**Obraz zawierający tekst, Czcionka, logo, wizytówka

Opis wygenerowany automatycznie**

**Sieci Semantyczne**

*Grafowa baza danych o publikacjach naukowych*

Prowadzący: Autor:

dr inż. Wojciech Kozioł Filip Cyboran

122926

Kierunek: Informatyka, grupa: Lab 02

Rzeszów r.a. 2023/2024

Spis treści

[1. Opis Projektu 2](#_Toc170817440)

[2. Źródła danych 2](#_Toc170817441)

[3. Struktura bazy danych 2](#_Toc170817442)

[Węzły 2](#_Toc170817443)

[Relacje 4](#_Toc170817444)

[4. Sposób wypełnienia bazy danych 6](#_Toc170817445)

[5. Uruchomienie skryptów wypełniających bazę 8](#_Toc170817446)

[6. Zapytania do bazy w języku Cypher 8](#_Toc170817447)

[7. Możliwe wykorzystanie bazy 14](#_Toc170817448)

[8. Podsumowanie 14](#_Toc170817449)

## Opis Projektu

Projekt miał na celu stworzenie grafowej bazy danych zawierającej wybrane informacje o publikacjach naukowych. Baza zawiera liczne wpisy na temat publikacji, jak i autorów i daty w których te publikacje zostały opublikowane.

Dane zostasły pozyskane za pomocą scrappera napisanego w języku Python. Dodawanie danych do bazy danych jest realizowane w procesie scrappowania za pomocą biblioteki neo4j z GraphDatabase.

## Źródła danych

Źródłem dla projektu była strona [arxiv.org](https://arxiv.org/), która zawiera informacje na temat licznych publikacji w wielu dziedzinach nauki.

## Struktura bazy danych

Struktura stworzonej grafowej bazy danych stworzona jest z węzłów, zawierających tytuł publikacji, jej autora jak i rok publikacji, jak i również relacje, w których zawarte zostały pozostałe informacje.

## Węzły

#### Węzeł Author

Reprezentuje autora publikacji

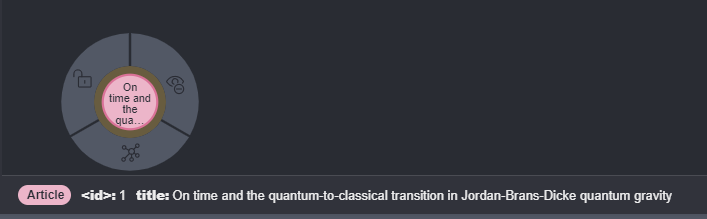
* title: Pierwsze litery imion i nazwisko naukowca.



