

Sieci semantyczne

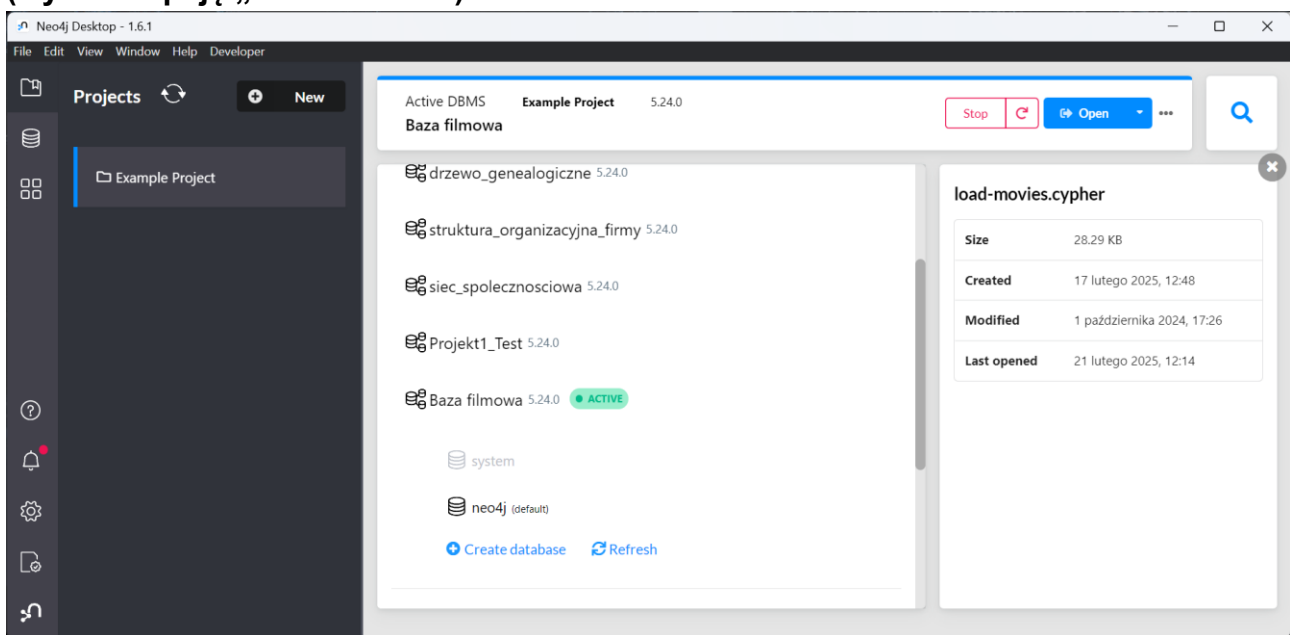
Laboratorium 4: Cypher – zapytania ciąg dalszy.

