MPT3. Escenario con sistemas NoSQL

En esta practica vamos a estudiar los usos que se le pueden dar a una base de datos NoSQL.

El uso en particular es la recopilación de datos a gran escala. Ejemplos claros en los cuales se utiliza una NoSQL es en Bet365, para la recogida de datos de las estadísticas de partidos deportivos de cualquier categoría; Riot Games, para la recogida de estadística de partidas, estadísticas de usuario, recogida del ELO (partidas ganadas o perdidas para luego pasarlo a un algoritmo de emparejamiento para las partidas).

En mi caso he elegido MongoDB y vamos a inyectar información a la base de datos de una API Restful de la pagina https://www.coingecko.com/es donde recogeremos datos sobre las distintas criptomonedas que hay, sus cambios y sus exchanges. La practica principal de esto es que con esta información podremos ver el margen del cambio de la moneda que queramos buscando el margen mínimo y máximo para luego comprar la moneda en el mínimo y venderla en el máximo y sacar beneficio. Es una técnica muy sencilla, pero sin automatizarla seria muy pesado.

1. Instalación de MongoDB

Lo primero que debemos hacer es configurar el repositorio oficial de MongoDB para Ubuntu 20.04 LTS. Primero instalamos la clave publica:

‘wget https://www.mongodb.org/static/pgp/server-4.4.asc -O- | sudo apt-key add -’

Creamos el archivo de repositorio **‘sudo nano /etc/apt/sources.list.d/mongodb-org.list’** y añadimos lo siguiente:

deb http://repo.mongodb.org/apt/ubuntu focal/mongodb-org/4.4 multiverse

Guardamos el archivo y hacemos **‘sudo apt update’**.
Una vez hecho todo esto podemos instalar MongoDB con **‘sudo apt install mongodb-org-server’**
Luego ponemos **‘sudo systemctl start mongod’** y para que se inicie cada vez que arranquemos la maquina **‘sudo systemctl enable mongod’**
Después instalamos el cliente usando **‘sudo apt install -y mongodb-org-shell’**

Para crear un usuario ponemos en la consola mongo y luego use admin. Creamos el usuario poniendo **db.createUser({user: “abraham”, pwd: “123456”, roles:[{role: “root”, db: “admin”}]})**

Y ya estaría listo para conectarnos con nuestro usuario pondriamos mongo -u abraham

2. API Restful

Vamos a realizar un .py con una conexión a nuestra base de datos de mongo y vamos a extraer los datos de la API para introducirlas. Necesitamos **‘pip3 install pycoingecko’** y **‘pip3 install pymongo’**

La documentación de la API para ver las funciones que tenemos a nuestro alcance https://www.coingecko.com/api/documentations/v3

```
from pycoingecko import CoinGeckoAPI
import os
import requests
import pymongo
from pymongo import MongoClient
#conexion
con = MongoClient('localhost', port=27017, username='abraham', password='123456')
db = con['crypto']#crea la base de datos y si esta pues se situa en ella
col = db['intercambios']#crea la coleccion intercambios y si esta se situa en ella

cg = CoinGeckoAPI()
monedas = cg.get_coins_list()


mis_monedas = []
for i in monedas:
    mis_monedas.append(i['id'])


#for z in mis_monedas:
    #print(mis_monedas) # lista con todas las monedas para insertar los tickets en intercambio de cada una de ellas
intercambio = cg.get_coin_ticker_by_id(id=['bitcoin'])


for j in intercambio['tickers']:#poner bucle anidado para el array
        col.insert_one({

                'Moneda': j['base'],
                'Cambio': j['target'],
                'Mercado': j['market']['name'],
                'URL': str(j['trade_url']),
                'LastChange': (j['converted_last']['usd']),
                'Spread': (j['bid_ask_spread_percentage']),
                'FechaHora': str(j['timestamp']),


                })
```

Nos pasamos el fichero con scp a nuestra maquina virtual. Y tiramos el .py con ‘python3 nombrefichero’

Nos metemos en mongo y nos posicionamos en la db crypto que se crea en el script:

```
abraham@mongo:~$ mongo -u abraham
MongoDB shell version v4.4.4
Enter password:
connecting to: mongodb://127.0.0.1:27017/?compressors=disabled&gssapiServiceName
=mongodb
Implicit session: session { "id" : UUID("b19d5a27-3791-4b18-ae40-447dcbc268b3")
}
MongoDB server version: 4.4.4
---
The server generated these startup warnings when booting:
        2021-05-25T18:57:58.741+00:00: Using the XFS filesystem is strongly reco
mmended with the WiredTiger storage engine. See http://dochub.mongodb.org/core/p
rodnotes-filesystem
---
> use crypto
switched to db crypto
```

Y hacemos una consulta a la colección intercambios:

```
> db.intercambios.find().pretty()
{
        "_id" : ObjectId("60acae8e25d1c5366f97301e"),
        "Moneda" : "BTC",
        "Cambio" : "USDT",
        "Mercado" : "Binance",
        "URL" : "https://www.binance.com/en/trade/BTC_USDT?ref=37754157",
        "LastChange" : 38786,
        "Spread" : 0.010026,
        "FechaHora" : "2021-05-25T07:43:58+00:00"
}
{
        "_id" : ObjectId("60acae8e25d1c5366f97301f"),
        "Moneda" : "BTC",
        "Cambio" : "USD",
        "Mercado" : "Bitfinex",
        "URL" : "https://www.bitfinex.com/t/BTCUSD",
        "LastChange" : 38835,
        "Spread" : 0.025773,
        "FechaHora" : "2021-05-25T07:44:15+00:00"
}
{
        "_id" : ObjectId("60acae8e25d1c5366f973020"),
```

Vemos que los datos se han introducido bien, solo faltaría introducir la función de agregación para ver cual es el margen (spread) mínimo y máximo, con eso tendríamos suficiente para ver la utilización. Si queremos emplear mas a fondo este proyecto podemos hacer otro script que manipule los datos de la colección y calcule el margen para ver si podemos obtener beneficio y esto como el script de la API programarlo cada cierto tiempo.