STUDIA POLITOLOGICZNE VOL. 80
DOI: 10.33896/SPolit.2026.80.1

*Dariusz Jaruga**

Baza grafowa w dydaktyce

Graph Database in Education

Słowa kluczowe: *baza grafowa, edukacja, dydaktyka, graf skierowany, graf nieskierowany, modelowanie grafowe*

Key words: *graph database, education, didactics, directed graph, undirected graph, graph modelling*

Abstrakt: *W artykule zasygnalizowano ogromny potencjał kryjący się w grafowych bazach danych w kontekście dydaktyki i nauki. Publikacja podkreśla, że mimo złożoności technologicznej, grafowe bazy danych mogą być intuicyjne i dostępne dla użytkowników niezależnie od ich poziomu zaawansowania w zakresie ICT. Szczególną uwagę poświęcono bazom grafowym, które umożliwiają graficzną wizualizację relacji pomiędzy obiektami, ułatwiając zrozumienie skomplikowanych struktur. Omówiony w artykule przykład praktyczny ilustruje, że przy pomocy prostych instrukcji możliwe jest efektywne korzystanie z tego typu narzędzi bez głębokiej wiedzy informatycznej. Wskazano, że pomimo wielu zalet, bazy grafowe nie zastępują innych typów baz danych, np. relacyjnych, lecz stanowią cenne ich uzupełnienie w zakresie analizy relacji i struktur. W artykule podkreślono, że włączenie baz grafowych w proces dydaktyczny pomaga w rozwoju kompetencji w zakresie prototypowania modeli badawczych, wspiera naukę myślenia relacyjnego i pozwala nieco inaczej kształtować umiejętności analityczne studentów i badaczy.*

* ORCID ID: https://orcid.org/0000-0001-7700-6249; doktor nauk społecznych w zakresie nauk o bezpieczeństwie, adiunkt Katedry Technologii Informacyjnych, Wydział Nauk Politycznych i Studiów Międzynarodowych, Uniwersytet Warszawski. E-mail: d.jaruga@uw.edu.pl.

Abstract: *The article highlights the enormous potential of graph databases in the context of teaching and learning. The publication emphasises that, despite technological complexity, graph databases can be intuitive and accessible to users regardless of their level of ICT proficiency. Particular attention is paid to graph databases that enable graphical visualisation of relationships between objects, facilitating the understanding of complex structures. The practical example discussed in the article illustrates that with simple instructions, it is possible to use such tools effectively without in-depth IT knowledge. It is pointed out that despite their many advantages, graph databases do not replace other types of databases, such as relational databases, but are a valuable complement to them in terms of analysing relationships and structures. The article emphasises that the inclusion of graph databases in the teaching process helps to develop competences in the field of research model prototyping, supports the learning of relational thinking and allows for a slightly different approach to shaping the analytical skills of students and researchers.*

Wprowadzenie

Człowiek jako istota rozumna jest ciekawy otaczającego go świata. Teza postawiona w pierwszym zdaniu to pewnego rodzaju uogólnienie, od którego zapewne możliwe jest znalezienie pewnych wyjątków, tym niemniej ludzie od niepamiętnych czasów poszukują rozwiązań, których celem jest pełniejsze poznanie otaczającej ich rzeczywistości. Ludzie poszukują także rozwiązań, które ułatwią codzienną egzystencję. Postęp i rozwój współczesnej cywilizacji możliwy jest m.in. dzięki nauce i powiązanej z nauką edukacji. Edukacja odgrywa istotną rolę w tym procesie. Człowiek rozumny stara się zmieniać swoje otoczenie poprzez budowanie kolejnych warstw wiedzy stanowiących fundamenty dzisiejszej szeroko rozumianej cywilizacji. Przykładem kluczowych zmian otoczenia człowieka było między innymi opanowanie ognia, wynalezienie koła, rozwój mowy i pisma. Idąc dalej mamy kolejne wynalazki, które zmieniły świat, jak: maszyna parowa, elektryczność, półprzewodniki, procesory, komputery, zawansowane oprogramowanie, bazy danych czy sztuczna inteligencja. Każda z nich na swój sposób przeorganizowała otaczającą nas rzeczywistość. Wymienione pokrótce wynalazki budziły i nadal budzą skrajne emocje, od tych negatywnych po te euforycznie pozytywne. Wymienione wybrane wynalazki mają jedną wspólną cechę. Każda z nich na swój sposób wspomaga człowieka eliminując jego ograniczenia biologiczne. Ludzie bardzo dobrze radzą sobie z wykształconymi w procesie ewolucji naturalnymi zagadnieniami związanymi z szeroko rozumianym środowiskiem naturalnym. Potrafimy obserwować otaczającą nas przyrodę i to co się w niej dzieje na poziomie niezbędnym do życia. W miarę dobrze potrafimy się poruszać, wcho-

dzić w interakcję z otaczającą nas przyrodą czy innymi ludźmi. Na swój sposób potrafimy sprawnie komunikować się i dbać o codzienne sprawy wymagające zwykłych umiejętności manualnych. Nieco gorzej sprawa wygląda, gdy coś musimy policzyć. Nie potrafimy tego robić z taką wydajnością z jaką robią to zwykłe kalkulatory, nie wspominając już o komputerach, czy super komputerach (KDM[1]). Kolejną ilustracją naszych ograniczeń jest umiejętność interpretacji danych. Jeśli jest ich sporo, uciekamy się do ich przedstawienia w postaci tabeli, wykresu czy diagramu, aby łatwiej nam było zrozumieć te dane przekształcając je na użyteczną dla nas informację. Nie potrafimy także zauważyć zależności w dużych zbiorach danych, jak i bez użycia dodatkowych narzędzi nie jesteśmy w stanie wyciągnąć nowej użytecznej dla nas informacji ze zbioru big data. W przywołanych na wstępie kilku przykładach wyraźnie widać, że nasze umiejętności ewolucyjne nie sprawdzają się w takich przypadkach. To co dla technologii ICT jest proste, dla ludzi zazwyczaj jest bardzo trudne. Wynika to po części z tego, że biologicznie zostaliśmy przez naturę dostosowani do czegoś innego.

Nie oznacza to jednak, że nie potrafimy się z tego typu problemami uporać. Silną stroną człowieka jest jego mózg i umiejętność logicznego rozumowania. Posiadamy jeszcze jedną niezwykle unikalną cechę – kreatywność, która pozwoliła nam być w tym miejscu, w którym się obecnie znajdujemy. Naszą niezwykle przydatną umiejętnością jest tworzenie nowych lepszych narzędzi przy pomocy narzędzi mniej doskonałych. Potrafimy także całkiem sprawnie przekazywać wiedzę nowym pokoleniom w procesie edukacji, aby mogły one tworzyć nowe, jeszcze lepsze narzędzia, które wprowadzą ludzkość na jeszcze wyższy poziom doskonałości. Badania naukowe i edukacja są bardzo blisko siebie, jeśli dodamy do tego jeszcze ciekawość poznania świata, to mamy trzy solidne filary prowadzące do pokonywania kolejnych biologicznych barier człowieka w poznawaniu otaczającego go świata. Wśród wielu narzędzi wspomagających te procesy znajdują się rozwiązania z obszaru ICT.

