

Wykorzystanie bazy grafowej Neo4j

Krzysztof Wyszyński

1. Wstęp

Natłok obowiązków związanych z innymi przedmiotami oraz napięte terminy zmusiły mnie do ograniczenia działań w niniejszym etapie projektu do minimalnych wymagań. Chętnie wrócę jednak do szerszej analizy bazy grafowej **Neo4j** we własnym zakresie w niedalekiej przyszłości.

2. Zbiór danych

W celu zapoznania się z bazą grafową **Neo4j** wykorzystany został zbiór danych dotyczących wpisów w platformie *Stack Overflow*. Składa się on z 6193 węzłów oraz 11540 krawędzi. Dotyczy pytań, odpowiedzi, tagów, komentarzy zawartych w platformie oraz relacji pomiędzy nimi. Zbiór znaleźć można pod adresem: <https://github.com/neo4j-graph-examples/stackoverflow>.

3. Instalacja bazy danych

W ramach projektu postanowiłem wykorzystać chmurową usługę *AuraDB*, która pozwala na zarządzanie bazą danych w środowisku sieciowym. W tym celu po utworzeniu darmowego konta pod adresem <https://console.neo4j.io/> rozpocząłem proces budowania bazy danych:



Let's get started!

Create a new fully-managed
Neo4j database in just a few
clicks.

Create a database

W ramach darmowej oferty *AuraDB* pozwala na stworzenie jednej bazy danych o ograniczonych wymiarach (50 tysięcy węzłów, 175 tysięcy relacji).

Let's create your first database

Step 1 of 1

Database type

less detail ^

AuraDB Free

For learning, prototyping and exploring Neo4j in a forever free instance with basic requirements.

- 1 forever free database
- Limits on graph size (50k nodes, 175k relationships)
- Standard procedure library (apoc-core)
- Auto-pause after 3 days of inactivity

FREE

No credit card required to start

AuraDB Professional

For production applications with scalability, disaster recovery and advanced development needs.

- Unlimited databases
- Starting at 1 GB of RAM
- Scalable on-demand
- Available in all global regions
- Daily backups with 7-day retention
- On-demand snapshots

Starting at

\$65/month

Database details

Database Name

stackoverflow

Następnie należy zapisać otrzymane dane dostępowe do bazy oraz potwierdzić, że zostały umieszczone w bezpiecznym miejscu.

Credentials for stackoverflow

Username

neo4j

Generated password

KPZWaaolTBDZiQeQDKC4uFTRZaHeju0cu441HN9NS8l

Once dismissed, these credentials will no longer be retrievable.
We recommend you update the password once your database is created.

☐ I have stored these credentials safely to use later

Continue

W tym momencie baza danych jest tworzona. Proces może potrwać do kilku minut.

[stackoverflow](#) FREE

Setting up your database

(this will take a few minutes)

Connection URI

Region Belgium (europa-west1)

Neo4j Version 4

Nodes 0 / 50000 (0%)

Relationships 0 / 175000 (0%)

Open with

Aby uzyskać dostęp do bazy danych można wykorzystać portal *Neo4j Browser*. Należy wprowadzić wcześniej zapisane dane w celu zalogowania się do środowiska.

Database access not available. Please use `:server connect` to establish connection. There's a graph waiting for you.

`$:server connect`

Connect to Neo4j

Database access might require an authenticated connection

Connect URL

neo4j+s://71adab9f.databases.neo4j.io:7687

Authentication type

Username / Password

Username

neo4j

Password

Connect

W przypadku pomyślnego logowania zwracana jest następująca informacja:

`$:server connect`

Connected to Neo4j


Nice to meet you.

You are connected as user `neo4j`
to `neo4j+s://71adab9f.databases.neo4j.io:7687`
Connection credentials are not stored in your web browser.

Baza danych jest gotowa do użytku. Poniżej przedstawiono informacje dotyczące jej stanu:

Database Information

Use database

neo4j 

Node Labels

There are no labels in database

Relationship Types

No relationships in database

Property Keys

There are no properties in database

Connected as


Username:

 neo4j

Roles:

 PUBLIC

Disconnect:

 :server disconnect

DBMS

Cluster role:

 LEADER

Version:

 4.4.0


Edition:

 Enterprise


Name:

 neo4j


Databases:

 :dbs

Information:

 :sysinfo

Query List:

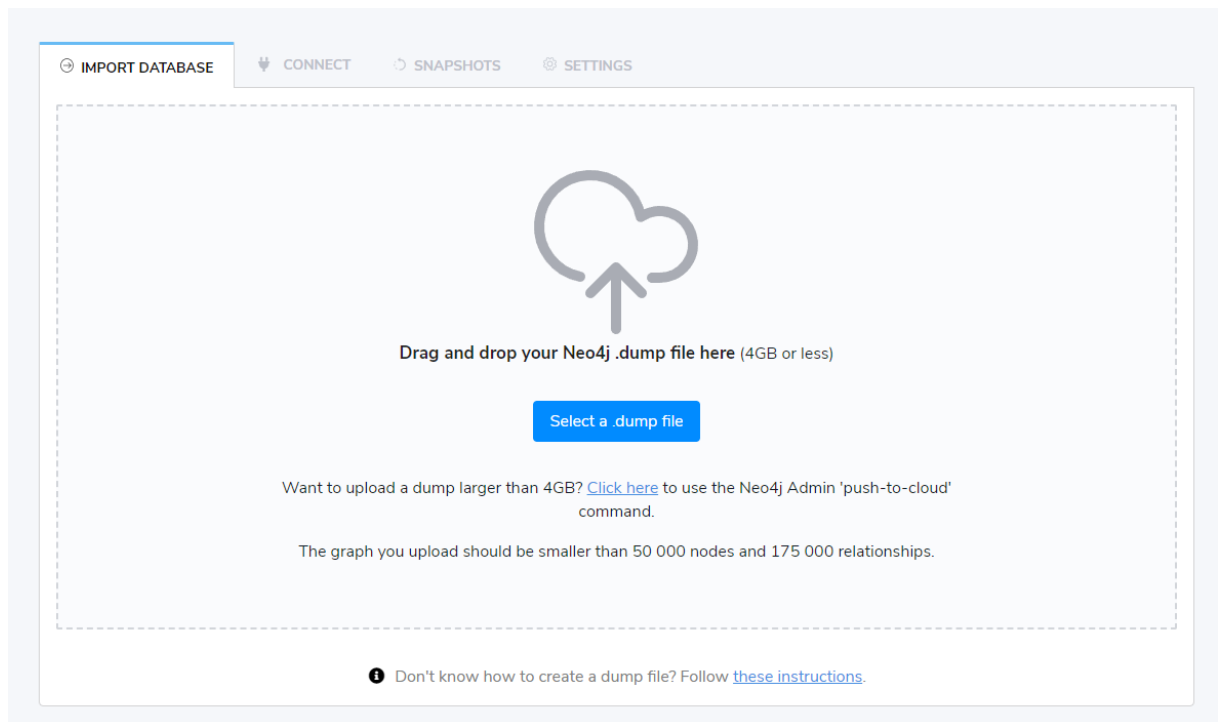
 :queries

Jak można zauważyć, aktualnie w bazie nie ma żadnych danych. Proces wprowadzania danych zostanie przedstawiony w następnym kroku.

4. Import danych

Narzędzie *AuraDB* pozwala na import danych w bardzo prosty sposób. Wystarczy dostarczyć do platformy odpowiedni plik z rozszerzeniem *.dump*. Dokumentacja

Neo4j zawiera również instrukcje dotyczące innych metod importu danych, np. za pomocą plików *.csv*.





Data Import

- Importing CSV
- Importing API Data
- Import: RDBMS to Graph
- How-To: Import Northwind Dataset
- How-To: Desktop CSV Import
- Example Datasets

W przypadku realizowanego projektu wykorzystany został plik *stackoverflow-4-3-1.dump*. Proces importu zajął kilka minut. Po zalogowaniu do platformy *Neo4j Browser* baza danych była już wypełniona danymi.

Database Information

Use database

neo4j  

Node Labels

*(6,193)

Answer

Comment

Question

Tag

User

Relationship Types

*(11,540)

ANSWERED

ASKED

COMMENTED

COMMENTED_ON

PROVIDED

TAGGED

Property Keys

accepted_answer_id

answer_count

body_markdown

creation_date

display_name

favorite_count

is_accepted

link

name

score

share_link

title


uuid

view_count

Connected as

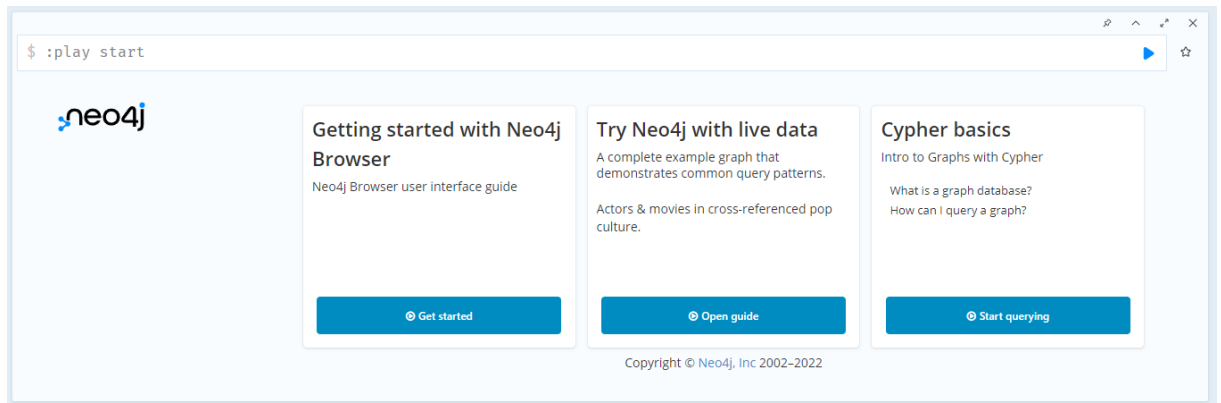
Username: neo4j

Roles: PUBLIC

Disconnect:  :server disconnect

5. Obsługa bazy danych

Portal *Neo4j Browser* oferuje kilka prostych tutoriali pozwalających na zaznajomienie się z podstawami obsługi bazy danych oraz języka zapytań *Cypher*.



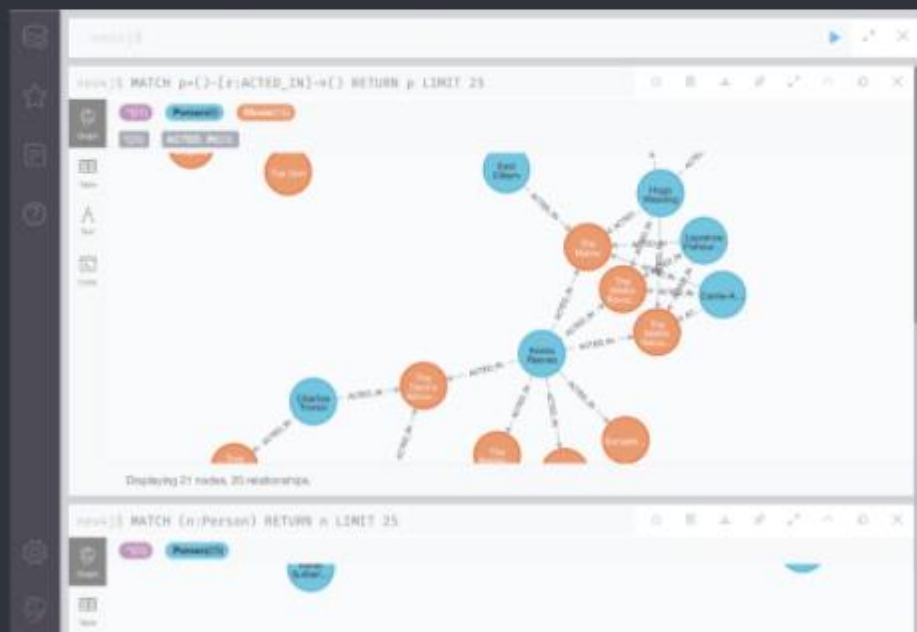
Intro Guide

Result frame

Most recently executed command or Cypher query

A result frame is created for each execution and added to the top of the stream to create a scrollable collection in reverse chronological order.