#### Węzeł Article

Reprezentuje publikacje naukową

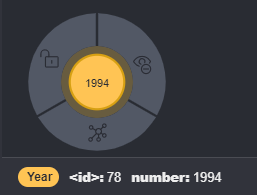
* title: Tytuł publikacji naukowej



#### Węzeł Year

Reprezentuje rok opublikowania

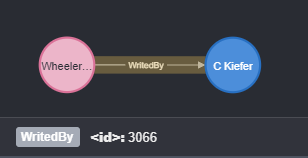
number: rok opublikowania



## Relacje

#### Relacja WritenBy

Relacja łączy publikacje z autorem

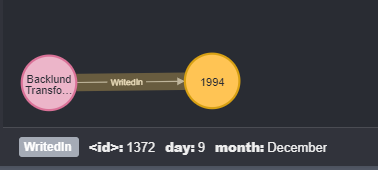


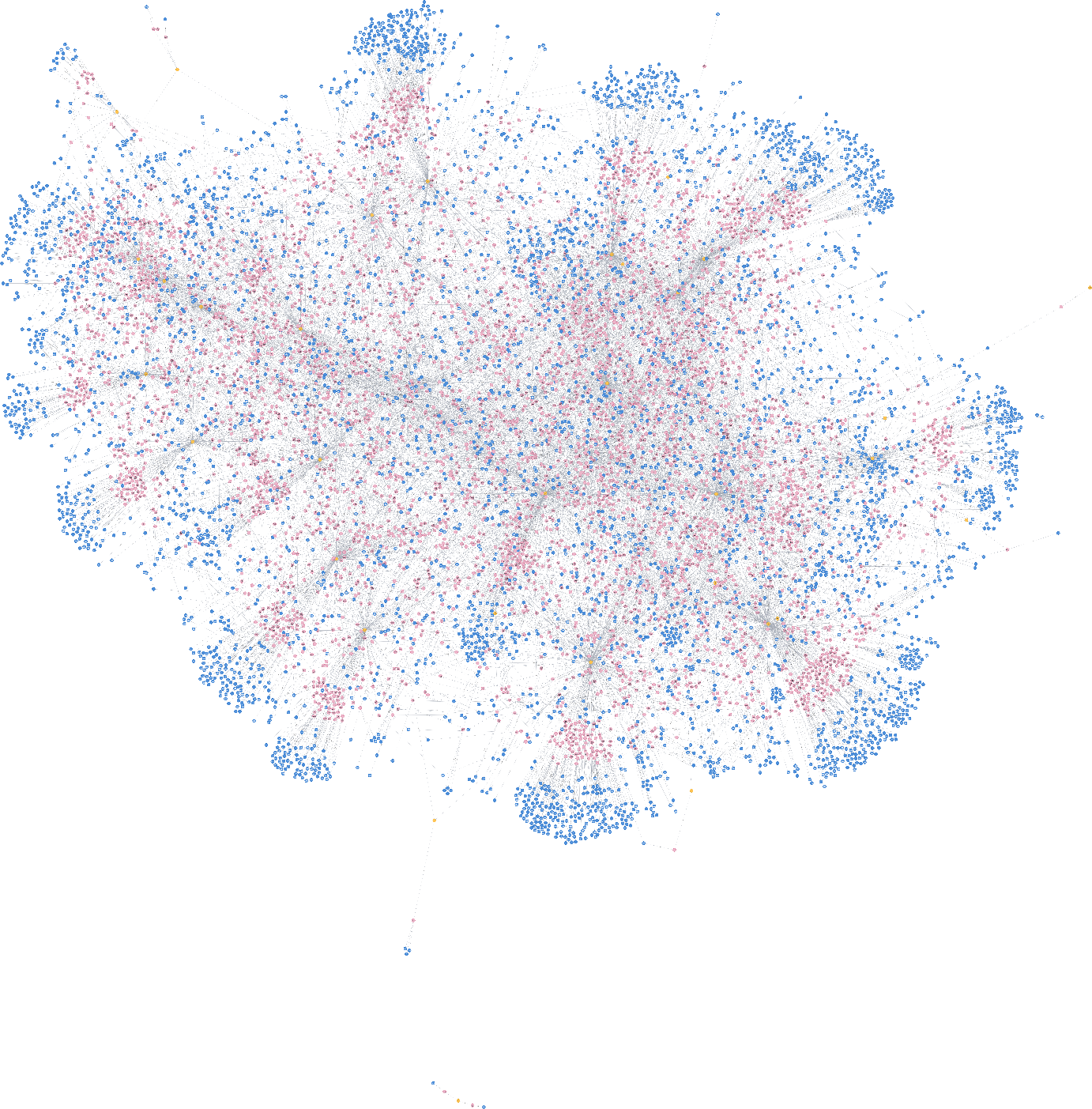
#### Relacja WriteIn

Relacja łączy publikacje z rokiem oopublikowania

Relacja posiada:

* label: month – miesiąc publikacji
* label: day – dzień publikacji





## Sposób wypełnienia bazy danych

Baza danych została wypełniona używając scrappera napisanego w języku Python, Napisany skrypt, przegląda stronę podaną w źródłach i wyciąga z niej informacje potrzebne do stworzenia bazy, po czym przekształca je na Crypt Querry, które następnie jest wysyłane do bazy.