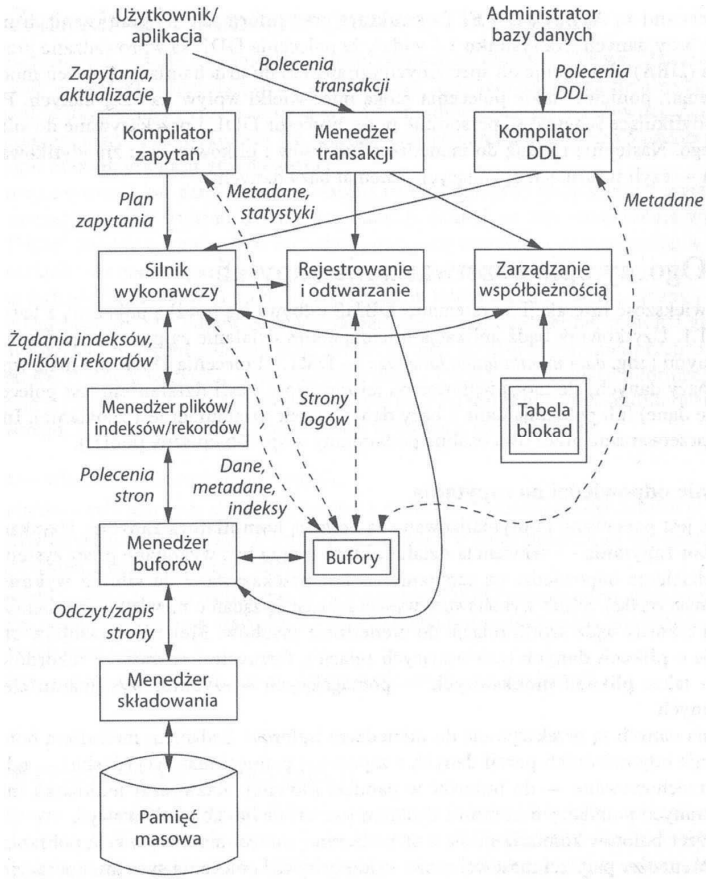
To one w znaczącym stopniu pozwoliły na gwałtowny rozwój współczesnej cywilizacji. Wśród nich znajdują się systemy baz danych i powiązane z nimi systemy zarządzania bazami danych DBMS (ang. Database Management Systems). Warto zaznaczyć, że DBMS są uważane za najbardziej złożone oprogramowanie współczesnych czasów[2]. Złożoność typowego systemu zarządzania bazą danych przedstawiono na rysunku 1.

---

1 KDM – Komputery Dużej Mocy.

2 J.D. Ullman, R. Meryk, J. Widom, *Podstawowy kurs systemów baz danych*, Gliwice 2011, s. 21.

Rysunek 1. Komponenty systemu zarządzania bazą danych



Źródło: J.D. Ullman, R. Meryk, J. Widom, *Podstawowy kurs systemów baz danych*, Gliwice 2011, s. 25.

Za złożonością systemu do zarządzania bazą danych kryje się ogromna siła sprawnego zarządzania danymi, począwszy od niewielkich zbiorów a skończywszy na zasobach danych cyfrowych określanych jako big data. Dodatkowo format przechowywanych danych może być dostosowany do ich indywidualnych cech i struktury. To właśnie między innymi te cechy spowodowały, że bazy danych są tak szeroko stosowane w niezliczonej ilości obszarów, jak: bankowość, przemysł, internet, wojsko, administracja publiczna, badania naukowe i edukacja.

Stąd niezwykle ważnym jest włączyć bazy danych do programów nauczania na różnych szczeblach edukacji stosownie do wieku. Bazy danych, choć są złożonym oprogramowaniem, to z perspektywy użytkownika mogą być stosunkowo proste w obsłudze jako narzędzie. Stąd na potrzeby niniejszego opracowania postawiono tezę, że grafowa baza danych (jeden z wielu rodzajów baz danych) jest łatwym do opanowania narzędziem dla studentów dowolnego

kierunku studiów, włączając w to studentów nauk niezwiązanych w bezpośredni sposób z naukami ścisłymi.

Mając na względzie postawioną tezę warto na chwilę spojrzeć na problematykę baz danych z pewnego dystansu. Patrząc z perspektywy historycznej, ludzie już od zarania dziejów starali się zapisywać różne informacje w taki sposób, aby przetrwały one w niezmienionej postaci przez długi czas. Początkowo były to malunki na ścianach, potem pojawiły się tabliczki gliniane, stelle, zwoje skórzane, papirusy itp. Jeszcze w latach 50. ubiegłego wieku dane cyfrowe zapisywano na specjalnie do tego celu przygotowanych kartach dziurkowanych czy dziurkowanych taśmach papierowych. Dalszy rozwój elektroniki przyczynił się do tego, że dane w postaci plików były zapisywane na nośnikach magnetycznych. Część z tych plików posiadała wewnętrzną strukturę, która zawierała zapis danych cyfrowych w postaci ustrukturyzowanych rekordów, będących protoplastą struktur wykorzystywanych we współczesnych rozwiązaniach bazodanowych. Prawdziwa rewolucja w tym zakresie dokonała się w 1970 roku za sprawą Teda Codda, który opublikował artykuł dotyczący relacyjnego modelu danych[3]. Przedmiotowy artykuł stał się motorem napędowym do zbudowania powszechnie występujących dziś relacyjnych baz danych opartych o tabele i relacje. Wśród przedstawicieli rozwiązań tej grupy warto wymienić bazę Oracle, IBM DB2, MySQL, MSSQL, PostgreSQL, Fire Bird, SQLite i wiele innych[4]. Zdominowały one w swoim czasie rynek przechowywania danych. W późniejszym okresie, rozwój technologii stosowanych w internecie oraz pojawienie się mediów społecznościowych przyczyniły się do powstania nierelacyjnych baz danych (1998 r.), w których przyjęto inne niż tabelaryczne sposoby przechowywania danych[5]. W skrócie, opracowane nierelacyjne bazy danych pozwalały przechowywać ogromne ilości danych zorganizowanych w różny sposób na wielu serwerach jednocześnie. Relacyjne bazy danych ze względu na ACID nie są aż tak bardzo podatne na proste skalowanie horyzontalne[6]. Wśród nierelacyjnych baz danych, określanych często mianem NoSQL,

---

3 E.F. Codd, *A relational model of data for large shared data banks*, «Communications of the ACM» 1970, t. 13, nr 6, doi: 10.1145/362384.362685.

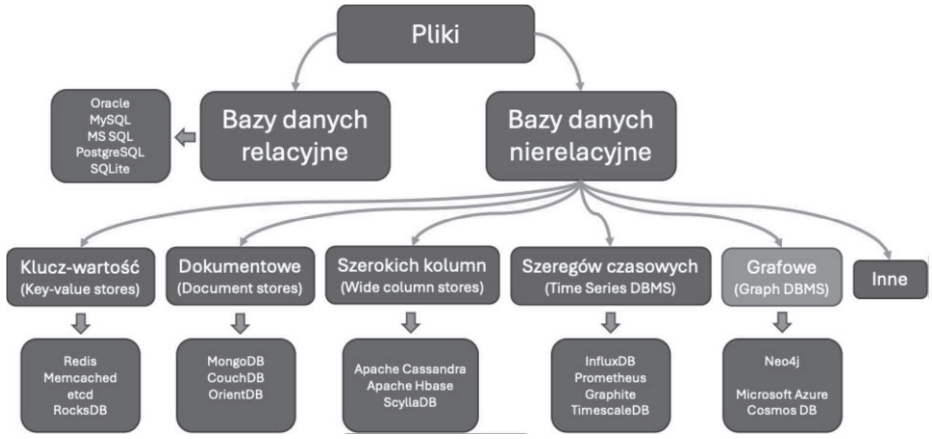
4 Więcej szczegółowych informacji na temat systemów baz danych jest dostępna we wskazanym źródle: *DB-Engines – Knowledge Base of Relational and NoSQL Database Management Systems*, https://db-engines.com/en/ (30.12.2025).

