AKADEMIA GÓRNICZO-HUTNICZA IM. STANISŁAWA STASZICA W KRAKOWIE
WYDZIAŁ ZARZĄDZANIA

KATEDRA INFORMATYKI STOSOWANEJ

PRACA DYPLOMOWA LICENCJACKA

Nierelacyjne bazy danych w analizie danych gospodarczych

Non-relational databases in the analysis of economic data

Autor: Małgorzata Maria Dyląg
Kierunek studiów: Informatyka i Ekonometria
Opiekun pracy: dr hab. inż. Andrzej Paliński

Kraków, 2020

OŚWIADCZENIE AUTORA PRACY DYPLOMOWEJ

„Oświadczam, że dokumentacja pracy dyplomowej nie narusza praw autorskich w rozumieniu ustawy z dnia 4 lutego 1994 roku o prawie autorskim i prawach pokrewnych (Dz. U. z 2006 r. Nr 90 poz. 631 z późniejszymi zmianami) oraz dóbr osobistych chronionych prawem cywilnym. Nie zawiera ona również danych i informacji, które uzyskałam w sposób niedozwolony. Wersja elektroniczna pracy dołączona przeze mnie na nośniku CD/DVD jest w pełni zgodna z wydrukiem pracy.

Zaświadczam także, że niniejsza praca dyplomowa nie była wcześniej podstawą żadnej innej urzędowej procedury związanej z nadawaniem dyplomów wyższej uczelni lub tytułów zawodowych."

…… ……

data podpis autora

OŚWIADCZENIE PROMOTORA PRACY

„Zatwierdzam do rejestracji i dopuszczam do obrony."

…… ……

data podpis autora

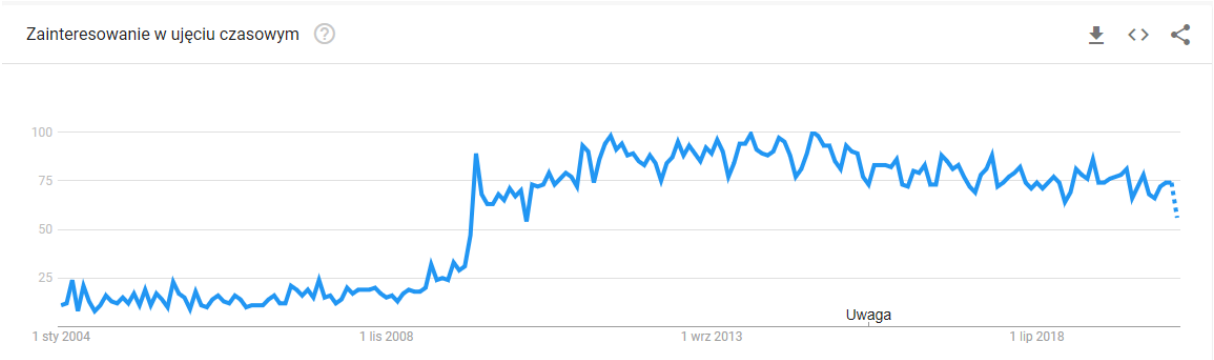
Spis Treści

Wstęp

Termin NoSQL (not only SQL) został po raz pierwszy użyty w dokumentacji w roku 1998 przez Carlo Strozziego.[1] Idea nierelacyjnych baz danych istniała jednak w świadomości programistów już wcześniej, kiedy prym na rynku wiodły ich starsze siostry – relacyjne bazy danych.

Rozwój nierelacyjnych baz danych, jaki obserwujemy przez ostatnie lata idzie w parze z niesamowitym rozwojem technologii Internetowych oraz Internet Of Things. Korzystając z narzędzia Google Trends zaobserwować możemy wzrost oraz utrzymującą się popularność wyszukiwania haseł związanych z nierelacyjnymi bazami danych. Na poniższych wykresach przykład dla ogólnego hasła „NoSQL" oraz dla jednego z systemów bazodanowych opartego na nierelacyjnym modelu danych „MongoDB".

