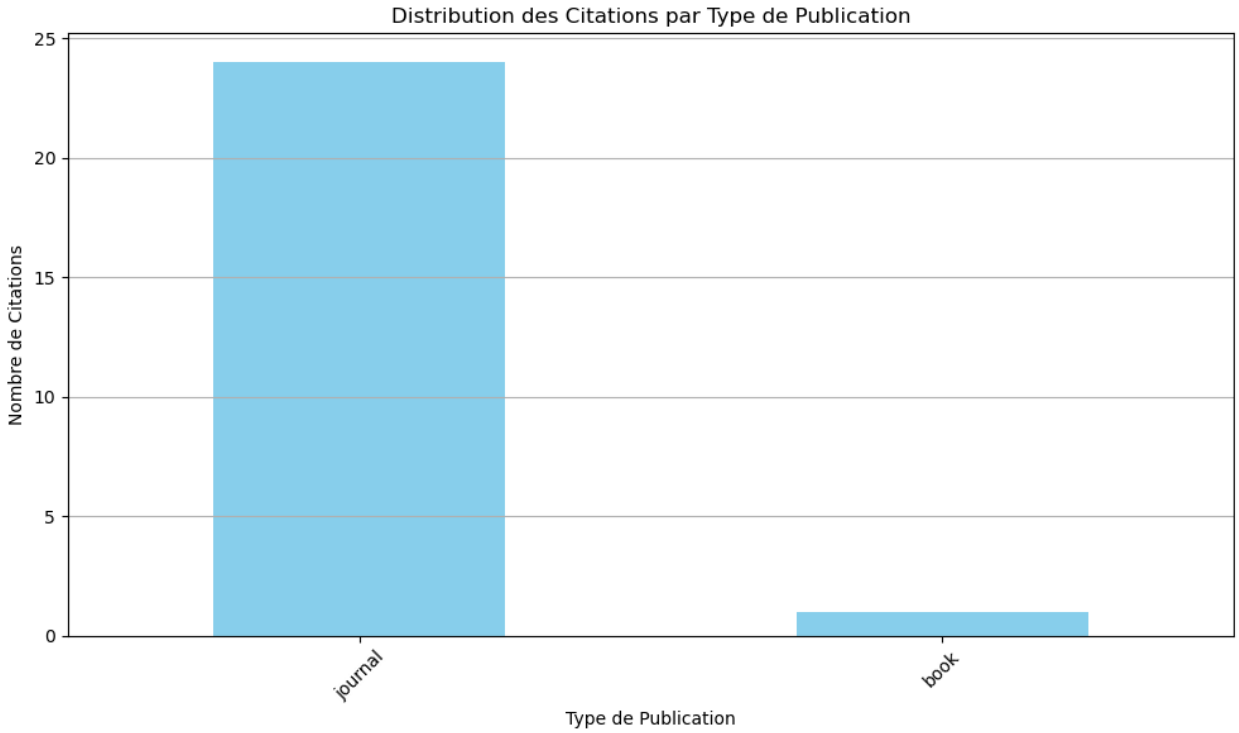
```
Répartition des publications par auteur :
dc:creator
Daneshvar M.                1
de la Barra Ortiz H.A.      1
De La Cruz-Vargas J.A.      1
Chakraborty S.              1
Peruchini M.                1
Abushamma F.                1
Abd El-Aal N.H.             1
Pranić S.                   1
Kazemi A.                   1
Zheng C.                    1
Name: count, dtype: int64
```



Visualisation de la distribution des citations par type de publication

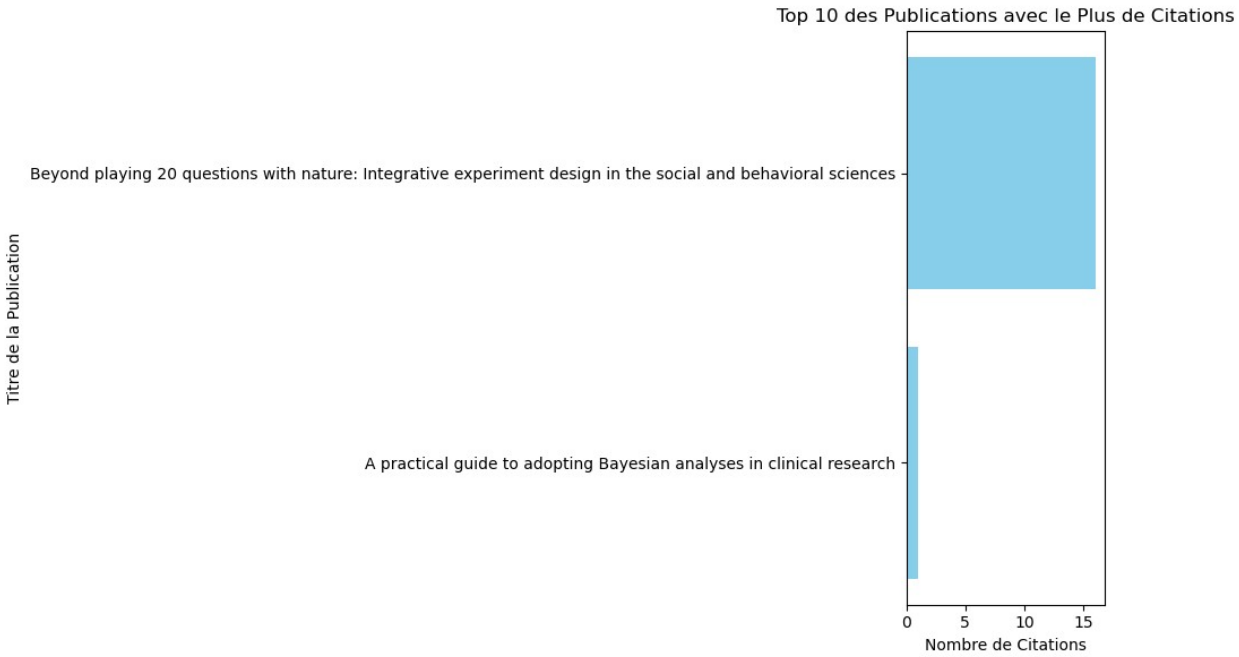
```
def distribution_citations_par_type(df):
    """Visualise la distribution des citations par type de
```

```
publication."""
    if 'citedby-count' in df.columns and 'prism:aggregationType' in
df.columns:
        plt.figure(figsize=(10, 6))
        df_filtered = df[df['prism:aggregationType'].notna()]
        df_filtered['prism:aggregationType'] =
df_filtered['prism:aggregationType'].str.lower()

df_filtered['prism:aggregationType'].value_counts().plot(kind='bar',
color='skyblue')
        plt.xlabel('Type de Publication')
        plt.ylabel('Nombre de Citations')
        plt.title('Distribution des Citations par Type de
Publication')
        plt.grid(axis='y')
        plt.xticks(rotation=45)
        plt.tight_layout()
        plt.show()
    else:
        print("Les colonnes nécessaires ('citedby-count' et
'prism:aggregationType') ne sont pas présentes dans le DataFrame.")

# Appel de la fonction pour visualiser la distribution des citations
par type de publication
distribution_citations_par_type(df)
```



