

# Création d'un framework BI pour l'analyse de données BIG Data



Mini  
Projet

2021-2022



Encadré Par : Mr. FENNAN Abdelhadi

Réalisé Par :

ACHATIBI Hajar

ZEROUALI Hajar

# Tables de matières

OBJECTIF.....	3
I. INTRODUCTION .....	4
II. TACHES DE REALISATION DE PROJET.....	5
III. COLLECTION DES DONNEES.....	6
a. Description .....	6
b. Installation des outils .....	7
c. Création du projet « Scrapy » .....	9
d. Structure de projet « Scrapy » .....	9
e. Création du modèle.....	10
f. Génération des spiders : .....	10
IV. STOCKAGES DES DONNEES .....	18
a. Description .....	18
b. Installation des outils .....	18
c. Enregistrement des données.....	18
d. Fichier de paramétrage .....	19
e. Affichage des résultats d'enregistrement .....	20
V. ANALYSE & VISUALISATION SPARK .....	21
a. Description .....	21
b. Installation des outils .....	21
c. Connexion à la base de données via « Apache Spark » .....	23
d. Chargement des données .....	24
e. Traitement des données.....	24
VI. ANALYSE BI & VISUALISATION .....	30
a. Description .....	30
b. Installation des outils .....	30
c. Schéma en étoile .....	32
d. Visualisation et reporting avec Microsoft Power BI.....	39
VII. Versions des technologies .....	47
VIII. PROBLEMES RENCONTRES.....	48
IX. Conclusion.....	49
X. REFERNCES.....	50

## OBJECTIF

Notre travail a pour but la création d'un framework BI pour l'analyse Big Data en abordant le sujet des recherches scientifiques effectuées dans les domaines de « Blockchain » et « Bitcoin » afin de mettre en lumière l'intérêt de ces domaines tout en extrair, collectant et analysant les données à l'aide d'ensemble des méthodologies offert par les technologies BI et Big Data.

## I. INTRODUCTION

La Business Intelligence est un processus technologique d'analyse des données et de présentation d'informations qui sert à aider les dirigeants, managers et les décideurs de l'entreprise à prendre des décisions business éclairées via les rapports, des tableaux de bord obtenus.

Face à une croissance impressionnante des données, les entreprises se retrouvent face à un challenge technique de taille : continuer à valoriser les données, tout en intégrant une quantité toujours plus grande d'informations. Les entreprises ont donc réussi à concilier BI et Big Data. Leur combinaison permet d'augmenter les sources d'informations disponibles, et donc de dépasser le cadre, souvent restreint, de l'organisation grâce à leurs capacités d'identifier et d'extraire des informations précieuses des grands volumes de données qu'elles stockent. Ces outils permettent d'en tirer des informations tels que des veilles concurrentielles et les tendances du marché, ainsi que des informations internes tel que trouver les raisons des opportunités perdues.

En effet, parmi les tendances de domaine informatique nous citons le « Bitcoin » et la technologie « Blockchain ». Les entreprises qui visent à s'investir ou savoir l'influence de ces technologies doivent penser à voir une veille scientifique et concurrentielle qui sert à capter le maximum d'informations sur tous ce qui concerne la recherche scientifique dans ces domaines à l'échelle mondiale.

## II. TACHES DE REALISATION DE PROJET

Dans le but de réussir les objectifs de ce projet nous allons exécuter ces tâches :

- ➊ Récupération des données en implémentant la méthode d'extraction des données depuis les journaux scientifiques avec l'outil Scrapy.
- ➋ Stockages des données massives collectés avec MongoDB et Hadoop.
- ➌ Traitement et analyse des données à travers Apache Spark en nettoyant les données afin d'obtenir des vrais résultats.
- ➍ Mise en place d'un entrepôt de données et analyse par la plateforme Pentaho.
- ➎ Visualisation des données et générations des rapports via la plateforme BI, Microsoft Power BI.

### III. COLLECTION DES DONNEES

#### a. Description

Cette phase sert à récupérer l'ensemble des articles scientifiques dans les domaines de recherche « Blockchain » et le « Bitcoin » à partir des trois journaux scientifiques situées ci-dessous :

- ✿ ACM (Association for Computing Machinery) : une association internationale basée aux États-Unis dont l'objectif est le développement et le soutien à la science et à la technologie informatique. ACM met à la disposition une bibliothèque numérique sous forme de plate-forme de recherche, de découverte et de mise en réseau.

Lien: <https://dl.acm.org/>

- ✿ ScienceDirect : un site web géré par l'éditeur Elsevier. Lancée en mars 1997, la plateforme permet d'accéder à plus de 3 800 revues académiques qui forment plus de 14 millions de publications scientifiques en sciences dures et appliquées, en économie, en psychologie, en médecine et dans une moindre mesure dans les autres disciplines revues par des pairs.

Lien: <https://www.sciencedirect.com/>

- ✿ IEEE Xplore : cette bibliothèque numérique est une base de données de recherche pour la découverte et l'accès à des articles de revues, des actes de conférence, des normes techniques et des documents connexes sur l'informatique, le génie électrique et l'électronique et les domaines connexes.

Lien: <https://ieeexplore.ieee.org/Xplore/home.jsp>

Dans ce qui suit nous définissons l'ensemble des informations qu'on vise à collecter de chaque article :

- ✿ Les informations liées aux auteurs :

- Nom et Prénom
  - Laboratoire ou Université
  - Pays

- ✿ Les informations liées aux articles :

- Titre
  - Sujet
  - Doi
  - Date de publication
  - Abstract
  - References
  - Citation

- Nombre de téléchargement
- ✿ Les informations liées aux journaux :
  - Publisher
  - Issn
  - Indexation
  - Impact factor

Les données sont extraites via l'outil « Scrapy ».

## b. Installation des outils

### Scrapy :

Scrapy est un framework python open source et collaboratif qui sert à faire l'extraction des données sur le web. Scrapy est adapté aux grands projets de web scraping. En effet, les projets Scrapy ont une structure assez claire qui facilite la maintenance et le passage à l'échelle. C'est une base sur les spiders qui jouent le rôle de des robots autonomes qui reçoivent certaines instructions.



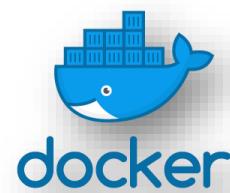
```
PS C:\Users\Probook\PycharmProjects\BIBigDataProject> pip install scrapy
Collecting scrapy
  Using cached Scrapy-2.5.1-py2.py3-none-any.whl (254 kB)
Collecting itemloaders>=1.0.1
  Using cached itemloaders-1.0.4-py3-none-any.whl (11 kB)
Collecting itemadapter>=0.1.0
  Using cached itemadapter-0.4.0-py3-none-any.whl (10 kB)
Collecting cssselect>=0.9.1
  Using cached cssselect-1.1.0-py2.py3-none-any.whl (16 kB)
```

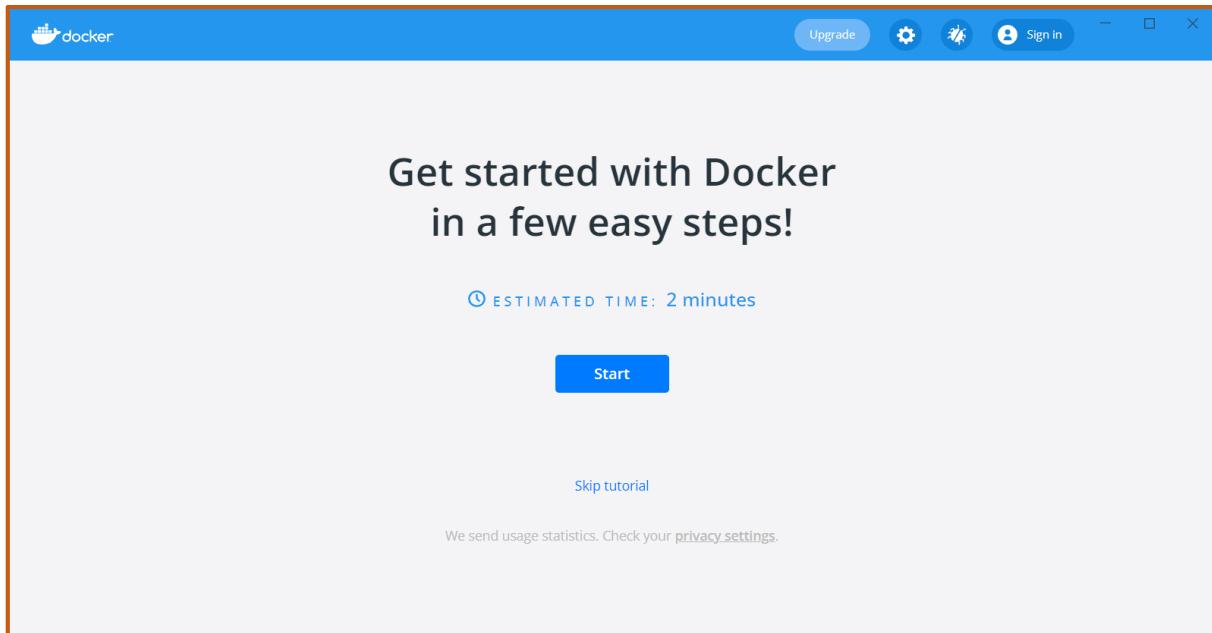
```
PS C:\Users\Probook\PycharmProjects\BIBigDataProject> scrapy startproject scrapingIEEE
New Scrapy project 'scrapingIEEE', using template directory 'c:\users\probook\pycharmprojects\bibigdataproj
C:\Users\Probook\PycharmProjects\BIBigDataProject\scrapingIEEE

You can start your first spider with:
  cd scrapingIEEE
  scrapy genspider example example.com
PS C:\Users\Probook\PycharmProjects\BIBigDataProject>
```

### Docker :

Docker est une plateforme de conteneurs lancée en 2013 ayant largement contribué à la démocratisation de la conteneurisation. Elle permet de créer facilement des conteneurs et des applications basées sur les conteneurs.





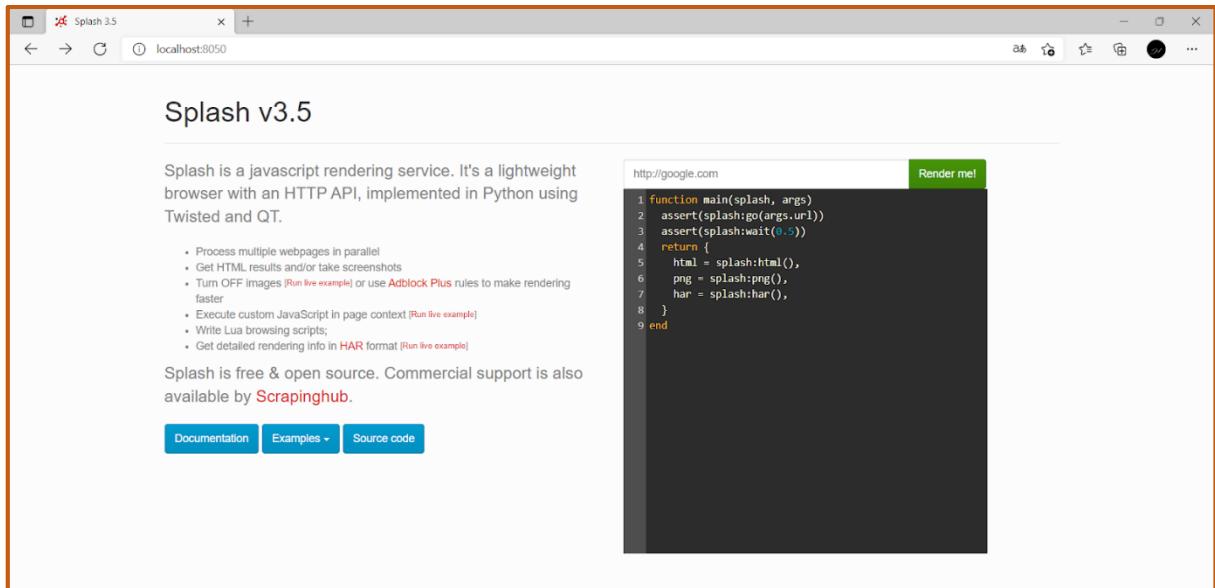
## Splash :

Splash est un navigateur léger avec une API HTTP, implémenté en python 3 à l'aide de Twisted et QT5. Il est rapide, léger et sans état, ce qui le rend facile à distribuer.

```
Collecting scrapy-splash
  Downloading scrapy_splash-0.8.0-py2.py3-none-any.whl (27 kB)
Installing collected packages: scrapy-splash
Successfully installed scrapy-splash-0.8.0
```

```
PS C:\Users\hajar\OneDrive\Bureau\S5\securite2\pyCharmProjects\BI_BIG-DATA> docker run -p 8050:8050 scrapinghub/splash
Unable to find image 'scrapinghub/splash:latest' locally
latest: Pulling from scrapinghub/splash
7595c8c21622: Pulling fs layer
d13af8ca898f: Pulling fs layer
70799171ddba: Pulling fs layer
b0c12202c5ef: Waiting
60a588d4f36e: Waiting
74efcc44bb0a: Waiting
630a03095961: Waiting
6ece3b427ef5: Pulling fs layer
a1fce8353093: Waiting
3933ca5f4768: Waiting
b0097a197686: Waiting
74efcc44bb0a: Downloading [=====>] 11.86MB/39.2MB
```

```
2021-12-13 15:51:20.219881 [-] Timing out client: IPv4Address(type='TCP', host='172.17.0.1', port=55544)
2021-12-13 15:51:20.230692 [-] Timing out client: IPv4Address(type='TCP', host='172.17.0.1', port=55548)
2021-12-13 15:51:20.248345 [-] Timing out client: IPv4Address(type='TCP', host='172.17.0.1', port=55532)
2021-12-13 15:51:20.267556 [-] Timing out client: IPv4Address(type='TCP', host='172.17.0.1', port=55536)
2021-12-13 15:51:20.299320 [-] Timing out client: IPv4Address(type='TCP', host='172.17.0.1', port=55552)
```



### c. Création du projet « Scrapy »

```

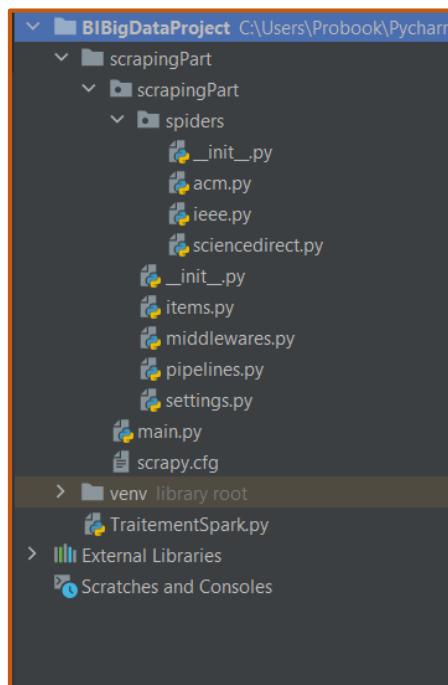
PS C:\Users\Probook\PycharmProjects\BIBigDataProject> scrapy startproject scrappingPart
New Scrapy project 'scrappingPart', using template directory 'c:\users\probook\pycharmproj
C:\Users\Probook\PycharmProjects\BIBigDataProject\scrappingPart

You can start your first spider with:
cd scrappingPart
scrapy genspider example example.com
PS C:\Users\Probook\PycharmProjects\BIBigDataProject> []

```

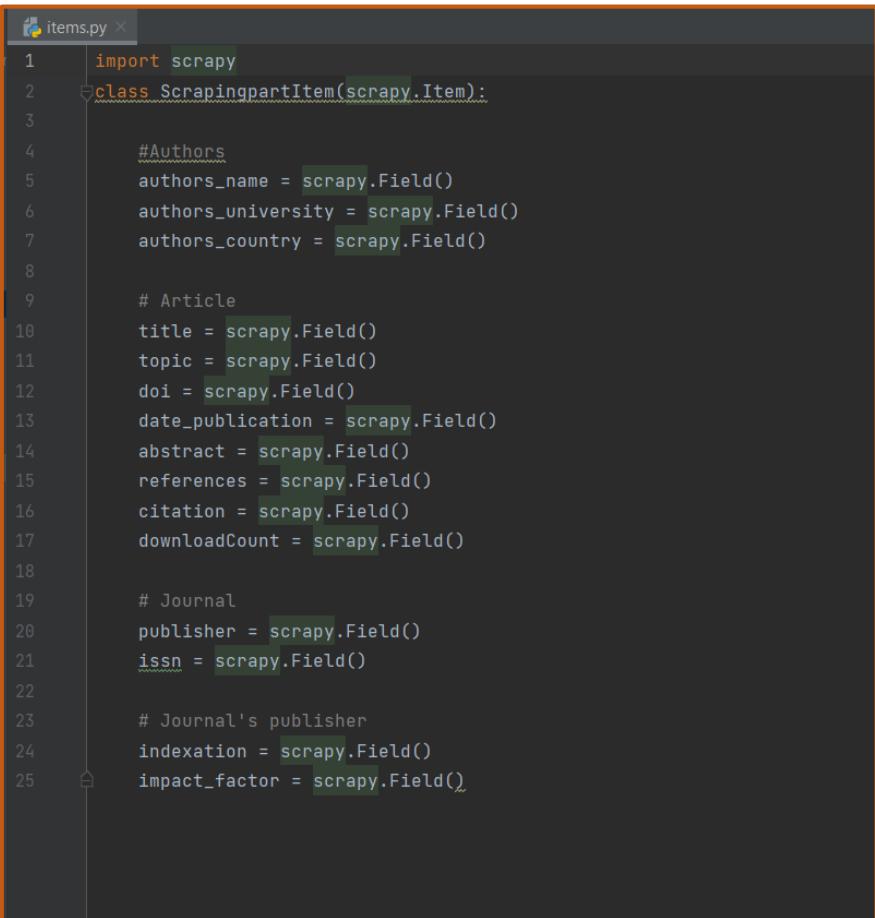
### d. Structure de projet « Scrapy »