- A pinned frame always stays in the same position.
- You can clear the stream of result frames by running the `:clear` command.
- The maximum number of result frames displayed in the stream is 30. You can change this number in the Browser Settings drawer.
- You can bring up the history of the executed commands and queries by running the `:history` command.



Cypher Guide

CREATE

Create a node

Let's use Cypher to generate a small social graph.

NOTE: This guide assumes that you use an empty graph.

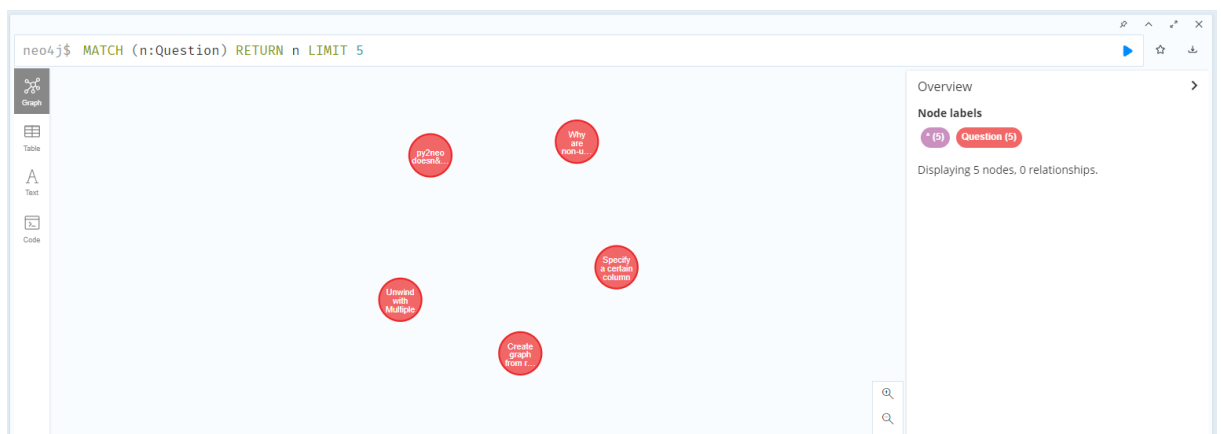
1. Click this code block and bring it into the Editor:

```
CREATE (ee:Person {name: 'Emil', from: 'Sweden', kloutScore: 99})
```

- **CREATE** creates the node.
 - **()** indicates the node.
 - **ee:Person** – **ee** is the node variable and **Person** is the node label.
 - **{}** contains the properties that describe the node.
2. Run the Cypher code by clicking the **Run** button.

Rezultat zapytań demonstrowany jest za pomocą interfejsu graficznego, lecz istnieje też możliwość wyboru formy odpowiedzi, np. tekstowej, tabelarycznej lub w formacie JSON.

Poniżej przedstawiono rezultat trywialnego zapytania, pobierającego 5 pytań zapisanych w bazie:



W polu tekstowym w górnej części ekranu widnieje treść zapytania w składni języka *Cypher*:

```
MATCH (n:Question) RETURN n LIMIT 5
```

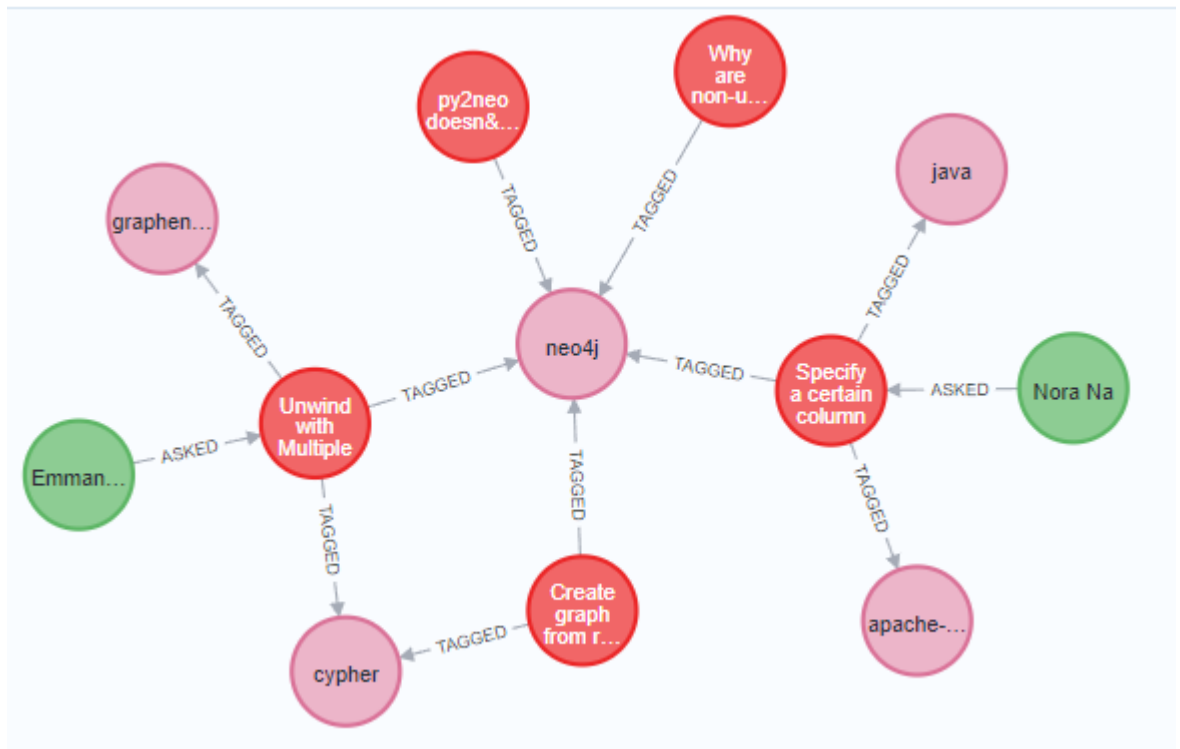
Zrzut ekranu przedstawia rezultat zapytania – 5 węzłów spełniających warunek zapytania. Za pomocą interfejsu graficznego można uzyskać szczegółowe dane dotyczące węzłów – umieszczenie kursora nad konkretnym węzłem wyświetla informacje w nim zawarte.

