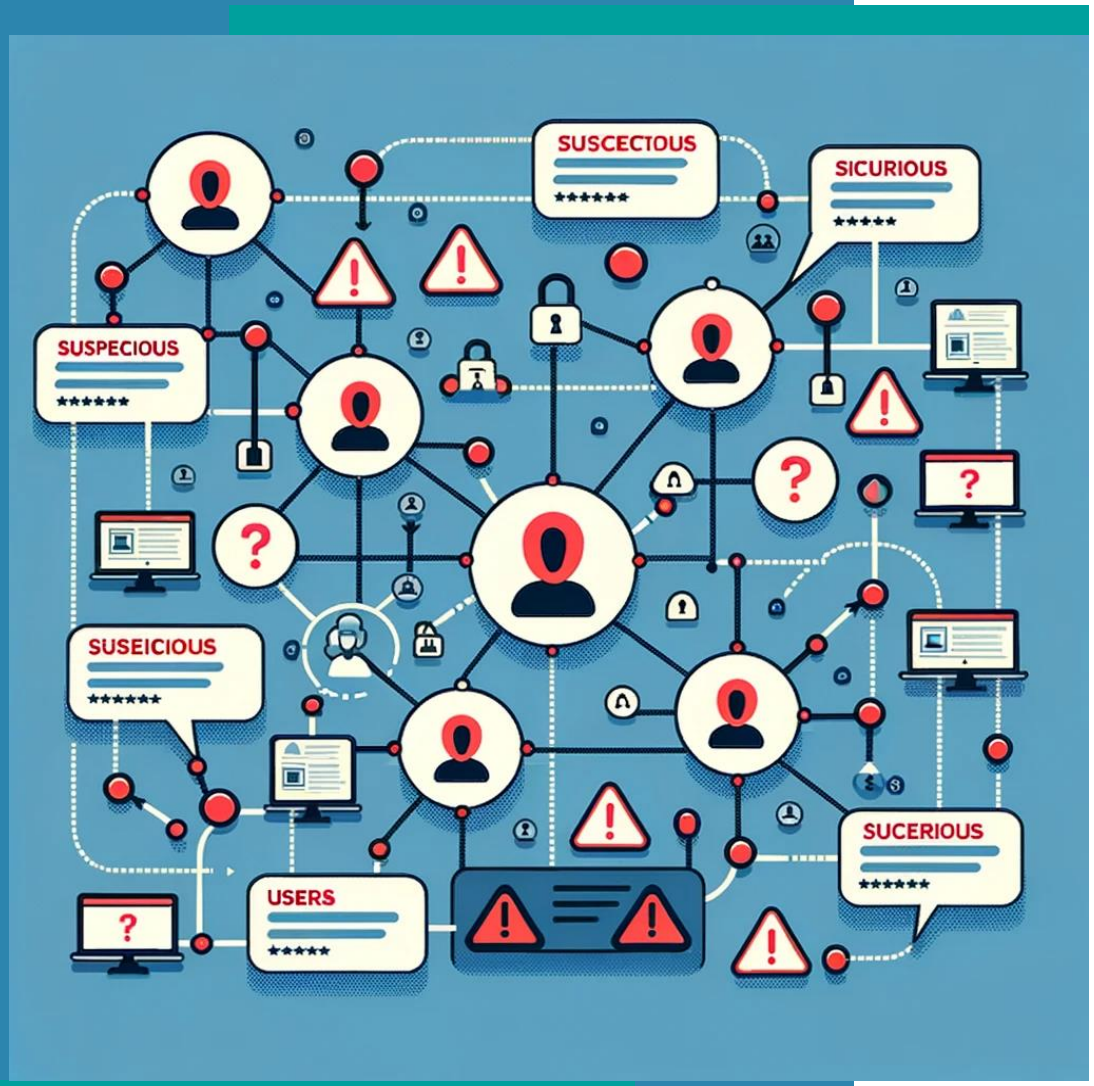
dr inż. Michał Malinowski
michal.malinowski@uth.edu.pl

https://www.linkedin.com/in/michał-malinowski-malkon/

Bazy grafowe

Ćwiczenie 09

ALGORYTMY
I STRUKTURY DANYCH



Zagadnienia (2h)

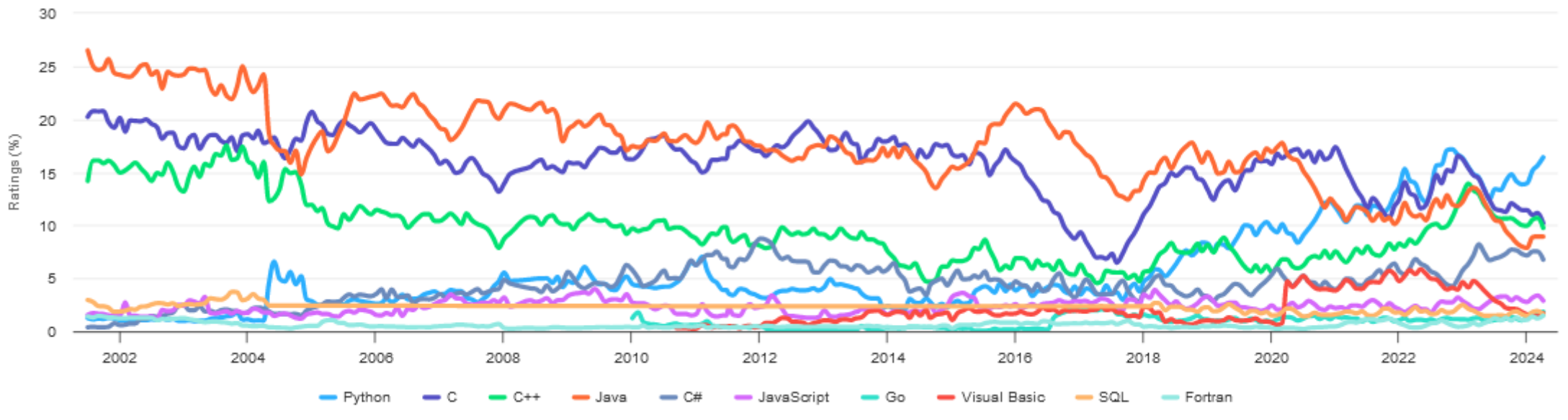
- Wprowadzenie do baz NoSQL
- Modelowanie węzłów i relacji
- Bazy grafowe vs. Bazy relacyjne
- Model grafu i dane grafu
- Baza grafowa Neo4j
  - Zadanie: Import danych do grafu
- Obszary zarządzania danymi
- Zapytania o dane CYPHER
  - Zadanie: System analizy danych
- Zadanie: Grafowy model danych

Ranking - Języki programowania

Apr 2024	Apr 2023	Change	Programming Language	Ratings	Change
1	1		Python	16.41%	+1.90%
2	2		C	10.21%	-4.20%
3	4	↑	C++	9.76%	-3.20%
4	3	↓	Java	8.94%	-4.29%
5	5		C#	6.77%	-1.44%
6	7	↑	JavaScript	2.89%	+0.79%
7	10	↑	Go	1.85%	+0.57%
8	6	↓	Visual Basic	1.70%	-2.70%
9	8	↓	SQL	1.61%	-0.06%
10	20	⇈	Fortran	1.47%	+0.88%
11	11		Delphi/Object Pascal	1.47%	+0.24%
12	12		Assembly language	1.30%	+0.26%
13	18	⇈	Ruby	1.24%	+0.58%
14	17	↑	Swift	1.23%	+0.51%
15	15		Scratch	1.14%	+0.35%
16	14	↓	MATLAB	1.11%	+0.25%
17	9	⇊	PHP	1.09%	-0.26%
18	38	⇈	Kotlin	1.05%	+0.80%
19	19		Rust	1.03%	+0.41%
20	16	⇊	R	0.84%	+0.09%