L'arborescence de ce projet est définie comme suit :



### e. Création du modèle

Le fichier responsable de la création de modèle est celui nommé : « Items.py ». Dans ce fichier nous définissons les champs qu'on souhaite extraire à partir des articles convenables.



```

items.py x
1 import scrapy
2 class ScrapingpartItem(scrapy.Item):
3
4     #Authors
5     authors_name = scrapy.Field()
6     authors_university = scrapy.Field()
7     authors_country = scrapy.Field()
8
9     # Article
10    title = scrapy.Field()
11    topic = scrapy.Field()
12    doi = scrapy.Field()
13    date_publication = scrapy.Field()
14    abstract = scrapy.Field()
15    references = scrapy.Field()
16    citation = scrapy.Field()
17    downloadCount = scrapy.Field()
18
19     # Journal
20    publisher = scrapy.Field()
21    issn = scrapy.Field()
22
23     # Journal's publisher
24    indexation = scrapy.Field()
25    impact_factor = scrapy.Field()

```

### f. Génération des spiders :

Afin d'extraire les données à partir de chacun des journaux scientifiques, nous avons créé des spiders qui explorent le web :

#### ❖ Création de spider pour « ACM » :

Cette commande permet la génération de spider en définissant lien d'ACM

```
scrapy genspider acm dl.acm.org
```

#### Développement :

Dans cette partie nous prenons en compte le mot clés mis en argument qui représente le sujet afin d'extraire les informations via les urls convenables.

Premièrement, nous avons créé une boucle afin de gérer la pagination qui permet de basculer entre les pages sans aucun besoin d'intervention humain.

```

import webbrowser
import scrapy
from scrapy import Selector
from scrapy.utils.response import open_in_browser
from scrapy_splash import SplashRequest
from scrapingPart.items import ScrapingpartItem

class AcmSpider(scrapy.Spider):
    name = 'acm'
    topic = "None"
    allowed_domains = ['dl.acm.org']
    start_urls = []

    handle_httpstatus_list = [302]
    handle_httpstatus_list = [301]

    def __init__(self, keyword=None, topic=None, *args, **kwargs):
        super(AcmSpider, self).__init__(*args, **kwargs)

        for i in range(50):
            self.start_urls += ['https://dl.acm.org/action/doSearch?AllFields=' + topic + '&startPage=' + str(i) + '&pageSize=' + str(20)]
        self.topic = topic

```

La fonction « parse » permet de gérer les réponses obtenues des requêtes déjà effectuées. Cette fonction retourne comme réponse le contenu de journal après la recherche de sujet désiré. Cette méthode serve à extraire les liens qui correspondent à l'ensemble des articles souhaités via la balise CSS `<a>` de l'attribut « href ».

```

def parse(self, response):
    for article in response.css("a::attr(href)"):
        if '/doi/' in article.extract():
            yield SplashRequest('https://dl.acm.org' + article.extract(), self.parse_article, args={'wait': 3})

```

La fonction « parse\_article » est appelée directement après la fonction « parse » afin de remplir la base de données par les données collectées à partir de réponse qui n'est rien d'autre que la page html d'un des articles.

```

def parse_article(self, response):
    item = ScrapingpartItem()

    # ---- Déclaration des attributs + Remplissage
    # Authors
    authors_name = response.css('.author-data').css('span::text').extract()
    authors_infos = response.css('.author-info__body').css('p::text').extract()
    authors_university = []
    authors_country = []

    # Article
    title = response.css('.citation__title::text').extract_first()
    topic = self.topic
    doi = response.css('.issue-item__doi::text').extract()
    date_publication = response.css('.CitationCoverDate::text').extract()

```

```

abstract = response.css('.abstractSection').css('p::text').extract()
references = response.css('.references__note::text').extract()
citation = ';' .join(response.css('.tooltip .citation').css('span::text').extract())
downloadCount = ';' .join(response.css('.tooltip .metric').css('span::text').extract())

# Journal
publisher = "ACM"
issn = "00045411, 1557735X"

# Journal's publisher
indexation = 134
impact_factor = 6.738

if len(authors_infos) != 0:
    for auth_info in authors_infos:
        if auth_info != "":
            s = auth_info.split(',')
            authors_university.append(s[0])

            if len(s) > 1:
                authors_country.append(s[len(s) - 1])
            else:
                authors_country = ""
        else:
            authors_university = ""
            authors_country = ""

authors_country = ';' .join(authors_country)

```

```

# ---- Affectation
# Authors
item['authors_name'] = authors_name
item['authors_university'] = authors_university
item['authors_country'] = authors_country.strip()

# Article
item['title'] = title
item['topic'] = topic
item['doi'] = doi[0]
try:
    item['date_publication'] = int(date_publication[0].split(' ')[-1])
except:
    item['date_publication'] = 0
item['abstract'] = abstract
item['references'] = references
item['citation'] = int(''.join(citation).split(';')[0].replace(',', ''))
item['downloadCount'] = int(''.join(downloadCount).split(';')[0].replace(',', ''))

# Journal
item['publisher'] = publisher
item['issn'] = issn

# Journal's publisher
item['indexation'] = indexation
item['impact_factor'] = impact_factor

yield item

```

**Exécution :**

Cette commande permet l'exécution de spider à condition de saisir le topic.

```
scrapy crawl acm -a topic="Bitcoin"
```

**⌚ Création de spider pour « IEEE Xplore » :**

Cette commande permet la génération de spider en définissant lien d'IEEE Xplore :

```
scrapy genspider ieee ieeexplore.ieee.org
```

**Développement :**

```
import requests
import scrapy
from scrapingPart.items import ScrapingpartItem

class IeeeSpider(scrapy.Spider):
    name = 'ieee'
    topic = None
    start_urls = None
    page_no = 1
    r = None
    headers = {
        "Accept": "application/json, text/plain, */*",
        "Origin": "https://ieeexplore.ieee.org",
        "Content-Type": "application/json",
    }
    payload = {"newsearch": True, "queryText": topic, "highlight": False, "returnFacets": ["ALL"], "returnType": "SEARCH", "pageNumber": page_no}

    # Constructeur
    def __init__(self, keyword=None, topic=None, *args, **kwargs):
        super(IeeeSpider, self).__init__(*args, **kwargs)
        self.start_urls = ['https://ieeexplore.ieee.org/rest/search']
        self.topic = topic
        self.payload['queryText'] = topic
        self.r = requests.post(
            "https://ieeexplore.ieee.org/rest/search",
            headers=self.headers,
            json=self.payload
        )
```

```
def parse(self, response):
    page_data = self.r.json()
    for record in page_data["records"]:

        # ---- Déclaration des attributs + Remplissage
        # Authors
        authors_name = []
        authors_infos = record["authors"]
        authors_university = ""
        authors_country = ""

        # Article
        title = record["articleTitle"]
        topic = self.topic
        try:
            doi = record["doi"]
        except:
            doi = ""
        date_publication = record["publicationYear"]
        abstract = record["abstract"]
        references = ""
```

```

citation = record["citationCount"]
downloadCount = record["downloadCount"]

# Journal
publisher = record["publisher"]
issn = "21682372"

# Journal's publisher
indexation = 24
impact_factor = 3.825

l = len(authors_infos)
for i in range(l):
    auth = authors_infos[i]
    authors_name.append(auth['preferredName'])

item = ScrapingpartItem()

# Authors
item['authors_name'] = authors_name
item['authors_university'] = authors_university
item['authors_country'] = authors_country

# Article
item['title'] = title
item['topic'] = topic
item['doi'] = doi

```

```

try:
    item['date_publication'] = date_publication
except:
    item['date_publication'] = 0
item['abstract'] = abstract
item['references'] = references
item['citation'] = citation
item['downloadCount'] = downloadCount

# Journal
item['publisher'] = publisher
item['issn'] = issn

# Journal's publisher
item['indexation'] = indexation
item['impact_factor'] = impact_factor

yield item

```

### Exécution :

Cette commande permet l'exécution de spider à condition de saisir le topic.

```
scrapy crawl ieee -a topic="Blockchain"
```

## 1. Création de spider pour « ScienceDirect »

Cette commande permet la génération de spider en définissant lien de ScienceDirect :

```
scrapy genspider sciencedirect www.sciencedirect.com
```

### Développement :

```
from selenium.webdriver.support import expected_conditions as EC
from selenium.webdriver.common.by import By
from selenium.common.exceptions import TimeoutException
from selenium.webdriver.support.wait import WebDriverWait
import os
import time
from selenium import webdriver
from selenium.webdriver.chrome.options import Options
import scrapy
from scrapingPart.items import ScrapingpartItem
from selenium.webdriver import ActionChains

class SciencedirectSpider(scrapy.Spider):
    name = 'sciencedirect'
    start_urls = None
    topic = ""
    chrome_options = Options()
    driver = webdriver.Chrome(executable_path=os.path.abspath("C:/Users/hajar/Downloads/chromedriver.exe"),
                               options=chrome_options)

    # Constructeur
    def __init__(self, topic=None, *args, **kwargs):
        super(SciencedirectSpider, self).__init__(*args, **kwargs)
        self.topic=topic
        self.start_urls = ['https://www.sciencedirect.com/search']
        self.driver.get("https://www.sciencedirect.com/search?qs="+topic)

    def parse(self, response):
```

```

    def parse(self, response):
        delay = 10 # seconds
        try:
            myElem = WebDriverWait(self.driver, delay).until(EC.presence_of_element_located((By.ID, 'srp-results-list'))))
            elements = self.driver.find_elements_by_css_selector("div.result-item-content h2 span a")
            for element in elements:
                article = element.get_attribute("href")
                driver1 = webdriver.Chrome(executable_path=os.path.abspath("C:/Users/haajar/Downloads/chromedriver.exe"),
                                            options=self.chrome_options)
                driver1.get(str(article))
                driver1.minimize_window()
                authors_name = []
                authors_university = []
                authors_country = []
                title = ""
                doi = None
                date_publication = 0
                abstract = ""
                references = []
                issn = ""
                indexation = 0
                impact_factor = 0
                affiliation = []
                try:
                    Elel = WebDriverWait(driver1, delay).until(EC.presence_of_element_located((By.ID, 'abstracts'))))
                    elements1 = driver1.find_elements_by_css_selector("span.title-text")

```

```

try:
    Elel = WebDriverWait(driver1, delay).until(EC.presence_of_element_located((By.ID, 'abstracts'))))
    elements1 = driver1.find_elements_by_css_selector("span.title-text")
    for element1 in elements1:
        title = element1.text
    elements1 = driver1.find_elements_by_css_selector(
        "div#author-group.author-group a.author.size-m.workspace-trigger span.content")
    for element1 in elements1:
        authors_name.append(element1.text)
    #-----Affiliation-----
    try:
        elems = driver1.find_elements_by_css_selector("button#show-more-btn")
        ActionChains(driver1).click(elems[0]).perform()
        elements1 = driver1.find_elements_by_css_selector("dl.affiliation dd")
        for element1 in elements1:
            affiliation.append(element1.text)
    except Exception:
        print(Exception)
    #-----
    elements1 = driver1.find_elements_by_css_selector("a.doi")
    for element1 in elements1:
        doi = element1.text
    elements1 = driver1.find_elements_by_css_selector("div#publication.Publication div.publication-volume div.text-xs")
    for element1 in elements1:
        date_publication = element1.text
    elements1 = driver1.find_elements_by_css_selector("div#abSTRACTS.Abstracts.u-font-serif")
    for element1 in elements1:
        abstract = element1.text

```

```
elements1 = driver1.find_elements_by_css_selector(
    "ul li.bib-reference.u-margin-s-bottom")
for element1 in elements1:
    references.append(element1.text)

except TimeoutException:
    print("Loading took too much time!")
driver1.quit()
date_publication = date_publication.split(',')[1]
date_publication = int(date_publication.split(' ')[-1])
item = ScrapingpartItem()
item['title'] = title
item['abstract'] = abstract
item['authors_name'] = authors_name
item['topic'] = self.topic
item['doi'] = doi
item['date_publication'] = date_publication
item['publisher'] = "ScienceDirect"
item['references'] = references
for inf in affiliation:
    y = inf.split(',')
    authors_university.append(y[0])
    authors_country.append(y[-1])
authors_country = ';' .join(authors_country)
item['authors_country'] = authors_country.strip()
item['authors_university'] = authors_university
item['issn'] = issn
item['impact_factor'] = impact_factor
item['indexation'] = indexation

yield item
except TimeoutException:
    print("Loading took too much time!")
self.driver.quit()
```

### Exécution :

```
scrapy crawl sciencedirect -a topic="Blockchain"
```

## IV. STOCKAGES DES DONNEES

### a. Description

Les données extraites sont sauvegardées sous MongoDB dans une base de données nommée : « ScrapingData » sous la collection « articles ».

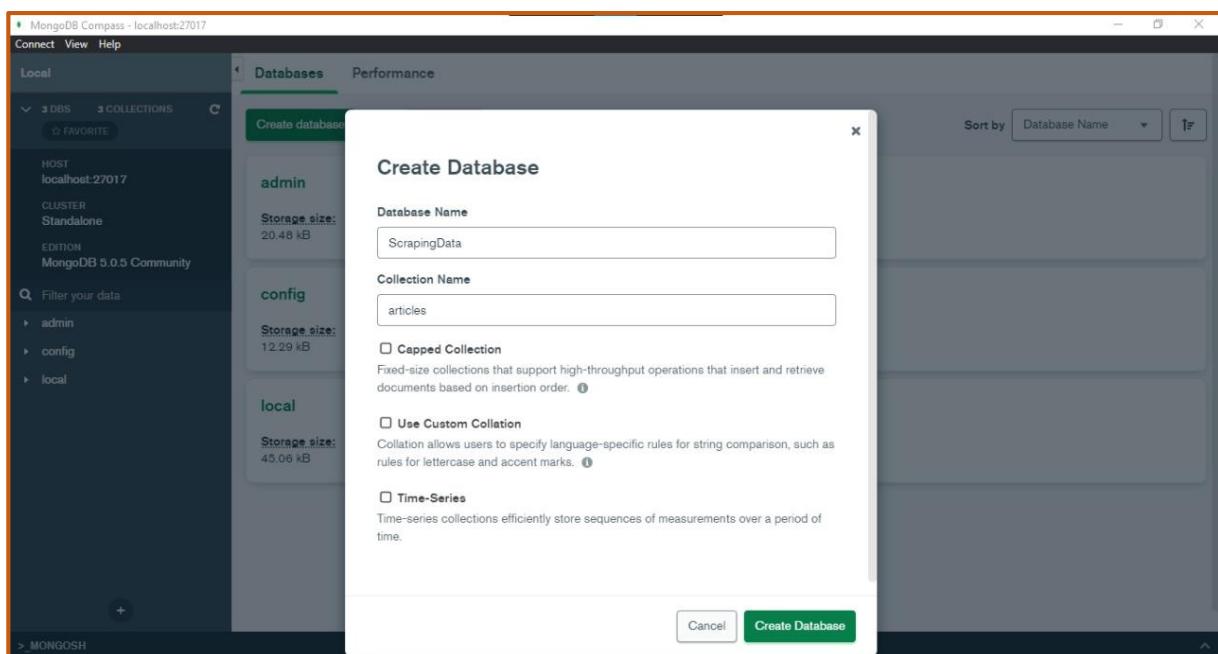
### b. Installation des outils

#### MongoDB :



MongoDB est une base de données NoSQL orientée document et Open Source. Elle se distingue des bases de données relationnelles par sa flexibilité et ses performances.