Node Properties

Question

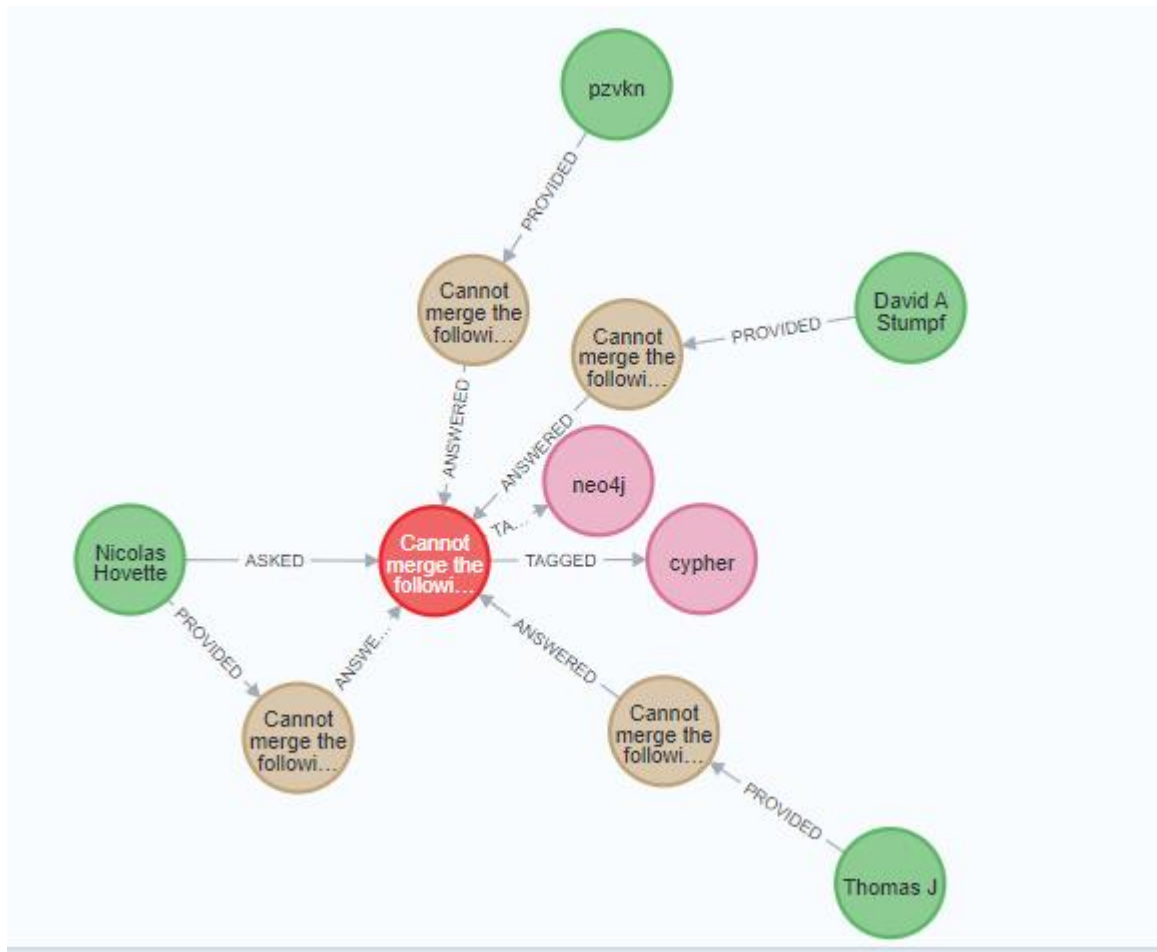
<id>	1	
answer_count	0	
body_markdown	I have a dataset in java which contains an "Id" column that provides a unique Id for each row. I want to write the contents of this dataset ... Show all	
creation_date	1610521682	
link	https://stackoverflow.com/q/65697147	
title	Specify a certain column as id values in Neo4j Spark connector	
uuid	65697147	
view_count	30	

Węzeł *Question* posiada pola dotyczące tytułu, treści, liczby wyświetleń, odpowiedzi itp. Naciśnięcie na węzeł powoduje wyświetlenie relacji z nim związanych.



Jak można zauważyć, pytania związane są z tagami relacją *TAGGED* oraz z użytkownikami relacją *ASKED*. W przypadku ukazanych pytań nie udzielono odpowiedzi, dlatego też liczba relacji jest ograniczona. Spróbujmy znaleźć bardziej rozbudowane pytania, np. takie, na które uzyskano więcej niż 3 odpowiedzi.