TIOBE Index for April 2024,
https://www.tiobe.com/tiobe-index/

Ranking - Języki programowania



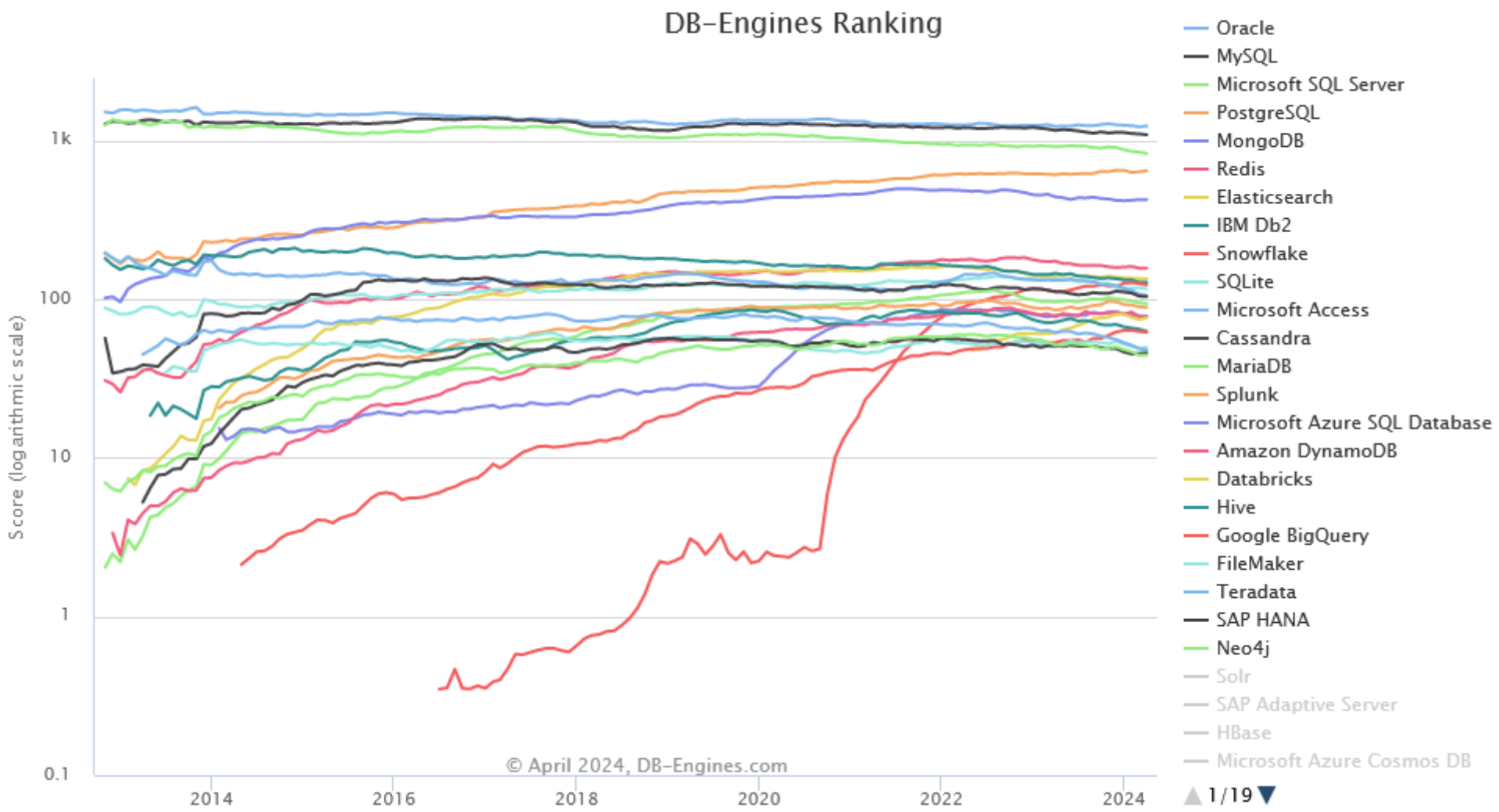
TIOBE Index for April 2024,
https://www.tiobe.com/tiobe-index/

Ranking - Systemy baz danych

Rank Apr 2024	Rank Mar 2024	Rank Apr 2023	DBMS	Database Model	Score Apr 2024	Score Mar 2024	Score Apr 2023
1.	1.	1.	Oracle	Relational, Multi-model	1234.27	+13.21	+5.99
2.	2.	2.	MySQL	Relational, Multi-model	1087.72	-13.77	-70.06
3.	3.	3.	Microsoft SQL Server	Relational, Multi-model	829.80	-16.01	-88.73
4.	4.	4.	PostgreSQL	Relational, Multi-model	645.05	+10.15	+36.64
5.	5.	5.	MongoDB	Document, Multi-model	423.96	-0.57	-17.93
6.	6.	6.	Redis	Key-value, Multi-model	156.44	-0.56	-17.11
7.	7.	↑ 8.	Elasticsearch	Search engine, Multi-model	134.78	-0.01	-6.29
8.	8.	↓ 7.	IBM Db2	Relational, Multi-model	127.49	-0.26	-18.00
9.	9.	↑ 12.	Snowflake	Relational	123.20	-2.18	+12.07
10.	10.	↓ 9.	SQLite	Relational	116.01	-2.15	-18.53
11.	11.	↓ 10.	Microsoft Access	Relational	105.40	-2.52	-25.97
12.	12.	↓ 11.	Cassandra	Wide column, Multi-model	103.86	-0.72	-7.94
13.	13.	13.	MariaDB	Relational, Multi-model	93.81	-1.22	-2.13
14.	14.	14.	Splunk	Search engine	88.71	-0.97	+3.27
15.	15.	15.	Microsoft Azure SQL Database	Relational, Multi-model	78.40	-0.11	-0.66
16.	16.	16.	Amazon DynamoDB	Multi-model	77.57	-0.15	+0.12
17.	17.	↑ 19.	Databricks	Multi-model	76.33	+1.99	+15.36
18.	18.	↓ 17.	Hive	Relational	62.59	-2.24	-9.07
19.	19.	↑ 20.	Google BigQuery	Relational	61.90	-0.77	+8.59
20.	↑ 21.	↑ 23.	FileMaker	Relational	49.73	+0.92	-0.27
21.	↓ 20.	↓ 18.	Teradata	Relational, Multi-model	47.84	-1.10	-13.75
22.	22.	22.	SAP HANA	Relational, Multi-model	45.84	+0.35	-5.24
23.	23.	↓ 21.	Neo4j	Graph	44.47	+0.11	-7.13
24.	24.	24.	Solr	Search engine, Multi-model	44.28	+0.37	-3.94
25.	25.	25.	SAP Adaptive Server	Relational, Multi-model	37.12	-0.37	-5.75

DB-Engines Ranking April 2024,
https://db-engines.com/en/ranking

Ranking - Systemy baz danych



DB-Engines Ranking April 2024,
https://db-engines.com/en/ranking

Wprowadzenie do baz NoSQL

Definicja

- NoSQL (Not Only SQL) to termin odnoszący się do systemów zarządzania bazami danych (DBMS), które różnią się od klasycznych relacyjnych baz danych (RDBMS).
- Zaprojektowane z myślą o obsłudze dużej liczby danych, rozproszonych na wielu serwerach.
- Nie zawsze zapewniają pełne wsparcie dla standardów ACID*

***ACID** (akronim, który opisuje podstawowe właściwości transakcji w systemie zarządzania bazą danych)

- Atomicity (Atomowość): Transakcje są niepodzielne.
- Consistency (Spójność): Każda transakcja przekształca bazę danych z jednego stanu spójnego do innego stanu spójnego.
- Isolation (Izolacja): Każda transakcja powinna być wykonywana w sposób izolowany od innych transakcji.
- Durability (Trwałość): Po zakończeniu transakcji, jej efekty są trwałe i niezawodne, nawet w przypadku awarii systemu.