#### Funkcja używana do przygotowanie tekstu:

def clean\_string(raw: str) -> str:  
clean\_text = re.sub(r'\s+', ' ', raw).lstrip().rstrip()  
 clean\_text = re.sub(r'[/.,!?;:\"\']+', '', clean\_text)  
 return clean\_text

Ta funkcja po podaniu w atrybucie raw fragment tekstu pobranego ze strony, zwraca sformatowany tekst tekst, według określonych kryteriów:

* Pojedyncze spacje pomiędzy wyrazami
* Brak spacji na początku tekstu
* Brak spacji na końcu tekstu
* Brak znaków specjalnych

#### Główna funkcja:

def process() -> None:  
  
 driver = GraphDatabase.driver('neo4j://localhost:7687',  
 auth=('neo4j', '1234'))  
 with driver.session() as session:  
 session.run("MATCH (n) DETACH DELETE n")  
 driver.close()  
  
 page = 0  
 total\_counter = 0  
  
 stopper = -1 # quantity of pages to be scrapped -1 for all pages available  
 while stopper != 0:  
 print(f"Scrapping page {page+1}")  
 prompts = []  
 links = []  
  
 # accessing composers page  
 target\_url = f'https://arxiv.org/search/?query=%22quantum+gravity%22&searchtype=all&abstracts=show&order=announced\_date\_first&size=50&start={page\*50}'  
  
 try:  
 # accessing url and parsing html  
 response = requests.get(target\_url)  
 response.raise\_for\_status()  
 soup = BeautifulSoup(response.text, 'html.parser')  
  
 divs = soup.find\_all('li', class\_='arxiv-result')  
  
 if len(divs) < 50:  
 break  
  
 prompt\_list = []  
  
 for div in divs:  
 title = clean\_string(div.find('p', class\_='title is-5 mathjax').get\_text())  
  
  
 if title == "":  
 continue  
  
 authors\_div = div.find('p', class\_="authors")  
 authors\_list = authors\_div.find\_all('a')  
 authors = []  
  
 for author in authors\_list:  
 authors.append(clean\_string(author.get\_text()))  
  
  
 date\_div = div.find('p', class\_="is-size-7")  
 date\_text = clean\_string(date\_div.get\_text().split(";")[0])  
  
 date\_split = date\_text.split(" ")  
 date\_day = date\_split[1]  
 date\_month = date\_split[2]  
 date\_year = date\_split[3]  
  
 prompt = f"MERGE (A:Article {{title:\"{title}\"}})\n"  
  
 iterator = 0  
 for author in authors:  
 prompt += f"MERGE (P{iterator}:Author {{title:\"{author}\"}})\n"  
 prompt += f"Merge (A)-[:WritedBy]->(P{iterator})\n"  
 iterator +=1  
  
 prompt += f"MERGE (Y:Year {{number:\"{date\_year}\"}})\n"  
 prompt += f"Merge (A)-[:WritedIn {{month:\"{date\_month}\", day:\"{date\_day}\"}}]->(Y)\n"  
  
 prompt\_list.append(prompt)  
  
 driver = GraphDatabase.driver('neo4j://localhost:7687',  
 auth=('neo4j', '1234'))  
 for pr in prompt\_list:  
  
 try:  
 driver.session().run(pr)  
 except ValueError as e:  
 print(e)  
 continue  
  
 driver.close()  
  
 except Exception as e:  
 print(e)  
  
 stopper -= 1  
 page += 1  
 print(f"finished scrapping page {page}")

Funkcja ta używa powyższej funkcji, do pobierania danych ze strony, wyciągnięciu wybranych danych, zbudowania querry w oparciu o nie, po czym wysyła to querry do bazy.

## Uruchomienie skryptów wypełniających bazę

Aby prawidłowo stworzyć oraz wypełnić bazę danych należy wykonać następujące kroki:

#### Krok 1: Przygotowanie środowiska

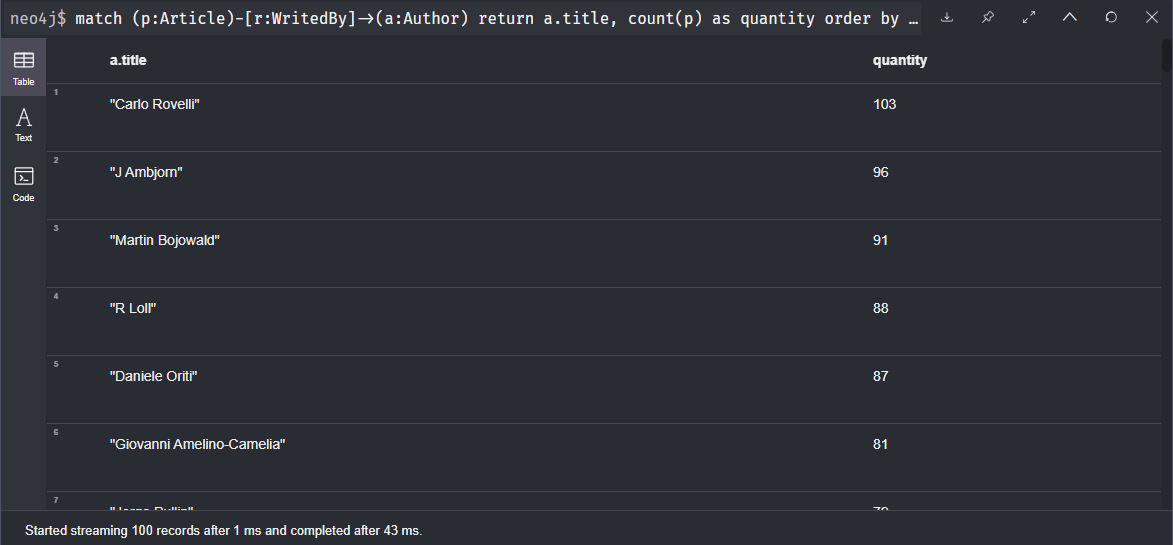
1. **Należy posiadać zainstalowany Python3, jak i wszystkie wymagane biblioteki,**
2. **Należy posiadać zainstalowany Neo4J i utworzoną bazę danych.**
3. **Stałe połączenie z internetem**

#### Krok 2: Uruchomienie Skryptu

**Aby poprawnie uruchomić skrypt, należy się upewnić że baza jest uruchomiona, i że mamy do niej dostęp, a następnie uruchomić funkcję process().**

## Zapytania do bazy w języku Cypher

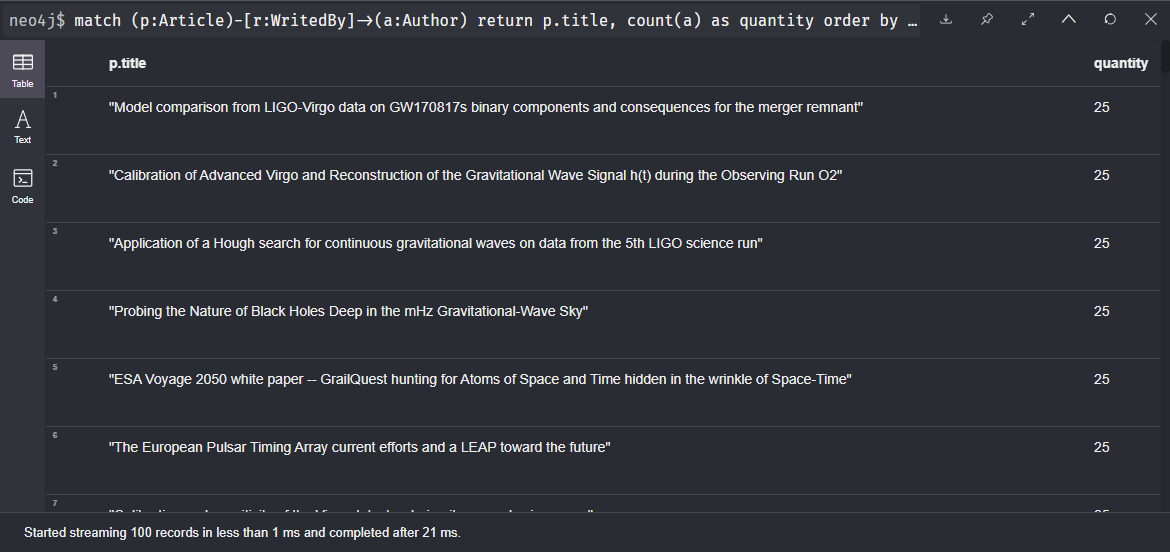
#### Zapytanie zwracające ilość publikacji napisanych przez każdego z naukowców

match (p:Article)-[r:WritedBy]->(a:Author) return a.title, count(p) as quantity order by quantity desc limit 100