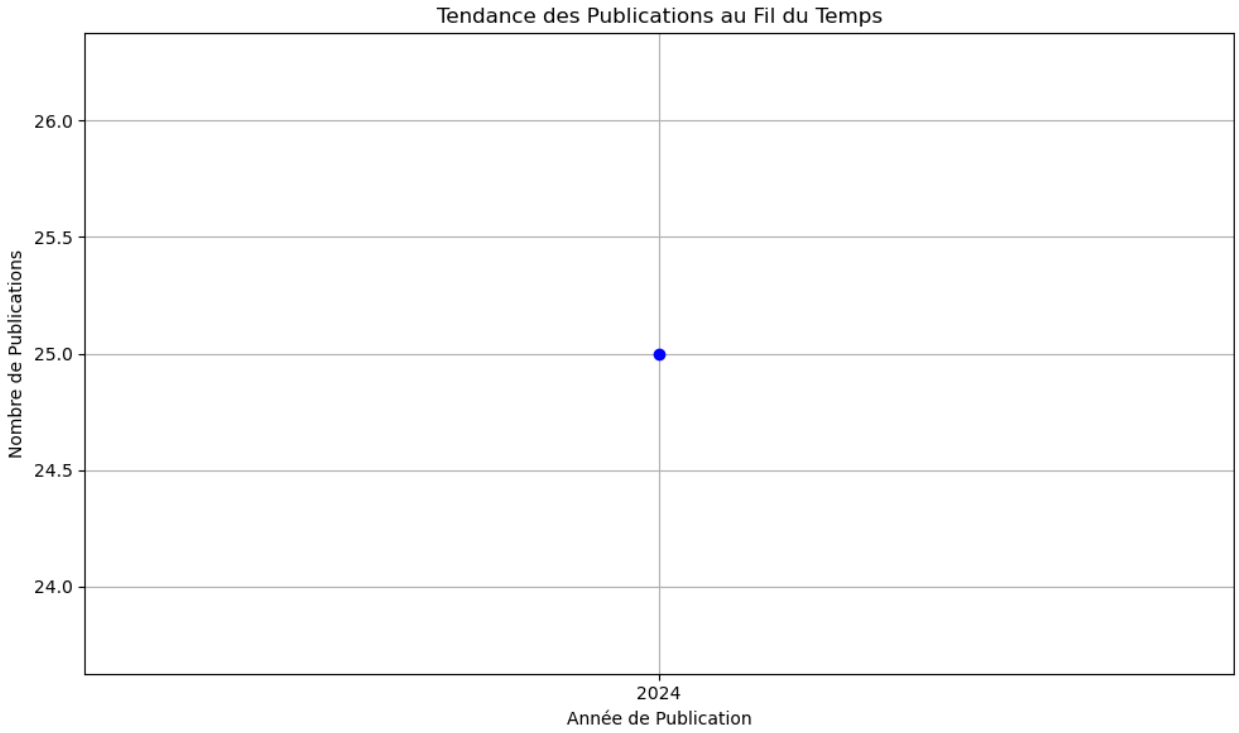
Visualisation du Nombre de Citations par Publication

```
import matplotlib.pyplot as plt

def plot_citations_per_publication(df):
    """Plot the number of citations per publication."""
    if 'citedby-count' in df.columns:
        # Filtrer les publications avec un nombre de citations non nul
        df_filtered = df[df['citedby-count'] > 0]

        # Tri des publications par nombre de citations (top 10)
        top_publications = df_filtered.nlargest(10, 'citedby-count')

        # Création du graphique à barres
        plt.figure(figsize=(10, 6))
        plt.barh(top_publications['dc:title'],
top_publications['citedby-count'], color='skyblue')
        plt.xlabel('Nombre de Citations')
        plt.ylabel('Titre de la Publication')
        plt.title('Top 10 des Publications avec le Plus de Citations')
        plt.gca().invert_yaxis()  # Inverser l'ordre des publications
pour afficher du plus grand au plus petit
        plt.tight_layout()
        plt.show()
    else:
        print("La colonne 'citedby-count' n'est pas présente dans le
DataFrame.")

# Appel de la fonction pour visualiser le nombre de citations par
publication
plot_citations_per_publication(df)
```