Prowadzący: pracownik UR

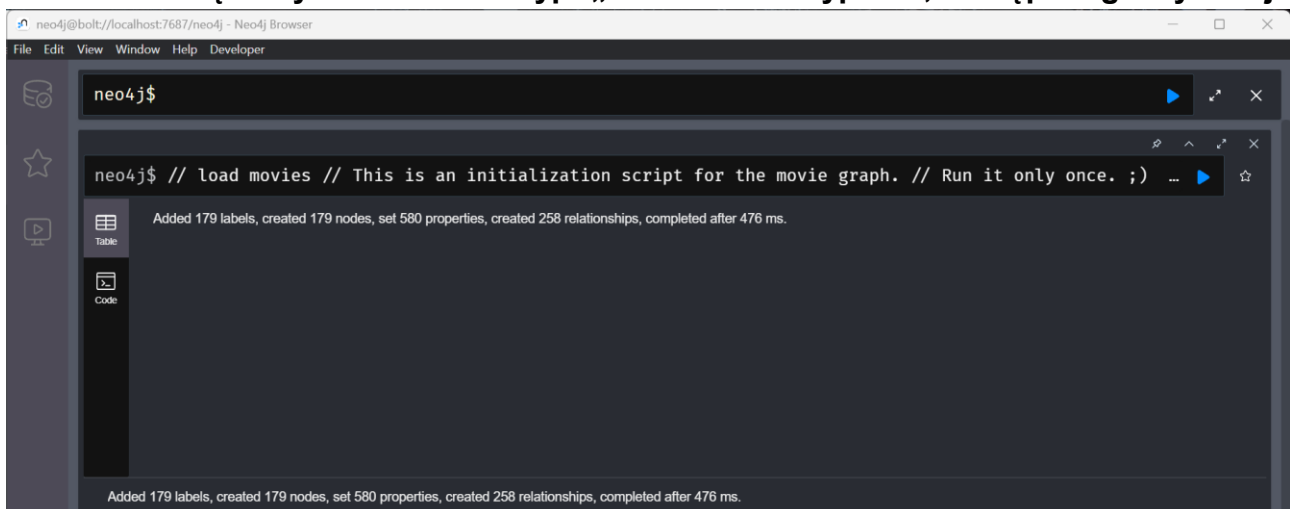
Wykonał: Piotr Rojek, pr125159

Zadanie 1

Utwórz nową bazę danych o nazwie „Baza filmowa” przy użyciu przycisku „Add” (wybierz opcję „Local DBMS”).



Uruchom bazę danych i otwórz skrypt „load-movies.cypher”, następnie go wykonaj.



Zadanie 2

Wyszukaj wszystkie filmy z bazy i ogranicz ich liczbę na wyjściu do 10.

Kod zapytania:

```
MATCH (n:Movie) RETURN n LIMIT 10
```

Wynik zapytania:



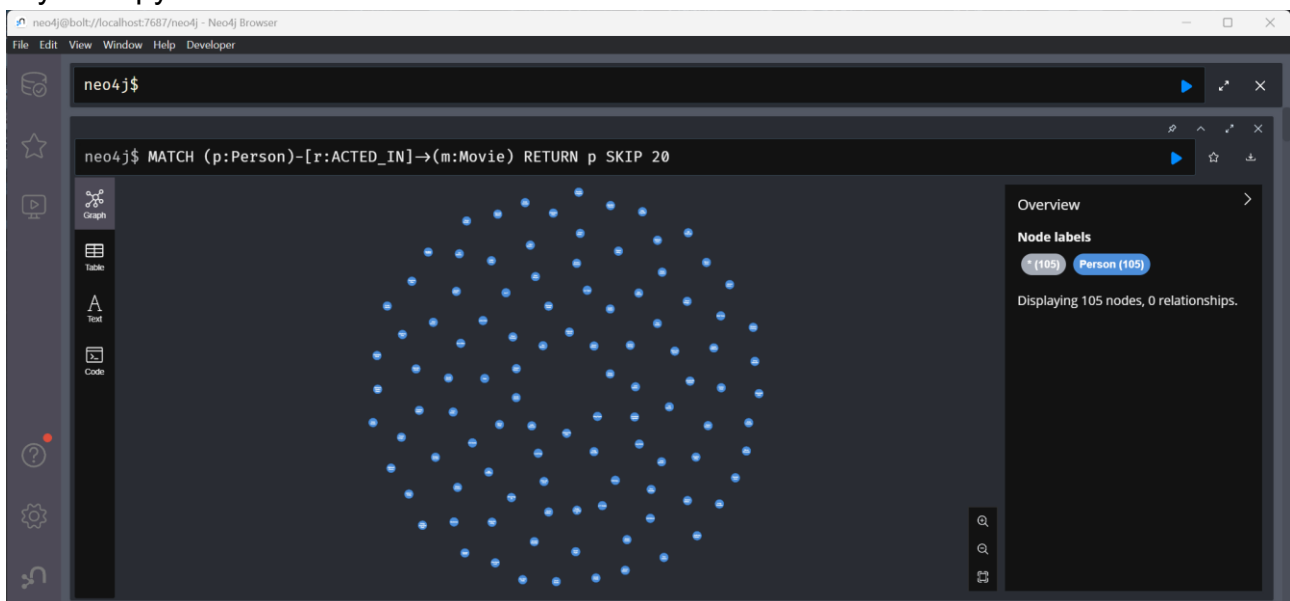
Zadanie 3

Wyszukaj wszystkich aktorów i rozpocznij włączanie na wyjście od 20 węzła.

