Build a Knowledge Graph using Neo4J

Jay Doshi · Follow
Published in Analytics Vidhya
4 min read · Jun 6, 2020

For the past decade or so, Knowledge Graphs have been sneaking into our daily lives, be it through voice assistants (such as Alexa, Siri, or Google Assistant), intuitive search results, or even personalized shopping experiences on Amazon. We constantly interact with Knowledge Graphs on a daily basis.

However, Knowledge Graphs and underlying graph databases are still a mystery, most of us aren't even aware of how dependent we are on the technology — or worse, how we have come to expect a certain quality and standard that we are now so reliant on.

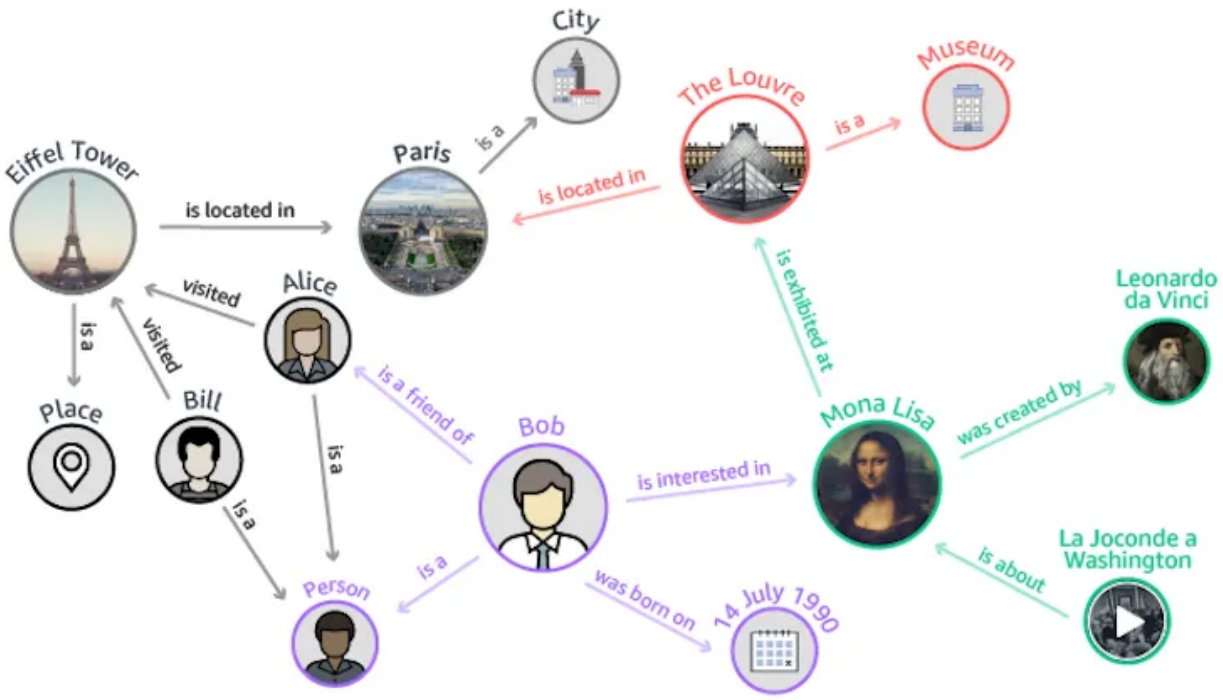
Several organizations are already using this technology to stay ahead of the curve. Furthermore, there are use cases for Knowledge Graphs and graph databases in all types of industries, ranging from banking, the auto industry, oil and gas to retail, pharmaceutical, media, etc.

What is a Knowledge Graph?

A Knowledge Graph is nothing but the graphical representation of underlying data that models entities within the data as nodes and the relations between those entities as edges/links between the nodes.

It can help us understand the linkages within our data at scale — be it structured or unstructured.

Consider the following example of a movie knowledge graph:



source: https://www.kaggle.com/ferdzso/knowledge-graph-analysis-with-node2vec

Python with Neo4J

Here we will use ***Py2neo,*** a client library, and toolkit for working with Neo4j from within Python applications and from the command line.