MATCH (q:Question)-[a:ASKED]-(u:User) **where** q.answer_count > 3 **return** q,u,a

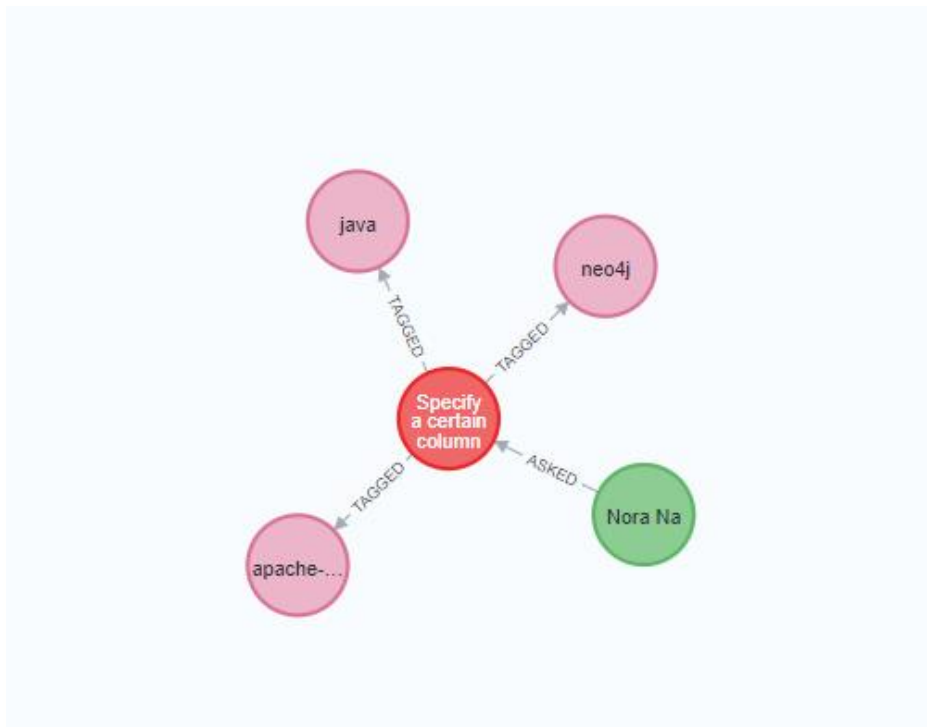


W tym przypadku można rozwinąć strukturę otrzymanej odpowiedzi o węzeł *Answered* oraz użytkowników, którzy udzielili odpowiedzi połączonych z węzłem *Answer* relacją *PROVIDED*.

Aby pozyskać wszystkie relacje pierwszego rzędu związane z danym węzłem można wykonać przykładowe zapytanie:

```
match(q:Question{title:'Specify a certain column as id values in Neo4j Spark connector'})-[r]-
(b) return r,q,b
```

W rezultacie zwrócony zostanie następujący wynik:



6. Proste zapytania

Zapytanie w języku Cypher rozpoczyna się słowem kluczowym *match*. Następnie podajemy węzły, których dotyczy zapytanie oraz relacje pomiędzy nimi. Relacja może być nieskierowana (--) bądź skierowana (->)(<-). Każde zapytanie musi zwrócić rezultaty, a zwracane elementy umieścić należy po słowie kluczowym *return*.

Spróbujmy wykonać proste zapytania:

1. Znajdź 20 najbardziej popularnych tagów wraz z sumą pytań dotyczących każdego z nich oraz sumą odpowiedzi udzielonych w ramach wszystkich zapytań dotyczących danego tagu.

```

match(t:Tag)--(q:Question)--(a:Answer)
return t.name as Tag,count(q) as QuestionCount, sum(q.answer_count) as AnswerCount
order by QuestionCount desc limit 20
  
```

"Tag"	"QuestionCount"	"AnswerCount"
"neo4j"	1318	1776
"cypher"	993	1385
"graph-databases"	105	141
"neo4j-apoc"	101	125
"graph"	61	87

"python"	61	87
"database"	53	75
"java"	42	54
"spring-data-neo4j"	42	62
"redisgraph"	29	37
"node.js"	28	38
"py2neo"	27	35
"spring-boot"	23	35
"javascript"	23	31
"graphql"	22	24
"csv"	19	19
"spring"	18	30
"json"	18	24
"docker"	16	16
"relationship"	16	26

2. Znajdź 10 użytkowników o największej sumarycznej liczbie punktów uzyskanych za odpowiedzi na pytania.

```
match(u:User)--
(a:Answer) return u.display_name as User, sum(a.score) as Score order by Score desc limit 10
```


"User"	"Score"
"cybersam"	132
"Graphileon"	70
"InverseFalcon"	51
"Tomaž Bratanič"	51
"jose_bacoy"	41
"Christophe Willemsen"	28
"fbiville"	19
"Luanne"	19
"SWilly22"	18
"Rajendra Kadam"	15

3. Znajdź wszystkie pytania i wszystkie odpowiedzi oraz zwróćmy je w formie jednej kolumny. Jest to operacja bardzo podobna do języka SQL, należy wykorzystać słowo kluczowe *UNION* oraz upewnić się, że oba zapytania zwracają wynik oznaczony tą samą nazwą.

```

1 match (q:Question) return q.title as QA
2 union all
3 match (a:Answer) return a.title as QA

```

Table

Text

Code

"QA"

"Create graph from recursive JSON data using apoc.load.json and use UNWIND and FOREACH for setting property"

"Specify a certain column as id values in Neo4j Spark connector"

"Why are non-unique indexes dropped and created when restarting the server?"