Kod zapytania:

```
MATCH (p:Person)-[r:ACTED_IN]->(m:Movie) RETURN p SKIP 20
```

Wynik zapytania:



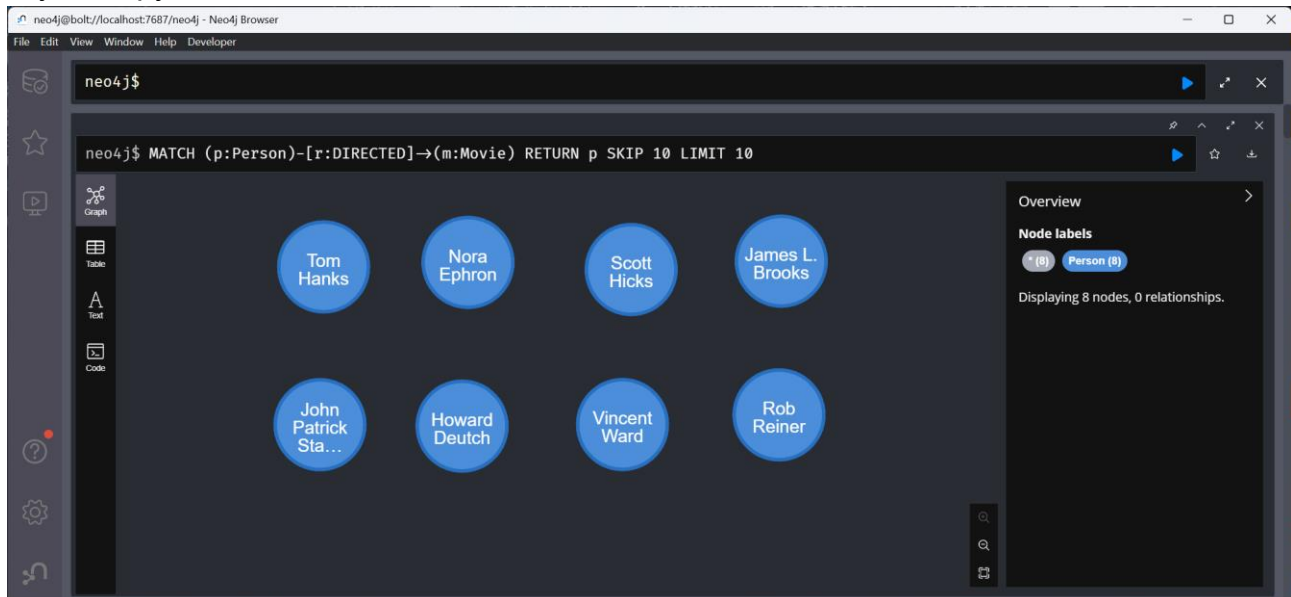
Zadanie 4

Wyszukaj wszystkich reżyserów i rozpocznij ich włączanie na wyjście od 10 węzła i ogranicz liczbę węzłów do 10.

Kod zapytania:

```
MATCH (p:Person)-[r:DIRECTED]->(m:Movie) RETURN p SKIP 10 LIMIT 10
```

Wynik zapytania:



Zadanie 5

Wyszukaj i wyświetl nazwiska aktorów, którzy urodzili się w latach 1931, 1944, 1954, 1956, 1967.