We will have a simple use case of building a knowledge graph containing news articles, article sources, authors, etc. For the purpose of getting data we will use event registry, an API that provides a collection of global news articles and events.

Let us now dive into the code, we can talk about the intricacies along the way.

Step 1) Import libraries

The py2neo library will help us write python commands to insert, update and query the neo4j graph database

```
import pickle
import requests
import json
from py2neo import Database, Graph, Node, Relationship
import re
import en_core_web_sm
import os
import random
import time
from datetime import datetime
import warnings
warnings.filterwarnings('ignore')
```

kg_imports.gist hosted with ❤ by **GitHub** **view raw**

Fetch News Articles

These will be the input to our knowledge graph.

The reason why we are using event-registry to gather news articles is that it provides us with the complete article-text free from any HTML tags, thus we do not need to parse anything. Additionally, it provides us with other article meta-data such as source of the article, authors, sentiment, language, etc.

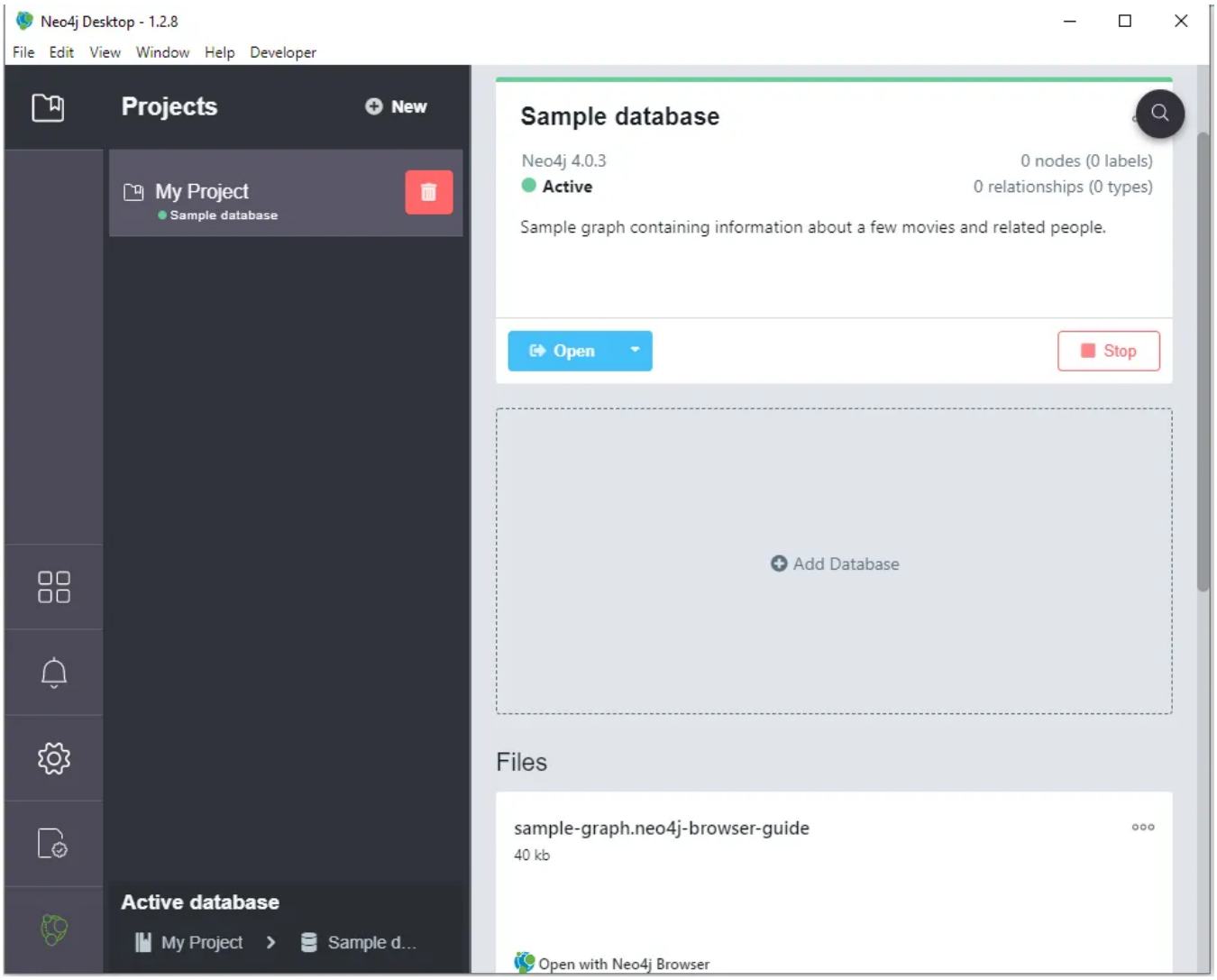
Once we register on https://eventregistry.org/(create a free account), we get an API key that must be sent within the body of the POST request to retrieve articles.

```
def get_articles_request(keywords, num_articles=100):
    print("[+] Getting articles for keywords:", keywords)
    articles = []
    url = "http://eventregistry.org/api/v1/article/getArticles"
    keywords = "technology"
    api_key = "API_KEY"
    payload = {"action": "getArticles", "keyword": keywords, "articlesPage": 1, "articl
               "articlesSortByAsc": False, "articlesArticleBodyLen": -1, "resultType":
               "dataType": ["news", "pr"], "apiKey": api_key,
               "forceMaxDataTimeWindow": 31,
               "articlesCount": num_articles}

    headers = {
        'Content-Type': 'application/json'
    }

    response = requests.request("POST", url, headers=headers, data=json.dumps(payload))
    try:
        articles = json.loads(response.text.encode('utf8'))
    except:
        print("Exception occured for keyword:", keywords)
        return list()

    return articles["articles"]["results"]
```