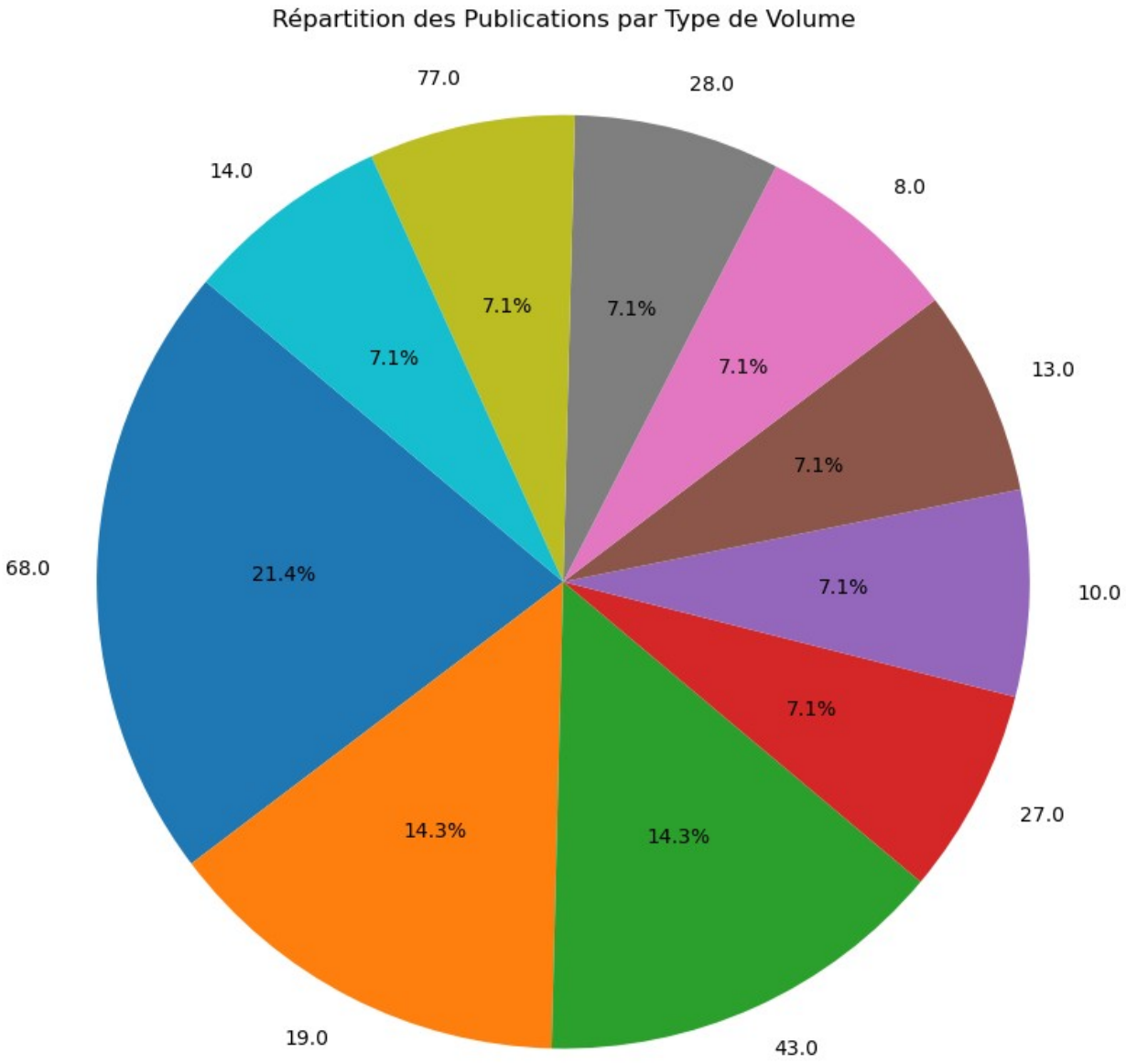
Tendance des Publications au Fil du Temps

```
def plot_publications_trend(df):
    """Plot the trend of publications over time."""
    if 'coverYear' in df.columns:
        # Compter le nombre de publications par année
        publications_per_year =
df['coverYear'].value_counts().sort_index()

        # Création du graphique linéaire
        plt.figure(figsize=(10, 6))
        plt.plot(publications_per_year.index,
publications_per_year.values, marker='o', linestyle='-', color='b')
        plt.xlabel('Année de Publication')
        plt.ylabel('Nombre de Publications')
        plt.title('Tendance des Publications au Fil du Temps')
        plt.grid(True)
        plt.xticks(publications_per_year.index)
        plt.tight_layout()
        plt.show()
    else:
        print("La colonne 'coverYear' n'est pas présente dans le
DataFrame.")

# Appel de la fonction pour visualiser la tendance des publications au
fil du temps
plot_publications_trend(df)
```



Répartition des Publications par Type de Volume

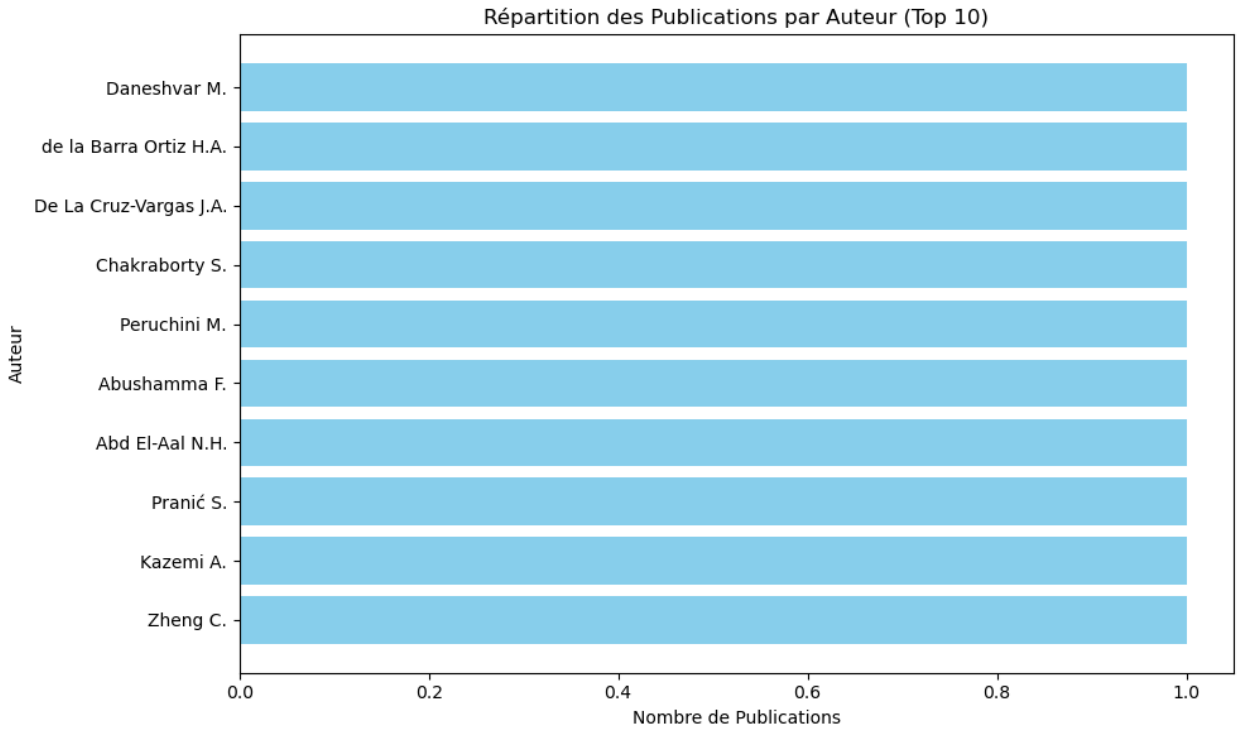
```
def plot_publications_by_volume(df):
    """Plot the distribution of publications by volume type."""
    if 'prism:volume' in df.columns:
        # Compter le nombre de publications par type de volume (top
10)
        publications_by_volume =
df['prism:volume'].value_counts().head(10)

        # Création du graphique à secteurs
        plt.figure(figsize=(8, 8))
        plt.pie(publications_by_volume.values,
labels=publications_by_volume.index, autopct='%1.1f%%',
startangle=140)
        plt.title('Répartition des Publications par Type de Volume')
        plt.axis('equal')
        plt.tight_layout()
        plt.show()
    else:
        print("La colonne 'prism:volume' n'est pas présente dans le
DataFrame.")

# Appel de la fonction pour visualiser la répartition des publications
par type de volume
plot_publications_by_volume(df)
```