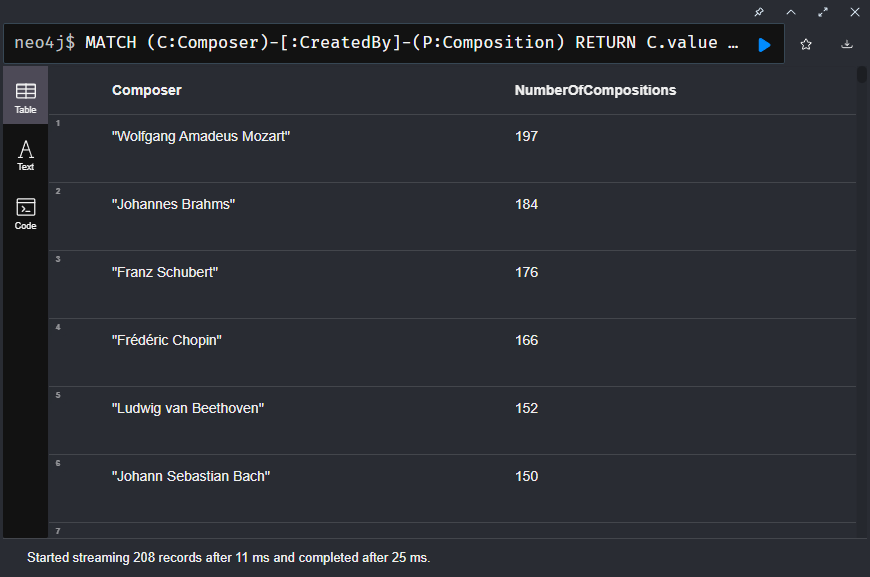
#### Zapytanie zwracające utwory napisane przez György Kurtág z gatunku instrumenty solowe

MATCH (C:Composer {value: "György Kurtág"})-[:CreatedBy]-(P:Composition)-[:InGenreOf]-(G:Genre {value: "Solo instrumental"})

RETURN P.title



#### Zapytanie zwracające ilość naukowców współpracujących nad daną publikacją

match (p:Article)-[r:WritedBy]->(a:Author) return p.title, count(a) as quantity order by quantity desc limit 100

#### Zapytanie zwracające listę publikacji pisanych przez jedną osobę

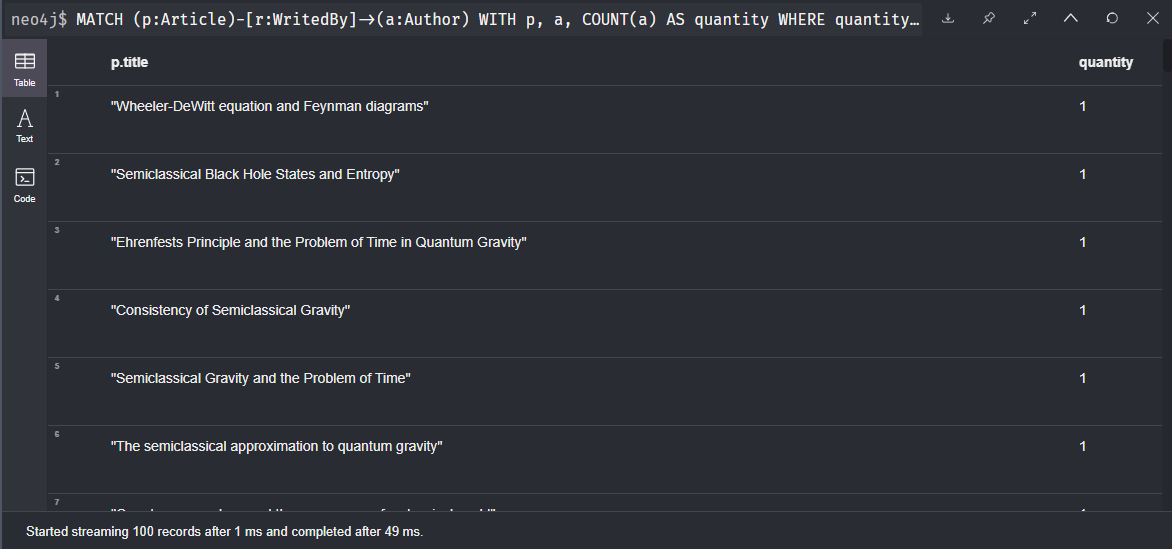
match (p:Article)-[r:WritedBy]->(a:Author)