Wprowadzenie do baz NoSQL

Rodzaje baz NoSQL

- Bazy klucz-wartość (Key-Value): najprostszy typ NoSQL, gdzie wszystko przechowywane jest jako para klucz-wartość.
  Przykłady: Redis, Amazon DynamoDB.
- Bazy dokumentowe (Document): przechowują informacje jako dokumenty, często w formatach takich jak JSON.
  Przykłady: MongoDB, Couchbase.
- Bazy kolumnowe (Columnar): dane są przechowywane i wyszukiwane na podstawie kolumn, a nie wierszy.
  Przykłady: Apache Cassandra, Google Bigtable.
- Bazy grafowe (Graph): skoncentrowane na przechowywaniu informacji o węzłach i relacjach między nimi.
  Przykład: Neo4j, JanusGraph

Wprowadzenie do baz NoSQL

Elementy bazy grafowej

Węzły (Nodes)

Reprezentują obiekty takie jak osoby, miejsca, wydarzenia.

Krawędzie (Edges, Relationships)

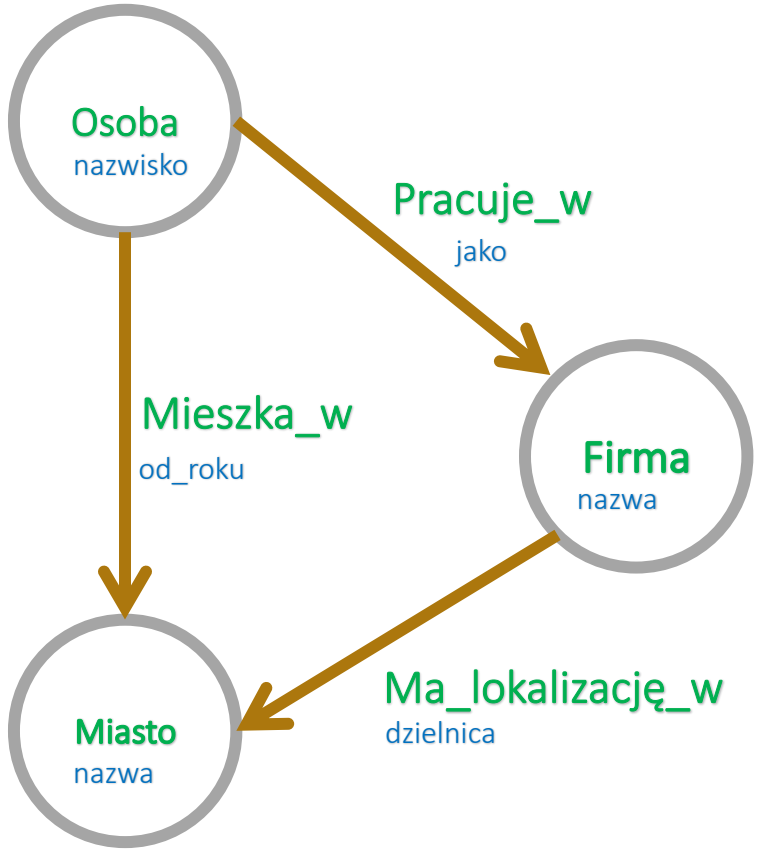
Reprezentują relacje między obiektami, np. znajomość, przepływ informacji.

Etykiety (Labels)

Klasyfikują węzły i krawędzie, ułatwiając organizację i wyszukiwanie danych.

Właściwości (Properties)

Informacje dodatkowe przypisane do węzłów i krawędzi, np. nazwa, data.



Modelowanie węzłów i relacji

Węzły

Węzły przeważnie będą odpowiadały rzeczownikom z opisu rzeczywistości a w szczególności z przypadków użycia.

Węzły będą otrzymywały etykiety odpowiadające zidentyfikowanym rzeczownikom lub ich synonimom (mianownik w liczbie pojedynczej).

Przypadki użycia

1. W jakiej **firmie** pracuje ta **osoba**?

2. Jaką **rolę** pełni ta **osoba**?
3. Jakie **składniki** są używane w **przepisie**?
4. Kto jest **mężem** tej **osoby**?
5. Jakie **osoby** zagrały w **filmie**?
6. Kim są **dziadkowie** tego **dziecka**?

Modelowanie węzłów i relacji

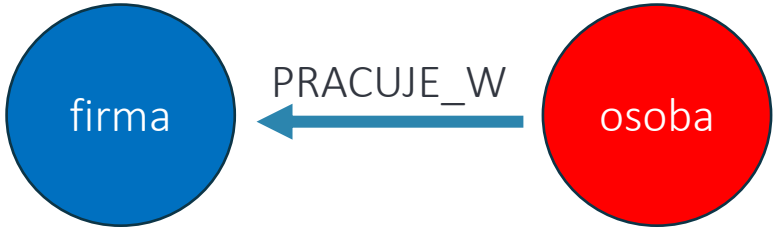
Relacje

Relacje przeważnie będą odpowiadały czasownikom z opisu rzeczywistości, a w szczególności z przypadków użycia.

Relacje będą otrzymywały etykiety odpowiadające zidentyfikowanym czasownikom lub ich synonimom (forma osobowa liczba pojedyncza).

Przypadki użycia

1. W jakiej **firmie** pracuje ta **osoba**?



2. Jaką **rolę** pełni ta **osoba**?
3. Jakie **składniki** są używane w **przepisie**?
4. Kto jest w związku małżeńskim z tą **osobą**?
5. Jakie **osoby** zagrały w **filmie**?
6. Kim są **dziadkowie** tego **dziecka**?

Bazy grafowe vs. Bazy relacyjne

Relacyjna baza danych

Osoba

id	nazwisko
1	Nowak
2	Kowalski

Mieszka_w

idosoba	idmiasto	od_roku
1	2	1990
2	1	2011

Firma

id	Nazwa
1	Alfa
2	Beta

Pracuje_w

idosoba	idfirma	jako
1	2	Mechanik
2	1	Księgowy

Miasto

id	nazwa
1	Warszawa
2	Kraków

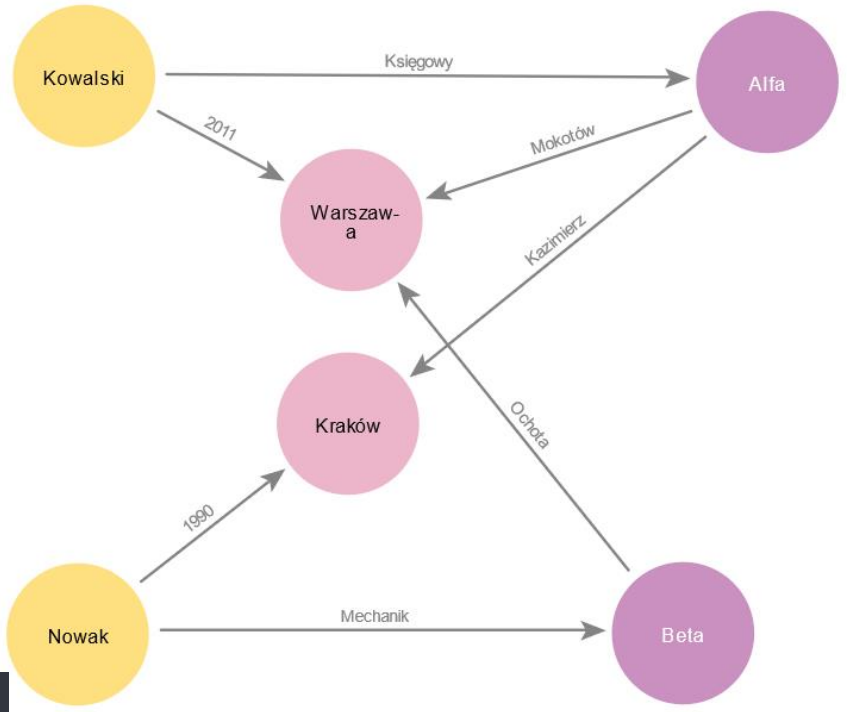
Ma_lokalizacje_w

idfirma	idmiasto	dzielnica
1	1	Mokotów
1	2	Kazimierz
2	1	Ochota

SQL

```
SELECT Firma.nazwa, Pracuje_w.jako FROM Firma
JOIN Pracuje_w ON Pracuje_w.idfirma = Firma.id
JOIN Osoba ON Osoba.id = Pracuje_w.idosoba
WHERE Osoba.nazwisko = "Kowalski"
```