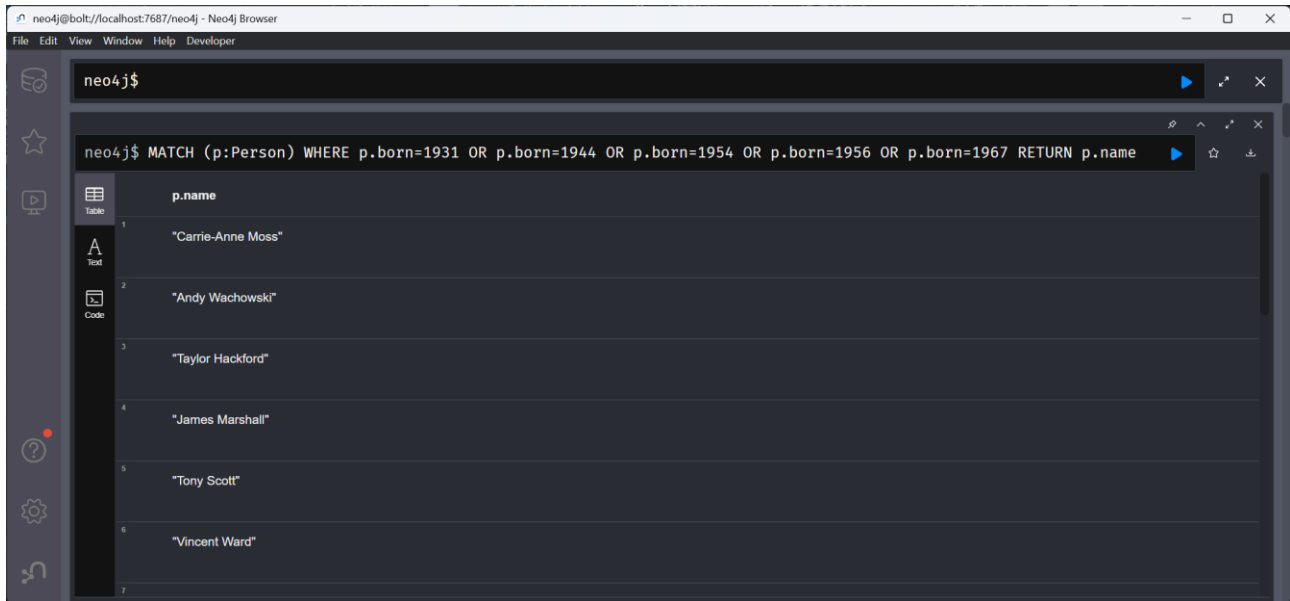
Przy wyszukiwaniu zastosuj:

- a) operator OR
- b) klauzulę UNWIND

Kod zapytania z użyciem OR:

```
MATCH (p:Person) WHERE p.born=1931 OR p.born=1944 OR p.born=1954 OR p.born=1956 OR p.born=1967 RETURN p.name
```

Wynik zapytania z użyciem OR:



Nazwiska z użyciem OR:

- "Carrie-Anne Moss"
- "Andy Wachowski"
- "Taylor Hackford"
- "James Marshall"
- "Tony Scott"
- "Vincent Ward"
- "Steve Zahn"
- "Tom Hanks"
- "Rita Wilson"
- "Nathan Lane"
- "Carrie Fisher"
- "Zach Grenier"
- "Mike Nichols"
- "Ron Howard"
- "Ben Miles"
- "Danny DeVito"
- "Philip Seymour Hoffman"
- "Julia Roberts"
- "Madonna"
- "Geena Davis"

Kod zapytania z użyciem UNWIND:

```
UNWIND [1931, 1944, 1954, 1956, 1967] AS year MATCH (p:Person {born:year}) RETURN p.name
```

Wynik zapytania z użyciem UNWIND:



Nazwiska z użyciem UNWIND:

- "Mike Nichols"
- "Taylor Hackford"
- "Tony Scott"
- "Danny DeVito"
- "Zach Grenier"
- "Ron Howard"
- "Madonna"
- "Vincent Ward"
- "Tom Hanks"
- "Rita Wilson"
- "Nathan Lane"
- "Carrie Fisher"
- "Geena Davis"
- "Carrie-Anne Moss"
- "Andy Wachowski"
- "James Marshall"
- "Steve Zahn"
- "Ben Miles"
- "Philip Seymour Hoffman"
- "Julia Roberts"

Który zapis zapytania jest bardziej zwięzły lub uniwersalny?

Bardziej zwięzły i uniwersalny jest zapis z UNWIND, ponieważ może przyjąć dynamicznie listę wartości, można łatwo dodawać nowe warunki do wyszukania oraz łatwo łączyć się z innymi zapytaniami.

Zadanie 6

Co zwróćą poniższe zapytania? Jaka jest różnica w wynikach?

Kod zapytania:

```
UNWIND ["Ala","Ola"] as Imie UNWIND [[1945, 1946,2020],[1923, 1945, 2025]] as Lata RETURN Imie, Lata
```

Wynik zapytania:

	Imie	Lata
1	"Ala"	[1945, 1946, 2020]
2	"Ala"	[1923, 1945, 2025]
3	"Ola"	[1945, 1946, 2020]
4	"Ola"	[1923, 1945, 2025]

Started streaming 4 records after 3 ms and completed after 3 ms.

