ANO
2024

UNINTER

CADERNO DE RESPOSTAS DA ATIVIDADE PRÁTICA DE:

NoSQL

ALUNO: WERRICSSON SANTOS - 4231021

Caderno de Resposta Elaborado por:
Prof. MSc. Guilherme Ditzel Patriota

Prática 01 – JSON COM NEO4J.

Questão 01 – DESCOBERTA DA HASHTAG PRINCIPAL.

ENUNCIADO: Veja o Roteiro da Atividade Prática para mais detalhes.

I. Apresentação dos comandos (códigos e queries) usados (não esquecer do identificador pessoal):

```
/*Criando a base de dados a partir dos arquivos JSON
  localizados na pasta import*/
CALL apoc.load.directory('*.json') YIELD value
WITH value AS arquivos
ORDER BY arquivos DESC
CALL apoc.load.json(arquivos) YIELD value
UNWIND value.data AS tweet
//Criando os nós de Tweet
MERGE (t:Tweet {tweet_id: tweet.id})
ON CREATE SET t += {
    texto: tweet.text,
    criado_em:
        date(datetime(tweet.created_at).year + "-" +
             datetime(tweet.created_at).month + "-" +
             datetime(tweet.created_at).day),
    lingua: tweet.lang,
    curtidas: tweet.public_metrics.like_count,
    retweets: tweet.public_metrics.retweet_count,
    respostas: tweet.public_metrics.reply_count,
    citacoes: tweet.public_metrics.quote_count,
    autor: tweet.author_id,
    geolocalizacao: tweet.geo.place_id
}
//Criando os nós de Hashtags
FOREACH (hashtag IN tweet.entities.hashtags |
    MERGE (h:Hashtag {hashtag: apoc.text.replace(
        apoc.text.clean(hashtag.tag), '[^a-zA-Z0-9]', '')})
    MERGE (h)<-[:POSSUI]-(t)
)
/*Identificando o tipo de Tweet, removendo dos nós de Tweet e
  recriando o nó com a etiqueta correta conforme o tipo */
FOREACH (ref_tweet IN tweet.referenced_tweets |
    SET t.tipo_ref = coalesce(t.tipo_ref, []) + [ref_tweet.type],
        t.id_ref = coalesce(t.id_ref, []) + [ref_tweet.id]
);
MATCH (t) WHERE "retweeted" in t.tipo_ref
REMOVE t:Tweet
SET t:Retweet;
MATCH (t) WHERE "replied_to" in t.tipo_ref
REMOVE t:Tweet
SET t:Replied_to;
MATCH (t) WHERE "quoted" in t.tipo_ref
REMOVE t:Tweet
SET t:Quot;

/*Criando o nós Aluno e numeroRU com o relacionamento RU
  (para melhor apresentação da identificação na imagem de grafos) */
CREATE (a:Aluno {nome: "Werricsson Santos"}) -[:RU]-> (r:numeroRU {RU: 4231021});

/*Identificando a hashtag principal e fazendo a consulta para criar a
  imagem de grafos conforme os requisitos da atividade*/
MATCH (h:Hashtag) <-[:POSSUI]- (t:Tweet)
WITH h.hashtag as tag, COUNT(t) AS Frequencia
ORDER BY Frequencia DESC LIMIT 1
Match (t:Tweet) -[:POSSUI]-> (h:Hashtag)
Match (a:Aluno) -[:RU]-> (r:numeroRU)
WHERE tag in h.hashtag AND date(t.criado_em) = date("2020-03-05")
RETURN t, h, a, r
```

Figura 1: Imagem do código que cria o banco de dados e gera a imagem de grafos exibindo a hashtag principal.

Observação:

Deixei comentários no código a fim de ser mais suscinto na descrição da figura.

II. Apresentação das Imagens/Print do resultado (não esquecer do identificador):

Figura 2: Imagem de grafos gerada através da consulta (imagem do código à esquerda), a partir da linha 52. A imagem apresenta o nó central que corresponde a hashtag principal “issoaglobonaomostra”, rodeado pelos nós dos Tweets que possuem essa hashtag principal contendo as arestas do relacionamento “Tweet –[:POSSUI]-> Hashtag”.

Observação:

Para gerar a imagem, rode o código a partir da linha 52, após ter rodado as linhas anteriores para popular o banco.

No rodapé da imagem está os nós Aluno e numeroRU, com a aresta indicando o relacionamento “Aluno –[:RU]-> numeruRU” (o nome do relacionamento está assim apenas para melhor representação na imagem).

III. Responda à pergunta: Qual foi a hashtag usada como filtro para coleta dos dados analisados?

Resposta:

A consulta realizada a partir da linha 52 finalizada na linha 54 tem como resposta a hashtag principal (mais frequente nos tweets), que é a hashtag “#issoaglobonaomostra”. Indicando que esse foi o filtro utilizado para a coleta dos dados analisados.

Prática 01 – JSON COM NEO4J.

Questão 02 – ANÁLISE DOS DADOS SEGUNDO VIÉS A SUA ESCOLHA.

ENUNCIADO: Veja o Roteiro da Atividade Prática para mais detalhes.