kg_get_articles hosted with ❤ by **GitHub** **view raw**

Configure Neo4J

Download Neo4J Desktop version from https://neo4j.com/download/ and install it on your machine. Once that is done you should see something like:



Click on “Add Database” -> “Create a local graph” -> change the name from “Graph” if you want and set a password.

Click the Start button to activate the database, then open the Neo4J browser to visualize the data.

Loading Data

The below code demonstrates how we can enter data in Neo4J.

We follow the below steps:

- Check if news_source has been added,
- if yes, then we just retrieve the node from the graph database

- else, we create a new node, assign a unique id to it.
- Once we have the node, we create a relationship between the article node and news-source node
- Store the relationship to neo4j

You can find the entire code in this Github repository

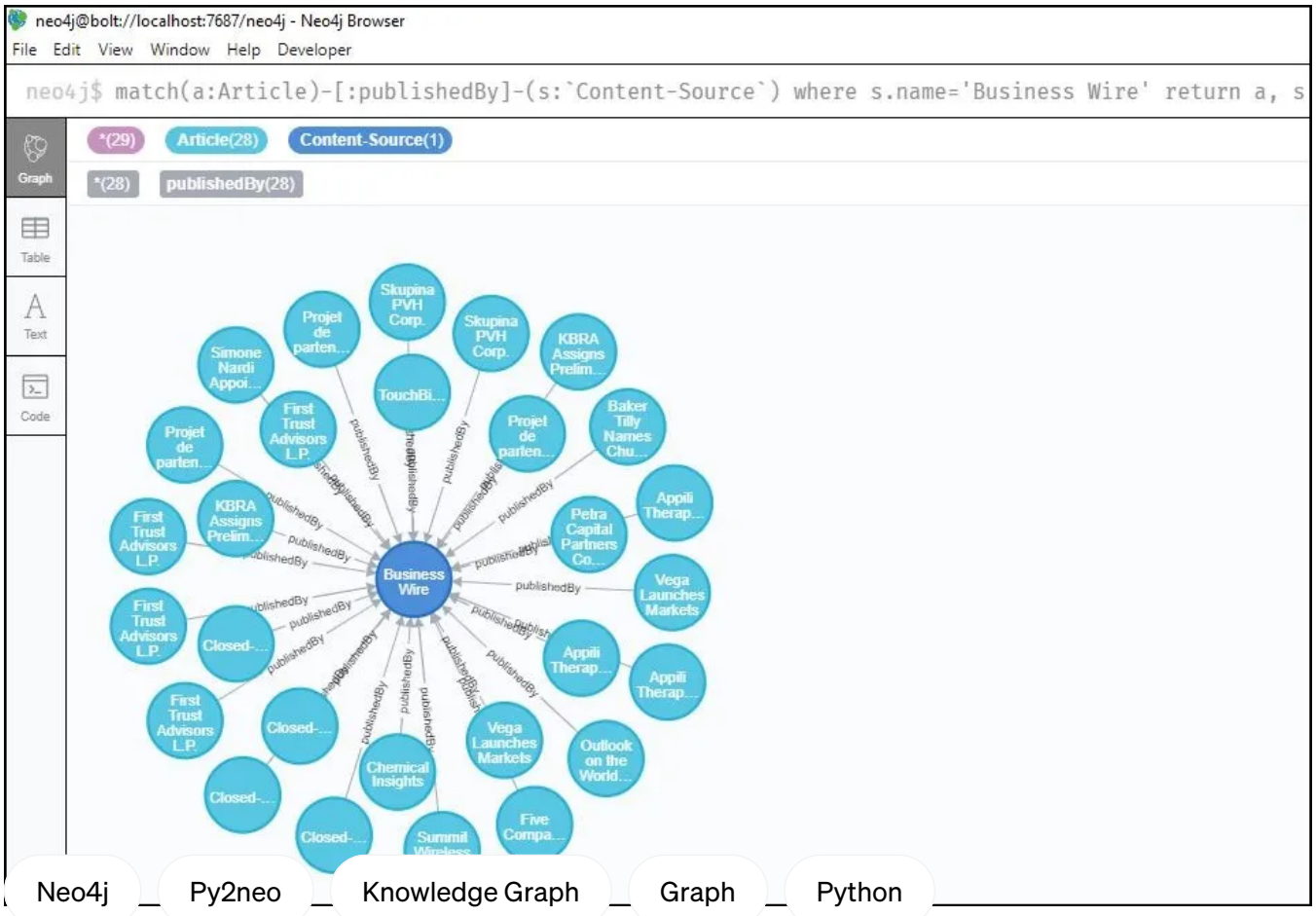
Visualization

Once we load all relevant information within the graph database, we can use the in-built Neo4J browser to visualize the data.

We can use cypher for this purpose, Cypher is quite similar to SQL and acts as an interface to the knowledge graph.