Kod zapytania:

```
UNWIND ["Ala","Ola"] as Imie UNWIND [[1945, 1946,2020],[1923, 1945, 2025]] as Lata UNWIND Lata as Rok RETURN Imie, Rok
```

Wynik zapytania:

	Imie	Rok
1	"Ala"	1945
2	"Ala"	1946
3	"Ala"	2020
4	"Ala"	1923
5	"Ala"	1945
6	"Ala"	2025
7	"Ala"	1923

Started streaming 12 records after 3 ms and completed after 4 ms.

Jaka jest różnica w wynikach?

W zapytaniu pierwszym „UNWIND [\"Ala\", \"Ola\"] AS Imie” rozdziela listę na osobne wiersze (powstaną dwa wiersze), „UNWIND [[1945, 1946, 2020], [1923, 1945, 2025]] AS Lata” też rozdziela listę na osobne wiersze (tutaj powstaną też dwa wiersze), „RETURN Imie, Lata” łączy każdy wiersz z pierwszego UNWIND z każdym wierszem z drugiego UNWIND, powstaną cztery kombinacje ($2 * 2$). W drugim zapytaniu dzieje się to samo, ale dodatkowo „UNWIND Lata AS Rok” rozdziela listy z Lat na osobne wiersze (więc z dwóch wierszy powstanie sześć), dlatego w „RETURN Imie, Rok” powstaje dwanaście kombinacji ($2 * 2 * 3$).

Zadanie 7

Utwórz węzły osób z właściwością „name”. Węzły te połącz relacją „likes” z węzłami odpowiednich filmów, dla których rok produkcji jest rokiem ulubionych filmów danej osoby. Użyj w tym celu klauzuli UNWIND. Dla każdej osoby utwórz osobne zapytanie. Użyj danych z tabeli. Wyświetl utworzone w ten sposób ścieżki (Person)-[likes]→(Movie).

name	ulubione filmy
Jan	1975, 1990, 1995, 2003
Aleksander	1999, 2004, 2006
Maciej	2003, 2006
Ewa	1992, 1996, 2003
Filip	2000, 2012
Agata	2003, 2004, 1975
Wojciech	2008, 2019, 2006, 2004
Aneta	2000
Anna	2012, 2019
Ula	2004

Kody zapytań (każde uruchomione osobno):

```
MERGE (p:Person { name:"Jan" }) WITH p UNWIND [1975, 1990, 1995, 2003] AS year MATCH (m:Movie {released:year}) CREATE (p)-[r:likes]->(m) RETURN p, m, r
```

```
MERGE (p:Person { name:"Aleksander" }) WITH p UNWIND [1999, 2004, 2006] AS year MATCH (m:Movie {released:year}) CREATE (p)-[r:likes]->(m) RETURN p, m, r
```

```
MERGE (p:Person { name:"Maciej" }) WITH p UNWIND [2003, 2006] AS year MATCH (m:Movie {released:year}) CREATE (p)-[r:likes]->(m) RETURN p, m, r
```

```
MERGE (p:Person { name:"Ewa" }) WITH p UNWIND [1992, 1996, 2003] AS year MATCH (m:Movie {released:year}) CREATE (p)-[r:likes]->(m) RETURN p, m, r
```

```
MERGE (p:Person { name:"Filip" }) WITH p UNWIND [2000, 2012] AS year MATCH (m:Movie {released:year}) CREATE (p)-[r:likes]->(m) RETURN p, m, r
```

```
MERGE (p:Person { name:"Agata" }) WITH p UNWIND [2003, 2004, 1975] AS year MATCH (m:Movie {released:year}) CREATE (p)-[r:likes]->(m) RETURN p, m, r
```

```
MERGE (p:Person {name:"Wojciech"}) WITH p UNWIND [2008, 2019, 2006, 2004] AS year MATCH (m:Movie {released:year}) CREATE (p)-[r:likes]->(m) RETURN p, m, r
```