La création de la base de données :



### c. Enregistrement des données

Pour enregistrer les données extraites, il est obligatoire de se connecter via un pipeline qui permet d'établir une connexion avec la base de données MongoDB.

```

pipelines.py
10 import pymongo
11
12 class MongoPipeline(object):
13
14     def __init__(self, mongo_uri, mongo_db, mongo_collection):
15         print()
16         self.mongo_uri = mongo_uri
17         self.mongo_db = mongo_db
18         self.mongo_collection = mongo_collection
  
```

 A screenshot of a code editor window titled 'pipelines.py'. The code defines a class 'MongoPipeline' that inherits from 'object'. The '\_\_init\_\_' method takes three parameters: 'mongo\_uri', 'mongo\_db', and 'mongo\_collection'. Inside the method, there is a single line of code: 'print()'. The code editor has syntax highlighting for Python, with keywords like 'import', 'class', and 'def' in blue, and strings in red.

```

19
20     @classmethod
21     def from_crawler(cls, crawler):
22         return cls(
23             mongo_uri=crawler.settings.get('MONGO_URI'),
24             mongo_db=crawler.settings.get('MONGO_DB', 'items'),
25             mongo_collection=crawler.settings.get('MONGO_COLLECTION', 'items')
26         )
27
28     def open_spider(self, spider):
29         self.client=pymongo.MongoClient(self.mongo_uri)
30         self.db=self.client[self.mongo_db]
31
32     def close_spider(self, spider):
33         self.client.close()
34
35     def process_item(self, item, spider):
36         self.db[self.mongo_collection].insert_one(dict(item))
37         return item

```

Cette classe fait appel à des variables définies dans un autre fichier nommée « settings.py ».

#### d. Fichier de paramétrage

Au niveau de fichier « settings.py » nous définissons les champs qui seront utilisé par les différents composant de projet « Scrapy » surtout pour la connexion à la base de données MongoDB.

```

BOT_NAME = 'scrapingPart'

SPIDER_MODULES = ['scrapingPart.spiders']
NEWSPIDER_MODULE = 'scrapingPart.spiders'

SPLASH_URL = 'localhost:8050'
DUPEFILTER_CLASS = 'scrapy_splash.SplashAwareDupeFilter'
HTTPCACHE_STORAGE = 'scrapy_splash.SplashAwareFSCacheStorage'

ROBOTSTXT_OBEY = False
MEDIA_ALLOW_REDIRECTS = True
DOWNLOAD_DELAY = 3
SPIDER_MIDDLEWARES = {
    'scrapingPart.middlewares.ScrapingpartSpiderMiddleware': 543,
    'scrapy_splash.SplashCookiesMiddleware': 723,
    'scrapy_splash.SplashMiddleware': 725,
}
ITEM_PIPELINES = {'scrapingPart.pipelines.MongoPipeline': 300}
MONGODB_SERVER = "localhost"
MONGO_PORT = 27017
MONGO_DB = "ScrapingData"
MONGO_COLLECTION = "articles"
MONGO_URI = 'mongodb://localhost:27017'

```

## e. Affichage des résultats d'enregistrement

The screenshot shows the MongoDB Compass interface for a collection named "ScrapingData.articles". The top right corner displays statistics: DOCUMENTS 4.0k, TOTAL SIZE 22.0MB, AVG. SIZE 5.5KB; INDEXES 1, TOTAL SIZE 110.6KB, AVG. SIZE 110.6KB.

The main area shows a list of documents. A red arrow points to the status bar at the bottom right, which reads "Displaying documents 1 - 20 of 3999".

Two documents are partially visible:

```
_id: ObjectId("61df5ba03f520ed4ec33d9aa")
> authors_name: Array
> authors_university: Array
authors_country: "Australia; Australia; India; Australia"
title: "Blockchain Technology for Supply Chain Traceability, Transparency and ..."
topic: "Blockchain"
doi: "https://doi.org/10.1145/3381403.3381408"
date_publication: 2018
> abstract: Array
> references: Array
citation: 13
downloadCount: 1410
publisher: "ACM"
issn: "00045411, 1557735X"
indexation: 134
impact_factor: 6.738

_id: ObjectId("61df5ba43f520ed4ec33d9ab")
> authors_name: Array
> authors_university: Array
authors_country: "China; China; China"
title: "Blockchain-Enabled Trust Management in Service-Oriented Internet of Th..."
topic: "Blockchain"
doi: "https://doi.org/10.1145/3460537.3460544"
date_publication: 2021
> abstract: Array
> references: Array
citation: 0
downloadCount: 55
publisher: "ACM"
issn: "00045411, 1557735X"
```

## V. ANALYSE & VISUALISATION SPARK

### a. Description

Dans cette phase, nous allons effectuer un traitement sur les données collectées et stockées en avances dans la base de données MongoDB. Le but est de comparer le niveau de recherche scientifique pour les domaines « Blockchain » et « Bitcoin » selon certains critères.

### b. Installation des outils

#### Apache Spark :

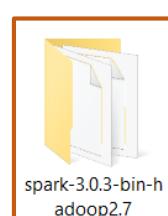
Il s'agit d'un projet de la fondation Apache issu de recherches à Berkeley. Apache Spark est un framework open source de traitement de données et ultra-rapide pour le traitement de données à grande échelle. Il est né d'une constatation simple : la technologie MapReduce.

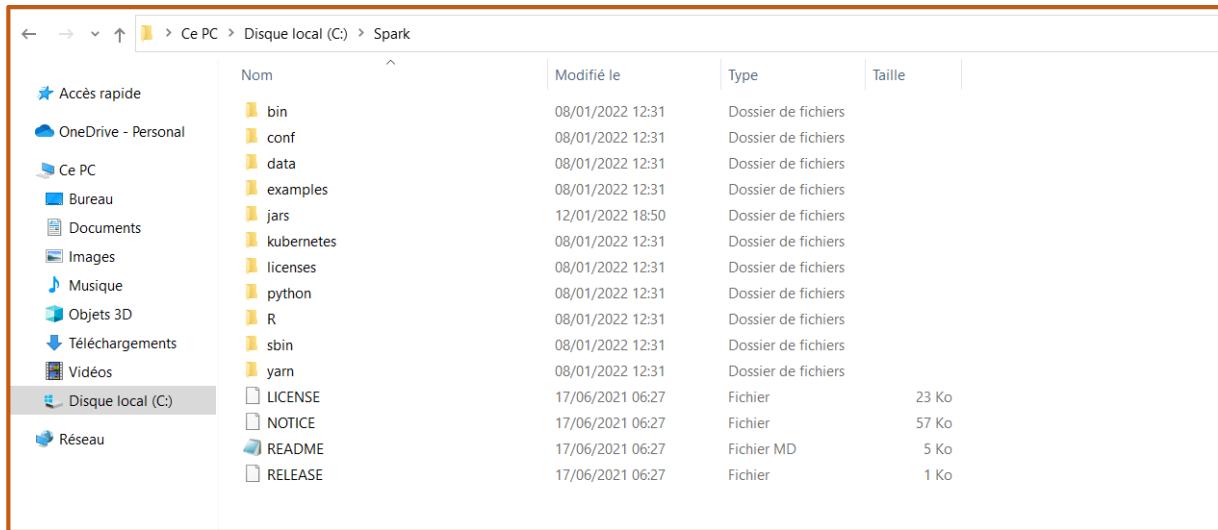


Spark permet de diviser par 100 les temps de calcul par rapport à l'utilisation de MapReduce sur du Hadoop. De plus, il permet d'appliquer des algorithmes impossibles à mettre en œuvre jusqu'ici avec les opérations de map et de reduce.

Pour installer Apache Spark, nous avons télécharger le fichier zip à partir de lien suivant : <https://spark.apache.org/downloads.html>

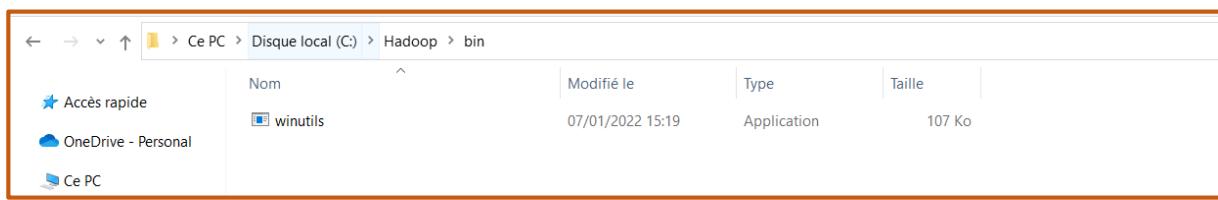
A screenshot of a web browser displaying the Apache Spark download page at spark.apache.org/downloads.html. The page has a blue header with the Apache Spark logo and navigation links for Download, Libraries, Documentation, Examples, Community, Developers, and Apache Software Foundation. The main content area shows a form for downloading Apache Spark 3.0.3 (Jun 23 2021) pre-built for Apache Hadoop 2.7. An orange arrow points from the word "Version" to the dropdown menu where "3.0.3 (Jun 23 2021)" is selected. Below the form, there's a section for "Link with Spark" with Maven dependency coordinates, "Installing with PyPi" (mentioning PySpark), "Release notes for stable releases" (listing Spark 3.2.0, 3.1.2, and 3.0.3), and "Archived releases". To the right, there's a sidebar with "Latest News" (listing releases from Oct 13, 2021, to May 17, 2021), a banner for APACHECON 2021, and sections for "Built-in Libraries" (SQL and DataFrames, Spark Streaming, MLlib, GraphX) and "Third-Party Projects".



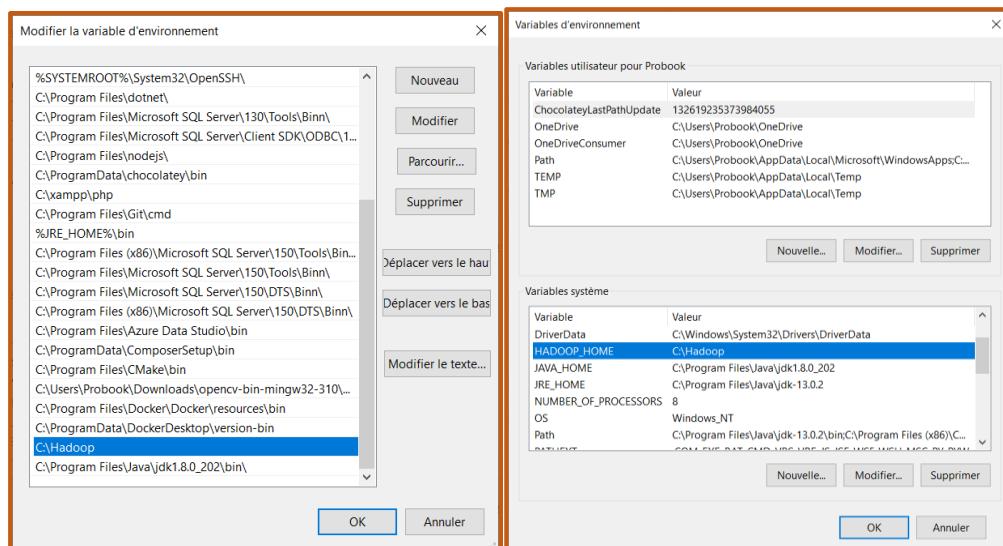


En effet, il est nécessaire d'ajouter certaines composantes afin que l'installation s'effectue avec succès :

#### ⚙️ L'ajout de fichier « winutils »



#### ⚙️ L'ajout de chemin de dossier « Hadoop » à la variable d'environnement :



#### ⚙️ L'ajout de fichier jar : mongo-spark-connector

```
PS C:\Spark> .\bin\pyspark --packages org.mongodb.spark: mongo-spark-connector_2.12-2.4.0
```



Après l'installation d'apache Spark, python et de Java developpement Kit (JDK), on lance la commande suivante :

```
PS C:\Spark> .\bin\pyspark
22/01/14 03:30:20 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Using Spark's default log4j profile: org/apache/spark/log4j-defaults.properties
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
Spark context Web UI available at http://DESKTOP-7N0G0OF.mshome.net:4040
Spark context available as 'sc' (master = local[*], app id = local-1642127427784).
Spark session available as 'spark'.
Welcome to

    ____
   / _ \_  ___ _ _ _ / _ \
  \ \ / / _ \ \ \ / \ \_ \
 / / / \ \ \ / / / \ \_ \
 /_/
Using Scala version 2.12.10 (Java HotSpot(TM) 64-Bit Server VM, Java 1.8.0_202)
Type in expressions to have them evaluated.
Type :help for more information.

scala> 22/01/14 03:30:37 WARN ProcfsMetricsGetter: Exception when trying to compute pagesize, as a result reporting of ProcessTree metrics is stopped
```

### PySpark :

PySpark est une bibliothèque de python. Il représente une collaboration entre Apache Spark et python.



L'installation de PySpark est réalisée par la commande suivante :

```
pip install pyspark
```

Lancement de PySpark :

```
PS C:\Spark> .\bin\pyspark
Python 3.9.4 (tags/v3.9.4:1f2e308, Apr  6 2021, 13:40:21) [MSC v.1928 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
22/01/14 03:27:29 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Using Spark's default log4j profile: org/apache/spark/log4j-defaults.properties
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
Welcome to

    ____
   / _ \_  ___ _ _ _ / _ \
  \ \ / / _ \ \ \ / \ \_ \
 / / / \ \ \ / / / \ \_ \
 /_/
Using Python version 3.9.4 (tags/v3.9.4:1f2e308, Apr  6 2021 13:40:21)
SparkSession available as 'spark'.
>>>
```

### c. Connexion à la base de données via « Apache Spark »

Afin de manipuler les données et effectuer les traitements et les analyses désirés, nous serons obligées d'établir une connexion entre la base de données et pyspark. Pour se faire, on ouvre une session Spark en faisant appel au module « SparkSession »

```
>>> spark = SparkSession.builder.appName("SparkApplication") \
...     .config("spark.mongodb.input.uri", "mongodb://localhost:27017/ScrapingData.articles") \
...     .config("spark.mongodb.output.uri", "mongodb://localhost:27017/ScrapingData.articles") \
...     .getOrCreate()
>>>
```

## d. Chargement des données

Une fois la connexion est ouverte, on peut charger la collection « articles » dans un DataFrame df.

Visualisation de schéma :

```
>>> df = spark.read.format("com.mongodb.spark.sql.DefaultSource").option("uri", "mongodb://localhost:27017/ScrapingData.articles").load()
22/01/14 15:18:59 WARN MongoInferSchema: Field 'abstract' contains conflicting types converting to StringType
22/01/14 15:18:59 WARN MongoInferSchema: Field 'authors_university' contains conflicting types converting to StringType
|-- authors_name: array (nullable = true)
|   |-- element: string (containsNull = true)
|-- authors_university: string (nullable = true)
|-- citation: integer (nullable = true)
|-- date_publication: string (nullable = true)
|-- doi: string (nullable = true)
|-- downloadCount: integer (nullable = true)
|-- impact_factor: double (nullable = true)
|-- indexation: integer (nullable = true)
|-- issn: string (nullable = true)
|-- publisher: string (nullable = true)
|-- references: string (nullable = true)
|-- title: string (nullable = true)
|-- topic: string (nullable = true)
```

## e. Traitement des données

L’analyse des données représentent la phase la plus intéressante dans ce projet. Pour ceci, nous avons mis en lumière certains critères.

### 1. Nombre de publications par année :

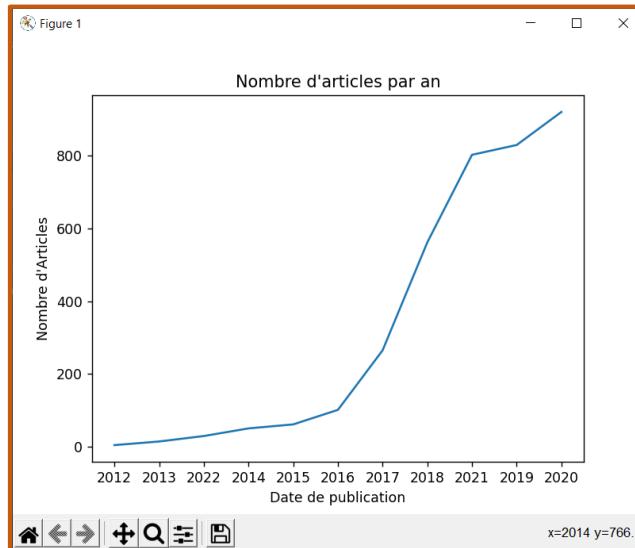
Nous allons afficher le nombre de publication scientifique réalisé par les chercheurs pendant la période de 2012 et 2021.