with p, a, count(a) as quantity

where quantity <= 1

return p.title, quantity

limit 100



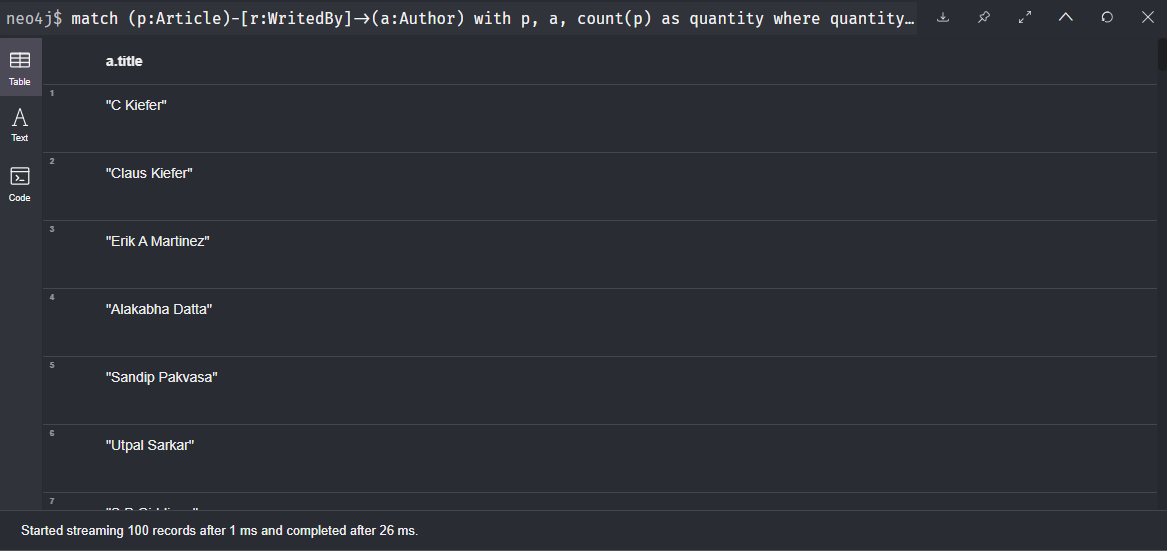
#### Zapytanie zwracające naukowców z jedną opublikowaną publikacją

MATCH (C:Composer)-[:CreatedBy]-(P:Composition)-[:ScoringOn]-(S:Scoring)

WITH P, COUNT(S) AS NumberOfScorings

WHERE NumberOfScorings > 1

RETURN P.title, NumberOfScorings



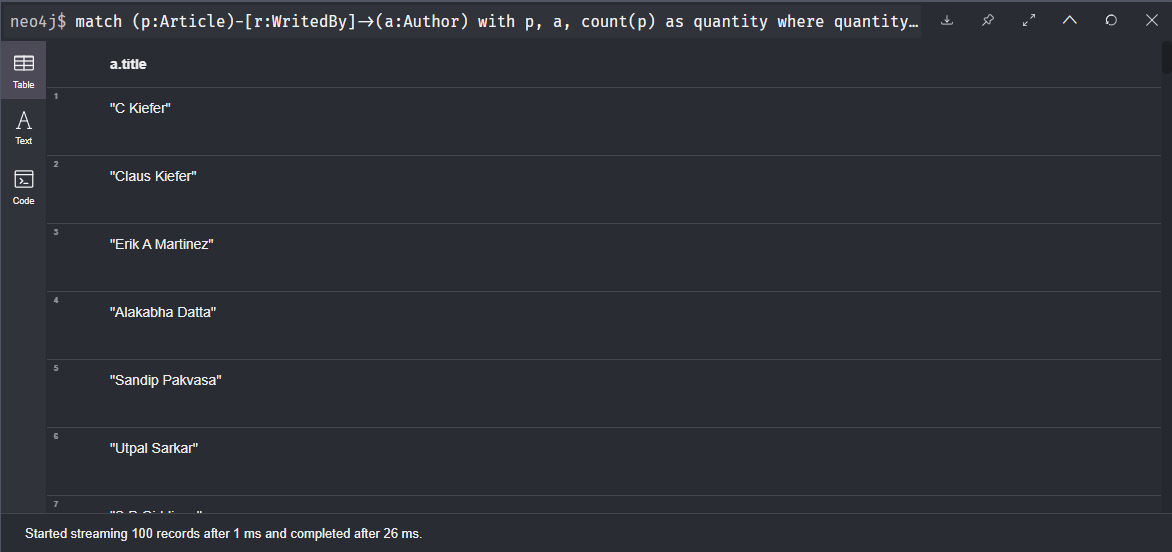
#### Zapytanie zwracające liste naukowców posiadających solowo opracowane publikacje