Répartition des Publications par Auteur (Top 10)

```
def plot_publications_by_author(df):
    """Plot the distribution of publications by author."""
    if 'dc:creator' in df.columns:
        # Compter le nombre de publications par auteur (top 10)
        publications_by_author =
df['dc:creator'].value_counts().head(10)

        # Création du graphique à barres horizontales
        plt.figure(figsize=(10, 6))
        plt.barh(publications_by_author.index,
```

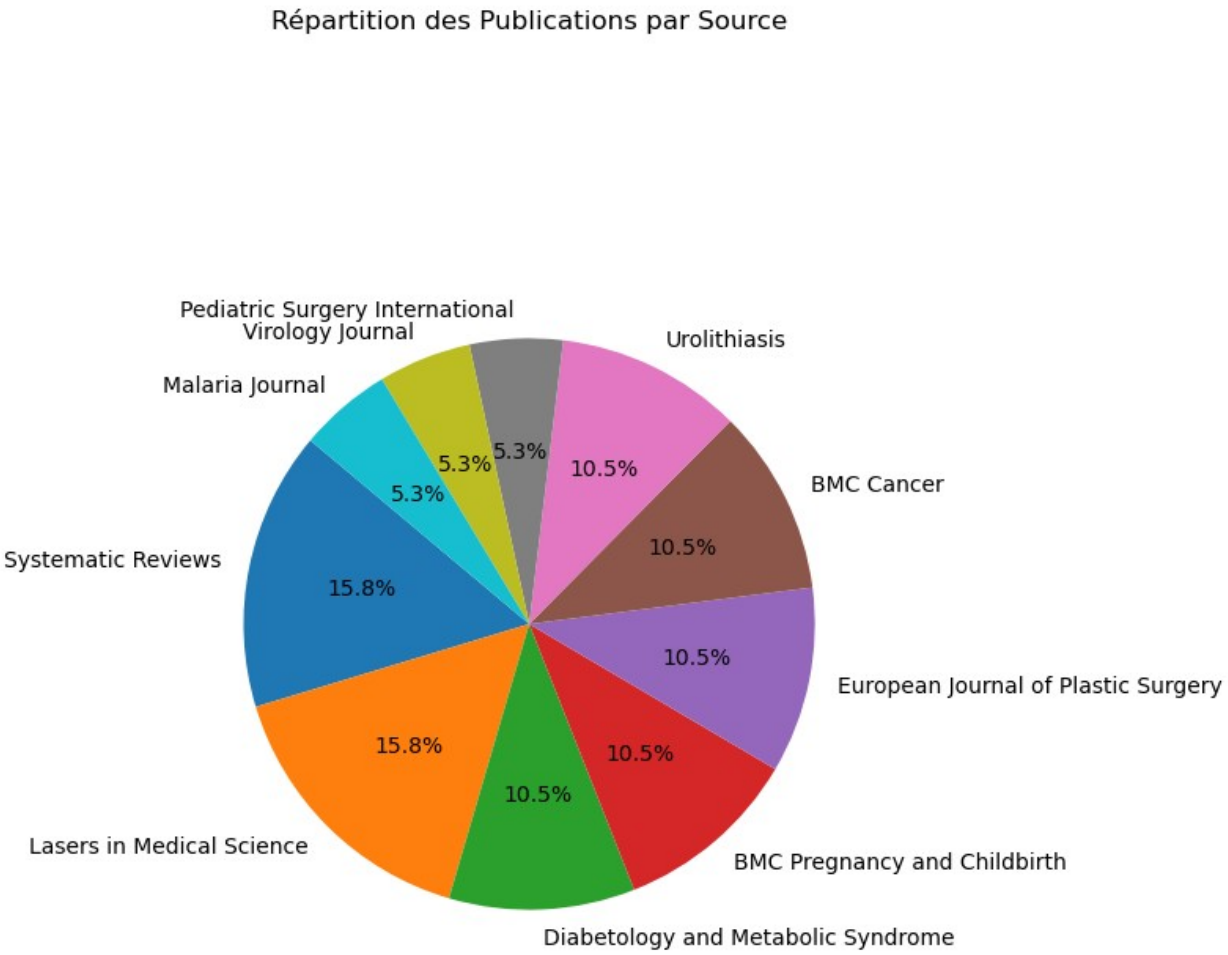
```
publications_by_author.values, color='skyblue')
        plt.xlabel('Nombre de Publications')
        plt.ylabel('Auteur')
        plt.title('Répartition des Publications par Auteur (Top 10)')
        plt.gca().invert_yaxis()  # Inverser l'ordre des auteurs pour
afficher du plus grand au plus petit
        plt.tight_layout()
        plt.show()
    else:
        print("La colonne 'dc:creator' n'est pas présente dans le
DataFrame.")

# Appel de la fonction pour visualiser la répartition des publications
par auteur
plot_publications_by_author(df)
```