#### Développement :

```
>>> #----- Les articles publiées par an
>>> import matplotlib.pyplot as plt
>>> import numpy as np
>>>
>>> df = spark.read.format("com.mongodb.spark.sql.DefaultSource").option("uri", "mongodb://localhost:27017/ScrapingData.articles").load()
22/01/14 15:34:24 WARN MongoInferSchema: Field 'abstract' contains conflicting types converting to StringType
22/01/14 15:34:24 WARN MongoInferSchema: Field 'authors_university' contains conflicting types converting to StringType
22/01/14 15:34:24 WARN MongoInferSchema: Field 'date_publication' contains conflicting types converting to StringType
22/01/14 15:34:24 WARN MongoInferSchema: Field 'references' contains conflicting types converting to StringType
>>>
>>> #---- Elimination de tous les articles et les dates nulles
>>> df = df[df['title'] != ""]
>>> df = df[df['date_publication'] != ""]
>>> group1 = df.groupBy("date_publication").count().sort("count", ascending=True)
>>> group1.show()
+-----+---+
|date_publication|count|
+-----+---+
|      2012|    4|
|      2013|   14|
|      2022|   29|
|      2014|   50|
|      2015|   61|
|      2016|  101|
>>> plt.xticks(np.arange(min(x), max(x)+1, 1.0))
```

```
>>> #plt.bar(x, y)
>>> plt.title("Nombre d'articles par an")
Text(0.5, 1.0, "Nombre d'articles par an")
>>> plt.xlabel("Date de publication")
Text(0.5, 0, 'Date de publication')
>>> plt.ylabel("Nombre d'Articles")
Text(0, 0.5, "Nombre d'Articles")
>>> plt.show()
```

### Résultat :



### Interprétation du résultat :

D'après cette figure nous voyons que la recherche scientifique réalisé pour les sujets « Bitcoin » et « Blockchain » a connu une croissance remarquable dès l'années 2016.

## 2. Nombre d'article publiée pour chaque sujet en fonction de la date de publication :

Pour illustrer l'évolution de la recherche dans les deux sujets nous avons implémenté le code ci-dessous :

### Développement :

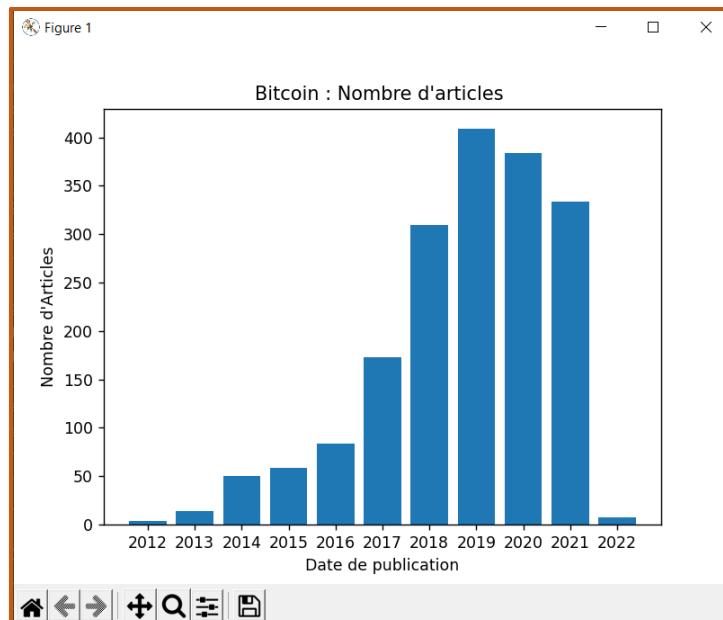
Pour choisir le sujet, il suffit juste de modifier `df['topic'] = 'nom_sujet'`

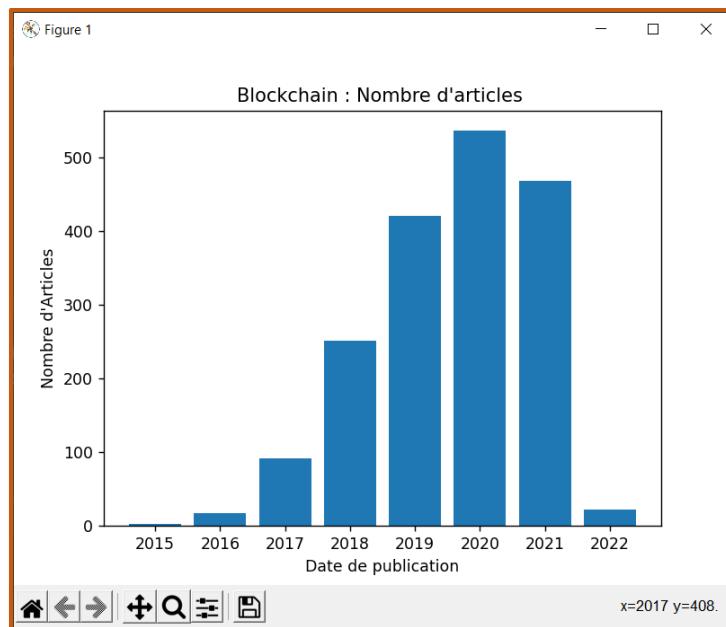
```
>>> #----- le nombre de publication pour un sujet = "Bitcoin" par rapport aux années
>>> import matplotlib.pyplot as plt
>>> import numpy as np
>>>
>>> df = spark.read.format("com.mongodb.spark.sql.DefaultSource").option("uri", "mongodb://localhost:27017/ScrapingData.articles").load()
22/01/14 16:12:05 WARN MongoInferSchema: Field 'abstract' contains conflicting types converting to StringType
22/01/14 16:12:05 WARN MongoInferSchema: Field 'authors_university' contains conflicting types converting to StringType
22/01/14 16:12:05 WARN MongoInferSchema: Field 'date_publication' contains conflicting types converting to StringType
22/01/14 16:12:05 WARN MongoInferSchema: Field 'references' contains conflicting types converting to StringType
>>>
>>> #----- Elimination de toutes articles nulles
>>> df = df[df['title'] != ""]
>>> df = df[df['date_publication'] != 0]
>>> df = df[df['topic'] == "Bitcoin"]
>>> group1 = df.groupBy("date_publication").count().sort("date_publication", ascending=True)
```

```
>>> group1.show()
+-----+----+
|date_publication|count|
+-----+----+
|      2012|    4|
|      2013|   14|
|      2014|   50|
|      2015|   59|
|      2016|   84|
|      2017| 173|
|      2018| 310|
|      2019| 409|
|      2020| 384|
|      2021| 334|
|      2022|    7|
+-----+----+
>>> group1 = group1.toPandas()
>>> x = group1['date_publication'].values.tolist()
>>> y = group1['count'].values.tolist()
```

```
>>> #---- Ajustement de l'echelle
>>> plt.xticks(np.arange(min(x), max(x)+1, 1.0))
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: can only concatenate str (not "int") to str
>>> plt.bar(x, y)
<BarContainer object of 11 artists>
>>> plt.title("Bitcoin : Nombre d'articles ")
Text(0.5, 1.0, "Bitcoin : Nombre d'articles ")
>>> plt.xlabel("Date de publication")
Text(0.5, 0, 'Date de publication')
>>> plt.ylabel("Nombre d'Articles")
Text(0, 0.5, "Nombre d'Articles")
>>> plt.show()
```

### Résultat :





### Interprétation du résultat :

Cette figure illustre l'évolution d'intérêt des chercheurs dans le domaine « Bitcoin » et aussi « Blockchain ». On remarque que 2019 a connu le maximum des publications pour le « bitcoin », par contre le blockchain a eu son top en 2020.

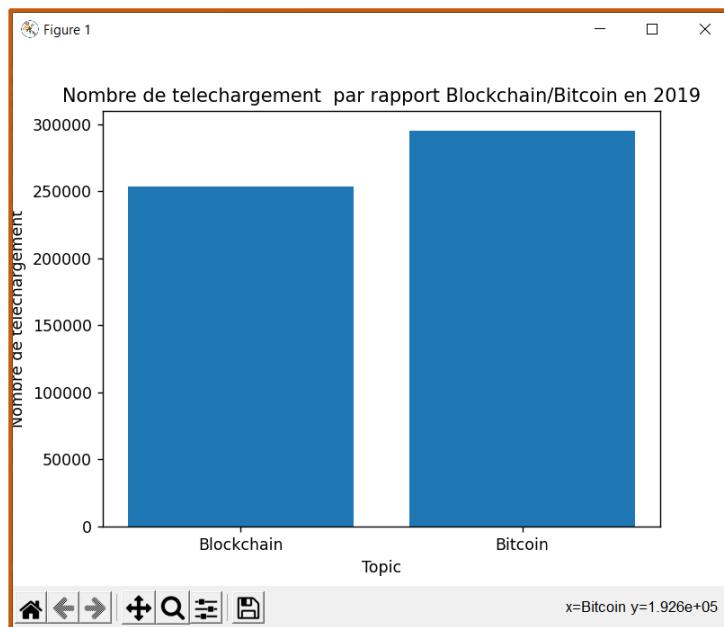
### **3. Comparaison de nombre de téléchargement des articles des deux sujets**

Pour savoir le sujet qui attire plus des lecteurs nous avons exécuté ce code.

#### Développement :

```
>>> #----- le nombre de downloads d'une article en 2019
>>> import matplotlib.pyplot as plt
>>> import numpy as np
>>>
>>> df = spark.read.format("com.mongodb.spark.sql.DefaultSource").option("uri", "mongodb://localhost:27017/ScrapingData.articles").load()
22/01/14 16:56:01 WARN MongoInferSchema: Field 'abstract' contains conflicting types converting to StringType
22/01/14 16:56:01 WARN MongoInferSchema: Field 'authors_university' contains conflicting types converting to StringType
22/01/14 16:56:01 WARN MongoInferSchema: Field 'date_publication' contains conflicting types converting to StringType
22/01/14 16:56:01 WARN MongoInferSchema: Field 'references' contains conflicting types converting to StringType
>>>
>>> #----- Elimination de toutes articles nulles
>>> df = df[df['title'] != ""]
>>> #----- Fixer la date
>>> df = df[df['date_publication'] == 2019]
>>> group1 = df.groupby(["topic"]).sum()
  File "<stdin>", line 1, in <module>
TypeError: can only concatenate str (not "int") to str
>>> plt.bar(x, y)
<BarContainer object of 2 artists>
>>> plt.title("Nombre de telechargement par rapport aux sujets")
Text(0.5, 1.0, 'Nombre de telechargement par rapport aux sujets')
>>> plt.xlabel("Topic")
Text(0.5, 0, 'Topic')
>>> plt.ylabel("Nombre de telechargement")
Text(0, 0.5, 'Nombre de telechargement')
>>> plt.show()
```

### Résultat :



### Interprétation du résultat :

En 2019, les articles qui traite le sujet « Bitcoin » a eu presque 300 000 téléchargement, qui est plus grand que celle de « Blockchain » qui n'a pas dépassé le seuil de 250 000 téléchargement.

## 4. Nombre de publications scientifiques par pays

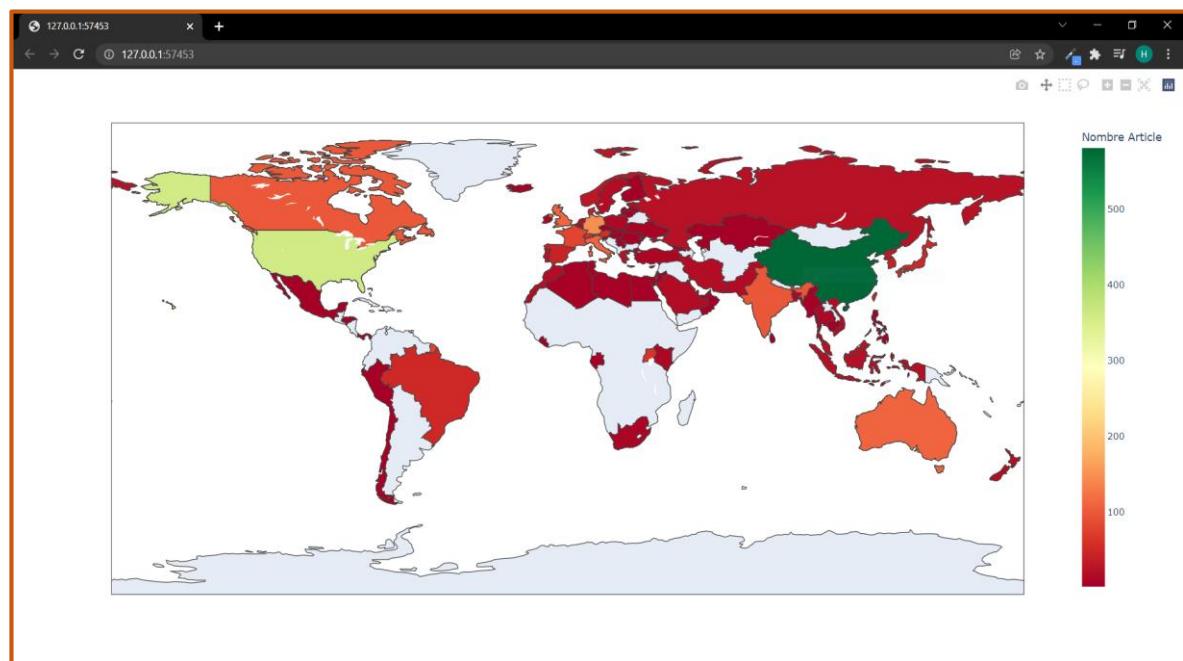
Dans ce stade, nous allons jeter un coup d'œil sur la recherche scientifique à l'échelle internationale afin de savoir les grands concurrents, on lance ce code.

### Développement :

```
>>> #----- le nombre de publication par pays
>>> import pandas as pd
>>> import pycountry as pycountry
>>> import plotly.express as px
>>>
>>> df = spark.read.format("com.mongodb.spark.sql.DefaultSource").option("uri", "mongodb://localhost:27017/ScrapingData.articles").load()
22/01/14 17:25:35 WARN MongoInferSchema: Field 'abstract' contains conflicting types converting to StringType
22/01/14 17:25:35 WARN MongoInferSchema: Field 'authors_university' contains conflicting types converting to StringType
22/01/14 17:25:35 WARN MongoInferSchema: Field 'date_publication' contains conflicting types converting to StringType
22/01/14 17:25:35 WARN MongoInferSchema: Field 'references' contains conflicting types converting to StringType
>>> df = df.toPandas()
>>>
>>> #---- Elimination de toutes articles nulles
>>> df = df[df['title'] != ""]
>>> #---- Elimination de toutes articles dont le pays est nul
>>> df = df[df['authors_country'] != ""]
>>> x = df[['authors_country']]
>>>
>>> #Mot: Country - Cle : count
>>> y = pd.DataFrame({'authors_country': x['authors_country'].apply(lambda x: pd.Series(list(set(x.split(','))))).stack().tolist()})
>>> y['count'] = 0
>>>
>>> counter_country = y.groupby("authors_country").count().sort_values(by=["count"], ascending=True).reset_index()
>>> list_counter_country = counter_country.values.tolist()
...
    try:
...
        country_info = pycountry.countries.search_fuzzy(country)
...
        country_code = country_info[0].alpha_3
```

```
...     country_code = country_info[0].alpha_3
...
...     countries_codes.update({country: country_code})
...
... except:
...     #print('Error : Country', country)
...     if country == "P.R.China" or country == "China Beijing" or country == "P.R. China":
...         countries_codes.update({country: 'CHN'})
...
...     if country == "U.S.A":
...         countries_codes.update({country: 'USA'})
...
...     if country == "Morroco":
...         countries_codes.update({country: 'MAR'})
...
...
>>> for k, v in countries_codes.items():
...     df_counter_country.loc[df_counter_country.Country == k], 'iso_alpha'] = v
...
>>> fig = px.choropleth(data_frame=df_counter_country, locations='iso_alpha', color="Nombre Article", hover_name="Country", color_continuous_scale='RdYlGn')
>>> fig.show()
>>>
```

### Résultat :



### Interprétation du résultat :

A partir de la figure ci-dessus, nous constatons la distribution des publications qui correspond à chaque pays. On peut dire que la Chine et le pays le plus actif puis l'Amérique, Canada, l'Australie et l'Inde, par contre les pays africains comme le Maroc, l'Algérie, La Libye, la Tunisie, l'Egypte n'en pas encore assez de publication dans ces domaines.

## VI. ANALYSE BI & VISUALISATION

### a. Description

A ce stade, nous arrivons à la phase d'analyse BI et la visualisation des données en misant en place un entrepôt de données dans une base de données relationnelles MySQL à l'aide de PDI (Pentaho Data Integration). L'analyse et le reporting sera réalisé via la plateforme Power BI.

### b. Installation des outils

#### Pentaho Data Integration

Pentaho est une solution d'informatique décisionnelle open source entièrement développée en Java. Elle porte sur toute la chaîne décisionnelle et utilise différents outils et composants :



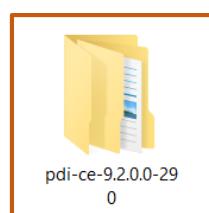
- Pour la collecte et l'intégration : les outils d'ETL Kettle ou Mondrian,
- Pour la diffusion : un serveur d'application JBoss ou TOMCAT,
- Pour la présentation : JFreeReport, BIRT ou encore JasperReport.