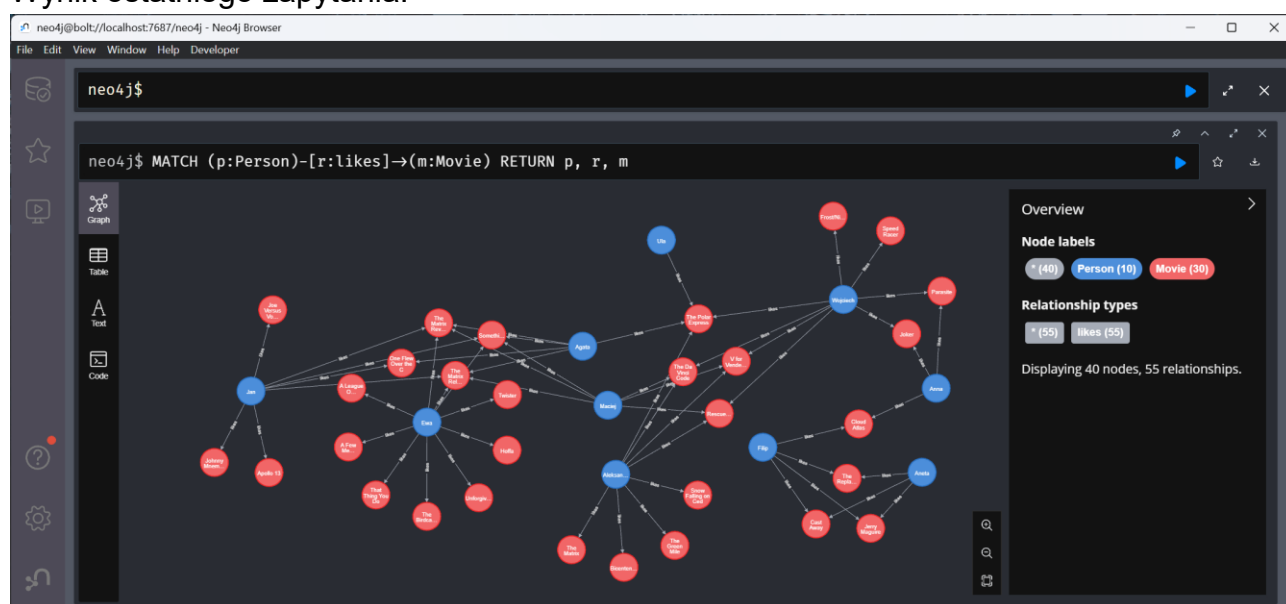
```
MERGE (p:Person {name:"Aneta"}) WITH p UNWIND [2000] AS year MATCH (m:Movie {released:year}) CREATE (p)-[r:likes]->(m) RETURN p, m, r
```

```
MERGE (p:Person {name:"Anna"}) WITH p UNWIND [2012, 2019] AS year MATCH (m:Movie {released:year}) CREATE (p)-[r:likes]->(m) RETURN p, m, r
```

```
MERGE (p:Person {name:"Ula"}) WITH p UNWIND [2004] AS year MATCH (m:Movie {released:year}) CREATE (p)-[r:likes]->(m) RETURN p, m, r
```

```
MATCH (p:Person)-[r:likes]->(m:Movie) RETURN p, r, m
```

Wynik ostatniego zapytania:



Zadanie 8

Wykonaj i przetestuj wszystkie przykłady z wykładu dotyczące klauzuli „MERGE”.

Kod zapytania:

```
MERGE (jan:Pisarz) RETURN jan, labels(jan)
```


Wynik zapytania:

The screenshot shows the Neo4j Browser interface. The query entered is `neo4j$ MERGE (jan:Pisarz) RETURN jan, labels(jan)`. The result is displayed as a graph with a single node labeled '189'. The node properties panel on the right shows the following details:

Node properties	
Pisarz	
<elementId>	4:4ee6b20a-d290-4c78-bca8-b231611479dd:189
<id>	189

Kod zapytania:

```
MERGE (osoba {imie:"Ola", wiek:10}) RETURN osoba
```

Wynik zapytania:

The screenshot shows the Neo4j Browser interface. The query entered is `neo4j$ MERGE (osoba {imie:"Ola", wiek:10}) RETURN osoba`. The result is displayed as a graph with a single node. The node properties panel on the right shows the following details:

Node properties	
<elementId>	4:4ee6b20a-d290-4c78-bca8-b231611479dd:190
<id>	190
imie	Ola
wiek	10

Kod zapytania:

```
MERGE (osoba:Osoba {imie:"Ania", wzrost:25}) RETURN osoba
```