Grafowa baza danych

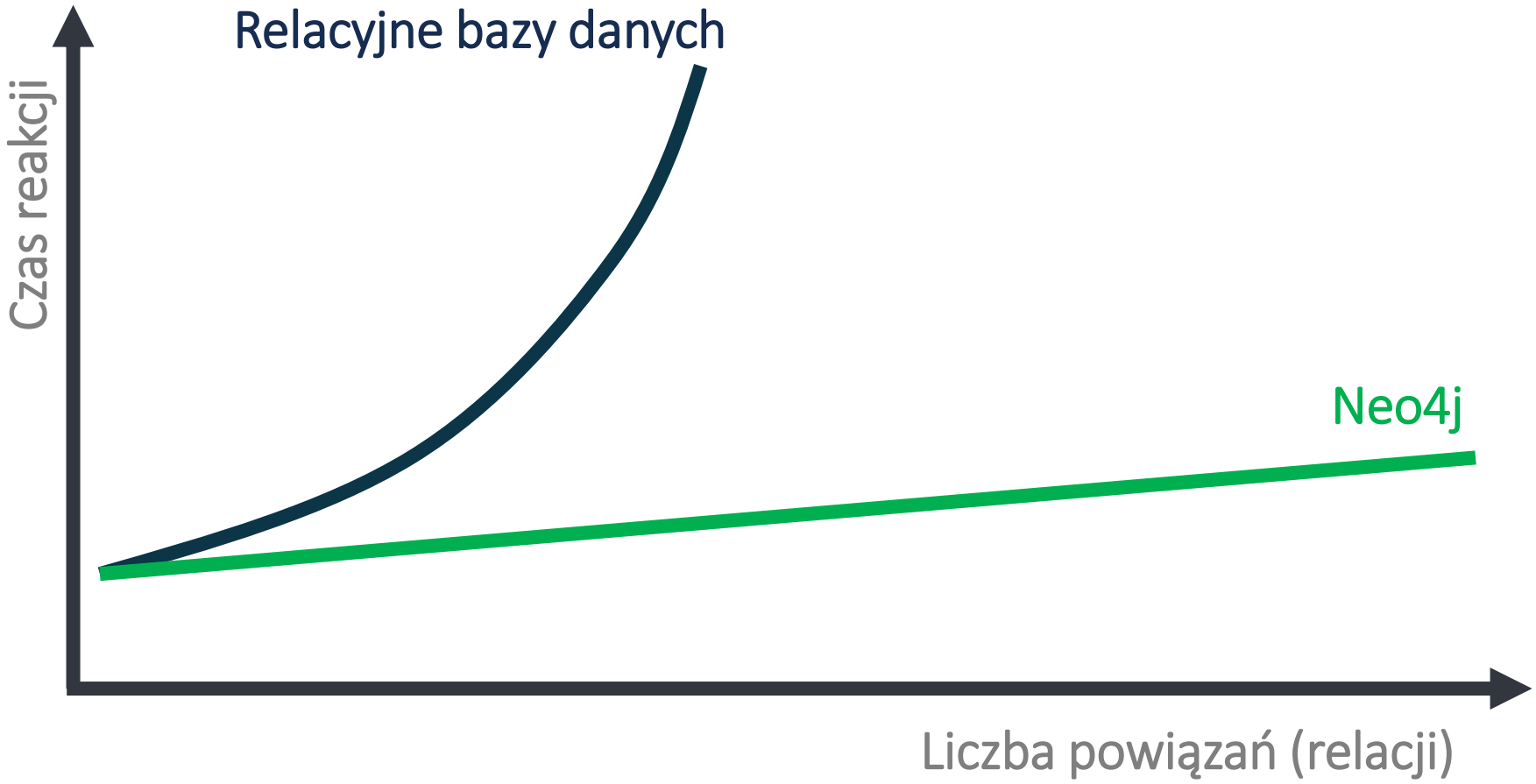


Cypher

```
MATCH (o:Osoba {nazwisko:'Kowalski'})-[p:Pracuje_w]->(f:Firma)
RETURN f.nazwa, p.jako
```