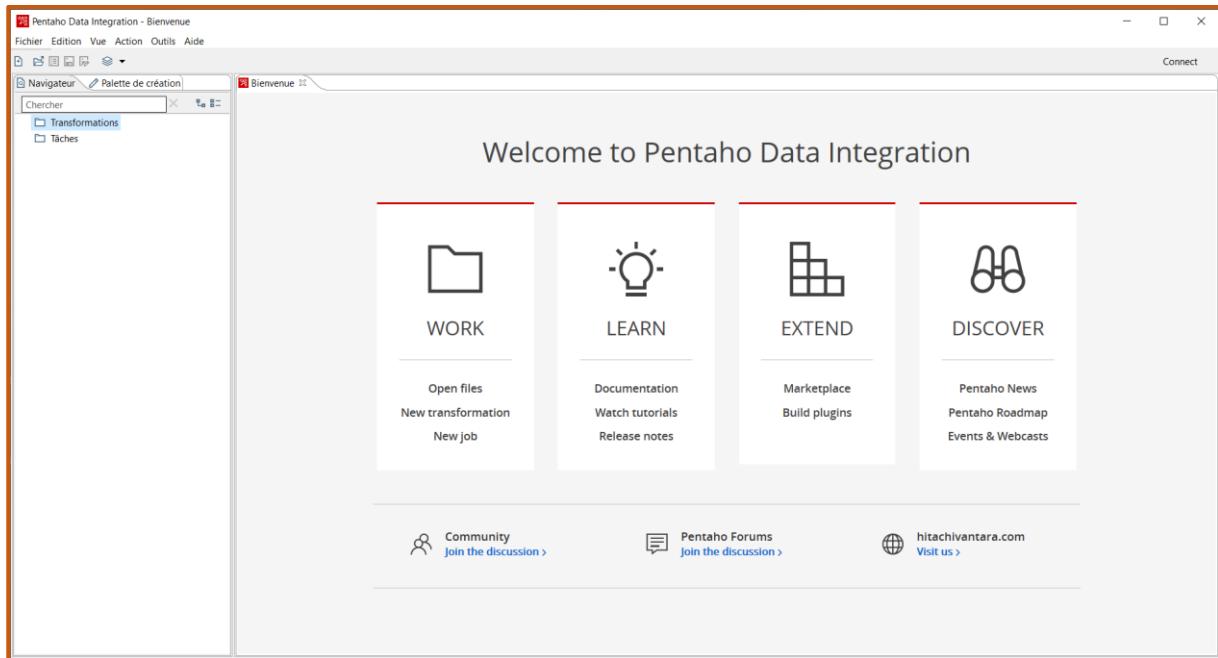
L'installation de PDI est réalisé via le site web suivant :

<https://sourceforge.net/projects/pentaho/>

The screenshot shows the SourceForge project page for 'Pentaho from Hitachi Vantara'. The page includes the following information:

- Project Summary:** Pentaho from Hitachi Vantara
- Downloads:** 7,478 This Week
- Last Update:** 2021-10-20
- Rating:** ★★★★☆ 67 Reviews
- Operating Systems:** Windows, Mac, Linux
- Links:** Summary, Files, Reviews, Support, Wiki, News
- Description:** End to end data integration and analytics platform
- Advertisements:**
  - Marketing Analytics Course (coursera.org)
  - chrome enterprise (Install & Configure Chrome)
  - Get latest updates about Open Source Projects, Conferences and News.



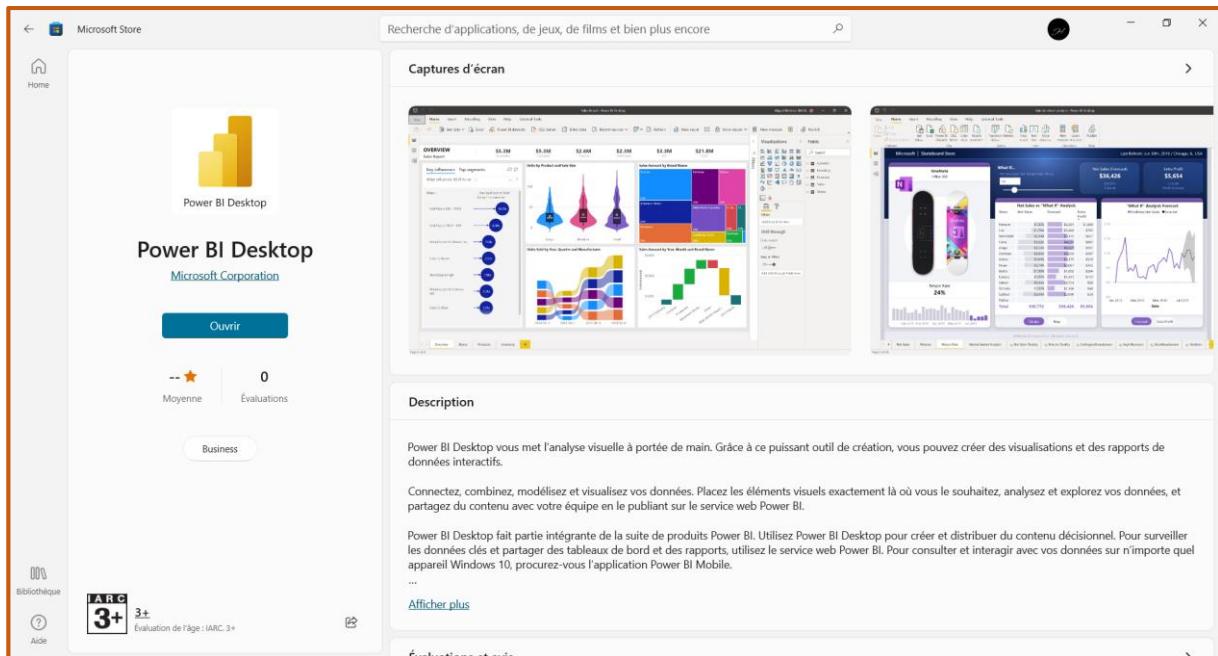


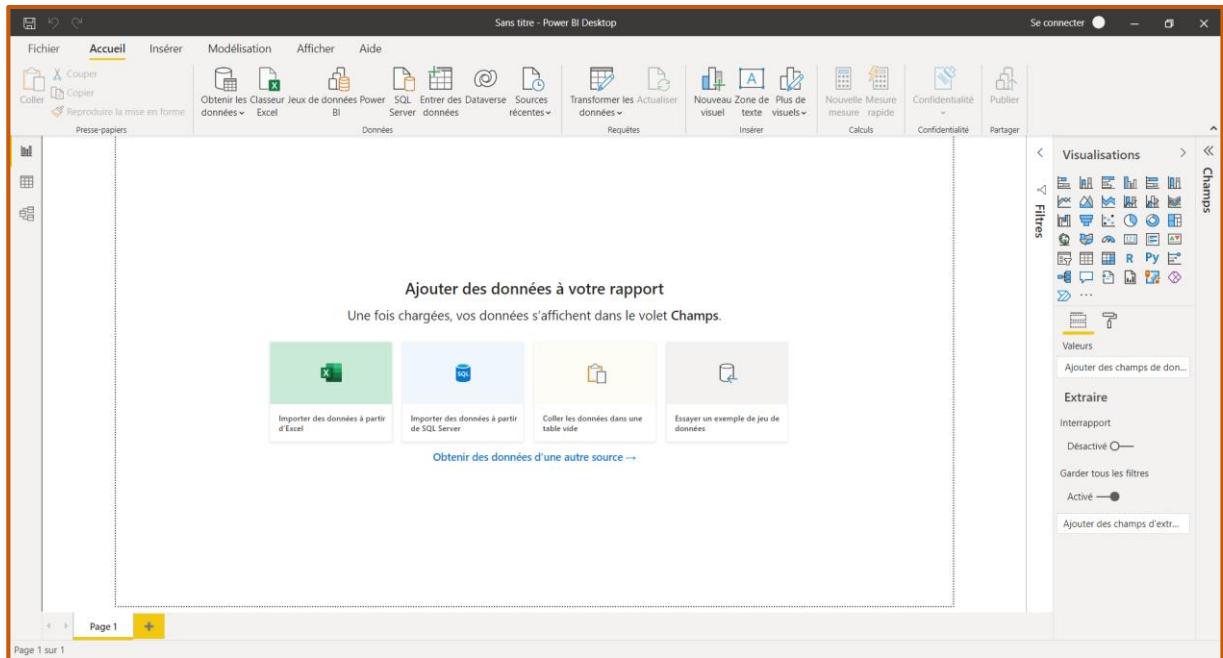
## Microsoft Power BI :



Power BI Desktop est une application gratuite fournie par Microsoft qui s'installe sur un ordinateur local et permet de se connecter à des données, de les transformer et de les visualiser.

L'installation de logiciel BI est accessible via ce lien :  
<https://powerbi.microsoft.com/en-us/desktop/>



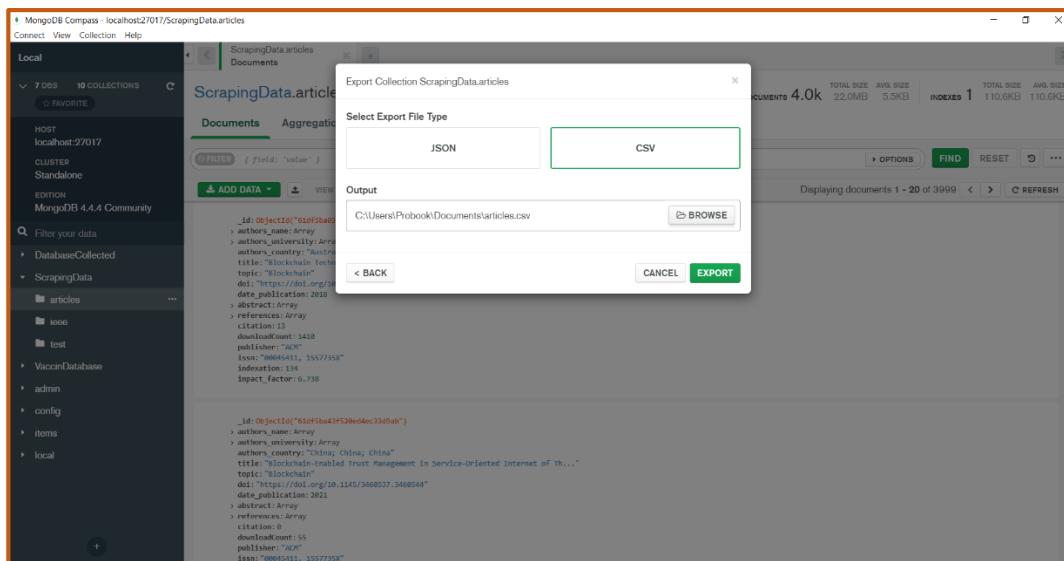


### c. Schéma en étoile

Il s'agit du schéma le plus simple et le plus efficace dans un entrepôt de données. Une table de faits au centre entourée de tables de dimensions multiples ressemble à une étoile.

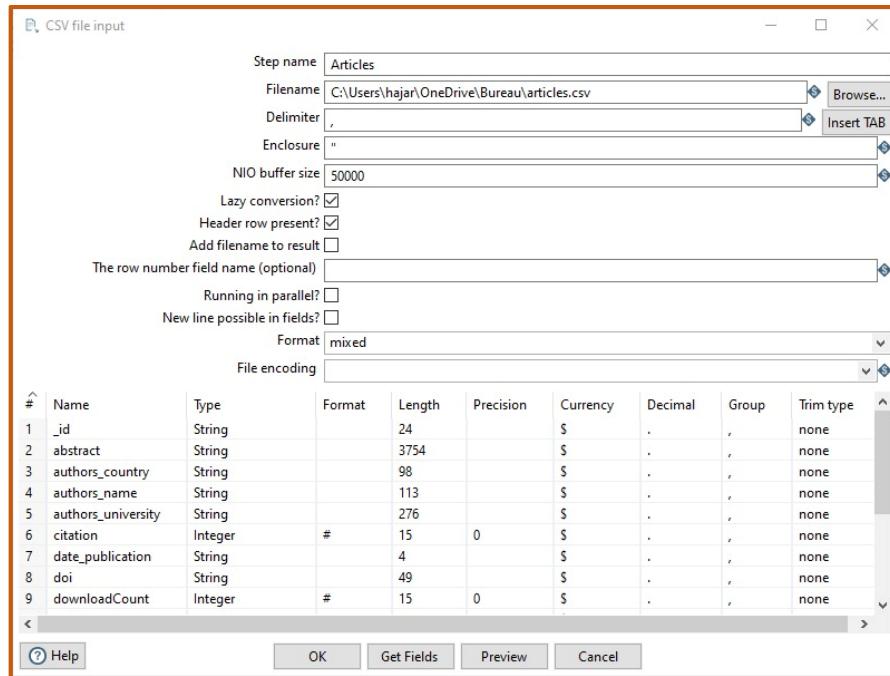
La table de faits maintient des relations un-à-plusieurs avec toutes les tables de dimension. Chaque ligne d'une table de faits est associée à ses lignes de table de dimension avec une référence de clé étrangère.

### Exportation des données sous forme d'un fichier csv :



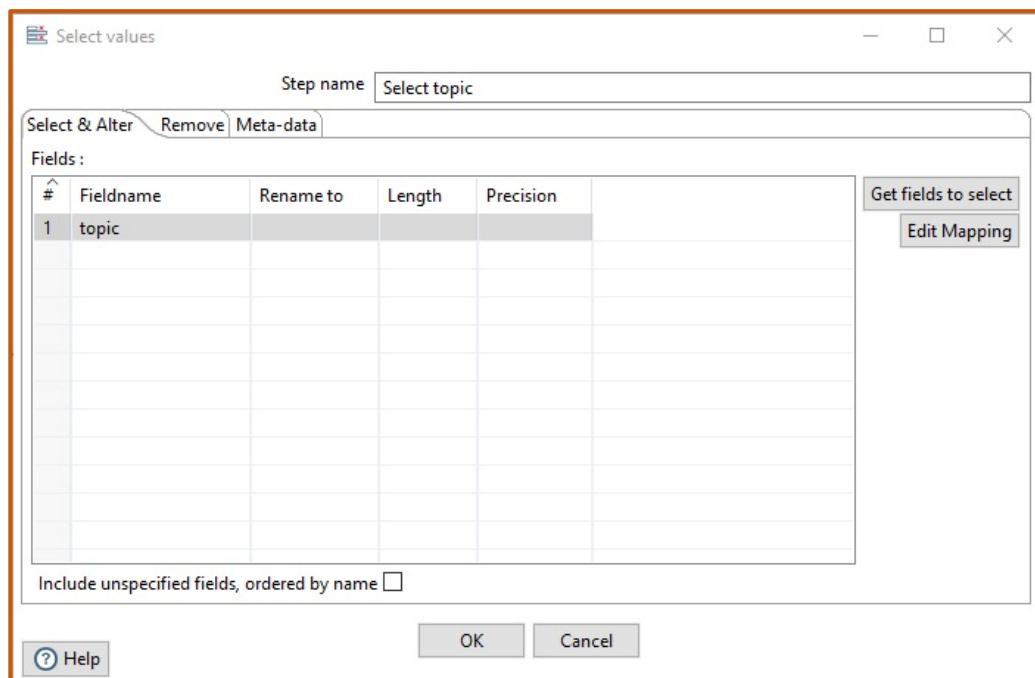
### Importation de la base de données :

Pour générer les dimensions on importe la base de données depuis le fichier csv exporté.

