Examen Interciclo

Inteligencia artificial

Esteban Sibri

Objetivo: Consolidar los conocimientos adquiridos en clase para los métodos de búsqueda y bases de datos orientadas a grafos.

Enunciado

Diseñe y desarrolle un sistema recopilador que permita obtener las noticias, Facebook, twitter de los alcaldes dentro de una base de datos orientados a grafos:
Webscraping es la técnica de extraer datos contenidos en un formato no estructurado en una página web y llevarlos a una estructura fácil de usar.
Es por ello, que se desea crear nuevos métodos que permitan la recopilación masiva de información para su posterior estudio y correlación en forma de big data.
En base a ello, vamos a obtener los datos de lo que esta hablando las noticias de los candidatos dentro del Ecuador y almacenar los datos dentro de una base de datos orientadas a grafos.

Funciones principales

Creamos los nodos

```python
from neo4j import GraphDatabase

class Neo4jService(object):

    def __init__(self, uri, user, password):
        self._driver = GraphDatabase.driver(uri, auth=(user, password))

    def close(self):
        self._driver.close()

    def crear_Alcaldes(self, tx, nombre):
        tx.run("CREATE (:Alcaldes {nombre: $nombre})", nombre=nombre)

    def crear_Alcalde(self, tx, nombre):
        tx.run("CREATE (:Alcalde {nombre: $nombre})", nombre=nombre)

    def crear_Tweets(self, tx, nombre):
        tx.run("CREATE (:Tweet {nombre: $nombre})", nombre=nombre)

    def crear_Tweet(self, tx, nombre, fuente, retuits):
        tx.run("CREATE (:Noticias {nombre: $nombre, fuente: $fuente, retuits: $retuits})", nombre=nombre,fuente=fuente,retui

```

Creamos las relaciones

```
#realciones central - alcandes
    def crear_relacion_Alcaldes(self, tx, nombre_alcalde, nombre_alcaldes):
        tx.run("MATCH (a:Alcaldes {nombre: $nombre_alcalde}) "
               "MATCH (b:Alcalde {nombre: $nombre_alcaldes}) "
               "MERGE (a)-[:Alcaldes]->(b)",
               nombre_alcalde=nombre_alcalde, nombre_alcaldes=nombre_alcaldes)

    def crear_relacion_TweetAl(self, tx, nombre_alcalde, nombre_tweet):
        tx.run("MATCH (a:Alcalde {nombre: $nombre_alcalde}) "
               "MATCH (b:Tweet {nombre: $nombre_tweet}) "
               "MERGE (a)-[:tweet_Alcalde]->(b)",
               nombre_alcalde=nombre_alcalde, nombre_tweet=nombre_tweet)

#realciones alcalde - noticia
    def crear_relacion_TitNot(self, tx, nombre_titulo,nombre_noticia):
        tx.run("MATCH (a:Tweet {nombre: $nombre_titulo}) "
               "MATCH (b:Noticias {nombre: $nombre_noticia}) "
               "MERGE (a)-[:Tweet_Noticia]->(b)",
               nombre_titulo=nombre_titulo, nombre_noticia=nombre_noticia)

print("Ejecucion correcta")
```