5 P.M. Tracz, M. Plechawska-Wójcik, *Comparative analysis of the performance of selected database management system*, «Journal of Computer Sciences Institute» 2024, t. 31, doi: 10.35784/jcsi.5927, akap. Charakterystyka systemów baz danych.

6 Więcej na temat baz danych w tym ACID: A. Silberschatz, H.F. Korth, S. Sudarshan, *Database system concepts*, New York, NY 2011, rozdz. 14; *ACID*, [w:] *Wikipedia*, 2025; *ACID Databases – Atomicity, Consistency, Isolation & Durability Explained*, 17.01.2024, https://www.freecodecamp.org/news/acid-databases-explained/ (30.12.2025).

wyróżnić można różne ich typy, takie jak bazy klucz-wartość, dokumentowe, bazujące na szerokich kolumnach, szeregach czasowych, grafowe itp. Każdy z wymienionych typów baz danych charakteryzuje się odmienną strukturą przechowywania danych i dedykowany jest do innych zastosowań. Schematycznie kolejne fazy rozwoju systemów baz danych, począwszy od plików a skończywszy na współczesnych rozwiązaniach, przedstawiono na rysunku 2.

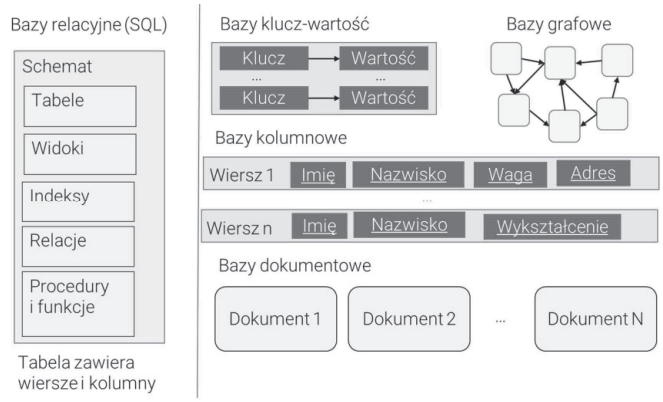
Rysunek 2. Techniki przechowywania danych od plików po różne modele systemów baz danych



Źródło: opracowanie własne.

Patrząc na bazy relacyjne (SQL) i nierelacyjne (NoSQL) w bardzo uproszczony sposób możliwe jest pokazanie struktur, w których przechowywane są dane. W przypadku baz relacyjnych dane przechowywane są w tabelach. Pomiędzy tabelami mogą być zdefiniowane relacje wskazujące na zależność pomiędzy danymi znajdującymi się przykładowo w dwóch tabelach. Takim klasycznym przykładem zależności pomiędzy tabelami może być tabela zawierająca dane osobowe człowieka, jak np. imię, nazwisko i druga tabela zawierająca dane dotyczące jego adresu zamieszkania. Z kolei baza klucz-wartość przechowuje pary parametrów, nazwę parametru i przypisaną do danego parametru wartość. Bazy kolumnowe przypominają w zakresie ich definicji tabele z baz relacyjnych, jednak to co je wyróżnia, to budowa. Tabele w bazach danych typu wide column stories mogą zawierać bardzo wiele kolumn (setki, tysiące…) gdzie każdy wiersz może zawierać inny zestaw kolumn. Inaczej niż w bazach relacyjnych, w których każdy wiersz składa się ze stałej liczby kolumn zdefiniowanej w schemacie tabeli. Baza dokumentowa w uproszczeniu przechowuje dokumenty w kolekcjach. Wyróżnia się tym, że każdy z dokumentów może mieć osobną strukturę i zawierać inne informacje.

Rysunek 3. Wybrane modele baz danych

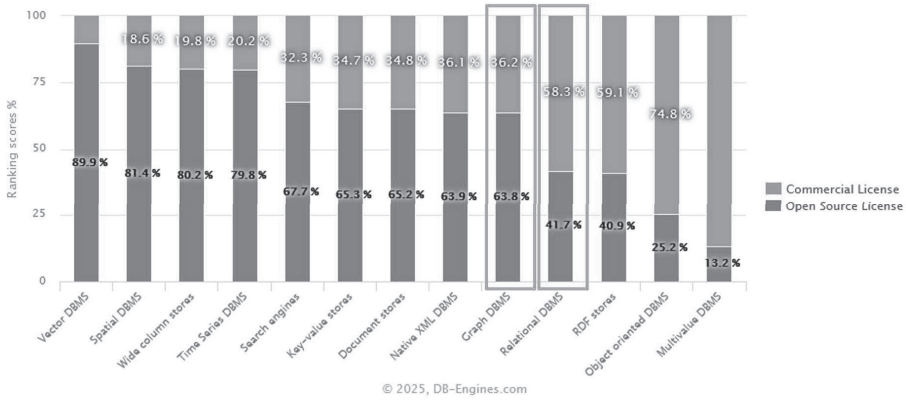


Źródło: opracowanie własne.

Na końcu mamy bazy grafowe, które przechowują informacje o obiektach i relacje pomiędzy nimi[7].

Poza tymi czterema typami baz danych mamy jeszcze inne, nie omówione w niniejszym artykule a wymienione na kolejnym wykresie. Zostały na nim przedstawione rodzaje baz danych wraz z udziałem rynkowym rozwiązań otwarto źródłowych i komercyjnych. Na przedmiotowym wykresie zaznaczono dwa rodzaje baz danych: relacyjne i grafowe. To, co warto zauważyć, to bazy

Rysunek 4. Ranking udziału poszczególnych modeli baz danych w rynku z podziałem na rozwiązania komercyjne i na otwartych licencjach



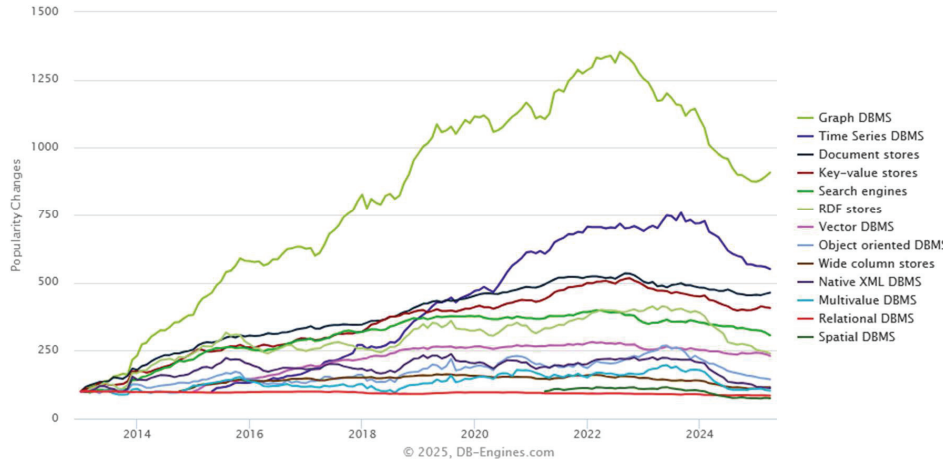
Źródło: https://db-engines.com/en/ranking_osvsc (1.03.2025).