match (p:Article)-[r:WritedBy]->(a:Author)

with p, a, count(p) as quantity

where quantity <= 1

return distinct a.title

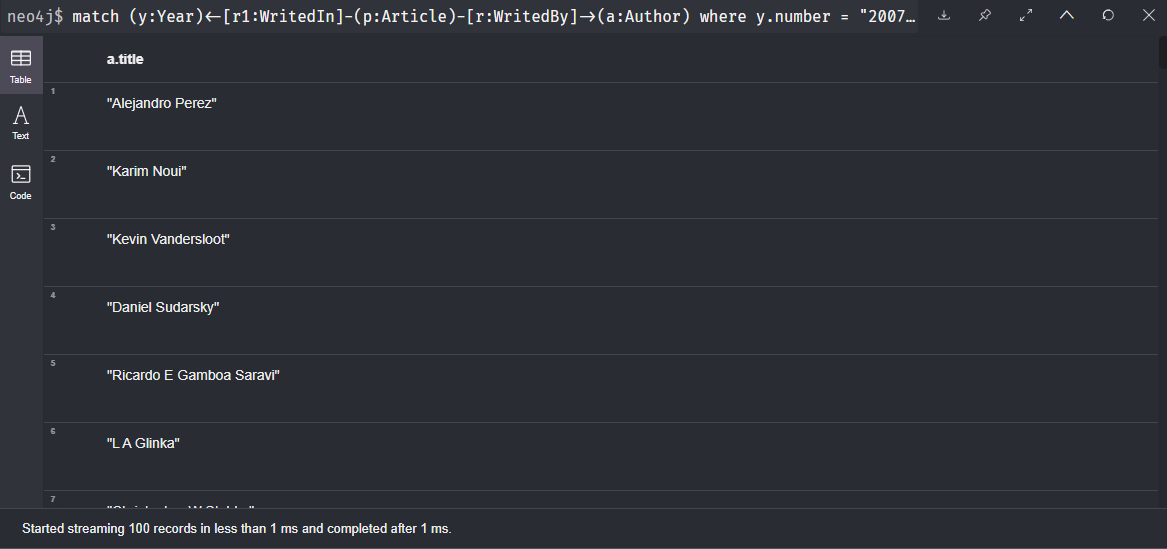
limit 100 

#### Zapytanie zwracające listę naukowców publikujących w 2007 roku

match (y:Year)<-[r1:WritedIn]-(p:Article)-[r:WritedBy]->(a:Author)

where y.number = "2007"

return distinct a.title

limit 100

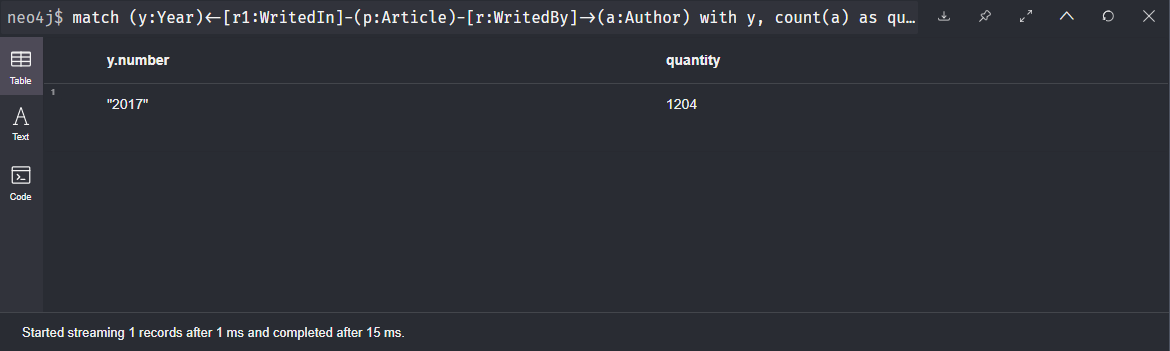
#### Zapytanie zwracające rok z nawiększą ilością publikujących autorów