Répartition des Publications par Source

```
def plot_publications_by_source(df):
    """Plot the distribution of publications by source."""
    if 'prism:publicationName' in df.columns:
        # Compter le nombre de publications par source (top 10)
        publications_by_source =
df['prism:publicationName'].value_counts().head(10)

        # Création du graphique à secteurs
        plt.figure(figsize=(8, 8))
```

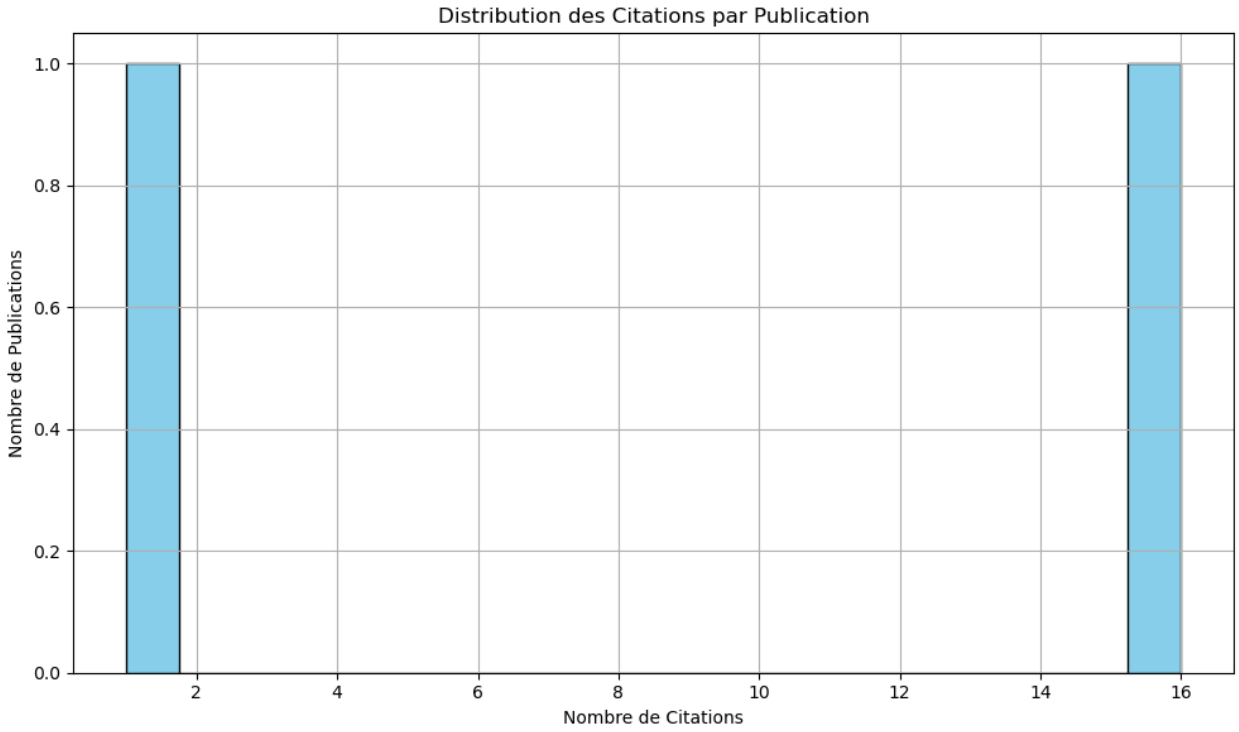
```
        plt.pie(publications_by_source.values,
labels=publications_by_source.index, autopct='%1.1f%%',
startangle=140)
        plt.title('Répartition des Publications par Source')
        plt.axis('equal')
        plt.tight_layout()
        plt.show()
    else:
        print("La colonne 'prism:publicationName' n'est pas présente
dans le DataFrame.")

# Appel de la fonction pour visualiser la répartition des publications
par source
plot_publications_by_source(df)
```



Distribution des Citations par Publication

```
def plot_citations_distribution(df):
    """Plot the distribution of citations per publication."""
    if 'citedby-count' in df.columns:
        # Filtrer les publications avec un nombre de citations non nul
        df_filtered = df[df['citedby-count'] > 0]

        # Création de l'histogramme des citations
        plt.figure(figsize=(10, 6))
        plt.hist(df_filtered['citedby-count'], bins=20,
color='skyblue', edgecolor='black')
        plt.xlabel('Nombre de Citations')
```

```
        plt.ylabel('Nombre de Publications')
        plt.title('Distribution des Citations par Publication')
        plt.grid(True)
        plt.tight_layout()
        plt.show()
    else:
        print("La colonne 'citedby-count' n'est pas présente dans le
DataFrame.")

# Appel de la fonction pour visualiser la distribution des citations
par publication
plot_citations_distribution(df)
```



Installation de la bibliothèque rdflib avec Python

```
 !pip install rdflib

Defaulting to user installation because normal site-packages is not
writeable
Requirement already satisfied: rdflib in c:\users\intel\appdata\
roaming\python\python311\site-packages (7.0.0)
Requirement already satisfied: isodate<0.7.0,>=0.6.0 in c:\users\
intel\appdata\roaming\python\python311\site-packages (from rdflib)
(0.6.1)
Requirement already satisfied: pyparsing<4,>=2.1.0 in c:\programdata\
anaconda3\lib\site-packages (from rdflib) (3.0.9)
```

```
Requirement already satisfied: six in c:\programdata\anaconda3\lib\
site-packages (from isodate<0.7.0,>=0.6.0->rdflib) (1.16.0)

DEPRECATION: Loading egg at c:\programdata\anaconda3\lib\site-
packages\vboxapi-1.0-py3.11.egg is deprecated. pip 24.3 will enforce
this behaviour change. A possible replacement is to use pip for
package installation.. Discussion can be found at
https://github.com/pypa/pip/issues/12330
```

Import des Bibliothèques RDFLib et Requests

```
import requests
import pandas as pd
from rdflib import Graph, Literal, RDF, URIRef, Namespace
from rdflib.namespace import DC
```