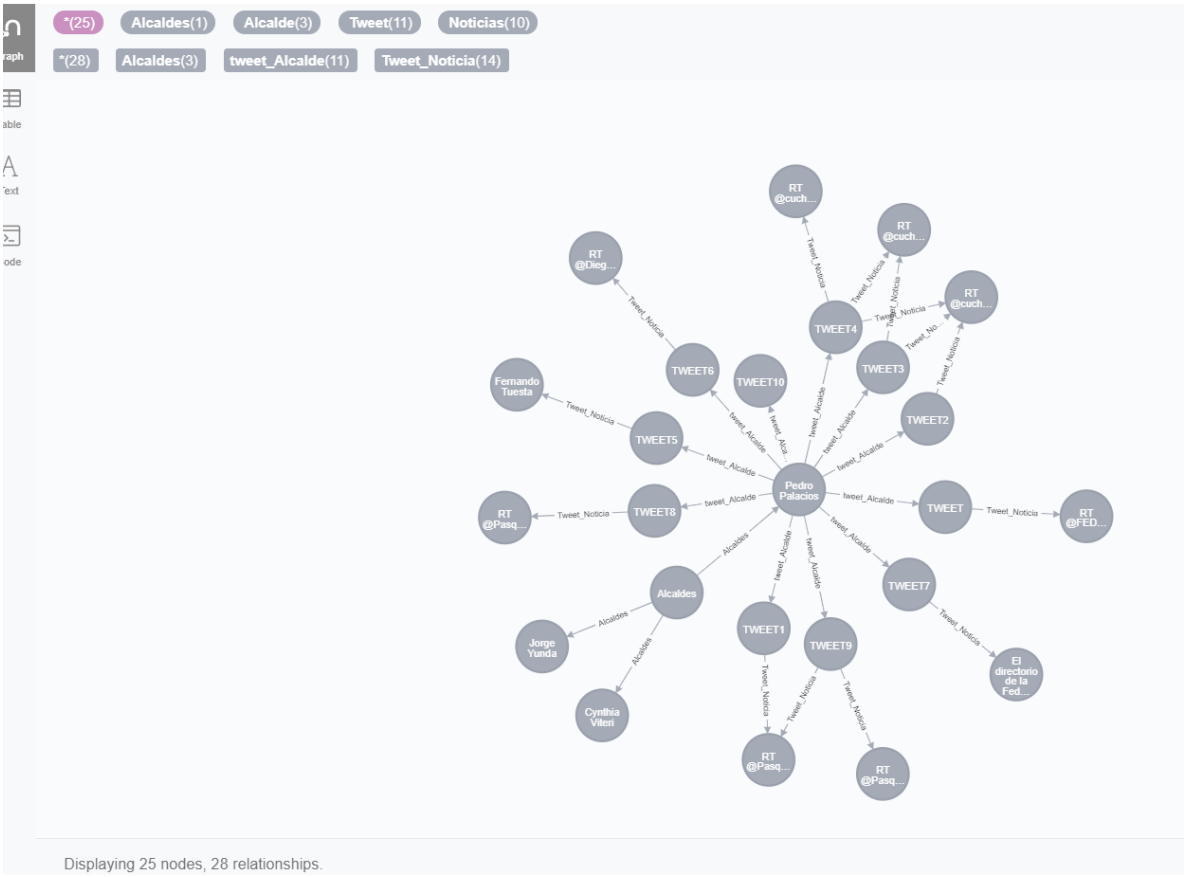
Datos

Usamos el siguiente código para extraer la información de twitter y hacemos la conexión con Neo4j para que guarde y grafique los datos obtenidos.

```
import tweepy
import time
neo4j = Neo4jService('bolt://localhost:7687', 'neo4j', 'examen')
with neo4j._driver.session() as session:

    session.write_transaction(neo4j.crear_Alcaldes , "Alcaldes")
    session.write_transaction(neo4j.crear_Alcalde , "Pedro Palacios")
    session.write_transaction(neo4j.crear_Alcalde , "Jorge Yunda")
    session.write_transaction(neo4j.crear_Alcalde , "Cynthia Viteri")

    session.write_transaction(neo4j.crear_relacion_Alcaldes, "Alcaldes", "Pedro Palacios")
    session.write_transaction(neo4j.crear_relacion_Alcaldes, "Alcaldes", "Jorge Yunda")
    session.write_transaction(neo4j.crear_relacion_Alcaldes, "Alcaldes", "Cynthia Viteri")

# 4 cadenas para la autenticacion
    consumer_key = "rwNrplwdiEpTFAc3WVeJmRHNI"
    consumer_secret = "vH6Hfo5kZHJHycymBGKQuFoqWWaKm9bXDb4jmvnCAVnEzW9h3d"
    access_token = "1302969322533519360-z5rquE3xp6bEaGROARjADTeAoqrrEt"
    access_token_secret = "yCm4m9MolHrS0QHX59N8pO11KFOMtbkiztb1acIttIw8V"

    auth = tweepy.OAuthHandler(consumer_key, consumer_secret)
    auth.set_access_token(access_token, access_token_secret)
    # con este objeto realizaremos todas las llamadas al API
    api = tweepy.API(auth,
                     wait_on_rate_limit=True,
                     wait_on_rate_limit_notify=True)
    a=0
    h=1
    nodo="TWEET"

    session.write_transaction(neo4j.crear_Tweets , nodo)
```

```
nodo="TWEET"

session.write_transaction(neo4j.crear_Tweets , nodo)
for tweet in tweepy.Cursor(api.search, q="Pedro Palacios", tweet_mode="extended").items(300):

    if(a==1):
        nodo="TWEET"+str(h)
        session.write_transaction(neo4j.crear_Tweets , nodo)
        a=0
        h=h+1
    else:
        a=a+1
        session.write_transaction(neo4j.crear_Tweet , tweet._json["full_text"], tweet._json["user"]["screen_name"], twee
        session.write_transaction(neo4j.crear_relacion_TitNot, nodo, tweet._json["full_text"])
        session.write_transaction(neo4j.crear_relacion_TweetAl, "Pedro Palacios", nodo)

time.sleep(65)
print("fin uno")

for tweet in tweepy.Cursor(api.search, q="Jorge Yunda", tweet_mode="extended").items(300):

    if(a==1):
        nodo="TWEET"+str(h)
        session.write_transaction(neo4j.crear_Tweets , nodo)
        a=0
        h=h+1
    else:
        a=a+1
        session.write_transaction(neo4j.crear_Tweet , tweet._json["full_text"], tweet._json["user"]["screen_name"], twee
        session.write_transaction(neo4j.crear_relacion_TitNot, nodo, tweet._json["full_text"])
        session.write_transaction(neo4j.crear_relacion_TweetAl, "Jorge Yunda", nodo)

time.sleep(65)
print("fin dos")
```

Datos generados en Neo4j



Conclusiones

Los datos recopilados de la aplicación de twitter nos ayuda a ver la cantidad de actividad y similitud que tienen algunas publicaciones de los alcaldes de Ecuador. La recopilación de datos

nos puede indicar el grado de popularidad y aceptación que tiene los tuits de los alcaldes con respecto a la población, ver si tiene gente que lo apoya o si hay gente que no lo apoya.

Recomendaciones

Tener en cuenta las llaves del api

Tener en cuenta el limite de peticiones por día que permite el api

Instalar las librerías adecuadas

Revisar la conexión de la base de Neo4j con Python