[7] *DB-Engines – Knowledge Base of Relational and NoSQL Database Management Systems*; A.B.M. Moniruzzaman, S.A. Hossain, *NoSQL Database: New Era of Databases for Big data Analytics – Classification, Characteristics and Comparison*, arXiv, 30.06.2013, http://arxiv.org/abs/1307.0191 (30.12.2025).

grafowe w prawie 64% stanowią rozwiązania otwarto źródłowe a nieco ponad 36% komercyjne. Zupełnie odwrotnie sytuacja przedstawia się w przypadku baz relacyjnych[8].

To, co warto także wyraźnie zaznaczyć, grafowe bazy danych począwszy od 2014 roku cieszą się największą dynamiką wzrostu w zakresie zastosowań. Oznacza to, że spora część użytkowników baz danych znajduje w nich wartościowy potencjał[9].

Rysunek 5. Zmiany popularności baz danych w poszczególnych kategoriach



Źródło: https://db-engines.com/en/ranking_categories (1.03.2025).

Baza grafowa

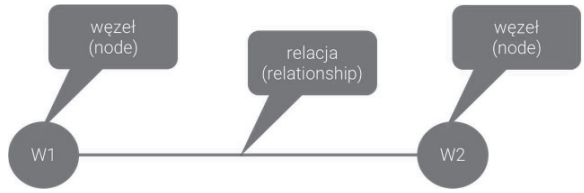
Przechodząc do głównego celu niniejszej publikacji, którym jest pokazanie, że bazy grafowe są dostępne i zrozumiałe dla każdego, warto w kilku zdaniach nieco je przybliżyć. W skrócie baza grafowa przechowuje informacje w węzłach i relacjach pomiędzy nimi. Każda relacja łączy ze sobą dwa węzły, tak jak to pokazano na rysunku 6.

Wśród grafów wyróżniamy grafy skierowane i nieskierowane. O tym, czy dana relacja jest skierowana lub nie, decyduje charakter relacji występujący pomiędzy węzłami. Jeśli węzeł W1 jest w relacji z W2 i jednocześnie węzeł W2 jest w relacji z W1, to wówczas możemy mówić o relacji dwukierunkowej

8 *DB-Engines Ranking Open Source vs. Commercial DBMS* [na:] https://db-engines.com/en/ranking_osvsc (30.12.2025).

9 *DB-Engines Ranking per database model category*, https://db-engines.com/en/ranking_categories (30.12.2025).

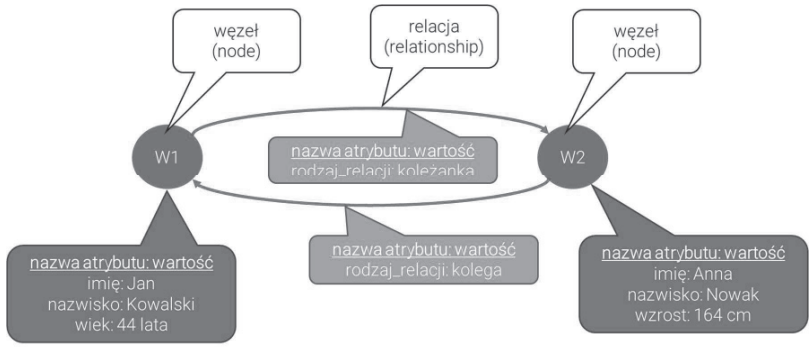
Rysunek 6. Najprostszy graf nieskierowany



Źródło: opracowanie własne.

i grafie nieskierowanym. W przeciwnym przypadku, gdy węzeł W1 jest w relacji z węzłem W2 i jednocześnie W2 nie jest w relacji z W1, wówczas mamy do czynienia z relacją jednokierunkową, czyli grafem skierowanym[10]. W dalszej części niniejszego artykułu została omówiona baza Neo4j, która obsługuje grafy skierowane[11]. Chcąc uzyskać w bazie danych Neo4j relację nieskierowaną, trzeba zdefiniować dwie relacje skierowane, jedną od węzła W1 do W2 i drugą od węzła W2 do W1. Tego typu podejście do sprawy tylko z pozoru jest utrudnieniem. Relacje skierowane w wielu przypadkach mogą być bardziej użyteczne aniżeli relacje nieskierowane. Oto prosty przykład, mamy dwie osoby, mężczyznę i kobietę, których łączą relacje koleżeńskie. Możemy zatem przyjąć, że jeśli węzeł W1 opisuje mężczyznę, a węzeł W2 opisuje kobietę, to pomiędzy nimi zachodzi podwójna relacja. W2 jest dla W1 koleżanką, a W1 dla W2 kolegą. Schematycznie taką sytuację przedstawiono na rysunku 7.

Rysunek 7. Przykład prostego dwuwęzłowego grafu skierowanego z atrybutami (Neo4j)



Źródło: opracowanie własne.

[10] S. Anuyah, V. Bolade, O. Agbaakin, *Understanding Graph Databases: A Comprehensive Tutorial and Survey*, arXiv, 15.11.2024, http://arxiv.org/abs/2411.09999 (30.12.2025).

[11] *Neo4j Graph Database & Analytics – The Leader in Graph Databases*, 6.01.2026, https://neo4j.com/ (22.02.2026).

Dodatkowo, co warto podkreślić, węzły zawierają dodatkowe informacje o kobiecie i mężczyźnie. Na temat mężczyzny wiemy, że nazywa się Jan Kowalski i ma 44 lata, w przypadku kobiety, Anny Nowak znamy jej wzrost. Widzimy także łączące ich relacje. One także mogłyby posiadać dodatkowe atrybuty, podobnie jak w przypadku węzłów. To, co powinno zwrócić uwagę, to fakt, że każdy z węzłów posiada nieco inny zestaw atrybutów. W1 posiada atrybut wiek, ale nie posiada atrybutu wzrost. W przypadku W2 sytuacja jest odwrotna. Warto w tym miejscu dodać, że węzły jak i relacje w bazie Neo4j mogą posiadać dowolną liczbę atrybutów[12]. Jest to bardzo przydatne, gdy budowany jest graf z niekompletnych informacji. Co do zasady przyjmuje się, że węzły to obiekty takie jak: osoby, miasta, budynki, serwery, firmy, organizacje, państwa. Z kolei relacjami mogą być: zależności służbowe, relacje społeczne, więzy rodzinne, droga (w rozumieniu odcinek łączący dwa punkty), reakcja chemiczna itp. Czasami pojawią się wątpliwości w zakresie, czy dany obiekt powinien być relacją, czy też węzłem. Dobrym tego przykładem może być umowa. Jeśli łączy tylko dwie osoby, można ją zapisać w bazie jako relację, ale w sytuacji, gdy stronami umowy jest więcej osób, albo osób i instytucji, wówczas umowa powinna być węzłem. Warto podkreślić aspekt, że w bazie Neo4j możemy przechowywać wiele różnych rodzajów obiektów (węzły), np. osoby, firmy, umowy i wiele różnych rodzajów relacji, np. służbowe, rodzinne etc. Do komunikacji z bazą grafową Neo4j używany jest dedykowany do tego celu język Cypher[13]. Ograniczona objętość niniejszego artykułu nie powala omówić wszystkich możliwości tego języka, ale kilka kluczowych informacji, niezbędnych do rozpoczęcia pracy z Neo4j, zostało przedstawionych w dalszej części artykułu. To co jest istotne, to fakt, że język ten jest dość prosty w zakresie dodawania do bazy danych węzłów i ich atrybutów. Na poniższym rysunku przedstawiono symbolicznie dane do dwóch węzłów W1 i W2 oraz dwa polecenia, przy pomocy których zostały one utworzone w bazie Neo4j.