I. Apresentação dos comandos (códigos e queries) usados (não esquecer do identificador pessoal):

```
1  // Buscando todos os tweets originais
2  MATCH (t:Tweet)
3  WITH t.tweet_id as id_original
4
5  // Buscando todos os retweets que referenciam esses tweets originais
6  MATCH (r:Retweet)
7  WHERE id_original in r.id_ref
8
9  // Criando a relação RETWEET_OF entre os retweets e os tweets originais
10 MERGE (r)-[:RETWEET_OF]→(t);
11
12 /* Buscando e contando os retweets para cada tweet original, com ordenação
13    decrescente na contagem de retweets, limitando a 1 nodo
14    (para trazer o Tweet mais retweetado) */
15 MATCH (r:Retweet)-[:RETWEET_OF]→(t:Tweet)
16 WITH t, COUNT(r) AS retweet_count
17 ORDER BY retweet_count DESC
18 LIMIT 1
19 /* Retornando o Tweet mais retweetado e seus respectivos retweets, com ordenação
20    pela data de criação mais recente, limitando em 12 nodos. */
21 MATCH (r:Retweet)-[:RETWEET_OF]→(t)
22 Match (a:Aluno) -[:RU]→ (ru:numeroRU)
23 RETURN t, r, a, ru ORDER BY r.criado_em DESC LIMIT 12;
24 RU = 4231021;
```

Figura 1: Código criado para gerar uma imagem de grafos representando como eixo central o Tweet mais retweetado. Para gerar a imagem, iniciamos criando uma relação entre Retweets e Tweets, após criado o relacionamento, identificamos o Tweet que mais teve retweets (linha 15 à 18) e fazemos o retorno do Tweet original, os retweets mais recentes relacionado a ele limitado a 12 nós e a minha identificação para a imagem (Aluno –[:RU] -> numeroRU).

Observação:

Na última linha estou repetindo meu RU, apenas para ter o número exibido também na imagem do código.

II. Apresentação das Imagens/Print do resultado (não esquecer do identificador):

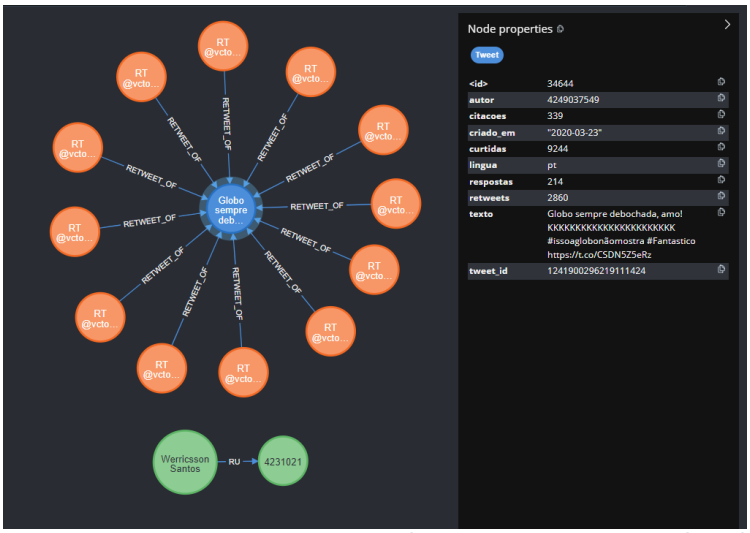

Figura 2: Imagem de grafos gerada através da consulta (imagem do código à esquerda) onde mostra o Tweet mais retweetado (nó central) e seus respectivos Retweets com as arestas indicando o relacionamento "RETWEET_OF".

Observação:

Para gerar a imagem, rode o código a partir da linha 15, após ter rodado as linhas anteriores para criar as relações.

A imagem está com o Tweet selecionado, sendo possível observar seus atributos ao lado direito da imagem de grafos.

No rodapé da imagem está os nós Aluno e numeroRU, com a aresta indicando o relacionamento "Aluno –[:RU]-> numeruRU" (o nome do relacionamento está assim apenas para melhor representação na imagem).

III. Responda à pergunta: Qual foi o comando usado por você para dar entrada dos dados em JSON no seu banco de dados Neo4j?

Resposta:

Conforme demonstrado na figura 1 (atividade 1), o comando usado para popular o banco de dados criando os nós através de arquivos JSON é feito através do uso da biblioteca APOC.

O primeiro passo é mover os arquivos desejados dentro da pasta import (localizada dentro da pasta do nosso projeto).

Após isso chamamos a função load da biblioteca com o código "call apoc.load.directory("*.json")". Essa função lê um diretório/pasta e retorna uma lista com todos os arquivos JSON contidos na pasta import o parâmetro "*.json" garante que iremos ler apenas arquivos desta extensão (evita erros caso haja algum arquivo inesperado).

O comando YIELD especifica qual coluna/valor queremos extrair, nesse caso, extraímos a coluna value que contém os arquivos JSON, renomeamos essa coluna para arquivos "WITH value as arquivos" e chamamos novamente a função load da biblioteca para ler cada um desses arquivos "call apoc.load.json(arquivos)" (desta vez usando "apoc.load.json" para ler os arquivos).

Nessa segunda chamada da função load, o comando YIELD value irá retornar tudo o que existe dentro de cada arquivo json, porém os dados principais que contém informações dos tweets estão dentro do container "data". Esse container possui uma estrutura de lista [] contendo vários objetos/documentos que são os tweets com seus respectivos atributos. Então usamos o comando "UNWIND value.data" para "desenrolar"/iterar sobre essa lista e poder criar os nós necessários, populando nosso banco.