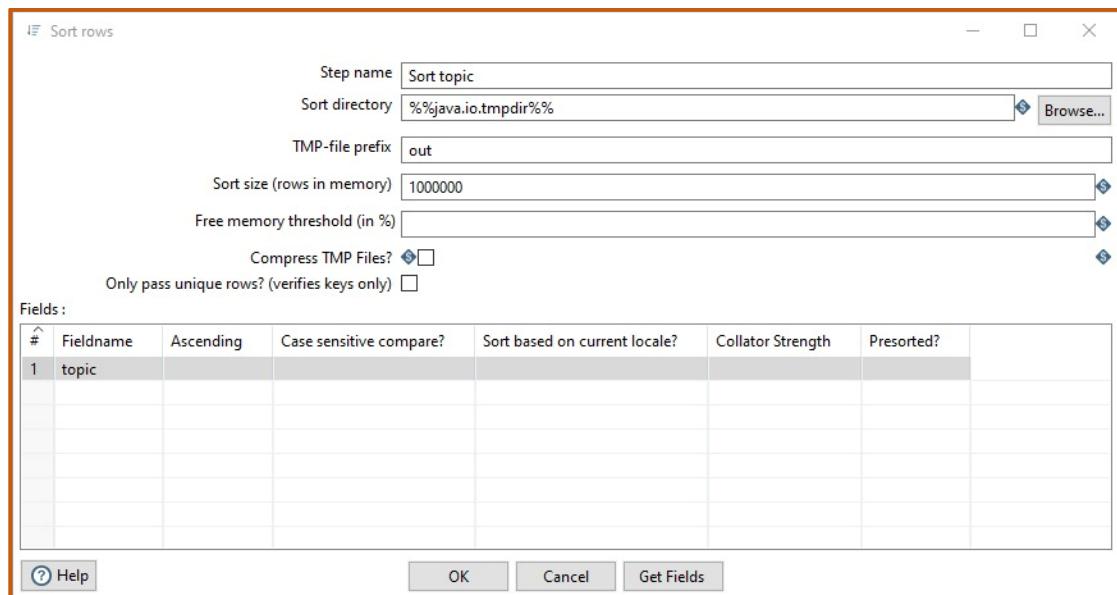


### Création des composants :

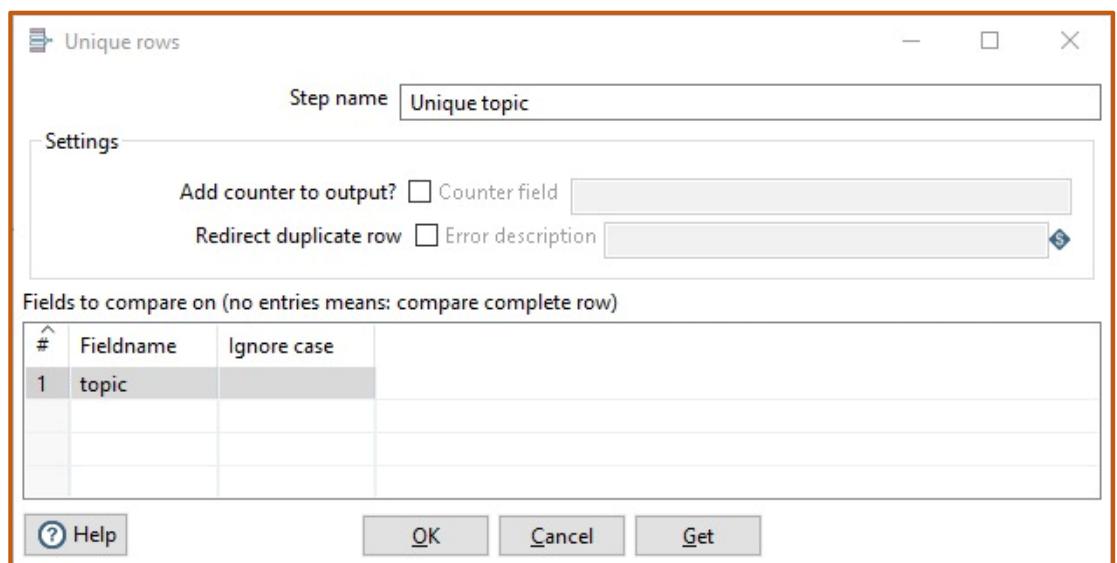
- On glisse sur « Select values » dans la fenêtre de transformation afin de sélectionner les valeurs à partir du fichier csv.



⚙ « Sort Rows »

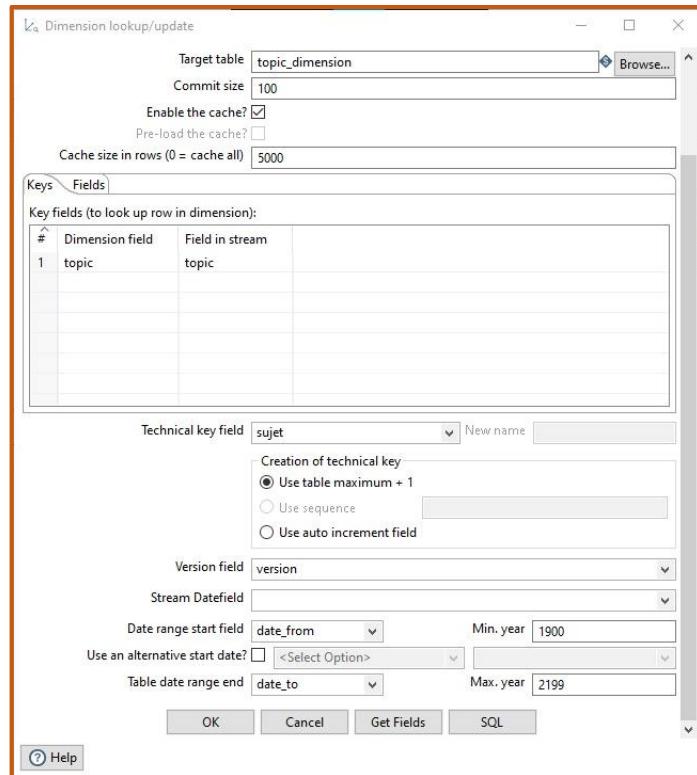


⚙ « Unique Rows »

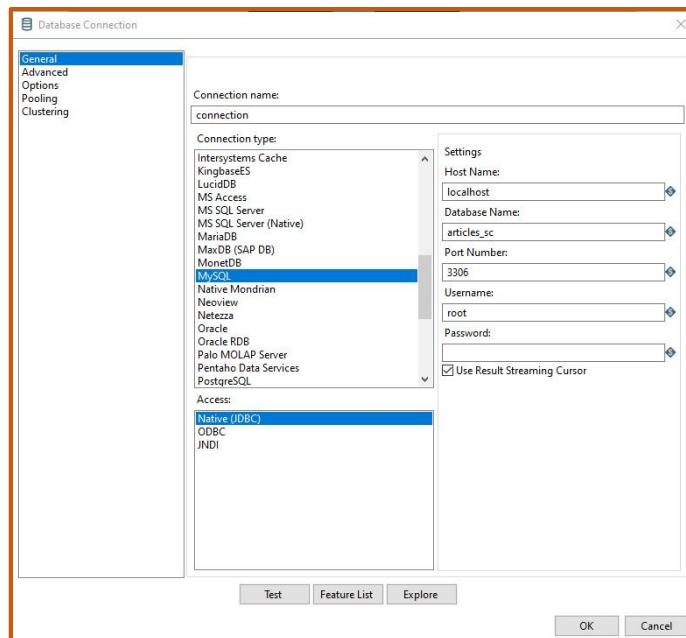


Chargement dans la base de données :

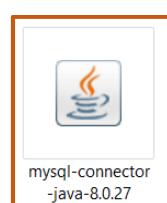
On clique sur « Dimension Lookup/Update » afin d'effectuer le chargement dans la base de données.



Création de connexion à la base de données MySQL.

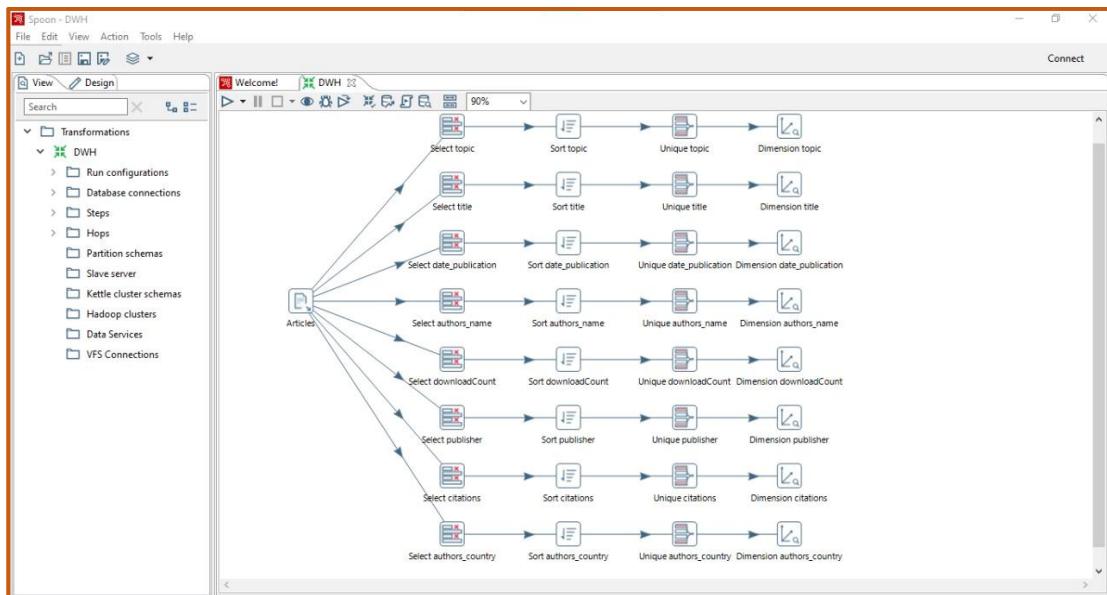


Etablir la connexion à la base de données MySQL demande un ajout d'un fichier jar.



Une fois la configuration est terminée, nous exécutons le code SQL sur la base de données pour créer la table de dimension « topic »

### Exécution des transformations :



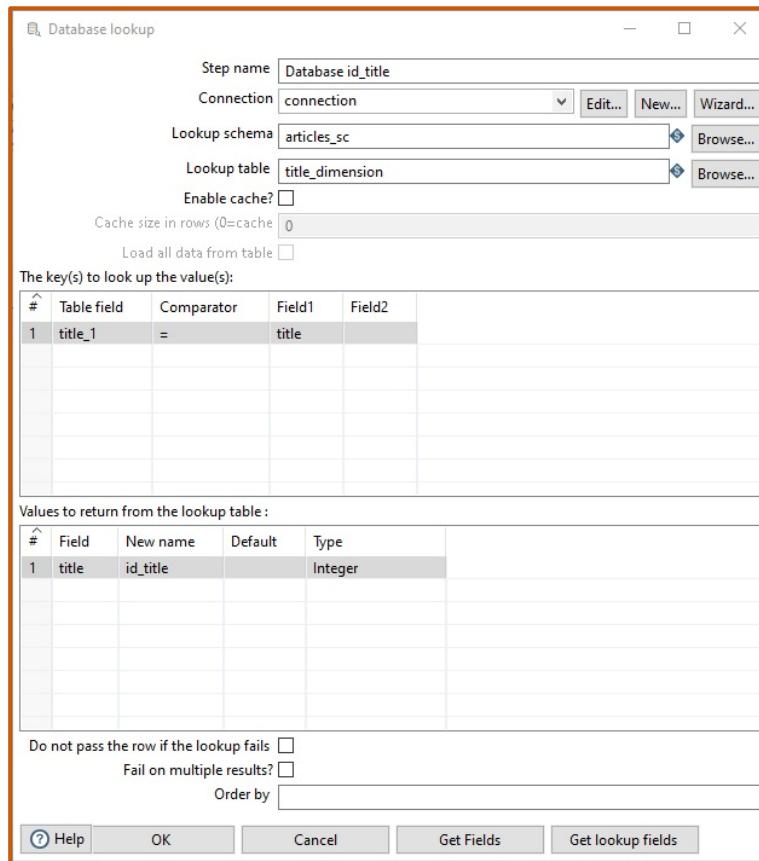
### Génération de la table de fait :



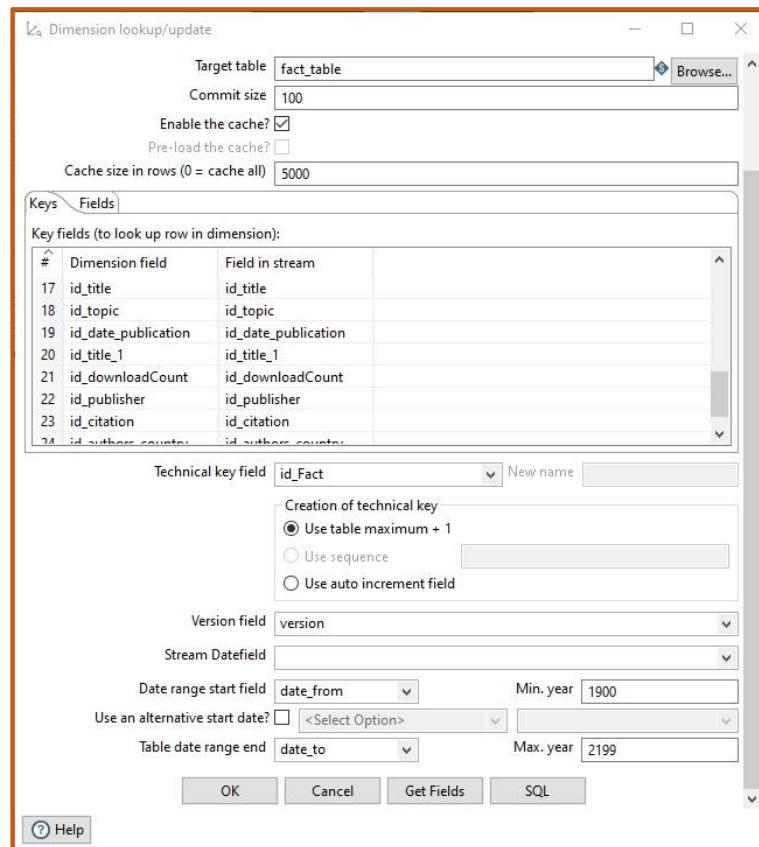
« Select values »  
Nous allons sélectionner toutes les valeurs.

#	Fieldname	Rename to	Length	Precision	
1	_id				
2	abstract				
3	authors_country				
4	authors_name				
5	authors_university				
6	citation				
7	date_publication				
8	doi				
9	downloadCount				
10	impact_factor				
11	indexation				
12	issn				
13	publisher				
14	references				
15	title				
16	topic				

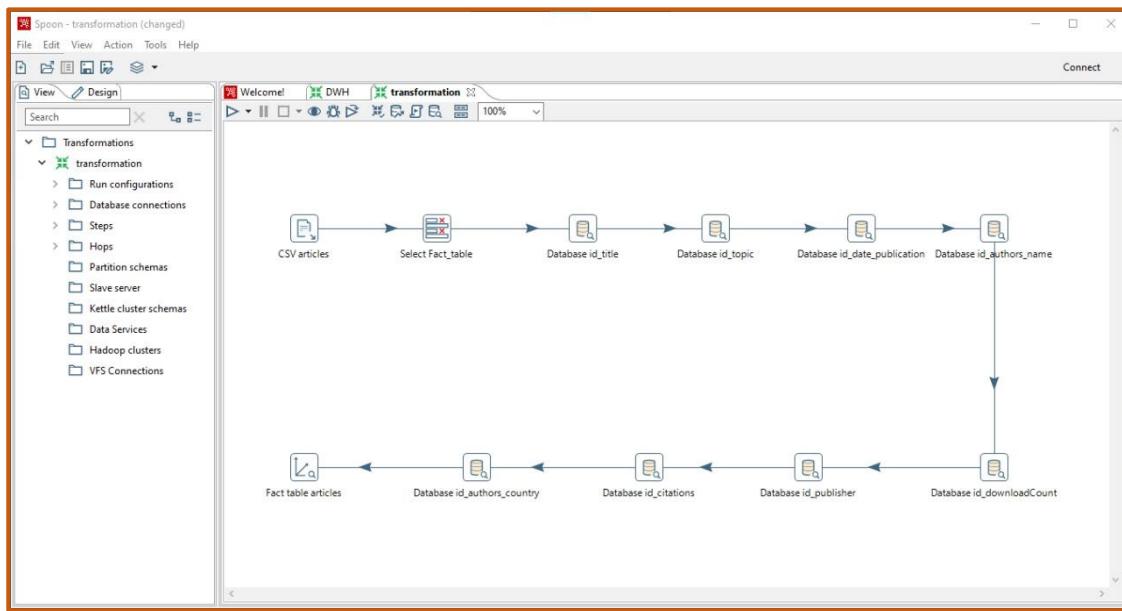
## ⌚ Recherche id



## ⌚ Dimension lookup/update



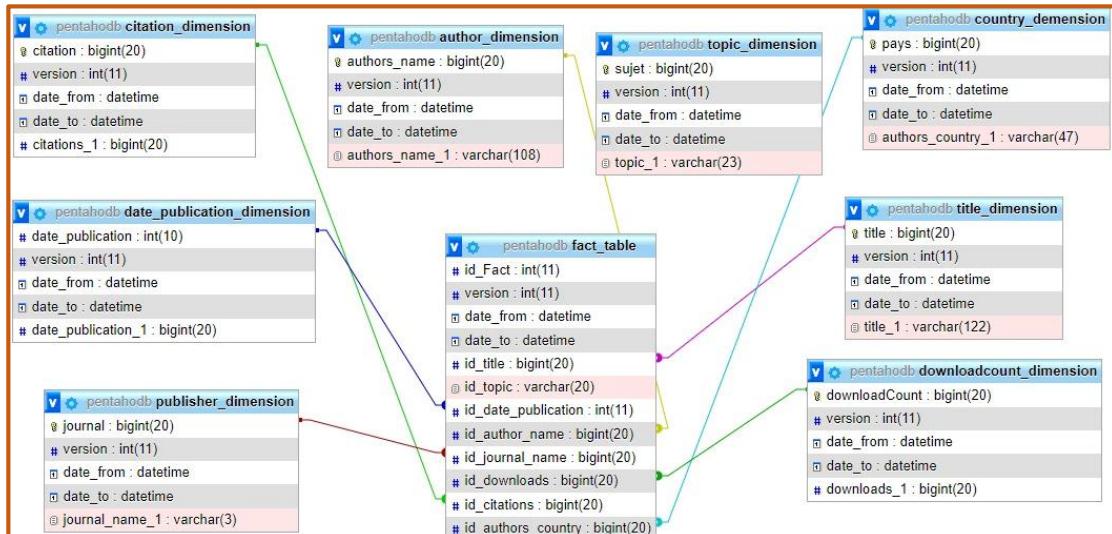
## Exécution de la transformation de la table de fait