Wynik zapytania:

neo4j\$

```
neo4j$ MERGE (osoba:Osoba {imie:"Ania", wzrost:25}) RETURN osoba
```

Node properties

Property	Value
<elementId>	4:4ee6b20a-d290-4c78-bca8-b231611479dd:191
<id>	191
imie	Ania
wzrost	25

Kod zapytania:

```
MERGE (o {imie:"Kazimierz"}) RETURN o
```

Wynik zapytania:

neo4j\$

```
neo4j$ MERGE (o {imie:"Kazimierz"}) RETURN o
```

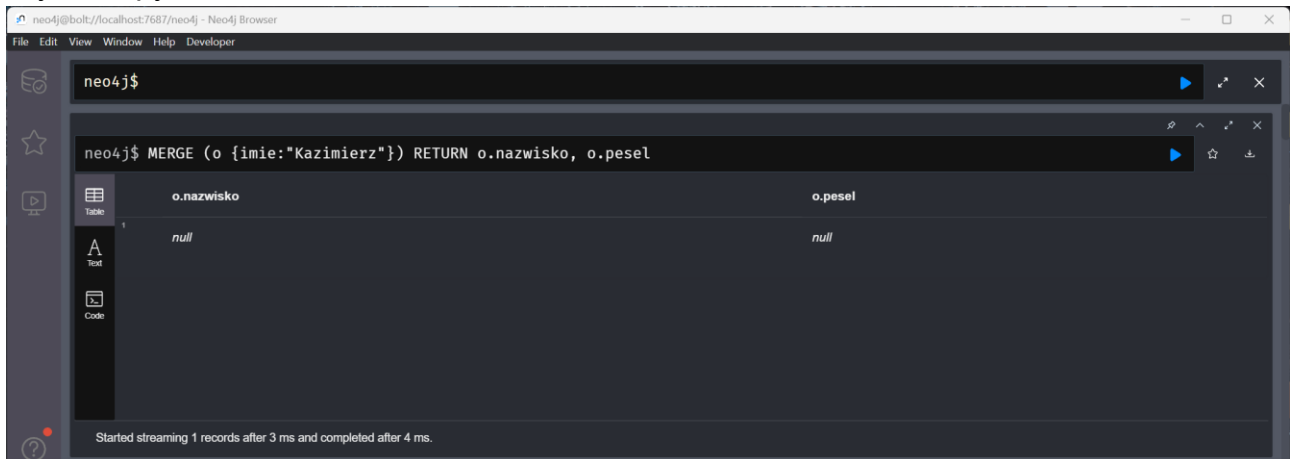
Node properties

Property	Value
<elementId>	4:4ee6b20a-d290-4c78-bca8-b231611479dd:192
<id>	192
imie	Kazimierz

Kod zapytania:

```
MERGE (o {imie:"Kazimierz"}) RETURN o.nazwisko, o.pesel
```

Wynik zapytania:

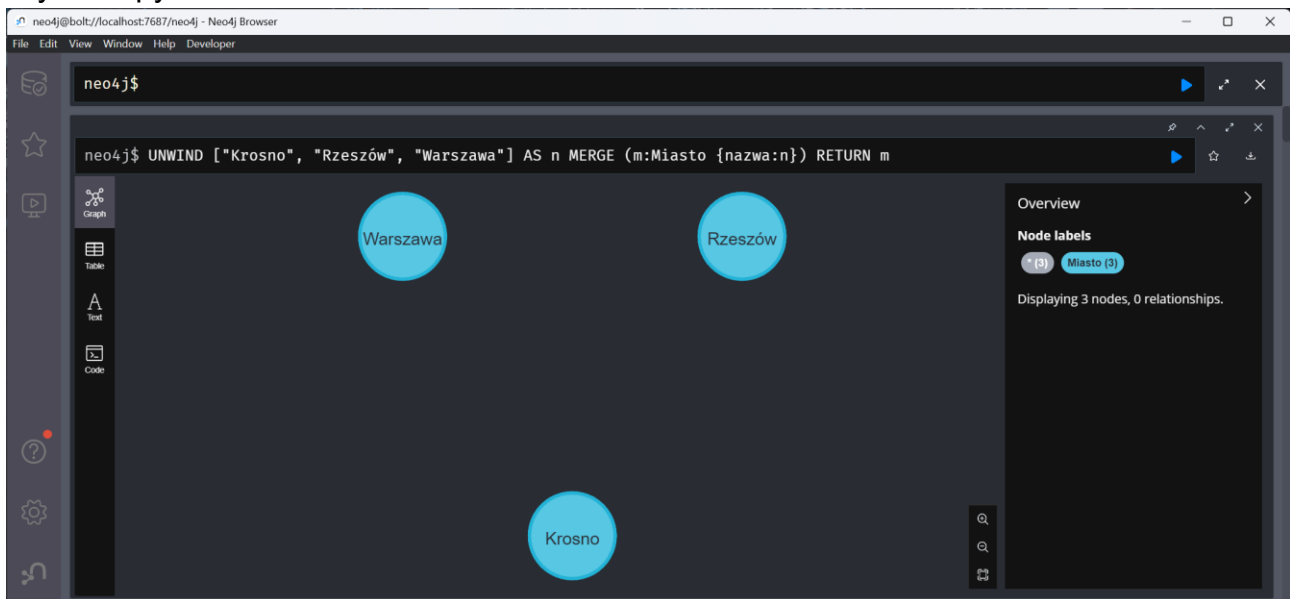


o.nazwisko	o.pesel
null	null

Kod zapytania:

```
UNWIND ["Krosno", "Rzeszów", "Warszawa"] AS n MERGE (m:Miesto {nazwa:n}) RETURN m
```

Wynik zapytania:



Overview

Node labels

(3) Miesto (3)

Displaying 3 nodes, 0 relationships.

Kod zapytania:

```
MERGE (o {imie:"Kazimierz"}) SET o.urodzony="Krosno" RETURN o
```

Wynik zapytania:

The screenshot shows the Neo4j Browser interface. The query bar contains the Cypher query: `neo4j$ MERGE (o {imie:"Kazimierz"}) SET o.urodzony="Krosno" RETURN o`. The main area displays a graph visualization with a central node and three surrounding nodes, each with an icon (a lock, a key, and a network symbol). The right sidebar shows the 'Node properties' for the selected node:

Property	Value
<elementId>	4:4ee6b20a-d290-4c78-bca8-b231611479dd:192
<id>	192
imie	Kazimierz
urodzony	Krosno

Kod zapytania:

```
MATCH (o {urodzony:o.urodzony}) MERGE (m:Miesto {nazwa:o.urodzony}) RETURN o, m
```

Wynik zapytania:

The screenshot shows the Neo4j Browser interface. The query bar contains the Cypher query: `neo4j$ MATCH (o {urodzony:o.urodzony}) MERGE (m:Miesto {nazwa:o.urodzony}) RETURN o, m`. The main area displays a graph visualization with a central node and three surrounding nodes, each with an icon (a lock, a key, and a network symbol). The right sidebar shows the 'Node properties' for the selected node:

Property	Value
<elementId>	4:4ee6b20a-d290-4c78-bca8-b231611479dd:192
<id>	192
imie	Kazimierz
urodzony	Krosno

Kod zapytania:

```
MERGE (p:Pracownik {imie:"Dariusz", urodzony:"Warszawa"}) RETURN p
```

Wynik zapytania:

neo4j\$

neo4j\$ MERGE (p:Pracownik {imie:"Dariusz", urodzony:"Warszawa"}) RETURN p

Node properties

Pracownik

<elementId>	4:4ee6b20a-d290-4c78-bca8-b231611479dd:196
<id>	196
imie	Dariusz
urodzony	Warszawa

Kod zapytania:

```
MATCH (o {urodzony:o.urodzony}) MERGE (m:Miesto {nazwa:o.urodzony}) RETURN o, m
```

Wynik zapytania:

neo4j\$

neo4j\$ MATCH (o {urodzony:o.urodzony}) MERGE (m:Miesto {nazwa:o.urodzony}) RETURN o, m

Node properties

Pracownik

<elementId>	4:4ee6b20a-d290-4c78-bca8-b231611479dd:192
<id>	192
imie	Kazimierz
urodzony	Krosno

Kod zapytania:

```
MATCH (o {urodzony:o.urodzony}) MERGE (m:Miesto {nazwa:o.urodzony}) MERGE (o)-[r:`urodzony w`]->(m) RETURN o, m, r
```

Wynik zapytania:

