Universitat Rovira i Virgili

Escola Tècnica Superior d'Enginyeria

SISTEMES DISTRIBUÏTS

Práctica 2

AUTORS:

DAVID NAVA FERNANDEZ

LLUIS ORIOL COLOM NICHOLS

DOCENT:

PEDRO ANTONIO GARCÍA LÓPEZ

JOSEP SAMPE DOMENECH

18/06/2021

2020 - 2021

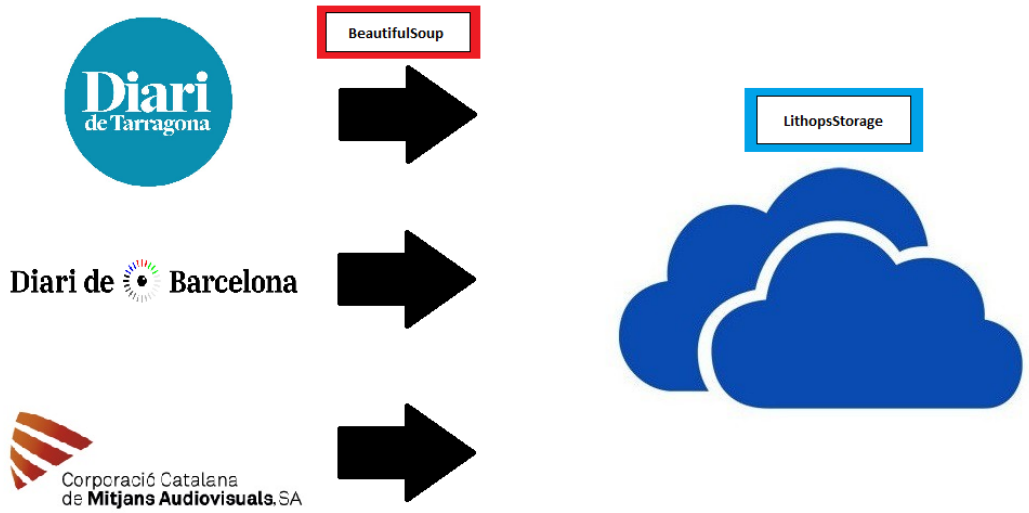
ÍNDICE

INTRODUCCIÓ

El nostre programa es basa en la idea de recopilar notícies de diferents diaris (diaridetarragona, diaridebarcelona i ccma), tractar-les de forma que poguem analitzar algunes dades básiques i guardar tota aquesta informació al cloud.

El nostre programa principal permetrà tant realitzar cerques per afegir noves noticies al nostre dataset, així com realitzar queries per buscar notícies en funció de diferents criteris.

ARQUITECTURA I DISSENY



Gràcies a l'import de BeautifulSoup som capaços d'obtindre els links de les diferents notícies que apareixen a qualsevol dels diaris mencionats anteriorment donat un tema en concret:

(en aquest cas primer obtenim el num. de pàgines de notícies que n'hi han i a continuació procedim a obtindrer tots els links)

```
def get_links():

    # Auxiliar variables.
    link_to_news = []

    # We create HTML parser.
    r = requests.get('https://www.diaridebarcelona.cat/search?q='+sys.argv[1], headers=header)
    soup = BeautifulSoup(r.text, 'html.parser')

    # Get the number of pages in the website.
    frame = soup.find(lambda tag: tag.name == 'li' and tag.get('class') == ['first'])
    if frame is not None:
        pages = frame.find("a").get('href')
        pages = pages.split("=")[-1]
    else:
        pages = 0

    # Get the links to the news.
    count = 0
    for i in range(int(pages)+1):
        r = requests.get('https://www.diaridebarcelona.cat/search?q='+SEARCH_KEY+'&start='+str(i), headers=header)
        soup = BeautifulSoup(r.text, 'html.parser')

        for news in soup.find_all(class_='col-sm-6 col-lg-3 mb-20px mb-lg-30px'):
            count +=1
            # We get the link to the news page.
            news = news.find(class_='h1 modul-petit')
            link_to_news.append(news.find("a").get('href'))

    return link_to_news,count
```

Un cop obtinguts tots els links referents a un tema, procedim a recollir totes les dades d'aquesta i analitzar-les, això ho farem amb diferents processos, de forma que podem paral·lelitzar l'execució de diferents notícies:

(podem veure en el cas següent com obtenim el títol d'una notícia i la entradeta corresponent)

```
def process_news(link):

    analyzer = SentimentIntensityAnalyzer()
    news_format = {}
    r = requests.get(link, headers=header)
    soup = BeautifulSoup(r.text, 'html.parser')

    # Put news link.
    news_format['link'] = link

    # Get news header.
    title = soup.find("div", class_='title-opening-section')
    if title is None:
        return          # Case we are threatting an opinion...
    news_format['title'] = title.text.replace("\n","").replace("\t","").replace("\"","")

    # Get news starter.
    description = soup.find("div", class_='description')
    if description is None:
        return          # Case we are threatting an interview...
    news_format['starter'] = description.text.replace("\n","").replace("\t","").replace("\xa0"," ").replace("\"","")
```

Podem paral·lelitzar aquesta execució gràcies a l'import de lithops.multiprocessing Pool:

```
    # Start cloud multiprocessing.
    with Pool() as pool:
        result = pool.map(process_news, link_to_news)
        print(result)
```

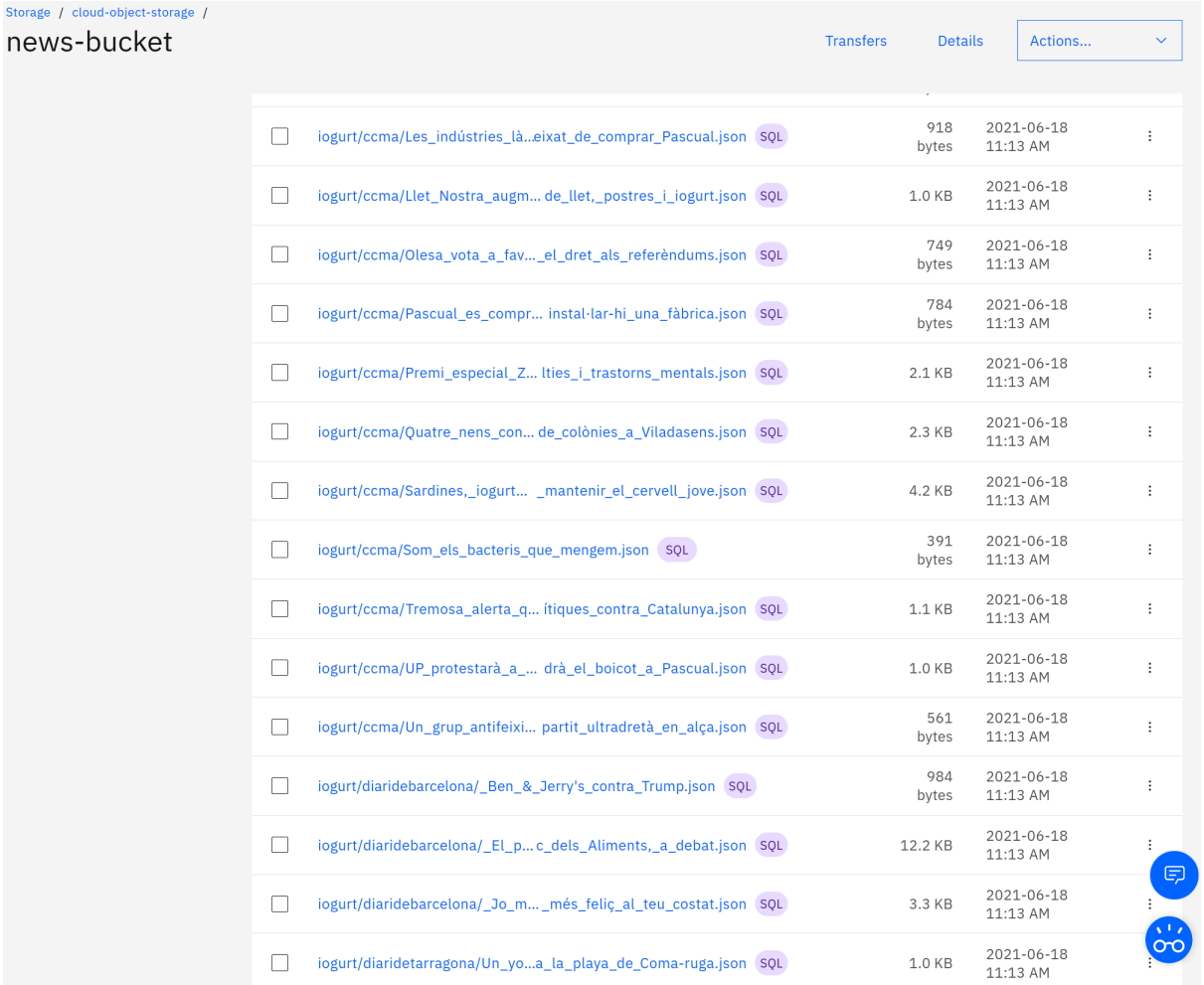
DATASET

El nostre dataset consta del conjunt de notícies cercades amb anterioritat. Aconseguim guardar-les mitjançant:

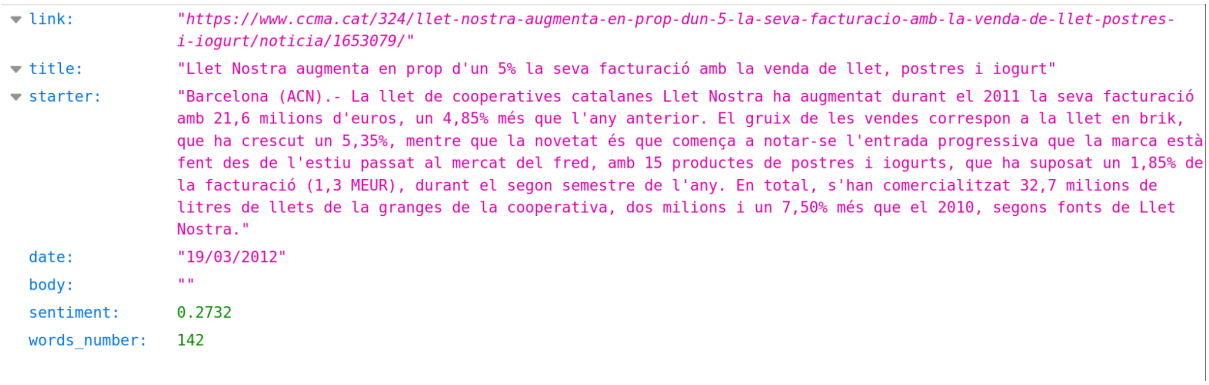
```
    # Store the news content to the cloud COS.
    storage = Storage()
    storage.put_object(bucket='news-bucket', key=SEARCH_KEY+'/diaridebarcelona/'+news_format['title'].replace(" ","_")+'.json', body = json.dumps(news_format))
```