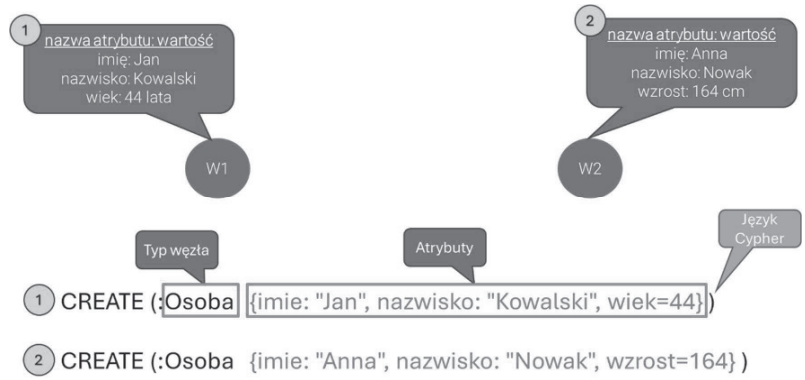
Utworzenie pierwszego węzła należy zacząć od polecenia CREATE, po czym, zachowując pokazaną na rysunku składnię, podać nazwę typu węzła, a następnie w nawiasach klamrowych odpowiednio nazwę atrybutu i jego wartość. W ten sposób zostaje utworzony pojedynczy węzeł. To samo należy zrobić z kolejnymi węzłami. W kolejnym kroku można przejść do utworzenia relacji pomiędzy tymi węzłami. Zgodnie z tym co zasygnalizowano wcześniej, zostaną utworzone dwie relacje: kolega i koleżanka.

---

12 *Graph database concepts – Getting Started*, https://neo4j.com/docs/getting-started/appendix/graphdb-concepts/ (30.12.2025); *What is a graph database – Getting Started*, https://neo4j.com/docs/getting-started/graph-database/, (30.12.2025); *Cypher – Neo4j Documentation*, https://neo4j.com/docs/cypher/ (30.12.2025).

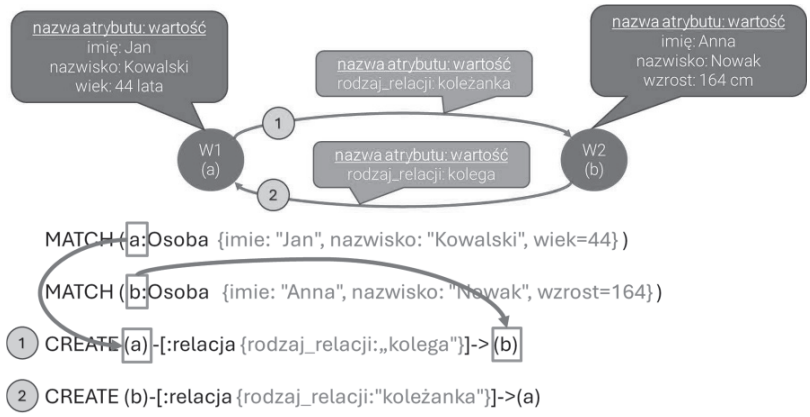
13 *Cypher – Neo4j Documentation*.

Rysunek 8. Tworzenie węzłów w bazie Neo4j



Źródło: opracowanie własne.

Rysunek 9. Tworzenie relacji w bazie Neo4j



Źródło: opracowanie własne.

Tutaj proces tworzenia relacji wymaga krótkiego komentarza. Otóż zanim zostanie utworzona pierwsza relacja, baza danych Neo4j musi „wiedzieć", jakie węzły chcemy taką relacją połączyć. Dlatego też dwa pierwsze polecenia MATCH wyszukują nam odpowiednie węzły i na chwilę przypisują je do dwóch zmiennych oznaczonych literkami „a" i „b"[14]. Te zmienne istnieją tylko przez chwilę. Dopiero po przypisaniu informacji o węźle W1 (do zmiennej a) i W2 (do zmiennej b) możliwe jest utworzenie relacji pomiędzy węzłami W1 i W2.

[14] W bazie Neo4j istnieją też inne metody wyszukiwania węzłów, które można połączyć relacją. Więcej informacji na ten temat jest dostępna na stronie https://neo4j.com/docs/ (22.12.2025).

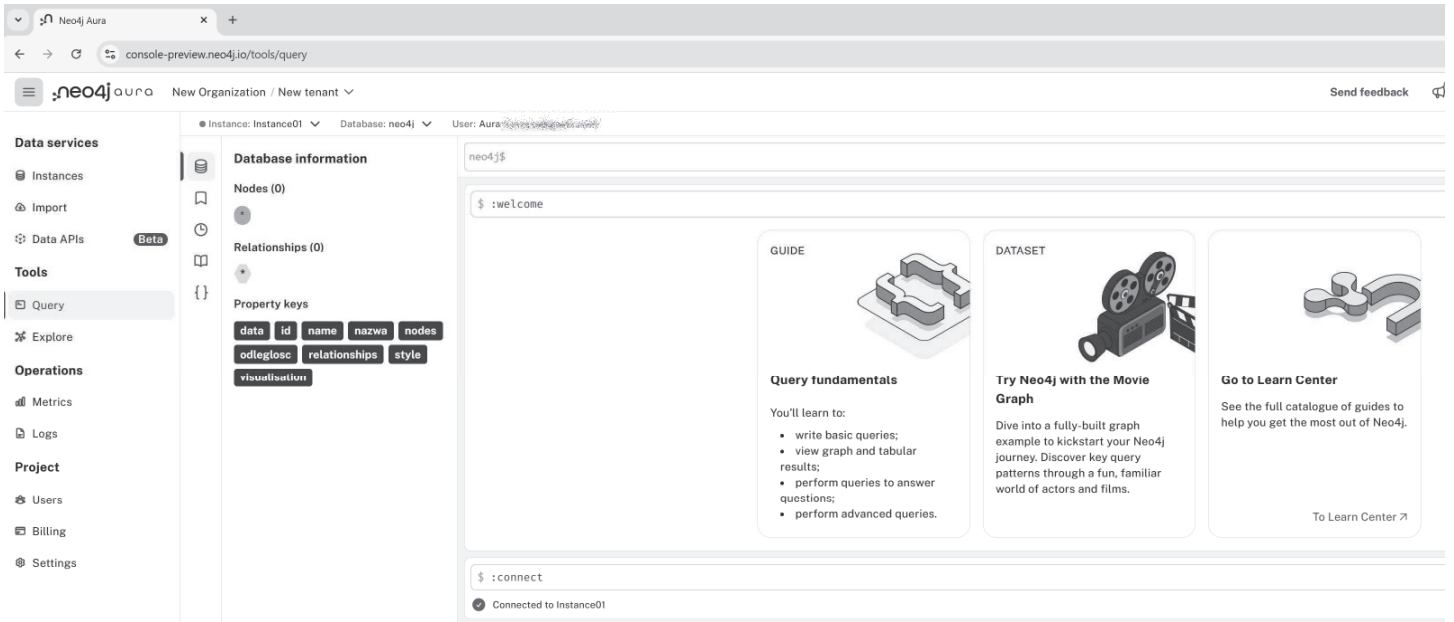
Podobnie jak w poprzednim przykładzie jest to realizowane poprzez polecenie CREATE, po którym zgodnie ze składnią w nawiasach okrągłych podajemy dane dotyczące węzła, np. „a", następnie w nawiasach kwadratowych definiujemy rodzaj relacji i w kolejnym kroku wskazujemy kierunek relacji, po czym instrukcję kończymy wskazaniem na drugi węzeł, w naszym przypadku „b". Tak samo postępujemy ze zdefiniowaniem w bazie drugiej relacji. W ten sposób grafowa baza danych może być rozbudowywana o kolejne węzły i relacje. Ważne jest to, że budując taką bazę, uczeń, naukowiec czy pracownik może skoncentrować się na małym fragmencie tematu, jakim się zajmuje. Przykładowo, można wyobrazić sobie sytuację, w której do bazy danych wprowadzane są kolejne osoby jako węzły i relacje między nimi, np. pokrewieństwa. W kolejnym kroku mając już wprowadzone wszystkie dane w zasadzie „od ręki" możliwe jest wyświetlenie drzewa zależności. Podobnie można uczynić z relacjami występującymi pomiędzy osobami a relacjami przynależności do np. partii politycznych. Warto w tym miejscu dodać, że w jednej bazie danych poszczególne węzły mogą być łączone różnymi rodzajami relacji. Jeszcze inny przykład to budowanie relacji pomiędzy wiadomościami w medium społecznościowym a kontami, które te wiadomości opublikowały lub udostępniły dalej celem identyfikacji źródła informacji, a być może i dezinformacji. Budując graficzną interpretację obiektów i zależności pomiędzy nimi łatwiej jest dostrzec pewne relacje występujące między nimi. Czasami nadmiar informacji dla ludzi jest trudny do zinterpretowania, a oprogramowanie komputerowe radzi sobie z tym zadaniem znacznie lepiej. Posiadanie umiejętności dodawania węzłów do bazy Neo4j i tworzenia między nimi relacji jest wystarczające by pracować w podstawowym zakresie z bazą grafową. Autor niniejszego opracowania wielokrotnie szkolił studentów z bazy Neo4j i z obserwacji własnych wynika, że dwie godziny dydaktyczne są wystarczające do tego, aby studenci, niezależnie od ich umiejętności technicznych w zakresie omówionym w niniejszym artykule, potrafili korzystać z bazy grafowej.