nazwa	jako
Alfa	Księgowy

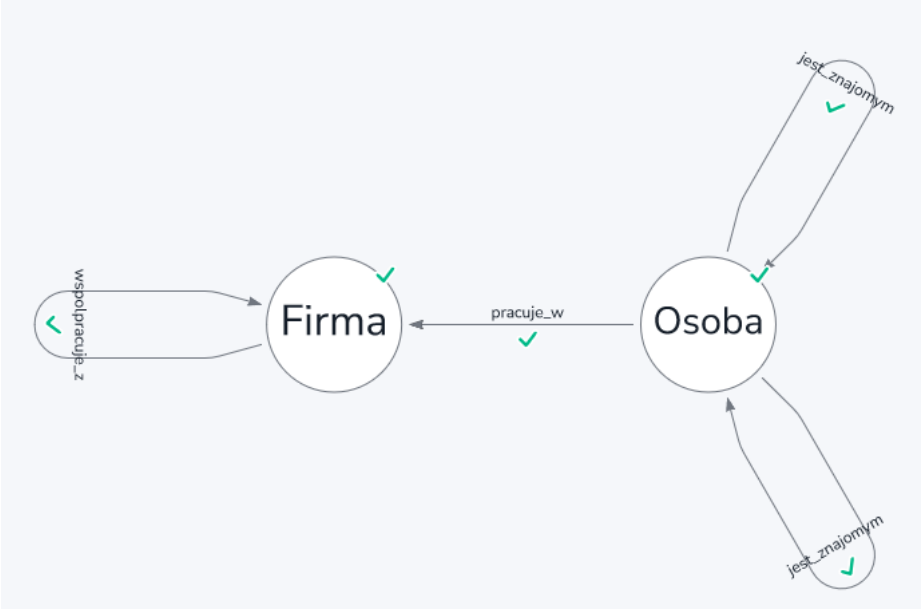
Bazy grafowe vs. Bazy relacyjne



Model grafu i dane grafu

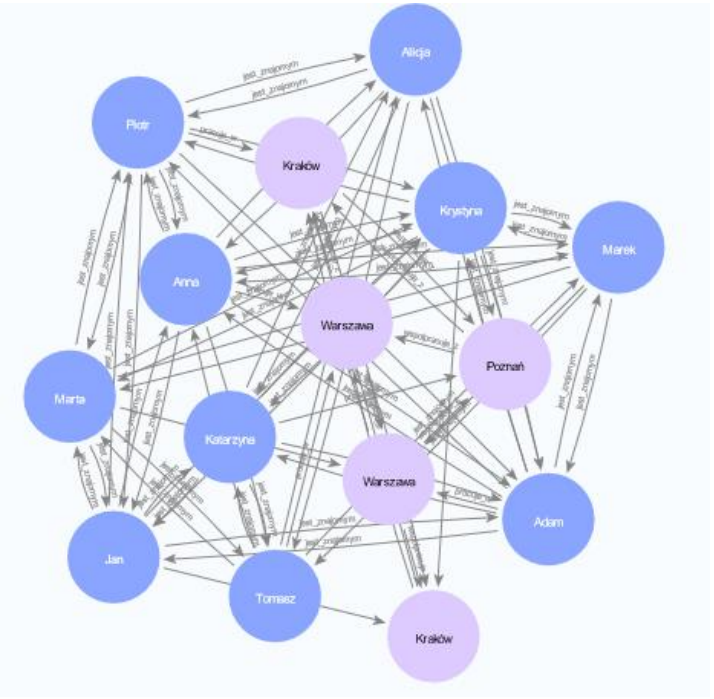
Model grafu

Definiuje schemat przechowywania danych w grafowej bazie danych

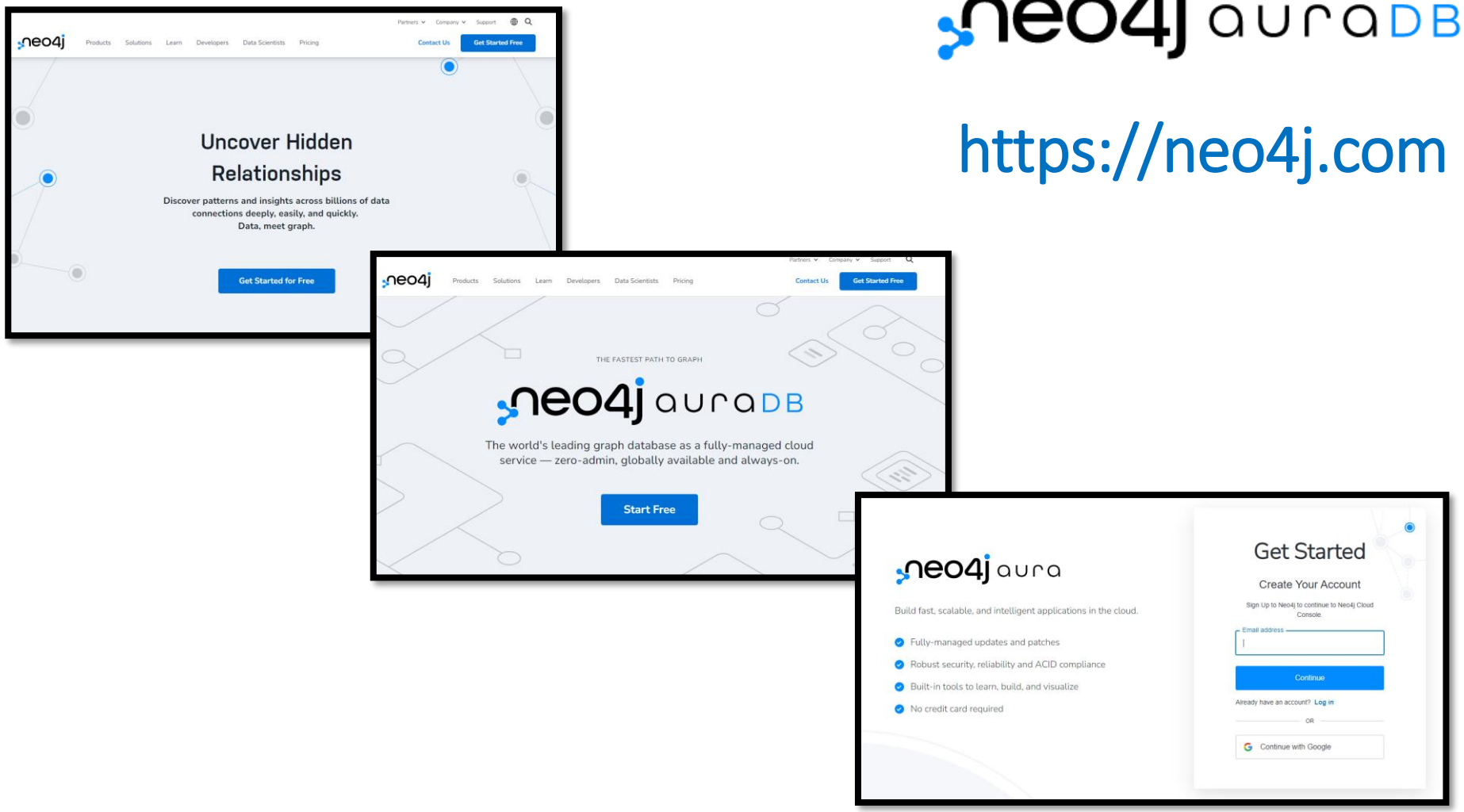


Dane grafu

Rzeczywiste informacje, które są przechowywane i reprezentowane w bazie na podstawie modelu grafu



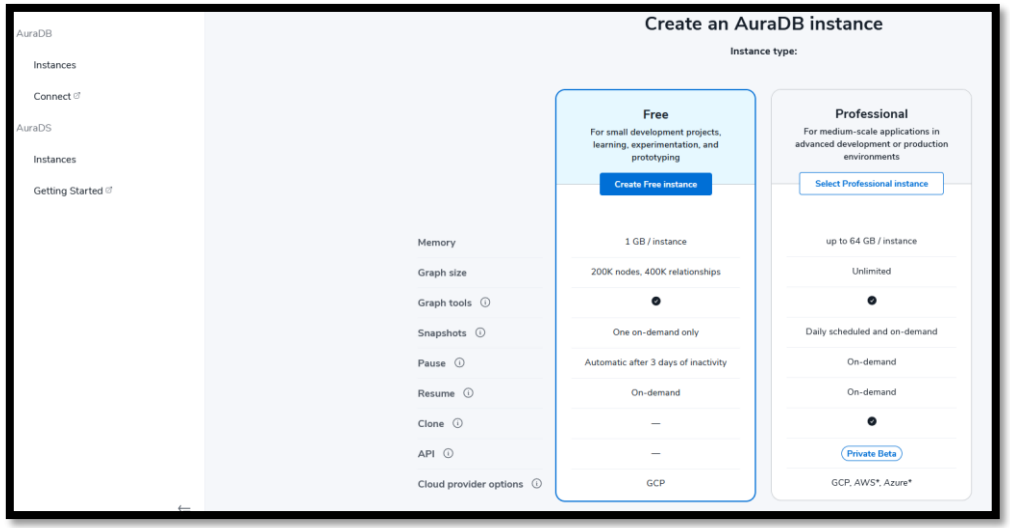
Baza grafowa Neo4j

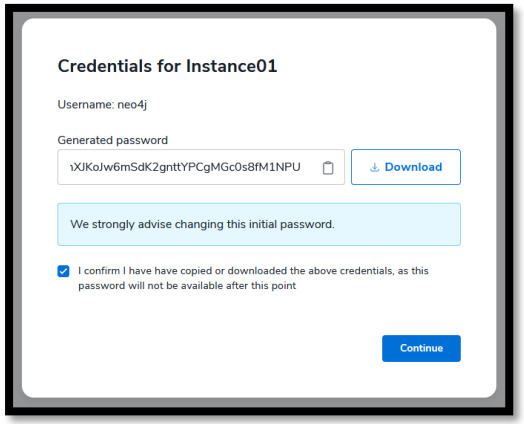


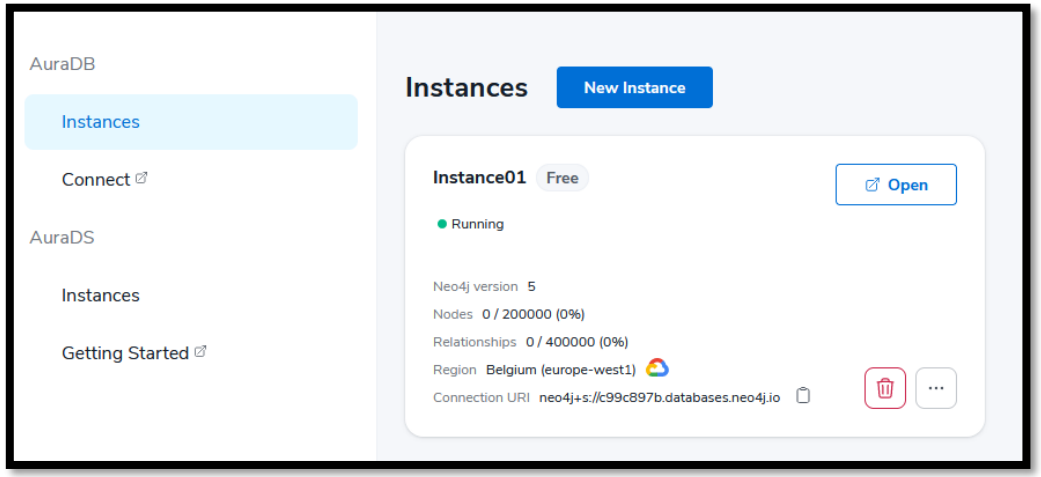
neo4j auraDB

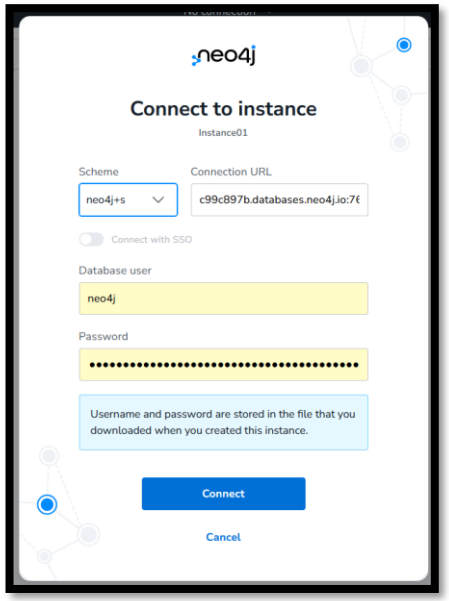
https://neo4j.com

Baza grafowa Neo4j









Zadanie: Import danych do grafu

Zasady

- Należy przygotować pliki CSV, które reprezentują węzły i relacje grafu, gdzie należy określić:
  - znak separatora pomiędzy danymi (znak przecinka - domyślny)
  - pierwszy wiersz zawierający informacje nagłówkowe
  - w kolejnych wierszach podawane są dane odpowiadające nagłówkom i oddzielone separatorem
- Pliki danych nazywać zgodnie z etykietami węzłów z jakimi dane będą związane
  - Węzły (:osoba) plik „osoba.csv”
  - Węzły (:firma) plik „firma.csv”
- Pliki przechowujące informacje o relacjach nazywać od etykiet węzłów, które będą łączyć
  - Relacje [:osoba -> :firma] plik „osoba_firma.csv”
- Wiersze mają zaczynać się od unikalnego klucza (id)

Zadanie: Import danych do grafu

Dane

osoba.csv

```
id,imie,nazwisko,dzien_urodzenia,student
1,Adam,Nowak,1990-04-03,true
2,Marek,Kowalski,1987-06-12,false
3,Krystyna,Zielinska,2001-11-11,true
4,Alicja,Habior,1996-03-24,false
```

firma.csv

```
id,nazwa,miejscowosc
1,GoodHome,Warszawa
2,Precless,Łódź
3,Rokitka,Warszawa
```

osoba_firma.csv

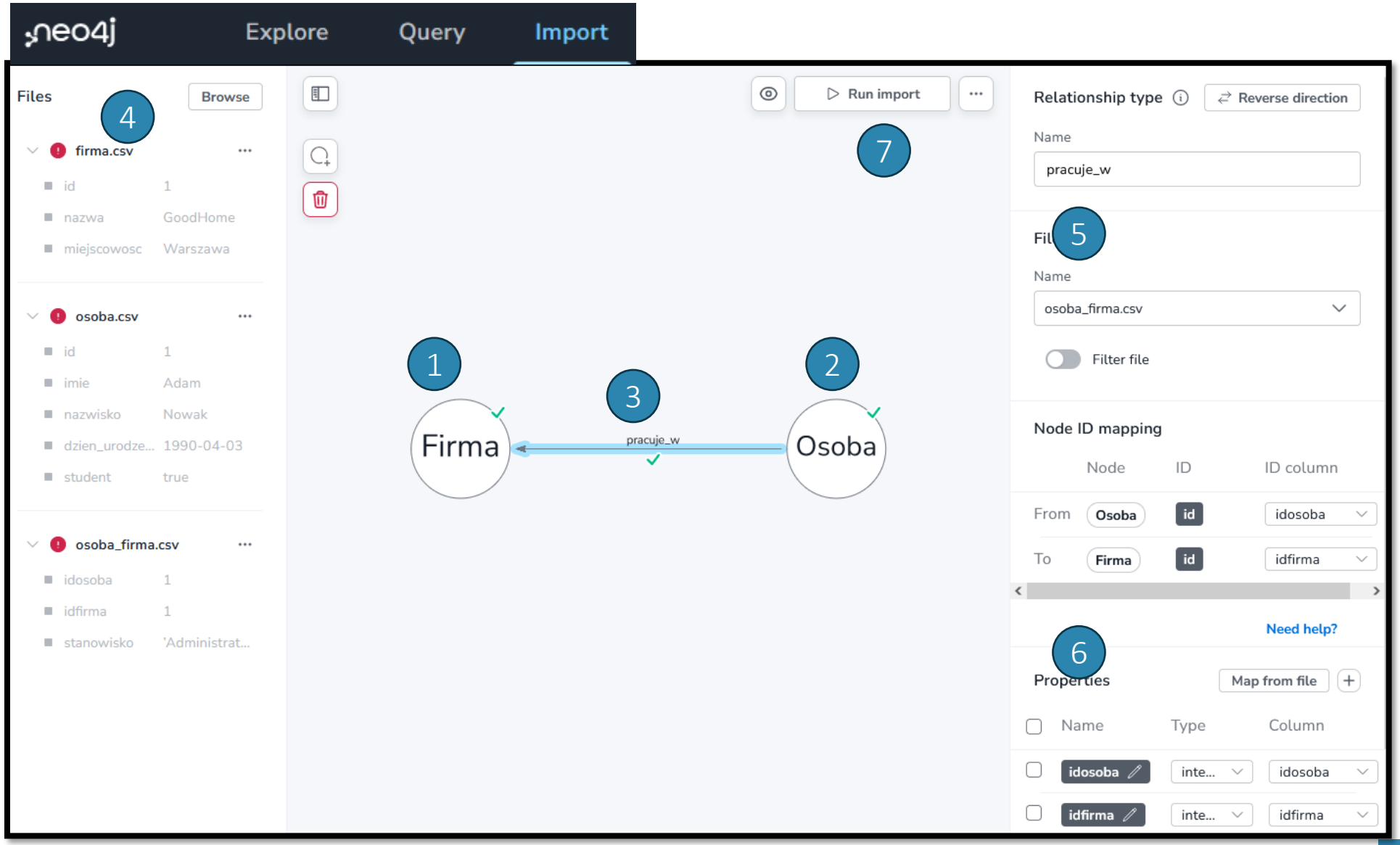
```
idosoba,idfirma,stanowisko
1,1,Administrator
2,1,Ksiegowy
3,2,Specjalista
4,3,Programista
```