"py2neo doesn't update database nodes"

"Unwind with Multiple OPTIONAL MATCH returns duplicates"

"Neo.ClientError.Statement.ExternalResourceFailed error on loading CSV file from local"

"LIMIT not working inside apoc procedure apoc.cypher.run"

"Neomodel: specify neo4j database name in connection string"

"Neo4j ETL takes time to migrate from RDBMS to Graph"

7. Indeksy

W bazie neo4j indeksy wykorzystywane są tylko do znalezienia punktu startowego – każdy węzeł odnosi się bezpośrednio do swoich sąsiadów, zamiast polegać na globalnym indeksie, tak jak działa to w bazach relacyjnych. Dlatego jedynym zadaniem indeksu w bazie grafowej będzie przyspieszenie znalezienia węzła, od którego rozpocząć będzie się trawersacja. Poniżej przedstawiono przykładową składnię tworzenia indeksu za pomocą języka Cypher:

Table 1. Syntax for managing indexes

Command	Description	Comment
<div> <div>Cypher</div> <div>Copy to Clipboard</div> </div> <pre> CREATE [BTREE] INDEX [index_name] [IF NOT EXISTS] FOR (n:LabelName) ON (n.propertyName) [OPTIONS "{ option: value[, ...] }"] </pre>	<p>Create a single-property index on nodes.</p> <p>Index provider and configuration can be specified using the <code>OPTIONS</code> clause.</p> <p>Explicit use of <code>BTREE</code> keyword or b-tree options are deprecated in 4.4 and will be replaced in 5.0.</p>	

Za pomocą dyrektywy *PROFILE* istnieje możliwość profilowania zapytań, czyli weryfikacji ich planu wykonania. Przeanalizujmy proste zapytanie, znajdujące użytkownika o danym pseudonimie:

```
1 profile
2 match(u:User{display_name: 'Simon'}) return u
```



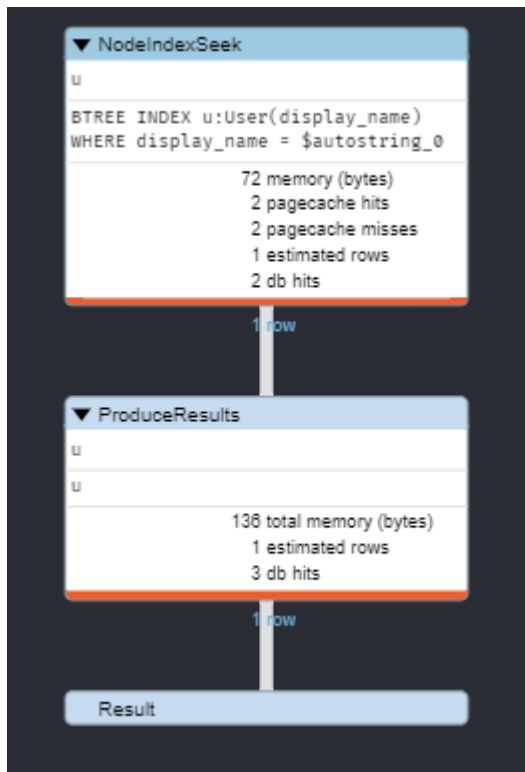
Cypher version: CYPHER 4.4, planner: COST, runtime: PIPELINED. 4099 total db hits in 39 ms.

Jak można zauważyć, w celu znalezienia użytkownika przeskanowano wszystkie 1365 węzłów. Spróbujmy założyć indeks na pole *display_name*:

```
1
2 CREATE INDEX
3 ON :User(display_name)
4
```

Added 1 index, completed after 3 ms.

Poniżej przedstawiono rezultat tego samego zapytania po nałożeniu indeksu:



Cypher version: CYPHER 4.4, planner: COST, runtime: PIPELINED. 5 total db hits in 31 ms.

Jak można zauważyć, różnica jest kolosalna – użytkownik został znaleziony natychmiastowo, poprzez wykonanie *NodeIndexSeek*, czyli przeszukania indeksu.

8. Procedury

Za pomocą komendy *show procedures* można wyświetlić listę procedur dostępnych w zainstalowanej wersji *neo4j*.

neo4j\$ show procedures

	name	description
	"dbms.cluster.role"	"The role of this instance in the cluster for the specified database."
181	"dbms.cluster.routing.getRoutingTable"	"Returns endpoints of this instance."
182	"dbms.components"	"List DBMS components and their versions."
183	"dbms.database.state"	"The actual status of the database with the provided name on this neo4j instance."
184	"dbms.functions"	"List all functions in the DBMS."
185	"dbms.info"	"Provides information regarding the DBMS."
186		

Started streaming 227 records after 49 ms and completed after 91 ms.