Co do samej bazy danych, Neo4j jest dostępna jako usługa w Cloud Computing, a osoby bardziej zaawansowane technicznie mogą ją zainstalować na własnym komputerze lub serwerze[15]. Niezależnie od sposobu uruchomienia bazy danych metoda pracy jest identyczna. Panel sterujący Neo4j zaraz po zalogowaniu się wygląda tak jak poniżej (rysunek 10).

Widać na nim najważniejsze elementy służące do obsługi bazy danych. Na szczególną uwagę zasługuje okienko ze znakiem zachęty „neo4j$" po którym należy wpisywać polecenia w języku Cipher. Zaraz obok znajduje się panel,

[15] Szczegółowe informacje na temat uruchomienia bazy danych są dostępne tu: *Neo4j Graph Database & Analytics – The Leader in Graph Databases*.

Rysunek 10. Panel do zarządzania bazą Neo4j



Źródło: zrzut ekranu z aplikacji Neo4j.

w którym prezentowana jest liczba węzłów i relacji jaką baza danych zawiera. Jest to też miejsce, w którym można zorientować się, z jak dużą bazą danych mamy do czynienia. W lewej skrajnej części znajdują się linki do najważniejszych funkcji bazy danych. Przedmiotowy panel stanowi narzędzie, przy pomocy którego, bez konieczności posiadania dodatkowych umiejętności, np. programowania, można w pełni pracować z grafową bazą danych. Za jego pośrednictwem możliwe jest wprowadzanie nowych danych jak i operowanie na nich.

Baza Neo4j – studium przypadku

W niniejszej części artykułu zaprezentowano bardzo prosty przykład użycia bazy grafowej polegający na znalezieniu najkrótszej drogi z miasta Kraków do miasta Szczecin. W tym celu do grafowej bazy danych wprowadzono węzły do dziewięciu miast. Wprowadzone węzły wraz z poleceniem ich utworzenia w bazie danych przedstawiono na rysunku 11.

Rysunek 11. Lista poleceń tworzących obiekty (miasta) w bazie Neo4j


```
create (KR:Miasto{nazwa:"Kraków"})
create (CZ:Miasto{nazwa:"Częstochowa"})
create (WR:Miasto{nazwa:"Wrocław"})
create (LO:Miasto{nazwa:"Łódź"})
create (WA:Miasto{nazwa:"Warszawa"})
create (PO:Miasto{nazwa:"Poznań"})
create (ZG:Miasto{nazwa:"Zielona Góra"})
create (BY:Miasto{nazwa:"Bydgoszcz"})
create (SZ:Miasto{nazwa:"Szczecin"})
```

Źródło: opracowanie własne, w tle zrzut mapy Polski z OpenStreetMap (https://www.openstreetmap.org/).

W kolejnym kroku, wprowadzono piętnaście relacji pomiędzy miastami, które ilustrują wybrane trasy, jakimi można się przemieścić pomiędzy kolejnymi węzłami, aby z Krakowa dojechać do Szczecina. Oczywiście nie są to wszystkie możliwe trasy jakimi można podróżować z Krakowa do Szczecina, wybrano

ich ograniczoną liczbę, aby przedstawiony przykład był czytelny. Każda z relacji posiada także zdefiniowany jeden atrybut zawierający liczbę kilometrów, jaką należy pokonać z jednego miasta do drugiego. Przykładowo z Warszawy do Bydgoszczy mamy 300 km. Mając już przygotowaną listę relacji, zapisano je w postaci języka Cipher. Niezbędne polecenia do utworzenia wszystkich 15 relacji przedstawiono na rysunku 12. W pierwszej części wskazujemy konkretne węzły i przypisujemy do nich zmienne tymczasowe, np. KR wskazuje na Kraków. W drugiej części poleceń tworzone są relacje pomiędzy miastami. To, na co warto zwrócić uwagę, to fakt, że każdą z relacji możemy tworzyć osobno albo wszystkie na raz. To od nas zależy jaką drogę wybierzemy.