Zadanie: Import danych do grafu

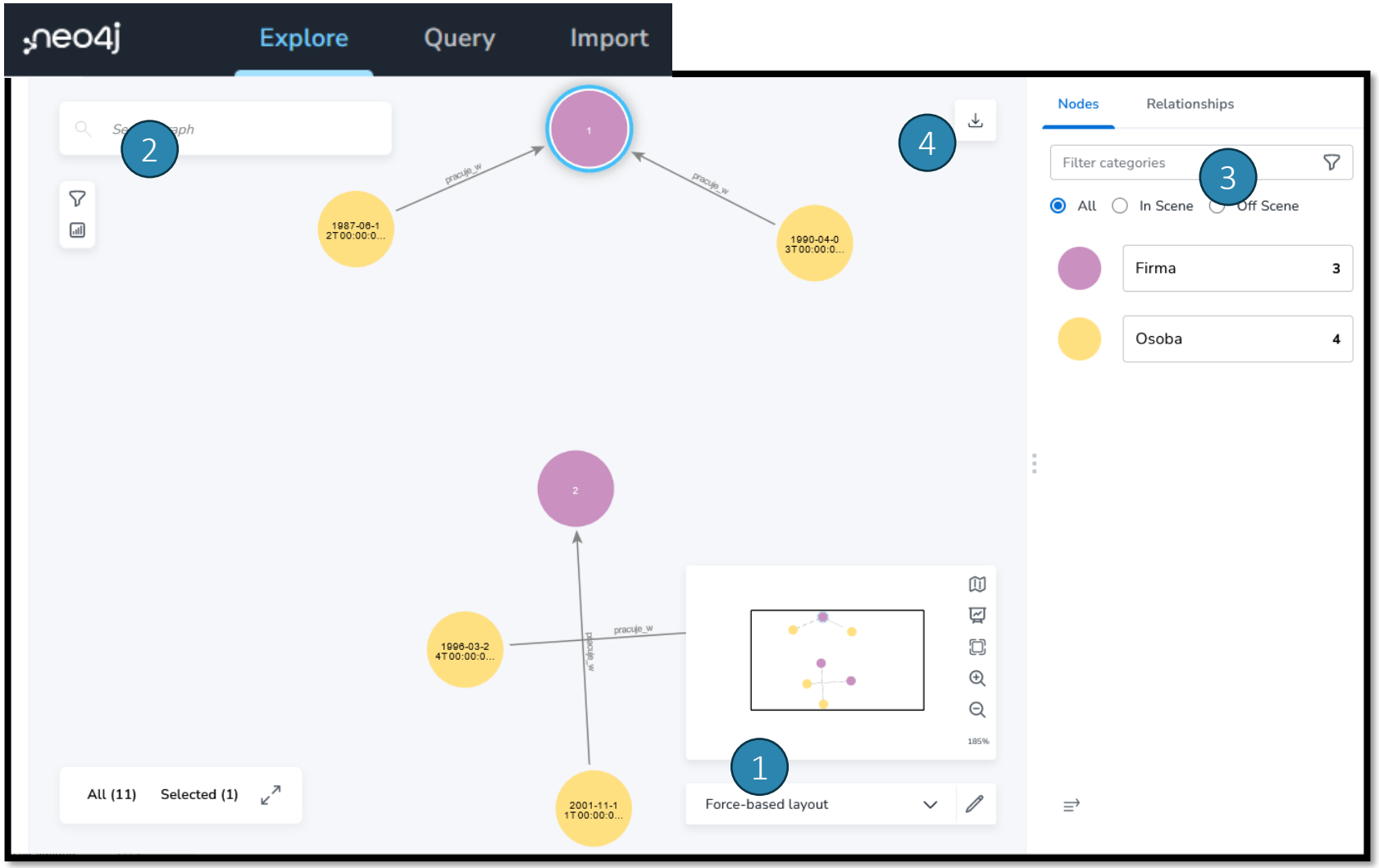
Import danych do grafu

Zaimportuj z wykorzystaniem „importera” przygotowane pliki CSV do bazy grafowej Neo4j.

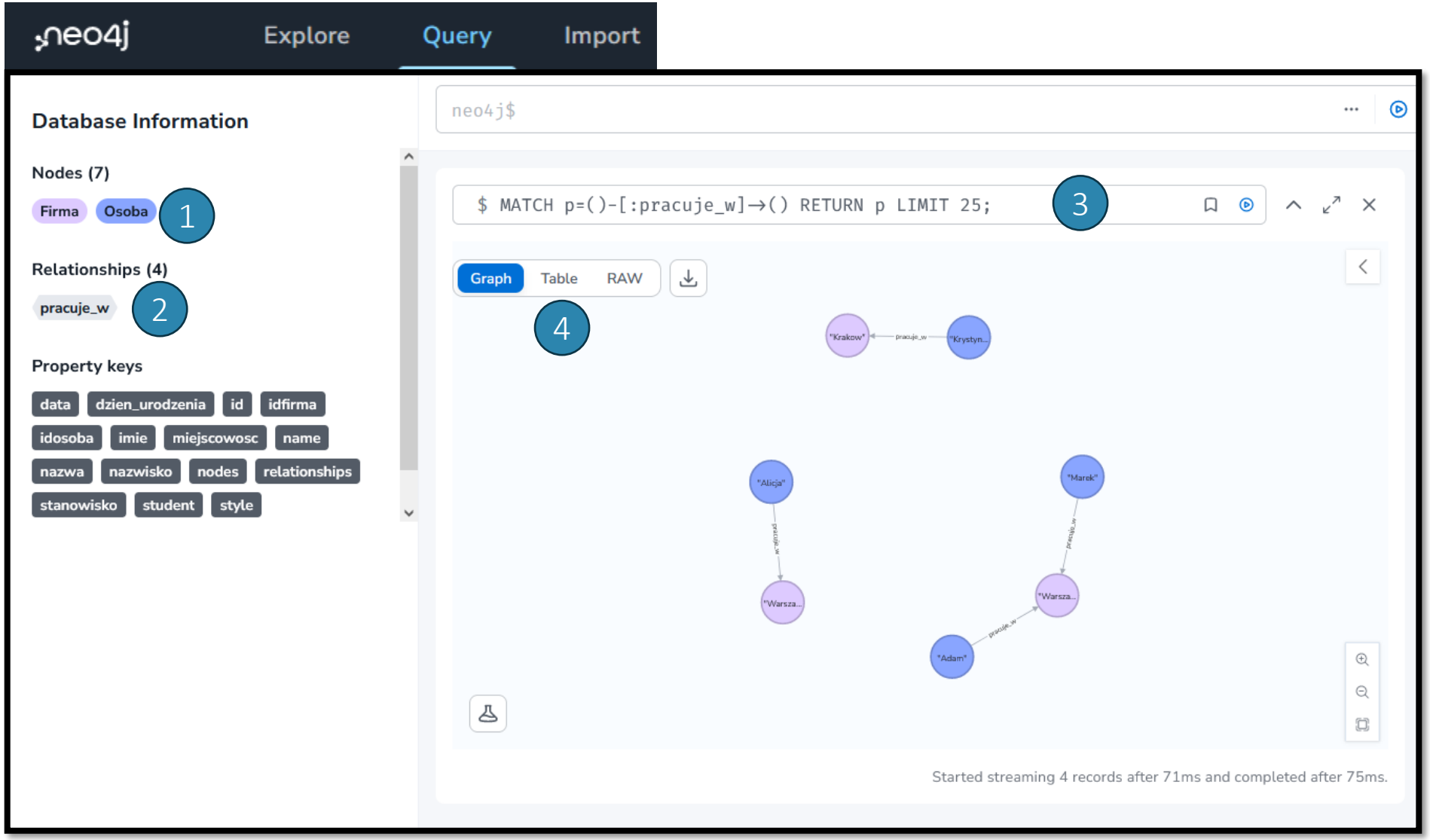
Obszary zarządzania danymi



Obszary zarządzania danymi



Obszary zarządzania danymi



Zapytania o dane CYPHER

Zapytanie o wszystkich studentów:

```
MATCH (o:Osoba)
WHERE o.student = true
RETURN o.imie, o.nazwisko, o.dzien_urodzenia
```

Zapytanie o wszystkie firmy z Warszawy:

```
MATCH (f:Firma)
WHERE f.miejscowosc = 'Warszawa'
RETURN f.nazwa
```

Zapytanie o osoby pracujące w firmie "GoodHome":

```
MATCH (o:Osoba)-[r:pracuje_w]->(f:Firma)
WHERE f.nazwa = 'GoodHome'
RETURN o.imie, o.nazwisko, r.stanowisko
```

Zapytania o dane CYPHER

Zapytanie o osoby, które nie są studentami i pracują w Warszawie:

```
MATCH (o:Osoba)-[r:pracuje_w]->(f:Firma)
WHERE o.student = false AND f.miejscowosc = 'Warszawa'
RETURN o.imie, o.nazwisko, f.nazwa, r.stanowisko
```