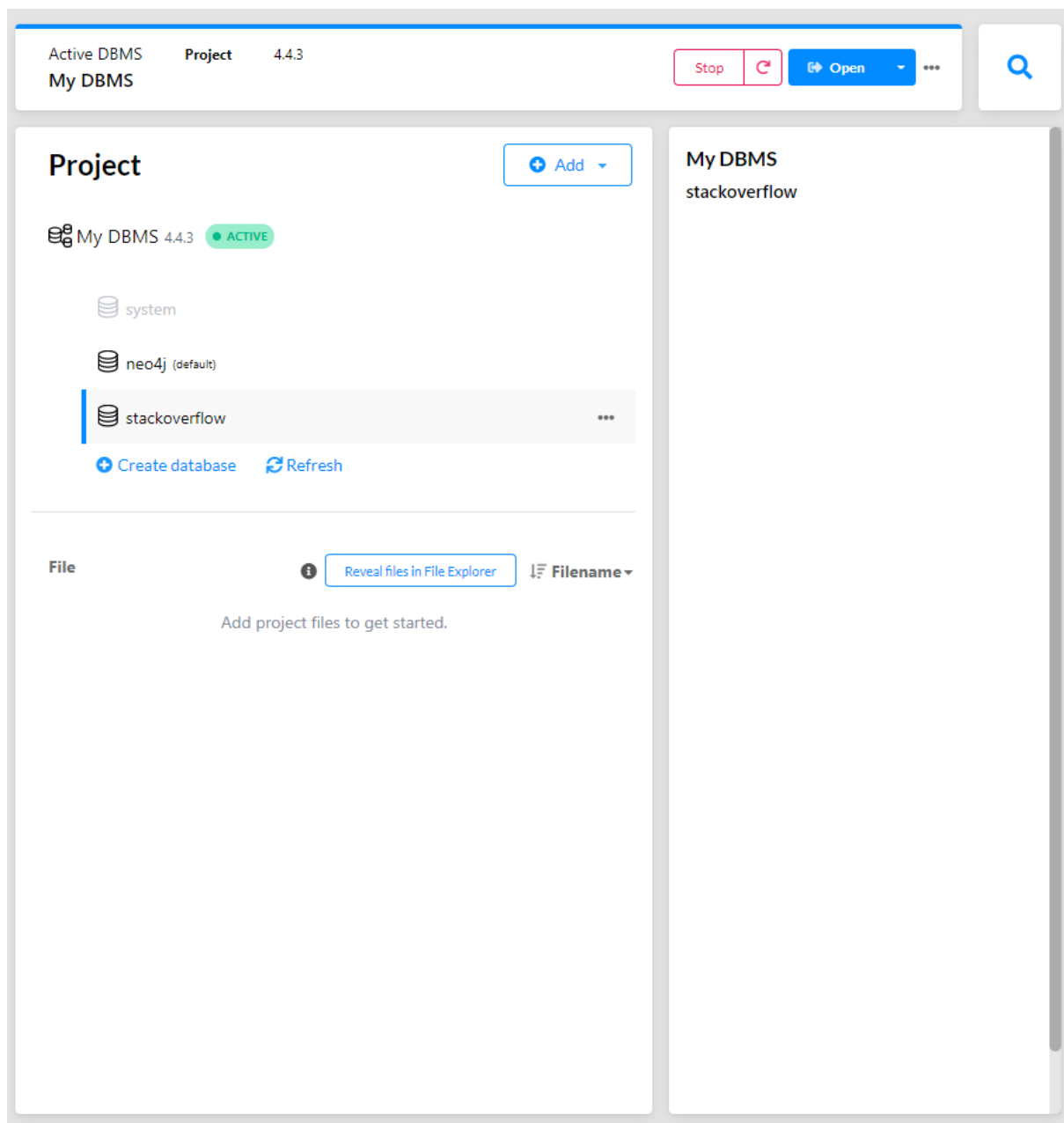
Według dokumentacji, procedura w *neo4j* to mechanizm, który pozwala *neo4j* być rozszerzonym poprzez generację kodu, który może być pozyskany bezpośrednio przez Cypher. Procedury mogą zawierać argumenty, wykonywać operacje na bazie danych i zwracać wyniki.

Procedury napisane są w języku Java oraz kompilowane do plików *.jar*. Mogą zostać wdrożone do bazy danych poprzez umieszczenie pliku *.jar* do odpowiedniego folderu *\$NEO4J_HOME/plugins*. Baza danych musi zostać zrestartowana, aby pozyskać nowe procedury. Przykładowym zastosowaniem procedur może być:

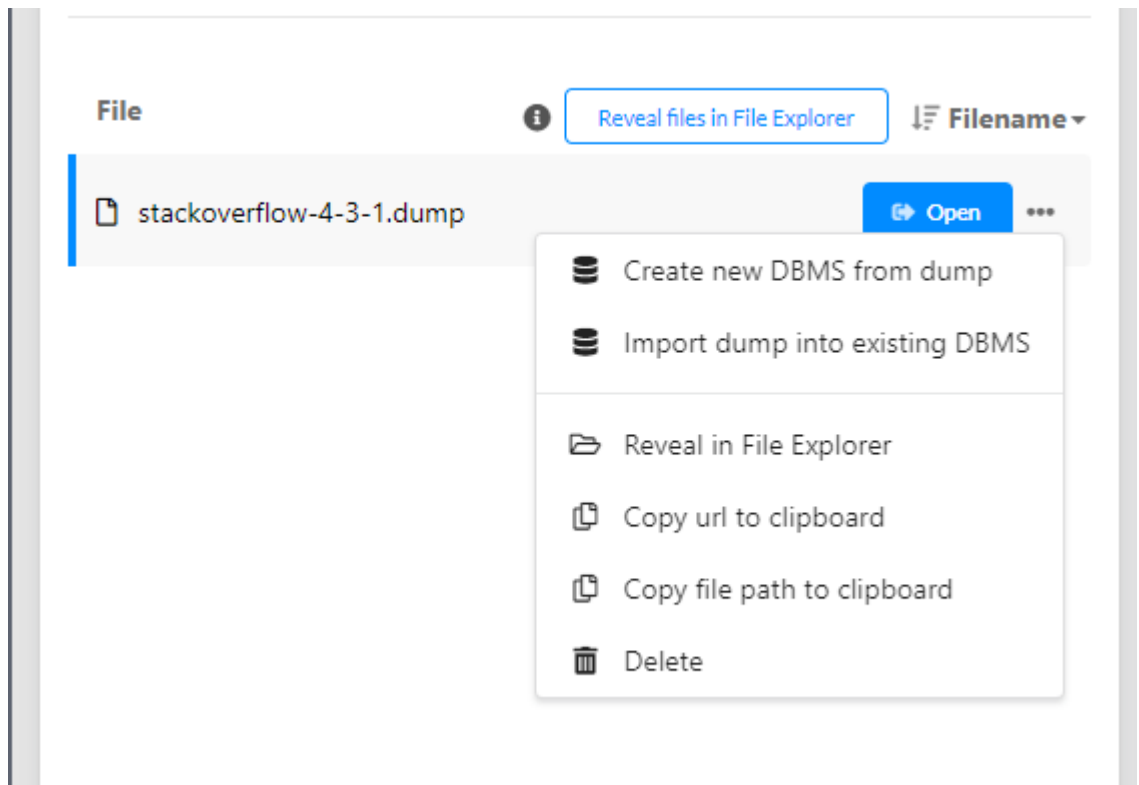
1. Umożliwienie dostępu do funkcjonalności niedostępnej w Cypher
2. Umożliwienie dostępu do innych systemów
3. Wykonywanie operacji globalnych na grafie, takich jak znajdowanie gęstych węzłów, zliczanie połączonych komponentów
4. Wyrażenie operacji proceduralnej, którą trudno wyrazić w Cypher

Ponieważ stworzenie procedury w Javie jest dosyć złożonym procesem, istnieje możliwość skorzystania z biblioteki APOC(Awesome Procedures On Cypher) w celu wygenerowania procedury za pomocą języka Cypher.