Rysunek 12. Lista poleceń tworzących relacje pomiędzy miastami (obiekty) w bazie Neo4j

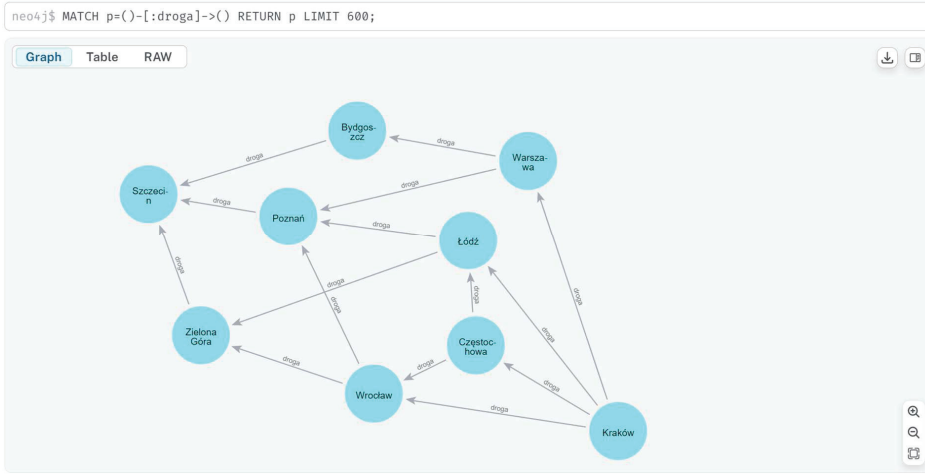

```
match (KR:Miasto {nazwa:"Kraków"})
match (CZ:Miasto {nazwa:"Częstochowa"})
match (WR:Miasto {nazwa:"Wrocław"})
match (LO:Miasto {nazwa:"Łódź"})
match (WA:Miasto {nazwa:"Warszawa"})
match (PO:Miasto {nazwa:"Poznań"})
match (ZG:Miasto {nazwa:"Zielona Góra"})
match (BY:Miasto {nazwa:"Bydgoszcz"})
match (SZ:Miasto {nazwa:"Szczecin"})

CREATE (KR)-[:droga {odleglosc:295}]->(WA)
CREATE (KR)-[:droga {odleglosc:210}]->(LO)
CREATE (KR) [:droga {odleglosc:140}] >(CZ)
CREATE (KR)-[:droga {odleglosc:270}]->(WR)
CREATE (CZ)-[:droga {odleglosc:190}]->(WR)
CREATE (CZ)-[:droga {odleglosc:150}]->(LO)
CREATE (WR)-[:droga {odleglosc:170}]->(PO)
CREATE (WR)-[:droga {odleglosc:150}]->(ZG)
CREATE (LO)-[:droga {odleglosc:200}]->(ZG)
CREATE (LO)-[:droga {odleglosc:220}]->(PO)
CREATE (WA)-[:droga {odleglosc:310}]->(PO)
CREATE (WA)-[:droga {odleglosc:300}]->(BY)
CREATE (PO)-[:droga {odleglosc:290}]->(SZ)
CREATE (BY)-[:droga {odleglosc:280}]->(SZ)
CREATE (ZG)-[:droga {odleglosc:190}]->(SZ)
```

Źródło: opracowanie własne, w tle zrzut mapy Polski z OpenStreetMap (https://www.openstreetmap.org/).

Po dodaniu do bazy Neo4j węzłów reprezentujących miasta oraz relacji pomiędzy nimi, struktura ta prezentuje się jako sieć połączonych ze sobą węzłów i krawędzi, odzwierciedlających trasy pomiędzy poszczególnymi miejscowościami (rysunek 13).

W górnej części rysunku widoczne jest polecenie, które pozwoliło znaleźć wszystkie relacje pomiędzy miastami. Ostatnie polecenie LIMIT 600 określa maksymalną liczbę krawędzi jaką chcemy, aby baza danych zwróciła w wyniku. W przypadku bardzo dużych baz danych mających tysiące węzłów i krawędzi, podanie takiego ograniczenia jest konieczne. Pozostał nam ostatni element, wyszukanie w takiej bazie optymalnych tras pod względem odległości, jakimi

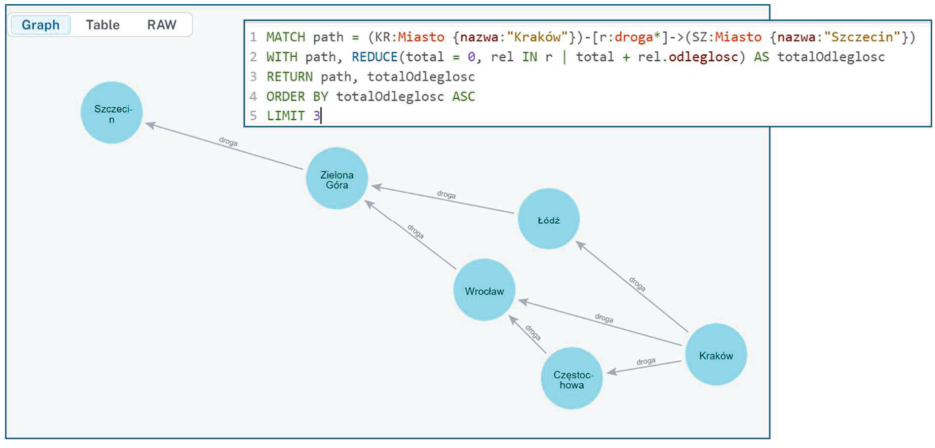
Rysunek 13. Węzły i relacje w bazie Neo4j



Źródło: opracowanie własne – zrzut ekranu z aplikacji Neo4j.

można przejechać z Krakowa do Szczecina. Do tego celu potrzebne jest już ułożenie bardziej złożonego zapytania do bazy Neo4j. Zapytanie to wskazuje, że interesuje nas dowolna droga z miasta Kraków do miasta Szczecin, przechodząca przez dowolną liczbę krawędzi i węzłów. Gdy takie wszystkie możliwe trasy zostaną odnalezione, to interesują nas tylko trzy najkrótsze pod względem sumarycznej odległości, posortowane od najkrótszej i dalej rosnąco do coraz dłuższych. Przykładowe polecenie i jego wynik zaprezentowano na rysunku 14.

Rysunek 14. Wybrane najbardziej optymalne (najkrótsze) trasy z grafu w Neo4j



Źródło: opracowanie własne – w tle zrzut ekranu z aplikacji Neo4j.

Powyżej przedstawiony wynik, można także zobaczyć w postaci tabeli, w której kolumna „path" zawiera informacje na temat przebiegu trasy przez kolejne węzły oraz kolumnę „totalOdleglosc", w której jest podana sumaryczna liczba kilometrów jaką dla każdej z tras należy przejechać, aby dotrzeć z Krakowa do Szczecina. Podsumowując, postawiony na początku problem znalezienia najkrótszej trasy został rozwiązany.

Rysunek 15. Dane szczegółowe wybranych tras z grafu w Neo4j – postać tabelaryczna

```
MATCH path = (KR:Miasto {nazwa:"Kraków"})-[r:droga*]->(SZ:Miasto {nazwa:"Szczecin"})
WITH path, REDUCE(total = 0, rel IN r | total + rel.odleglosc) AS totalOdleglosc
RETURN path, totalOdleglosc
ORDER BY totalOdleglosc ASC
LIMIT 3
```

Graph | Table | RAW

	path	totalOdleglosc
1	(:Miasto {nazwa: "Kraków"})-[:droga {odleglosc: 210}]->(:Miasto {nazwa: "Łódź"})-[:droga {odleglosc: 200}]->(:Miasto {nazwa: "Zielona Góra"})-[:droga {odleglosc: 190}]->(:Miasto {nazwa: "Szczecin"})	600
2	(:Miasto {nazwa: "Kraków"})-[:droga {odleglosc: 270}]->(:Miasto {nazwa: "Wrocław"})-[:droga {odleglosc: 150}]->(:Miasto {nazwa: "Zielona Góra"})-[:droga {odleglosc: 190}]->(:Miasto {nazwa: "Szczecin"})	610
3	(:Miasto {nazwa: "Kraków"})-[:droga {odleglosc: 140}]->(:Miasto {nazwa: "Częstochowa"})-[:	670

Źródło: opracowanie własne – zrzut ekranu z aplikacji Neo4j.