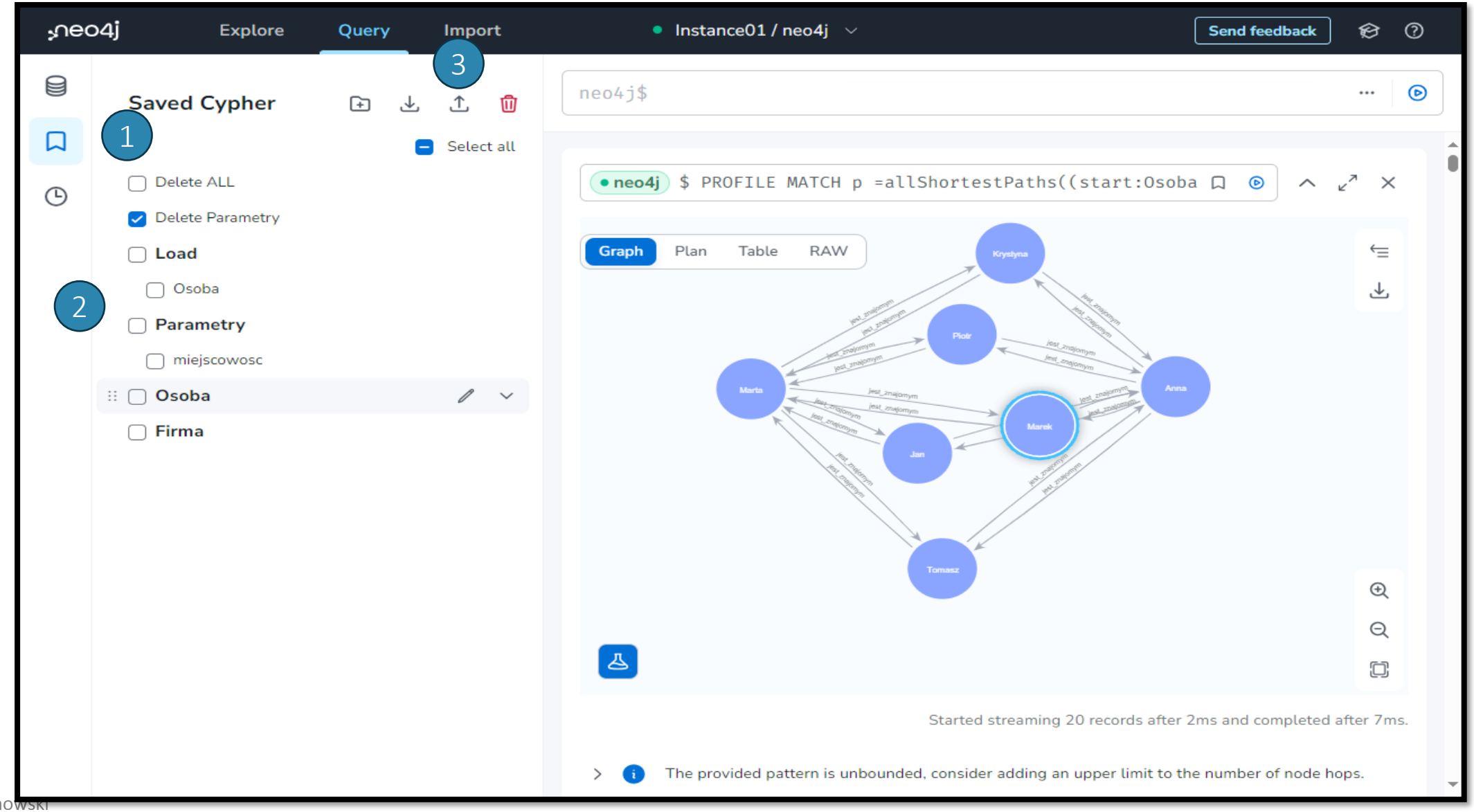
Zapytanie o firmy, w których pracują osoby urodzone po 1995 roku:

```
MATCH (o:Osoba)-[r:pracuje_w]->(f:Firma)
WHERE o.dzien_urodzenia >= '1995-01-01'
RETURN f.nazwa, COLLECT(o.imie + ' ' + o.nazwisko) AS pracownicy
```

Zapytanie o liczbę osób pracujących w każdej firmie:

```
MATCH (o:Osoba)-[r:pracuje_w]->(f:Firma)
RETURN f.nazwa, COUNT(o) AS liczba_pracownikow
```

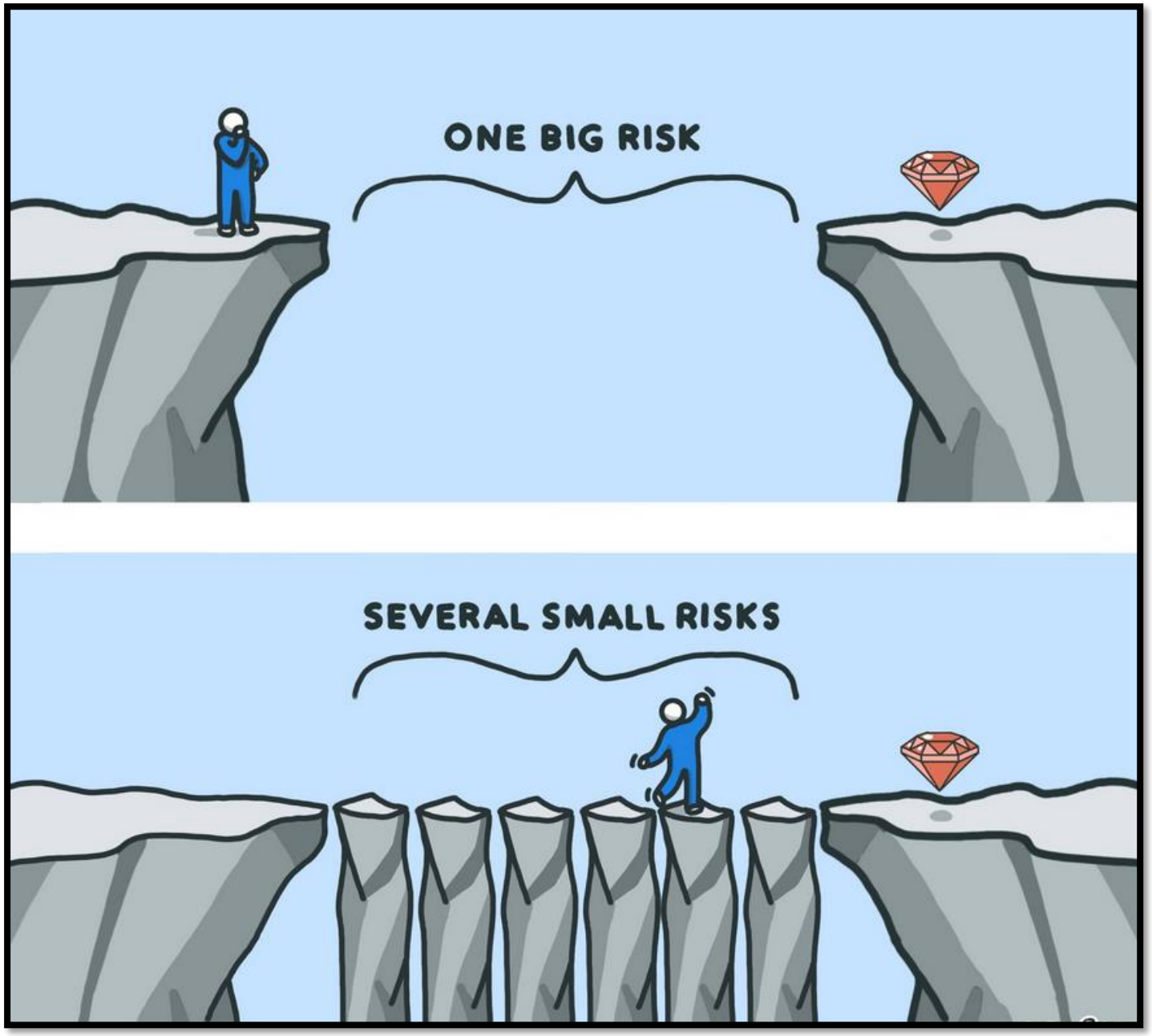
Zapytania o dane CYPHER



Zadanie: System analizy danych

System analizy danych

Zbuduj menu w bazie Neo4j bazując na przykładach zapytań CYPHER dla Systemu analizy danych


ONE BIG RISK
SEVERAL SMALL RISKS

Zadanie: Grafowy model danych

Zadanie (2 pkt)

Zamodeluj i wykonaj graf w bazie grafowej Neo4j reprezentujący .

Węzły (Nodes):
- Produkty: 10 elementów
  - nazwa: Nazwa produktu, np. "Smartfon"
  - sku: Unikalny kod produktu (SKU - Stock Keeping Unit)
- Magazyny: 3 elementy
  - lokalizacja: Lokalizacja magazynu, np. "Miasto A"
  - pojemnosc: Maksymalna pojemność magazynu

Krawędzie (Edges):
- Dostarczanie (DOSTARCZONO_DO): 20 elementów
  - data_dostarczenia: Data dostarczenia produktów do magazynu
  - ilosc_dostarczona: Ilość dostarczonych produktów

Przedstawienie wyniku

- Obrazek grafu w PDF
- Umieszczenie wyniku w Moodle

Termin

- Podany przy zadaniu w Moodle (około 2 tygodnie)