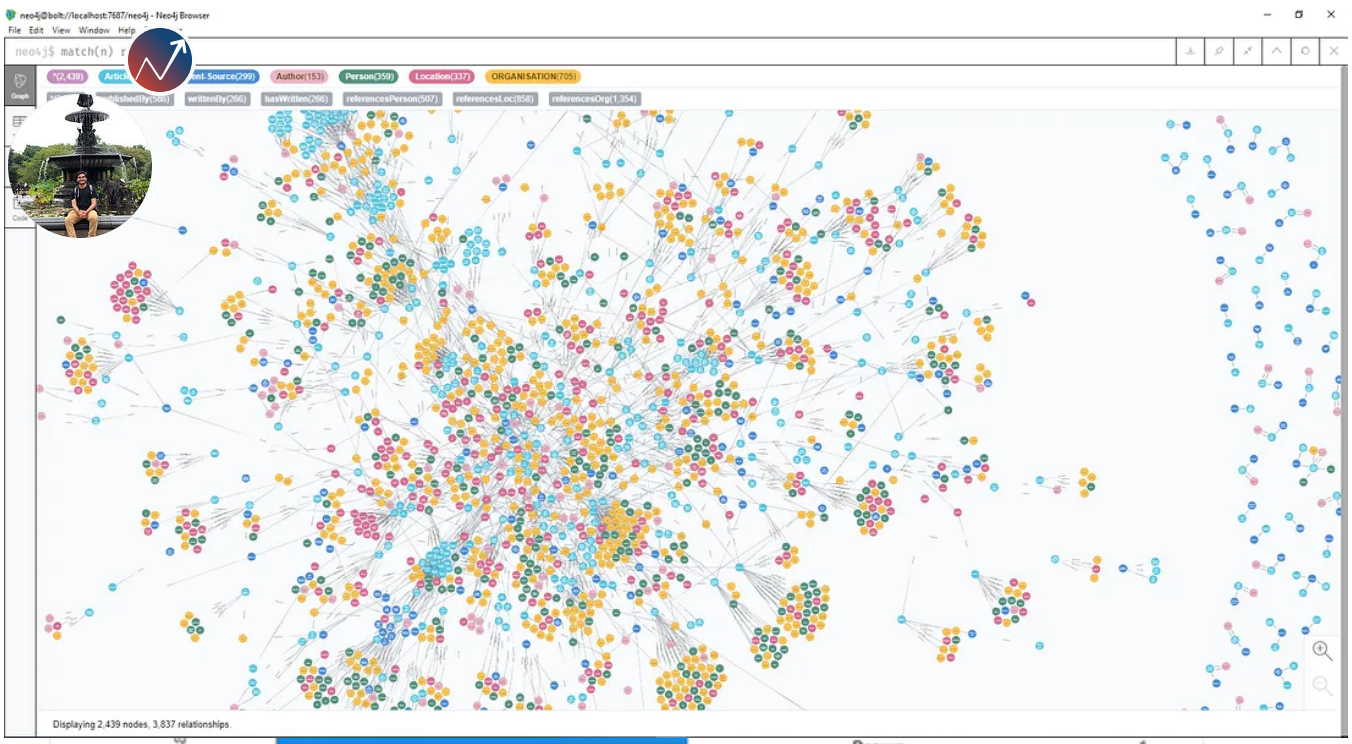
Let's say I want to look at all articles written by 'Business Wire', then we would execute the below query:

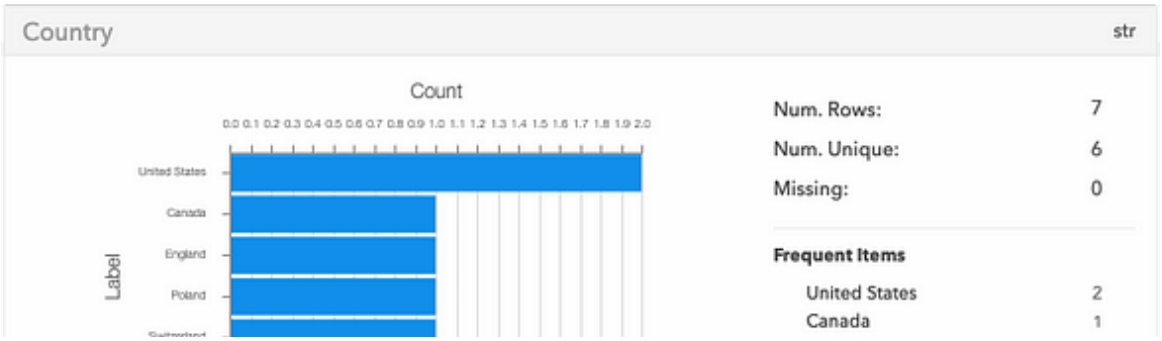
```
MATCH(a: Article)-[:publishedBy]-(s:`Content-Source`)
WHERE s.name='Business Wire'
RETURN a, s
```



Neo4j Py2neo Knowledge Graph Graph Python

Let us visualize everything! This is data related to 586 articles.





Jay Doshi in Analytics Vidhya

Is SFrame better than Pandas?:

Pandas is one of the most used and well-known python packages out there. Short for “Panel Data”; it makes working with tabular data easy.

2 min read · Nov 30, 2021

86 2