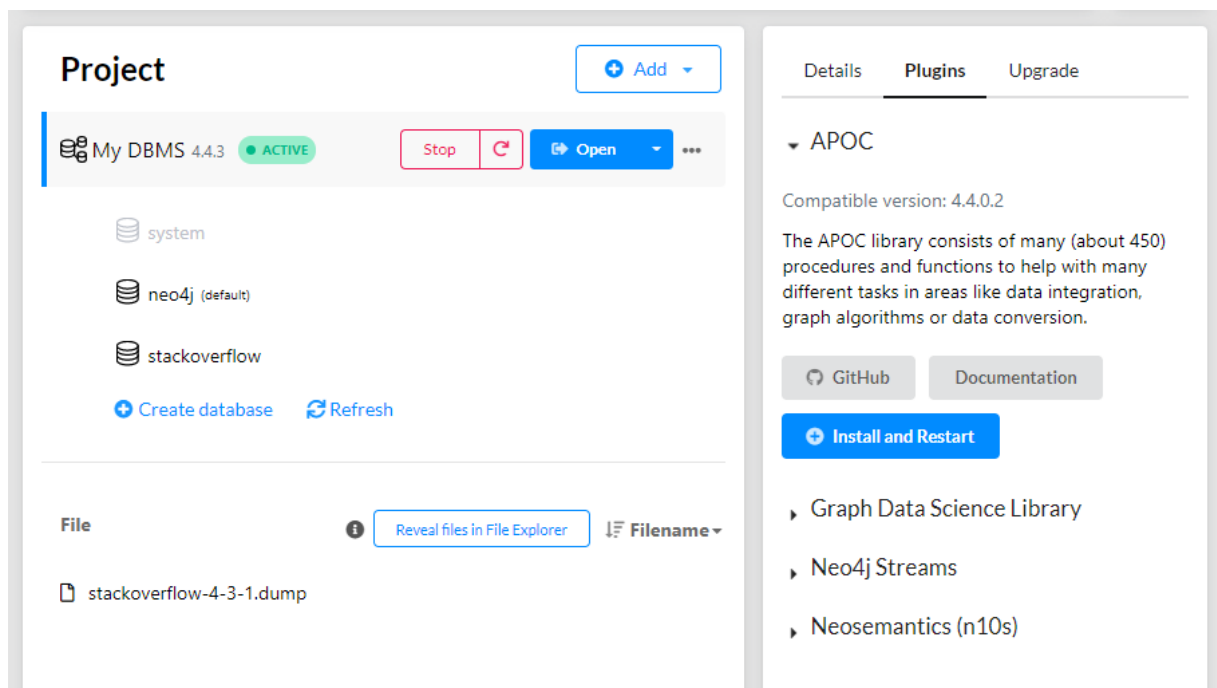
W celu stworzenia procedury koniecznym okazało się zainstalowanie aplikacji *Neo4j Desktop* oraz uruchomienia bazy danych w środowisku lokalnym.



Stworzyłem nowy system bazodanowy oraz zaimportowałem do niego bazę danych:



Kolejnym krokiem było zainstalowanie biblioteki APOC:



Następnie można już wykorzystać bibliotekę APOC do generacji procedur.

Stwórzmy prostą procedurę, która jako argumenty przyjmuje wartości uuid dwóch odpowiedzi i zwraca sumę ich punktów:

```

1 CALL apoc.custom.asProcedure(
2   'addScores',
3   'MATCH(a:Answer)
4   WHERE a.uuid IN [$uuid1, $uuid2]
5   | RETURN a.score AS scores',
6   'read',
7   [['scores', 'int']],
8   [['uuid1', 'int'], ['uuid2', 'int']]
9 )
10

```

```
stackoverflow$ CALL custom.addScores(68729855, 66767861);
```

	scores
1	6

9. Podsumowanie

Poniżej przedstawię swoje odczucia po dość krótkiej konfrontacji z bazą neo4j.

1. Baza neo4j, podobnie jak MongoDB zawiera nowoczesne rozwiązania chmurowe, pozwalające na szybkie i proste rozpoczęcie pracy.
2. Interfejsy graficzne *AuraDB*, *Neo4jBrowser* są przyjazne użytkownikowi. Dodatkowo, ciekawym pomysłem są krótkie poradniki wstępne dla początkujących.
3. Składnia języka Cypher w moim subiektywnym odczuciu ma dosyć wysoki próg wejścia. W bardzo łatwy sposób można konstruować za pomocą tego języka proste zapytania, lecz w przypadku bardziej skomplikowanych operacji, trudno znaleźć prawidłowe rozwiązanie.
4. Dokumentacja *neo4j* nie pozwoliła mi na szybkie znalezienie odpowiedzi w przypadku wątpliwości. Być może poświęciłem jej zbyt mało czasu. Również znalezienie rozwiązania za pomocą platformy *Stack Overflow* sprawiało wiele problemów.

5. Baza *neo4j*, zgodnie z założeniami teoretycznymi baz grafowych, jest niewydajna w przypadku zapytań globalnych, np. pobrania wszystkich węzłów. Natomiast posiada bardzo dobre mechanizmy indeksowania oraz sprawdza się świetnie przy wyszukiwaniu lokalnych rozwiązań.