Podsumowanie

Omówiony w niniejszym artykule przykład użycia bazy grafowej ilustruje, że bardzo dobrze sprawdzi się tam, gdzie poza danymi równie ważne są relacje. Bazy grafowe, inaczej niż np. relacyjne bazy danych, pozwalają zgromadzić duże ilości danych, które mogą być zwizualizowane w taki sposób, aby były łatwiejsze do zrozumienia dla odbiorcy niż klasyczne tabele czy wykresy. Omówiona w artykule baza Neo4j charakteryzuje się niskim progiem wejścia. Wystarczy znajomość kilku podstawowych instrukcji, by całkiem sprawnie z niej korzystać, nie posiadając zaawansowanej wiedzy z obszaru ICT. Z pewnością bazy grafowe nie zastąpią relacyjnych baz danych tak w dydaktyce jak i w na-

uce, ale mogą stanowić bardzo dobre uzupełnienie umiejętności w zakresie odkrywania struktur i relacji. To, co zostało już powiedziane nieco wcześniej, węzły w obiektowej bazie danych odpowiadają konkretnym obiektom jak osoba, miasto, firma itp., natomiast relacje odpowiadają konkretnym faktom typu: pokrewieństwo, zależność służbowa, mieszka w, pracuje w, zna daną osobę, łączy dwa obiekty w określony sposób itp. Język Cipher, choć na początku może wydawać się nieco trudny składniowo, pozwala definiować relacje w sposób opisowy, bez używania złożonych konstrukcji stosowanych w relacyjnych bazach danych. W bazie obiektowej przy pomocy relatywnie prostego zapytania jesteśmy w stanie wyszukać osoby będące w jakiejś relacji, które pracują w danej firmie i jednocześnie mieszkają w określonych miastach. Bazy grafowe doskonale nadają się do analizy złożonych systemów, z którymi często spotykamy się w naukach społecznych. Pozwalają przyjrzeć się sieci społecznej w zakresie identyfikacji grup i struktur społecznych, analizować ewolucję powiązań w czasie czy odkrywać trudne do zauważenia zależności.

Powyżej zasygnalizowane cechy grafowych baz danych otwierają drogę do szybkiego prototypowania modeli badawczych. Baza grafowa uczy myślenia o badanych obiektach jako o sieci obiektów i powiązań pomiędzy nimi. Mamy zatem połączenie myślenia grafowego w sieciowej rzeczywistości, która pozwala nam w stosunkowo prosty sposób modelować zależności celem ich dokładniejszego zbadania. Jest to zupełnie inny sposób postrzegania rzeczywistości, w której odchodzimy na chwilę od tabelek na rzecz obiektów i zachodzących pomiędzy nimi relacji. Zatem model grafowy sprzyja intuicyjnemu odwzorowaniu rzeczywistych struktur społecznych, organizacyjnych, technicznych czy przyrodniczych. Wartością dodaną baz grafowych jest to, że atrybuty obiektów i zależności pomiędzy nimi nie są ukryte w tabelach, a występują razem z obiektem lub relacją. To ułatwia skoncentrowanie się nad ważnymi elementami bez zbędnego rozpraszania się na duże struktury tabel i relacji pomiędzy nimi. Baza grafowa pozwala skoncentrować się na jednym wybranym z niej fragmencie. Bez większych problemów możliwe jest rozbudowywanie bazy grafowej o nowe węzły i nowe relacje, by na podstawie dodanych nowych danych obserwować zachodzące w sieci zmiany. Model grafowy pozwala w sposób prosty i elastyczny rozszerzać strukturę danych o nowe obiekty i nowe relacje. Przykładowo, w zaprezentowanym w artykule grafie zawierającym miasta i połączenia pomiędzy nimi bez problemu można dodać inne relacje jak np. połączenia kolejowe. W tym miejscu warto postawić zasadnicze pytanie. Czy tego samego nie można uzyskać w klasycznych relacyjnych bazach danych typu SQL. Odpowiedź jest twierdząca, można, tyle tylko, że nakład pracy polegający na odwzorowaniu grafu w tabelach będzie złożony i trudny do zrealizowania, choć technicznie możliwy. Grafowe bazy danych

jak Neo4j charakteryzują się cechami, które w prosty sposób pozwalają odwzorować złożone struktury danych i relacje między nimi. Bazy grafowe są do tego celu stworzone. To potwierdza prawdziwość postawionej na początku artykułu tezy. W przypadku baz relacyjnych SQL, lepiej sprawdzą się tam, gdzie dane są ustrukturyzowane. Zatem nie należy rezygnować z baz relacyjnych SQL na rzecz grafowych, tylko w zależności od stojącego przed nami problemu dobrać odpowiednie narzędzie. Z pewnością znajdą się wśród nich i takie problemy, że zaistnieje konieczność użycia obydwu rodzajów baz danych. Podsumowując, warto to mocno podkreślić, bazy grafowe, której przedstawicielem jest Neo4j stanowi wartościowe narzędzie dydaktyczne i badawcze. Wspierane przez bazy grafowe myślenie grafowe i sieciowe, pozwala intuicyjnie modelować złożone zależności w prosty i czytelny dla człowieka sposób. Dzięki temu możliwe jest nauczanie alternatywnych paradygmatów modelowania danych jak i prowadzenie badań, w których kluczowe znaczenie stanowi analiza powiązań pomiędzy badanymi obiektami.

Bibliografia

ACID Databases – Atomicity, Consistency, Isolation & Durability Explained, 17.01.2024, https://www.freecodecamp.org/news/acid-databases-explained/ (30.12.2025).

Anuyah S., Bolade V., Agbaakin O., *Understanding Graph Databases: A Comprehensive Tutorial and Survey*, arXiv, 15.11.2024, http://arxiv.org/abs/2411.09999 (30.12.2025).

Codd E.F., *A relational model of data for large shared data banks*, «Communications of the ACM» 1970, t. 13, nr 6, doi: 10.1145/362384.362685.

Cypher – Neo4j Documentation, https://neo4j.com/docs/cypher/ (30.12.2025).

DB-Engines – Knowledge Base of Relational and NoSQL Database Management Systems, https://db-engines.com/en/ (30.12.2025).

DB-Engines Ranking Open Source vs. Commercial DBMS, https://db-engines.com/en/ranking_osvsc (30.12.2025).

DB-Engines Ranking per database model category, https://db-engines.com/en/ranking_categories (30.12.2025).

Graph database concepts – Getting Started, Neo4j Graph Data Platform, https://neo4j.com/docs/getting-started/appendix/graphdb-concepts/ (30.12.2025).

Moniruzzaman A.B.M., Hossain S.A., *NoSQL Database: New Era of Databases for Big data Analytics – Classification, Characteristics and Comparison*, arXiv, 30.06.2013, http://arxiv.org/abs/1307.0191 (30.12.2025).

Neo4j Graph Database & Analytics – The Leader in Graph Databases, 6.01.2026, https://neo4j.com/ (22.02.2026).

Silberschatz A., Korth H.F., Sudarshan S., *Database system concepts*, New York, NY 2011.

Tracz P.M., Plechawska-Wójcik M., *Comparative analysis of the performance of selected database management system*, «Journal of Computer Sciences Institute» 2024, t. 31, doi: 10.35784/jcsi.5927.

Ullman J.D., Meryk R., Widom J., *Podstawowy kurs systemów baz danych*, Gliwice 2011.

What is a graph database – Getting Started, https://neo4j.com/docs/getting-started/graph-database/ (30.12.2025).