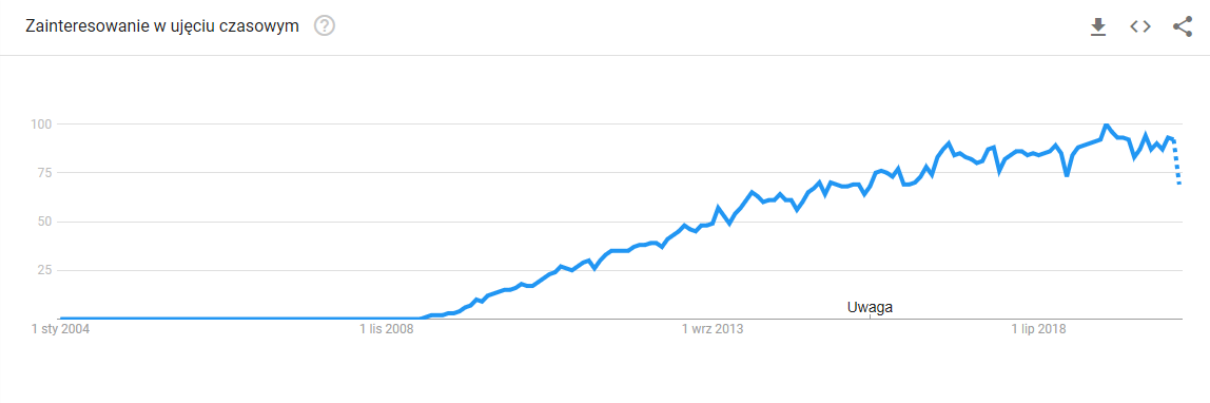
Tak nieoczekiwany nagły rozwój oraz wciąż pojawiające się nowe innowacje na rynku mogą powodować ogromy chaos informacyjny szczególnie dla osoby niewtajemniczonej w sam proces projektowania baz danych. Wybór odpowiedniego systemu bazodanowego jest niezwykle ważny na początku ścieżki rozwoju firmy.



Rysunek 1 Popularność wyszukiwania hasła "NoSQL" w latach 2004-2020

Źródło: https://trends.google.com/trends/explore?date=all&q=NoSQL

[1] NoSQL, https://pl.wikipedia.org/wiki/NoSQL, dostęp w dniu 04.08.2020



Rysunek 2 Popularność wyszukiwania hasła "MongoDB" w latach 2004-2020

Źródło: https://trends.google.com/trends/explore?date=all&q=MongoDB

Celem pracy jest opisanie poszczególnych systemów nierelacyjnych baz danych oraz opisanie aspektów pomocnych projektantom baz danych w dopasowaniu systemu do poszczególnych problemów rynku, a także wykonanie systemu bazodanowego wykorzystującego technologię NoSQL.

Praca oprócz wstępu składa się z trzech rozdziałów i zakończenia.

a. Pierwszy rozdział zawiera opis początków baz danych oraz szczegóły trzech rewolucji bazodanowych.
b. Rozdział drugi przybliża charakterystykę czterech głównych systemów nierelacyjnych baz danych, zawiera również porównania każdego z systemów z relacyjną bazą danych.
c. Rozdział trzeci zawiera analizę wykorzystania poszczególnych systemów NOSQL oraz projekt bazy nierelacyjnej.
d. Zakończenie zawiera podsumowanie pracy.

1. Historia i trzy rewolucje związane z bazami danych

Pojęcie bazy danych w świadomości społecznej pojawiło się w roku 1970, kiedy to Edgar F. Codd stworzył ramy teoretyczne dla modelu relacyjnej bazy danych.[2] Samo hasło w słowniku języka polskiego figuruje z następującym objaśnieniem: baza danych jest to „zbiór informacji na jakiś temat przechowywanych i przetwarzanych przez komputer".[3] Jednakże historia gromadzenia oraz gospodarowania danymi sięga swoimi początkami czasów o wiele odleglejszych. Mnisi, którzy w swoich zakonach w czasie średniowiecza nie byli świadomi, że zbiór woluminów, który skrupulatnie przepisują będzie mógł zostać nazwany w przyszłości „zbiorem danych". Rozwój pisma oraz druku stały się początkiem „produkcji" swoistych baz danych zawierających informacje na temat odkryć, sposobów pracy na roli czy zbiorów modlitw. Szczególną kategorią fizycznych zbiorów danych są słowniki oraz encyklopedie, biblioteki zaś określić można jako pierwsze systemy baz danych, powstałe na długo przed rewolucją przemysłową.