Instrucció la qual ens assegura que les notícies obtingudes pel diaridebarcelona tindran el format en el qual primerament s'indicarà el nom de la cerca realitzada, a continuació el nom del diari i finalment el títol de la notícia obtinguda i el format .json.

(aquest n'ès un exemple del nostre dataset donada la cerca de "iogurt" i diferents resultats obtinguts pels diferents diaris)



Si obrim qualsevol notícia, aquesta serà tota la informació que podrem observar d'ella:



Aquesta conté el link d'obtenció de la notícia, el títol i l'entradeta d'aquesta, la data de posting, la descripció de la notícia (que en aquest cas no en té), el sentiment que suggereix (entre -1 i 1) i el número de paraules que conté.

NUM. FUNCIONS

Al llarg del nostre programa podem observar diferents situacions en les quals utilitzem threads o la llibreria pool per tal de facilitar la computació de certes tasques.

En un principi, podem observar com creem tres fils per tal de d'encarregar-se de la execució de cadascun dels diaris que tractem:

```
ccma_thread = Thread(target=ccma_query)
ccma_thread.start()
dtg_thread = Thread(target=dtg_query)
dtg_thread.start()
dbc_thread = Thread(target=dbc_query)
dbc_thread.start()
```

A més, dins d'aquests threads realitzarem el pool per processar les notícies més ràpidament:

```
def dbc_query():
    link_to_news = dbc_get_links()

    # Start cloud multiprocessing.
    with Pool() as pool:
        result = pool.map(dbc_process_news, link_to_news)
```

Finalment, també utilitzarem el pool per fer el load de les notícies que hi ha al cloud referents al tema buscat:

```
storage = Storage()
news_list = storage.list_keys(BUCKET,SEARCH_KEY+'/')
with Pool() as pool:
    news = pool.map(get_object_cloud, news_list)
return news
```