Fonction pour Récupérer les Données depuis l'API Scopus

```
def fetch_scopus_data(api_key, query, count=25):
    """Fetch data from the Scopus API."""
    url = 'https://api.elsevier.com/content/search/scopus'
    params = {
        'apiKey': api_key,
        'query': query,
        'count': count
    }

    try:
        response = requests.get(url, params=params)
        response.raise_for_status()
        data = response.json()

        if 'search-results' in data and 'entry' in data['search-
results']:
            return data['search-results']['entry']
        else:
            print("La structure de la réponse JSON ne contient pas les
clés attendues.")
            return None
    except requests.exceptions.HTTPError as http_err:
        print(f'Erreur HTTP {response.status_code}:
{response.reason}')
        print(response.text)
    except requests.exceptions.RequestException as req_err:
        print(f'Erreur de requête: {req_err}')
    except Exception as err:
        print(f'Erreur: {err}')
```

Fonction pour Parser les Valeurs FreetoRead

```
def parse_freetoread(value):
    if isinstance(value, list):
        return ', '.join([item['$'] for item in value])
    return value
```

Fonction pour Nettoyer et Sauvegarder les Données

```
def clean_and_save_data(entries, filename):
    if entries:
        df = pd.json_normalize(entries)

        if 'freetoread.value' in df.columns:
            df['freetoread.value'] =
df['freetoread.value'].apply(parse_freetoread)

        pd.set_option('display.max_rows', None)
        pd.set_option('display.max_columns', None)
        pd.set_option('display.width', None)
        pd.set_option('display.max_colwidth', None)

        #print(df)

        df.to_csv(filename, index=False)
        print(f"Les données ont été nettoyées et sauvegardées dans le
fichier {filename}")
```

Fonction pour Créer un RDF à partir d'un CSV

```
from rdflib import Graph, Namespace, Literal, URIRef
from rdflib.namespace import RDF, DC
import pandas as pd

def create_rdf_from_csv(csv_file, rdf_file):
    df = pd.read_csv(csv_file)
    g = Graph()

    SCOPUS = Namespace('http://example.org/scopus/')
    g.bind('scopus', SCOPUS)
    g.bind('dc', DC)

    for index, row in df.iterrows():
        publication =
URIRef(f"http://example.org/scopus/publication/{index}")
        g.add((publication, RDF.type, SCOPUS.Publication))

        if 'dc:title' in row and pd.notna(row['dc:title']):
            g.add((publication, DC.title, Literal(row['dc:title'])))

        if 'citedby-count' in row and pd.notna(row['citedby-count']):
```

```
            g.add((publication, SCOPUS.citedbyCount,
Literal(row['citedby-count'])))

        if 'prism:publicationName' in row and
pd.notna(row['prism:publicationName']):
            g.add((publication, SCOPUS.publicationName,
Literal(row['prism:publicationName'])))

        if 'dc:creator' in row and pd.notna(row['dc:creator']):
            g.add((publication, DC.creator,
Literal(row['dc:creator'])))

        if 'prism:coverDate' in row and
pd.notna(row['prism:coverDate']):
            g.add((publication, DC.date,
Literal(row['prism:coverDate'])))

        if 'freetoread.value' in row and
pd.notna(row['freetoread.value']):
            g.add((publication, SCOPUS.freetoRead,
Literal(row['freetoread.value'])))

        # Ajout des autres colonnes spécifiées
        if 'prism:eIssn' in row and pd.notna(row['prism:eIssn']):
            g.add((publication, SCOPUS.eIssn,
Literal(row['prism:eIssn'])))

        if 'prism:volume' in row and pd.notna(row['prism:volume']):
            g.add((publication, SCOPUS.volume,
Literal(row['prism:volume'])))

        if 'prism:issueIdentifier' in row and
pd.notna(row['prism:issueIdentifier']):
            g.add((publication, SCOPUS.issueIdentifier,
Literal(row['prism:issueIdentifier'])))

        if 'prism:pageRange' in row and
pd.notna(row['prism:pageRange']):
            g.add((publication, SCOPUS.pageRange,
Literal(row['prism:pageRange'])))

        if 'prism:coverDisplayDate' in row and
pd.notna(row['prism:coverDisplayDate']):
            g.add((publication, SCOPUS.coverDisplayDate,
Literal(row['prism:coverDisplayDate'])))

        if 'prism:doi' in row and pd.notna(row['prism:doi']):
            g.add((publication, SCOPUS.doi,
Literal(row['prism:doi'])))
```

```
        if 'affiliation' in row and pd.notna(row['affiliation']):
            g.add((publication, SCOPUS.affiliation,
Literal(row['affiliation'])))

        if 'prism:aggregationType' in row and
pd.notna(row['prism:aggregationType']):
            g.add((publication, SCOPUS.aggregationType,
Literal(row['prism:aggregationType'])))

        if 'subtype' in row and pd.notna(row['subtype']):
            g.add((publication, SCOPUS.subtype,
Literal(row['subtype'])))

        if 'subtypeDescription' in row and
pd.notna(row['subtypeDescription']):
            g.add((publication, SCOPUS.subtypeDescription,
Literal(row['subtypeDescription'])))

        if 'article-number' in row and pd.notna(row['article-
number']):
            g.add((publication, SCOPUS.articleNumber,
Literal(row['article-number'])))

        if 'source-id' in row and pd.notna(row['source-id']):
            g.add((publication, SCOPUS.sourceId, Literal(row['source-
id'])))

        if 'openaccess' in row and pd.notna(row['openaccess']):
            g.add((publication, SCOPUS.openAccess,
Literal(row['openaccess'])))

        if 'openaccessFlag' in row and
pd.notna(row['openaccessFlag']):
            g.add((publication, SCOPUS.openAccessFlag,
Literal(row['openaccessFlag'])))

        if 'freetoread.value' in row and
pd.notna(row['freetoread.value']):
            g.add((publication, SCOPUS.freetoRead,
Literal(row['freetoread.value'])))

        if 'freetoreadLabel.value' in row and
pd.notna(row['freetoreadLabel.value']):
            g.add((publication, SCOPUS.freetoReadLabel,
Literal(row['freetoreadLabel.value'])))

        if 'prism:issn' in row and pd.notna(row['prism:issn']):
            g.add((publication, SCOPUS.issn,
Literal(row['prism:issn'])))
```

```
        if 'pubmed-id' in row and pd.notna(row['pubmed-id']):
            g.add((publication, SCOPUS.pubmedId, Literal(row['pubmed-
id'])))

        if 'coverYear' in row and pd.notna(row['coverYear']):
            g.add((publication, SCOPUS.coverYear,
Literal(row['coverYear'])))

    g.serialize(rdf_file, format='turtle')
```