## Résultat final dans MySQL

Table	Action	Rows	Type	Collation	Size	Overhead
author_dimension	★ Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_general_ci	16.0 KiB	-
citation_dimension	★ Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_general_ci	64.0 KiB	-
country_dimension	★ Browse Structure Search Insert Empty Drop	2	InnoDB	utf8mb4_general_ci	48.0 KiB	-
date_publication_dimension	★ Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_general_ci	64.0 KiB	-
downloadcount_dimension	★ Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_general_ci	64.0 KiB	-
fact_table	★ Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_general_ci	32.0 KiB	-
publisher_dimension	★ Browse Structure Search Insert Empty Drop	5	InnoDB	utf8mb4_general_ci	48.0 KiB	-
title_dimension	★ Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_general_ci	64.0 KiB	-
topic_dimension	★ Browse Structure Search Insert Empty Drop	5	InnoDB	utf8mb4_general_ci	48.0 KiB	-
9 tables	Sum	12	InnoDB	utf8mb4_general_ci	448.0 KiB	0 B

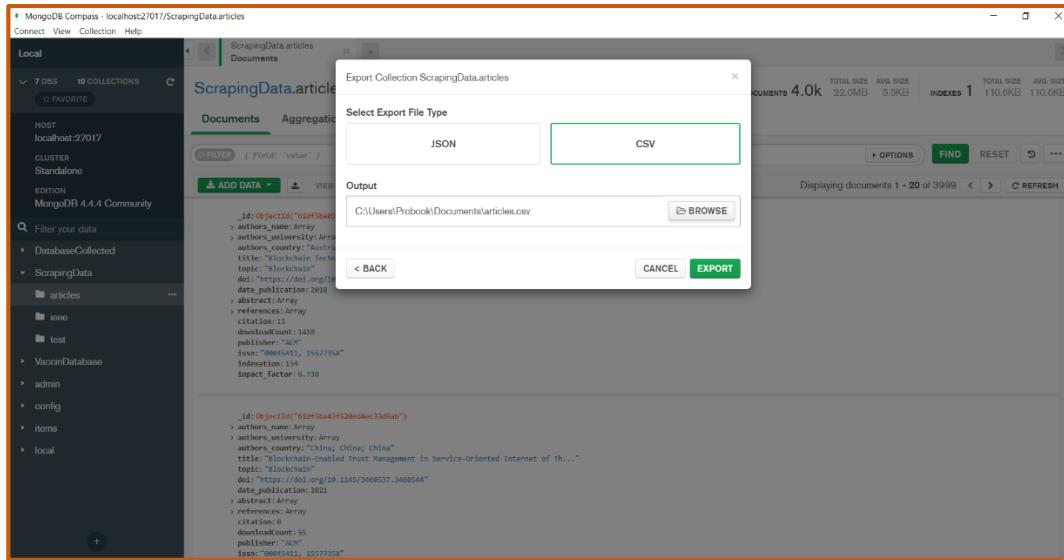
## Résultat final dans MySQL



## d. Visualisation et reporting avec Microsoft Power BI

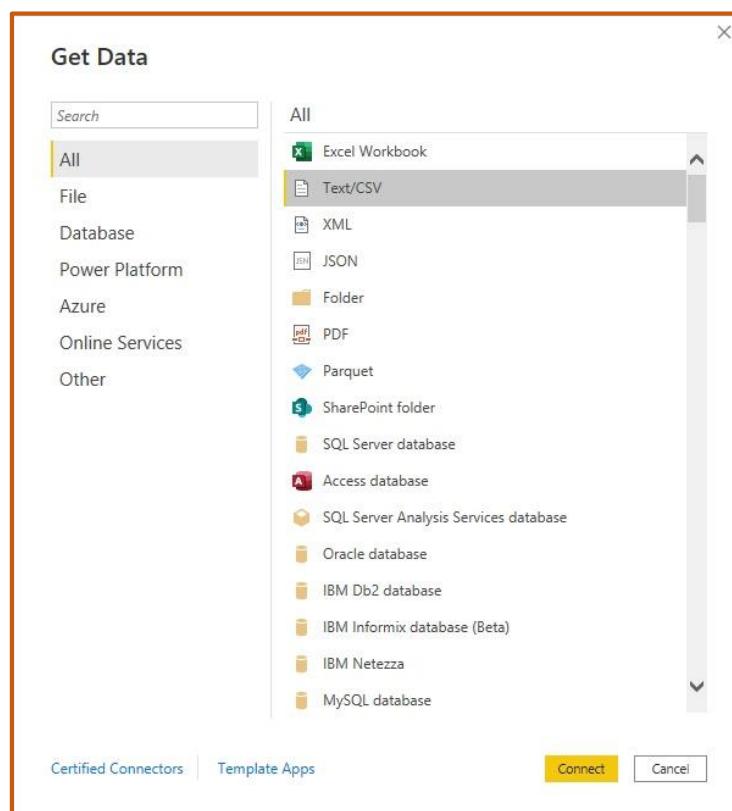
### Etablir la connexion avec la base de données :

- Etape 1 : Exportation du data sous forme d'un fichier csv



- Etape 2 :

Sur Power BI Desktop on importe le fichier csv, en cliquant sur :  
Get Data → Text/csv → Connect



## Préparation des données :

La préparation des données est nécessaire dans le but d'avoir des vrais résultats.

### 1. Transformation des données

Avant de charger les données sur Power BI, on effectue des transformations pour rendre les données plus exploitables.

#### ○ Etape 1

On clique sur "transform data" pour ouvrir "Power Query Editor"

_id	abstract	authors_country	authors_country1
61df5ba03f520ed4ec33d9aa	["The mining and metals industry is a critical component of the global economy."]	Australia; Australia; India; Australia	["Suruchi Ma...
61df5ba93f520ed4ec33d9ab	["With the development of information and computer technology, the...	China; China; China	["Lijun Wei";...
61df5ba93f520ed4ec33d9ac	["A blockchain is a gradually increasing list of "blocks" containing inf...	Malaysia; Malaysia; Malaysia	["Zi Hau Chin";...
61df5baa3f520ed4ec33d9ad	["With the development of Internet technology, all walks of life are...	China; China; Malaysia; Malaysia	["Lei Pan"; "Yi...
61df5bb23f520ed4ec33d9ae	["The Industrial Internet of Things (IIoT) is a typical infrastructure...	China; China; China; China; China	["Jiang Tan";...
61df5bb63f520ed4ec33d9af	["Blockchain is regarded as one of the most promising technologies in th...	China; China; China; China; China	["Shuliguang D...
61df5bb93f520ed4ec33d9b0	["With the rapid evolution of technological, economic, and regulatory env...	Australia; Australia; Germany	["HMN Dilum...
61df5bb93f520ed4ec33d9b1	["One of the major challenges in the healthcare industry is the need to...	United States; United States; United States; United States	["Jhanvi Deva...
61df5bc23f520ed4ec33d9b2	["Blockchain is perceived as a revolutionary technology offering consid...	Sadat Academy for Management Sciences Cairo, Egypt; Egypt	["Dina Salah";...
61df5bc63f520ed4ec33d9b3	["Blockchain is attracting attention from academia and industry."]	Pakistan; Pakistan; Pakistan; Pakistan; Pakistan	["Muhammad I...
61df5bc93f520ed4ec33d9b4	["Stealth address has been an effective way to protect privacy of users...	Kenya; Kenya; Kenya; Kenya; Kenya	["Jia Fan"; "Zh...
61df5bd3f520ed4ec33d9b5	["Several initiatives have been proposed to collect, report, and analyze...	India; India; India	["Nelson Borr...
61df5bd13f520ed4ec33d9b6	["Blockchain is viewed as one of the most promising technologies in the i...	Australia; Australia; Australia	["Ke Wang";...
61df5bd43f520ed4ec33d9b7	["As the underlying technology of cryptocurrency, blockchain is changing...	China; China; China; China	["Pengting Du";...
61df5bd73f520ed4ec33d9b8	["The forgery of certificates is a long-term problem in the academic co...	Egypt; Egypt; Egypt; Egypt; Egypt	["Alley El-Dor...
61df5bd3f520ed4ec33d9b9	["Cloud is widely used for data storage. A user who has uploaded his/h...	India; India; India	["Syed Saud H...
61df5bd3f520ed4ec33d9b0	["No abstract available."]		
61df5be23f520ed4ec33d9b1	["Cloud is widely used for data storage. A user who has uploaded his/h...	India; India; India	["Syed Saud H...
61df5be53f520ed4ec33d9b2	["The emergence of electronic medical records has provided great conveni...	China; China	["Sihua Wu";...
61df5be93f520ed4ec33d9b3	["The emergence of electronic medical records has provided great conveni...	China; China	["Sihua Wu";...

#### ○ Etape 2

On effectue des transformations selon le besoin

Les transformations effectuées dans notre cas :

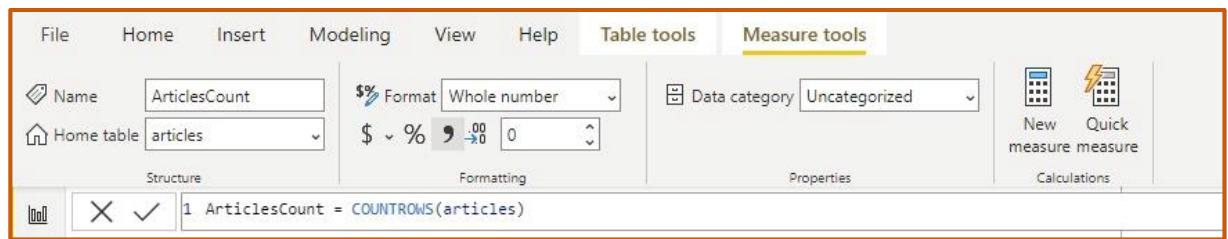
Source	Changed Type6
Promoted Headers	Removed Columns3
Changed Type	Split Column by Position2
Split Column by Delimiter	Changed Type7
Changed Type1	Removed Columns4
Split Column by Delimiter1	Split Column by Position3
Changed Type2	Changed Type8
Removed Columns	Removed Columns5
Split Column by Delimiter2	Split Column by Position4
Changed Type3	Changed Type9
Removed Columns1	Removed Columns6
Split Column by Delimiter3	Split Column by Position5
Changed Type4	Changed Type10
Renamed Columns	Removed Columns7
Split Column by Position	Split Column by Position6
Changed Type5	Changed Type11
Removed Columns2	Removed Columns8
Split Column by Position1	Split Column by Position7
Changed Type6	Changed Type12

## 2. Définition des catégories

Dans cette étape on définit les catégories des colonnes si nécessaires, dans notre cas on définit les colonnes qui correspondent aux pays des auteurs comme "country category".

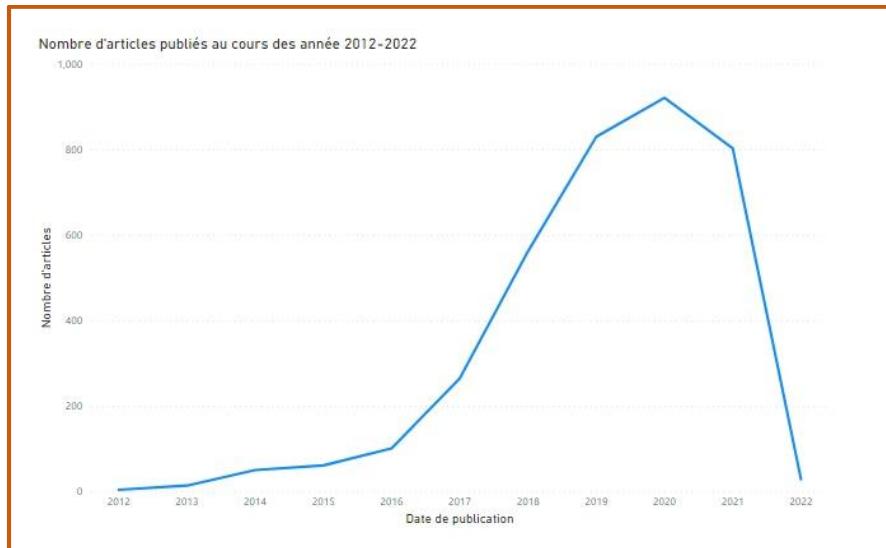
## 3. Ajout des mesures

Dans cette étape on ajoute des mesures si nécessaires, dans notre cas on a ajouté la mesure "*ArticlesCount = COUNTROWS(articles)*" qui correspond aux nombre d'articles publiés.



### Visualisation :

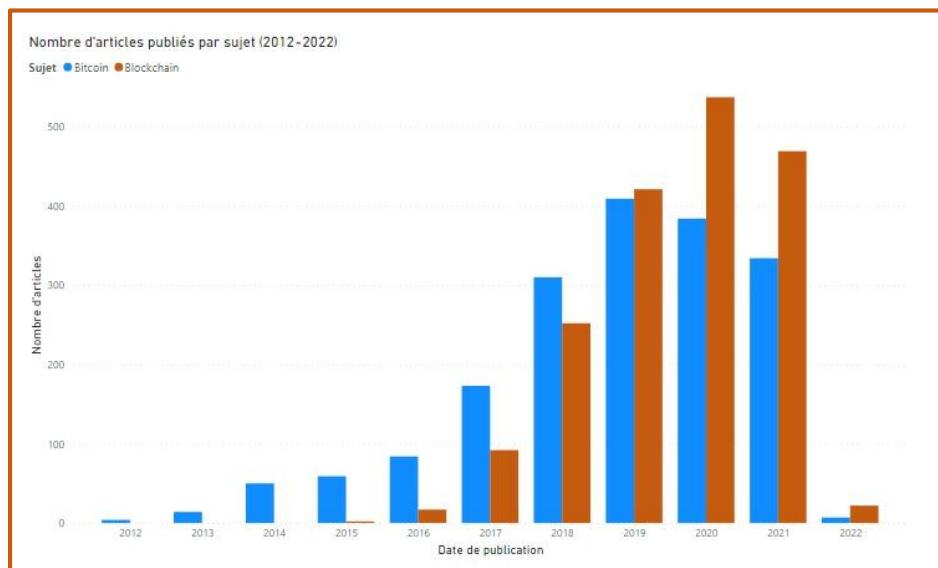
#### 1. Visualisation de nombre de publications au cours des années 2012-2022



### Interprétation :

Au cours des années 2012 – 2015 le bitcoin et la technologie blockchain ont été pauvre au niveau de la recherche scientifique. En revanche, ces deux disciplines ont eu une évolution presque exponentielle à partir de 2016.

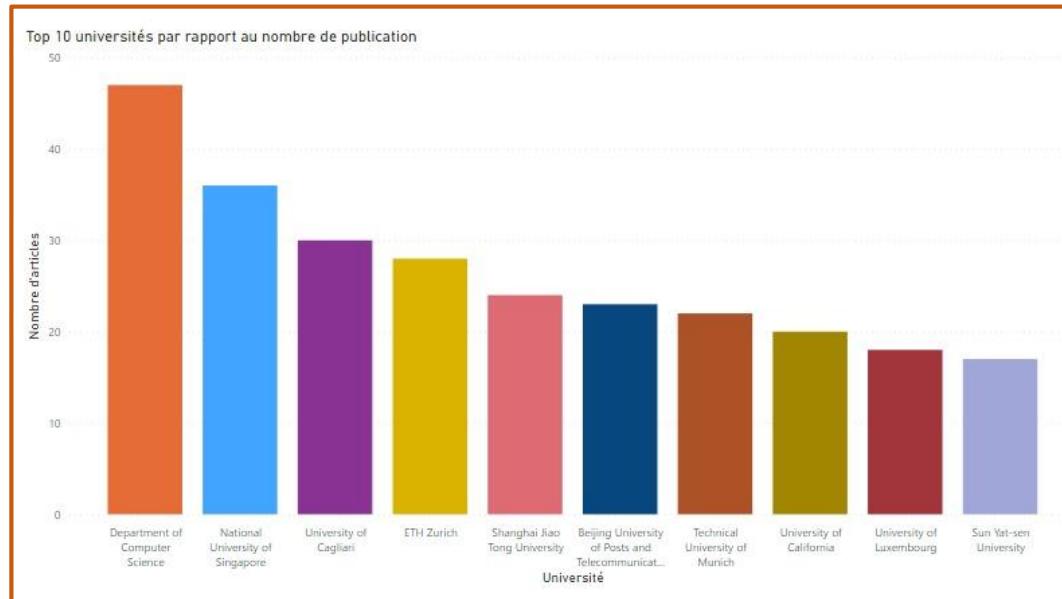
#### 2. Présentation de nombre d'article publiées par sujet entre 2021-2022



### Interprétation :

La croissance des nombres de publication durant ces 10 ans est très visible dans ce graphe pour les deux sujets.