match (y:Year)<-[r1:WritedIn]-(p:Article)-[r:WritedBy]->(a:Author)

with y, count(a) as quantity

return y.number, quantity

order by quantity desc

limit 1

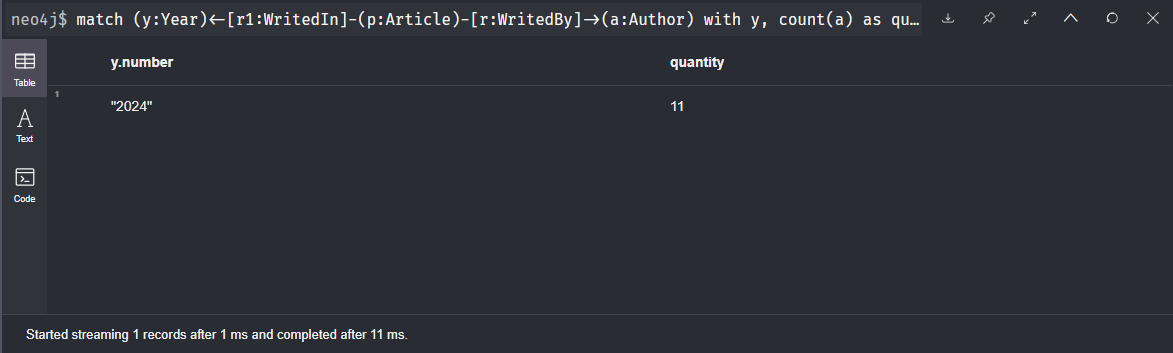
#### Zapytanie zwracające rok o najmniejszej ilości publikujących autorów

match (y:Year)<-[r1:WritedIn]-(p:Article)-[r:WritedBy]->(a:Author)

with y, count(a) as quantity

return y.number, quantity

order by quantity

limit 1

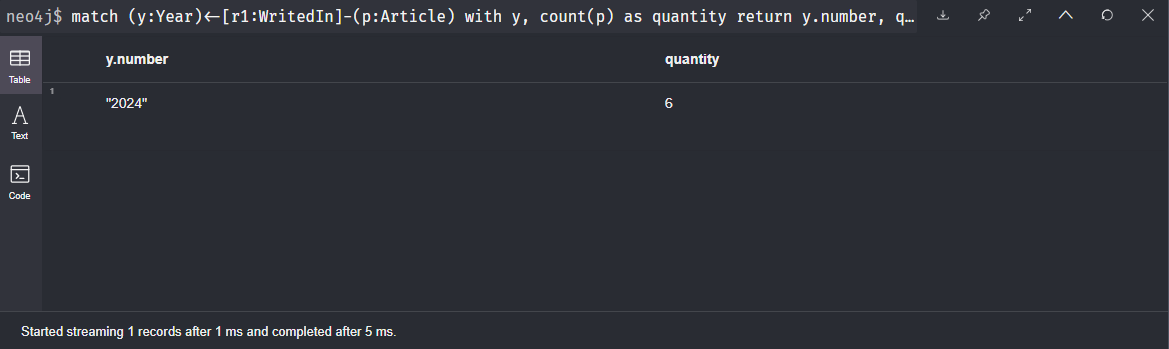
#### Zapytanie zwracające rok o najmniejszej ilości publikacji

match (y:Year)<-[r1:WritedIn]-(p:Article)

with y, count(p) as quantity

return y.number, quantity

order by quantity

limit 1

## Możliwe wykorzystanie bazy

Stworzona przeze mnie baza danych może zostać wykorzystana do wyszukiwania publikacji, i naukowców je opracowujących. Może również posłużyć do analizy przebiegu publikacji na przestrzeni lat jak i przedstawieniu trendów i innych statystyk dotyczących publikacji naukowych.

## Podsumowanie

Grafowa baza danych stworzona w ramach tego projektu gromadzi dane o autorach, publikacjach jak i datach publikacji została stworzona za pomocą scrappera napisanego w języku Python. Stworzona baza danych ma potencjalne zastosowania w analizie publikacji naukowych, umożliwiając badanie trendów publikacyjnych, identyfikację współpracujących naukowców oraz monitorowanie aktywności publikacyjnej w różnych latach. Może być ona również wykorzystana jako narzędzie do szybkiego wyszukiwania informacji o publikacjach i ich autorach.