Utilisation de l'API Scopus, Nettoyage des Données, Création d'un RDF et Requête SPARQL

```
api_key = '9aebde1fa88b0b7325c7d8054dd3e754'
query = 'KEY(scopus)'
filename = 'api_scopus_data.csv'
rdf_filename = 'scopus_data.ttl'

# Récupération et nettoyage des données
entries = fetch_scopus_data(api_key, query)
clean_and_save_data(entries, filename)

# Création du fichier RDF à partir du CSV
create_rdf_from_csv(filename, rdf_filename)
print(f"Les données RDF ont été créées à partir du fichier
{filename}.")

def execute_sparql_query_and_style_results(rdf_filename,
sparql_query):
    g = Graph()
    g.parse(rdf_filename, format='turtle')

    # Execute the SPARQL query
    results = g.query(sparql_query)

    # Styling the results
    print("Les noms de publication et dates :\n")
    for idx, row in enumerate(results):
        publication_name = row['publicationName']
        cover_date = row['coverDate']

        # Print each result with styling
        print(f"{idx + 1}. Publication Name: {publication_name}")
        print(f"   Date: {cover_date}\n")


sparql_query_publications_info = """
PREFIX scopus: <http://example.org/scopus/>
PREFIX dc: <http://purl.org/dc/elements/1.1/>
```

```
PREFIX prism: <http://prismstandard.org/namespaces/basic/2.0/>

SELECT ?publicationName ?coverDate
WHERE {
    ?publication a scopus:Publication ;
                 dc:date ?coverDate ;
                 scopus:publicationName ?publicationName .
}
LIMIT 5
"""
execute_sparql_query_and_style_results(rdf_filename,
sparql_query_publications_info)

Les données ont été nettoyées et sauvegardées dans le fichier
api_scopus_data.csv
Les données RDF ont été créées à partir du fichier
api_scopus_data.csv.
Les noms de publication et dates :

1. Publication Name: Systematic Reviews
   Date: 2024-12-01

2. Publication Name: BMC Psychiatry
   Date: 2024-12-01

3. Publication Name: BMC Pregnancy and Childbirth
   Date: 2024-12-01

4. Publication Name: Thrombosis Journal
   Date: 2024-12-01

5. Publication Name: Diabetology and Metabolic Syndrome
   Date: 2024-12-01


def execute_sparql_query_authors_and_titles(rdf_filename,
sparql_query):
    g = Graph()
    g.parse(rdf_filename, format='turtle')

    # Execute the SPARQL query
    results = g.query(sparql_query)

    # Store results in a list of tuples
    data = [(row['creator'], row['title']) for row in results]

    return data

# Votre requête SPARQL
sparql_query_authors_and_titles = """
PREFIX scopus: <http://example.org/scopus/>
```

```
PREFIX dc: <http://purl.org/dc/elements/1.1/>

SELECT ?creator ?title
WHERE {
    ?publication a scopus:Publication .
    ?publication dc:creator ?creator .
    ?publication dc:title ?title .
}
LIMIT 5
"""

# Appel de la fonction pour exécuter la requête et obtenir les
résultats sous forme de liste de tuples
results = execute_sparql_query_authors_and_titles(rdf_filename,
sparql_query_authors_and_titles)

# Affichage des résultats sous forme de liste de tuples
print("\nCréateurs et titres des publications :")
for creator, title in results:
    print(f"Auteur: {creator}\nTitre: {title}\n")


Créateurs et titres des publications :
Auteur: Benavides-Gil G.
Titre: Mindfulness-based interventions for improving mental health of
frontline healthcare professionals during the COVID-19 pandemic: a
systematic review

Auteur: Bafkar N.
Titre: Efficacy and safety of omega-3 fatty acids supplementation for
anxiety symptoms: a systematic review and dose-response meta-analysis
of randomized controlled trials

Auteur: Moradkhani A.
Titre: Association of vitamin D receptor genetic polymorphisms with
the risk of infertility: a systematic review and meta-analysis

Auteur: Maghsudlu M.
Titre: Systematic review and meta-analysis of association between
plasminogen activator inhibitor-1 4G/5G polymorphism and recurrent
pregnancy loss: an update

Auteur: Miao Z.
Titre: Impact of frailty on mortality, hospitalization, cardiovascular
events, and complications in patients with diabetes mellitus: a
systematic review and meta-analysis


def execute_sparql_query_publications_by_year(rdf_file, sparql_query):
    # Fonction pour exécuter la requête SPARQL et afficher les
résultats
```

```
    g = Graph()
    g.parse(rdf_file, format='turtle')

    qres = g.query(sparql_query)

    # Affichage des résultats sous forme de tableau
    print("\nNombre de publications par année :")
    print("{:<10} {:<10}".format("Year", "Count"))
    print("="*25)
    for row in qres:
        print("{:<10} {:<10}".format(row['year'], row['count']))

sparql_query_publications_by_year = """
PREFIX scopus: <http://example.org/scopus/>
PREFIX dc: <http://purl.org/dc/elements/1.1/>
PREFIX prism: <http://prismstandard.org/namespaces/basic/2.0/>

SELECT ((?coverDate) AS ?year) (COUNT(?publication) AS ?count)
WHERE {
    ?publication a scopus:Publication ;
                 dc:date ?coverDate .
}
GROUP BY ?year
"""
# Appel de la fonction avec la nouvelle requête et le nom de fonction
modifié
execute_sparql_query_publications_by_year(rdf_filename,
sparql_query_publications_by_year)


Nombre de publications par année :
Year       Count
=========================
2024-12-01 25

def execute_sparql_query_doi_and_publication_name(rdf_file,
sparql_query):
    # Fonction pour exécuter la requête SPARQL et afficher les
résultats
    g = Graph()
    g.parse(rdf_file, format='turtle')

    qres = g.query(sparql_query)

    # Affichage des résultats sous forme de tableau
    print("\nAffichage des DOI et noms de publication :")
    print("{:<30} {:<70}".format("DOI", "Publication Name"))
    print("="*100)
    for row in qres:
        print("{:<30} {:<70}".format(row['doi'],
```

```
row['publicationName']))

sparql_query_doi_and_publication_name = """