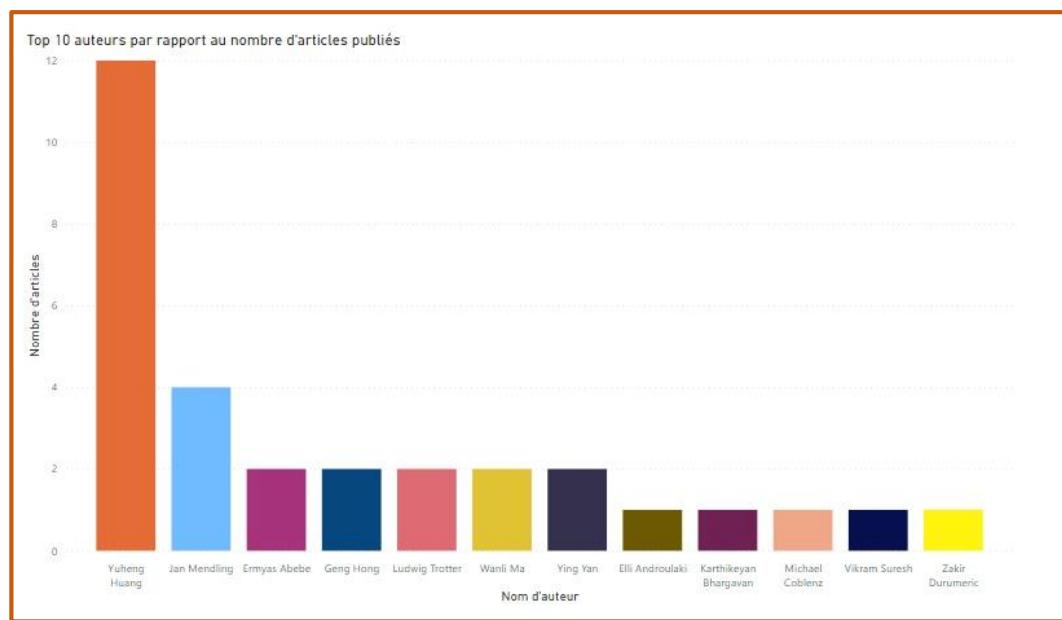
### 3. Top 10 universités par nombre de publication



### Interprétation :

Dans ce graphe nous découvrons le top 10 universités dans le monde dont le « Departement of Computer Science » est le leader.

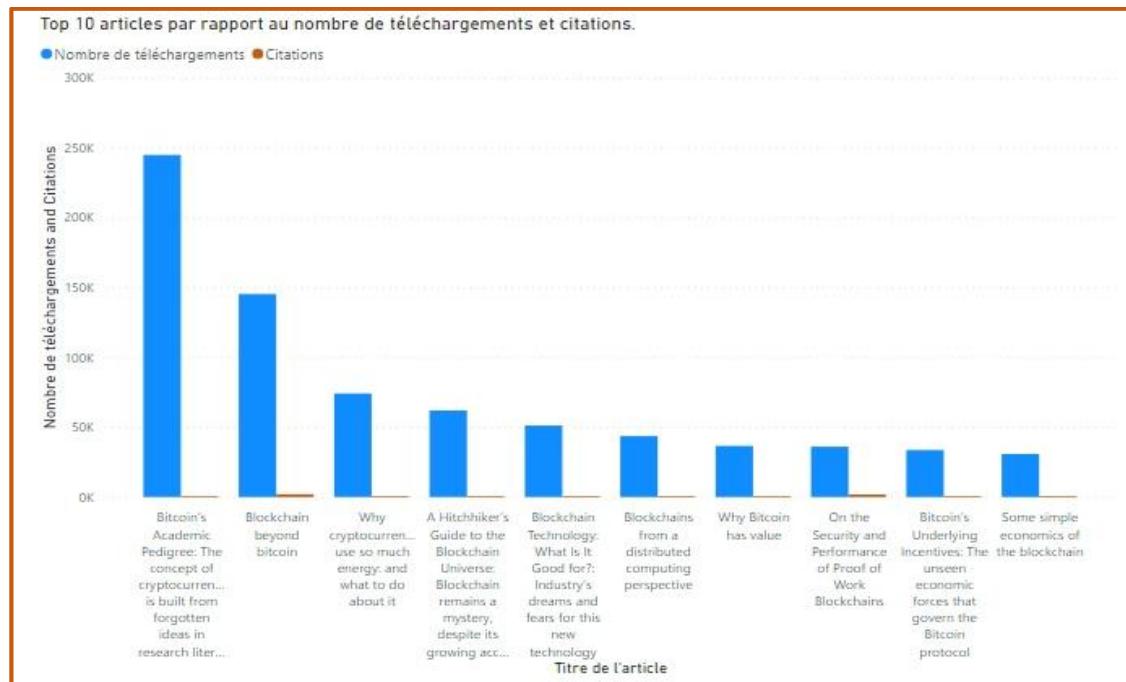
### 4. L'affichages de top 10 auteurs par rapport au nombre d'articles



### Interprétation :

Ce graphe illustre les auteurs ayant publié le nombre maximal d'article. Nous trouvons que « Mr. Yuheng Huang » est l'auteur le plus actif dans cette discipline.

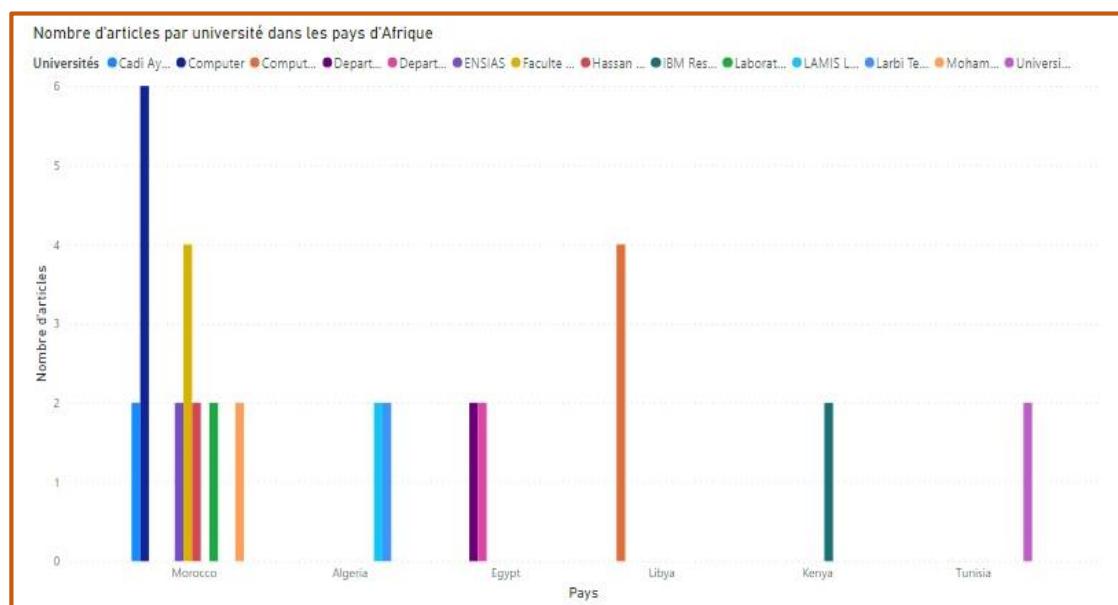
## 5. Exposition de top 10 articles en fonction de nombre de téléchargement et citation



### Interprétation :

L'article qui a peut capter plus d'attention des readers est « Bitcoin's Academic Predigree : the concept of cryptocurrency... »

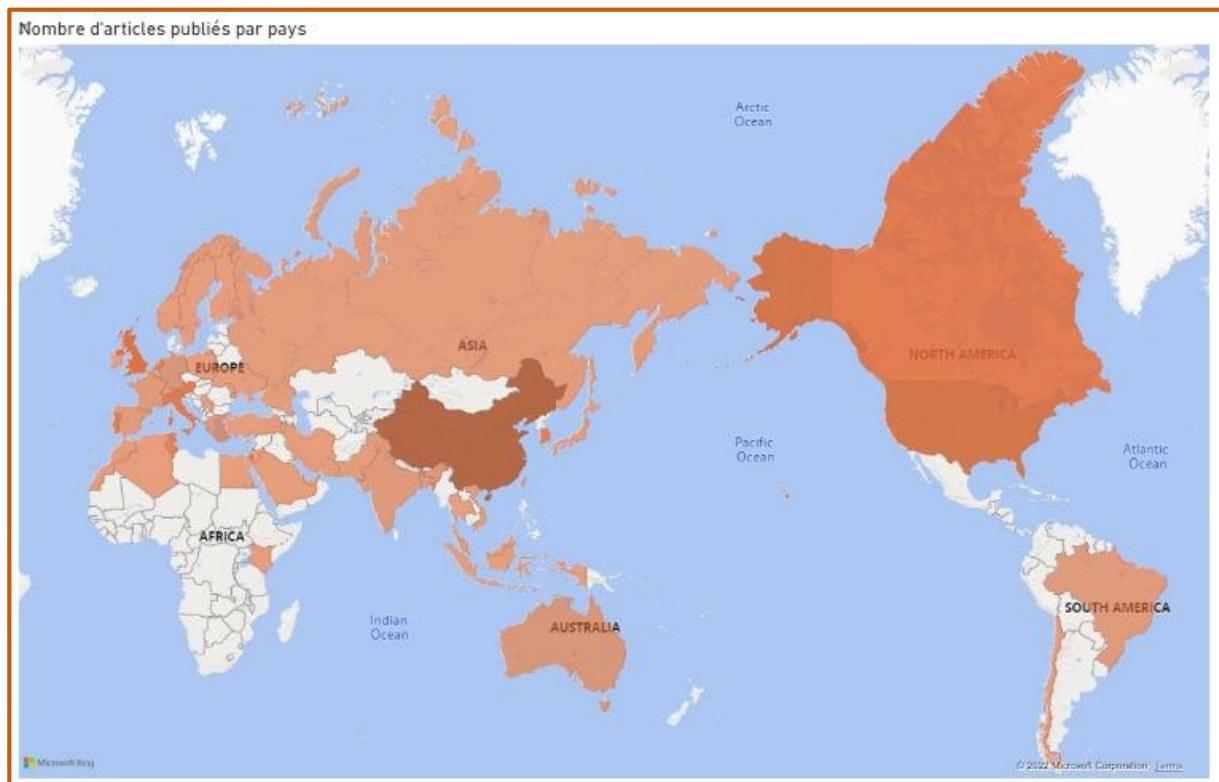
## 6. L'affichage de nombre d'articles publiées par les universités africaines



**Interprétation :**

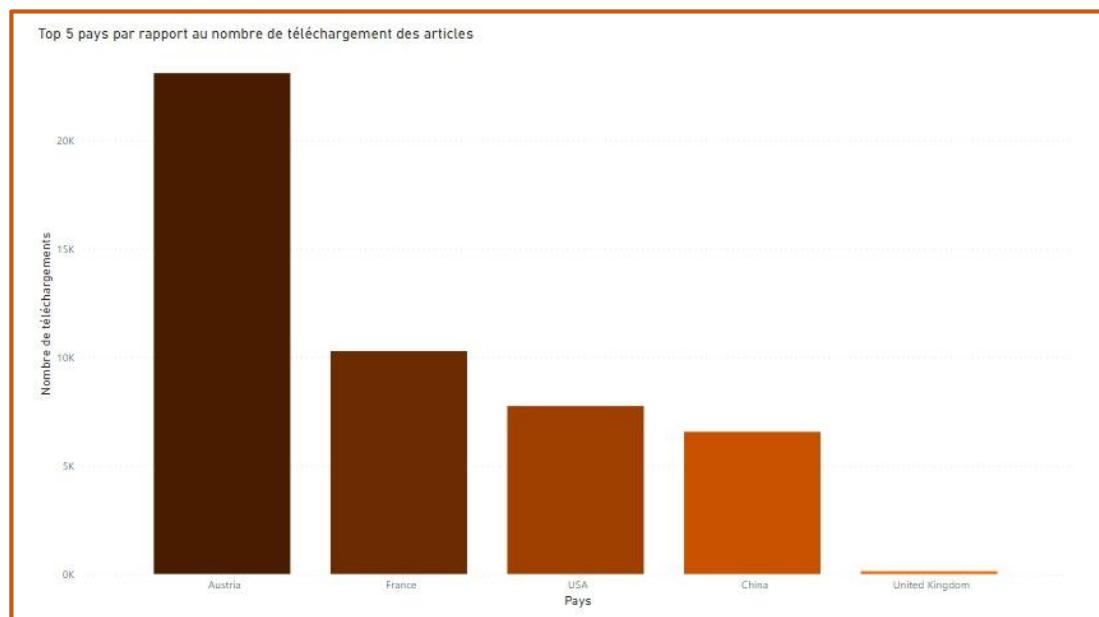
Ici, nous voyons que le Maroc est le leader des pays africain dans la recherche scientifique qui s'intéresse au « Blockchain » et au « Bitcoin » et parmi les universités charge de la recherche nous trouvons :

- ENSIAS
- Universite Hassan 2
- Faculté des sciences et techniques
- ...

**7. L'affichage de distribution de nombre de publication par pays****Interprétation :**

D'après cette figure on peut distinguer les pays les uns des autres selon le nombre d'articles publiées. Nous constatons que la chine, l'Amérique sont les pays les plus actifs, d'autre part le Maroc n'est pas assez intégré dans la recherche scientifique liée à ces domaines.

## 8. Top 5 pays par rapport au nombre de téléchargement d'article



### Interprétation :

Cette graphe nous permet de jeter un coup d'œil sur les cinq pays répertoriés par rapport au nombre des téléchargements effectués par les visiteurs des journaux. Nous constatons que l'Autriche a eu un nombre élevé de téléchargement par rapport aux autres pays.

## VII. Versions des technologies

- **Version de MongoDB :** Version 1.30.1
- **Version de Spark :** Version 3.03 avec Hadoop version 2.7 avec JDK 13
- **Version de PDI :** Version 9.2.0 avec JDK 1.8

## VIII. PROBLEMES RENCONTRES

Lors de la réalisation de projets nous avons rencontrés un ensemble des enjeux, comme :

- Le scraping pour les journaux « ScienceDirect » et « IEEE Xplore » n'était pas aisé, il nécessite des API Key.
- Le Blocage des adresses ip lors de scraping
- Apache Spark ne fonctionne qu'avec jdk 13
- Le problème de compatibilité de version de fichier jar « mongo-spark-connector » lors de la création de la connexion entre Apache Spark et MongoDB.

## IX. Conclusion

Nous avons introduit dans ce rapport la démarche effectuée pour la réalisation de notre projet dès l'extraction et la collection des données jusqu'à la visualisation de ces données par le biais des méthodologies offertes par les technologies de BI et Big Data. Ce travail nous a permis de suivre l'état des données au cours de tous les traitements jusqu'à l'état où elles deviennent des connaissances.

## X. REFERENCES

- <https://medium.com/@alexandrewrg/scrapinghub-scraper-les-donn%C3%A9es-de-plusieurs-pages-avec-scrapy-2e076ac7dc09>
- <https://datascientest.com/docker-guide-complet>
- <https://github.com/scrapinghub/splash>
- <https://datascientest.com/mongodb>
- [https://en.wikipedia.org/wiki/IEEE\\_Xplore](https://en.wikipedia.org/wiki/IEEE_Xplore)
- [http://www.democritique.org/IT/Association\\_for\\_Computing\\_Machinery.svg.xhtml](http://www.democritique.org/IT/Association_for_Computing_Machinery.svg.xhtml)
- <http://documentation.univ-rouen.fr/fiche-outils-n-51-zoom-sur-sciencedirect-458505.kjsp>
- <https://fr.myservername.com/schema-types-data-warehouse-modeling-star-snowflake-schema>
- <https://www.oracle.com/fr/database/business-intelligence-definition.html>