PREFIX scopus: <http://example.org/scopus/>
PREFIX dc: <http://purl.org/dc/elements/1.1/>
PREFIX prism: <http://prismstandard.org/namespaces/basic/2.0/>

SELECT ?doi ?publicationName
WHERE {
    ?publication a scopus:Publication ;
                 scopus:doi ?doi ;
                 scopus:publicationName ?publicationName .
}
LIMIT 6
"""
# Appel de la fonction avec la nouvelle requête et le nom de fonction
modifié
execute_sparql_query_doi_and_publication_name(rdf_filename,
sparql_query_doi_and_publication_name)


Affichage des DOI et noms de publication :
DOI                            Publication Name

================================================================================
==============================
10.1186/s13643-024-02574-5     Systematic Reviews

10.1186/s12888-024-05881-2     BMC Psychiatry

10.1186/s12884-024-06590-0     BMC Pregnancy and Childbirth

10.1186/s12959-024-00612-9     Thrombosis Journal

10.1186/s13098-024-01352-6     Diabetology and Metabolic Syndrome

10.1007/s00238-024-02190-5     European Journal of Plastic Surgery


def execute_sparql_query_volume(rdf_file, sparql_query):
    # Fonction pour exécuter la requête SPARQL et afficher les
résultats
    g = Graph()
    g.parse(rdf_file, format='turtle')

    qres = g.query(sparql_query)

    # Affichage des résultats sous forme de tableau
    print("\nAffichage des volumes des publications :")
    print("{:<10} {:<30}".format("Publication", "Volume"))
    print("="*50)
```

```
    for row in qres:
        print("{:<10} {:<30}".format(row['publication'],
row['volume']))

sparql_query_volume = """
PREFIX scopus: <http://example.org/scopus/>
PREFIX prism: <http://prismstandard.org/namespaces/basic/2.0/>

SELECT ?publication ?volume
WHERE {
    ?publication a scopus:Publication ;
                 scopus:volume ?volume .
}
LIMIT 10
"""
# Appel de la fonction avec la requête pour les volumes des
publications
execute_sparql_query_volume(rdf_filename, sparql_query_volume)


Affichage des volumes des publications :
Publication Volume
==================================================
http://example.org/scopus/publication/0 13
http://example.org/scopus/publication/1 24
http://example.org/scopus/publication/10 24

http://example.org/scopus/publication/11 22

http://example.org/scopus/publication/12 16

http://example.org/scopus/publication/13 47

http://example.org/scopus/publication/14 24

http://example.org/scopus/publication/15 13

http://example.org/scopus/publication/16 24

http://example.org/scopus/publication/17 39


from rdflib import Graph, Namespace

def execute_sparql_query_article_volume(rdf_file, sparql_query):
    # Chargement du fichier RDF
    g = Graph()
    g.parse(rdf_file, format='turtle')

    # Exécution de la requête SPARQL
    qres = g.query(sparql_query)
```

```
    # Affichage des résultats
    print("\nPublications de type 'Article' avec leur volume :")
    print("=" * 50)
    for row in qres:
        print(f"Titre de la publication : {row['title']}")
        print(f"Volume : {row['volume']}")
        print("-" * 50)

# Requête SPARQL pour récupérer les articles avec leur volume
sparql_query_article_volume = """
PREFIX scopus: <http://example.org/scopus/>
PREFIX prism: <http://prismstandard.org/namespaces/basic/2.0/>

SELECT ?title (GROUP_CONCAT(?volume; separator=", ") AS ?volume)
WHERE {
    ?publication a scopus:Publication ;
                 scopus:subtypeDescription "Article" ;
                 scopus:volume ?volume ;
                 dc:title ?title .
}
GROUP BY ?title
LIMIT 10
"""

# Appel de la fonction avec la requête pour les articles et leur
volume
execute_sparql_query_article_volume(rdf_filename,
sparql_query_article_volume)
```

```
Publications de type 'Article' avec leur volume :
==================================================
Titre de la publication : Mindfulness-based interventions for
improving mental health of frontline healthcare professionals during
the COVID-19 pandemic: a systematic review
Volume : 13
--------------------------------------------------
Titre de la publication : Efficacy and safety of omega-3 fatty acids
supplementation for anxiety symptoms: a systematic review and dose-
response meta-analysis of randomized controlled trials
Volume : 24
--------------------------------------------------
Titre de la publication : Association of vitamin D receptor genetic
polymorphisms with the risk of infertility: a systematic review and
meta-analysis
Volume : 24
--------------------------------------------------
Titre de la publication : Impact of frailty on mortality,
hospitalization, cardiovascular events, and complications in patients
```

with diabetes mellitus: a systematic review and meta-analysis
Volume : 16

Titre de la publication : Association of prothrombin time, thrombin time and activated partial thromboplastin time levels with preeclampsia: a systematic review and meta-analysis
Volume : 24

Titre de la publication : Protocol for a systematic review and meta-analysis on Janus kinase inhibitors in the management of vitiligo
Volume : 13

Titre de la publication : Analyzing global research trends and focal points in the utilization of laser techniques for the treatment of urolithiasis from 1978 to 2022: visualization and bibliometric analysis
Volume : 52

Titre de la publication : Barriers and facilitators to implementing workplace interventions to promote mental health: qualitative evidence synthesis
Volume : 13

Titre de la publication : The impact of immunosuppression on the mortality and hospitalization of Monkeypox: a systematic review and meta-analysis of the 2022 outbreak
Volume : 21

Titre de la publication : 75 years’ journey of malaria publications in English: what and where?
Volume : 23