1.1 Pierwsza rewolucja bazodanowa

Pierwsza rewolucja zapoczątkowana została przez pojawienie się oraz rozpowszechnienie pierwszych modeli komputerów po zakończeniu drugiej wojny światowej. Dwa modele, których rozwój przypadał na lata pierwszej rewolucji bazodanowej to model hierarchiczny oraz model sieciowej bazy danych (Rysunek 3 oraz Rysunek 4).

Oba modele nazywano również „nawigacyjnymi", bowiem poruszanie się po bazie, czyli dotarcie do konkretnej informacji wymagało nawigowania pomiędzy informacjami po gałęziach drzew. Czyli w przypadku z Rysunku 4, aby dostać się do informacji

[2] S. Rozmus, Projektowanie baz danych z pełną historią zmian danych – model bitemporalnej bazy danych i operacje zapisu, „Biuletyn WAT", 2016, Vol. LXV, Nr. 1

[3] Słownik Języka Polskiego PWN, hasło: baza danych, https://sjp.pwn.pl/sjp/baza-danych;2451201.html, dostęp w dniu 28.07.2020

o zamówieniu przeszukiwanie należy zacząć od sklepu, następnie udać się do klienta, aby na końcu dostać informację o zamówieniu.

W modelu hierarchicznym dane zgromadzone są w postaci drzewa, budowa bazy przypomina budowę drzewa folderów na dysku twardym komputera, a informacje odczytywać możemy zarówno z danych jak i ze struktury drzewa. W przypadku modelu hierarchicznego tabela nadrzędna jest przodkiem dla tabeli podrzędnej, która jest potomkiem, dodatkową zasadą jest możliwość posiadania wielu potomków przez jednego przodka przy jednoczesnym zachowaniu zasady jedynie jednego przodka dla każdego z potomków.

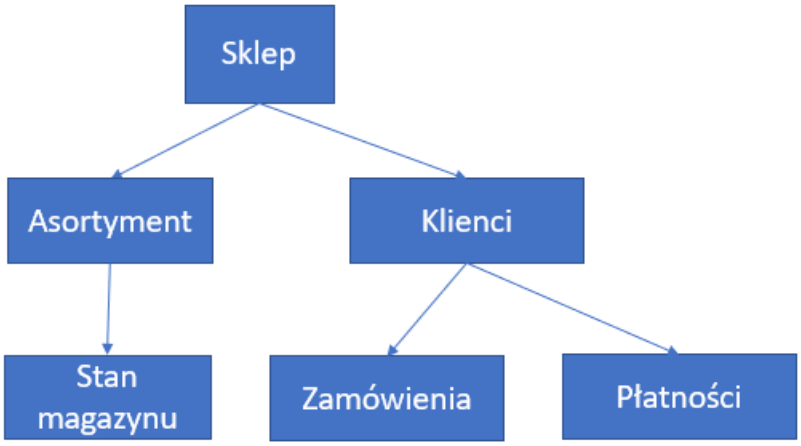
Główne ograniczenia modelu hierarchicznego wynikają głównie z zasady jeden potomek – jeden przodek. Przypadek, kiedy dwóch wspólników biznesowych (A i B) decyduje się na wspólny zakup sprzętu do swojego biura będzie bardzo trudna do obsłużenia za pomocą badanego modelu. Powstaną dwie bliźniacze pożyczki, jedna jako potomek Wspólnika A i jedna jako potomek Wspólnika B, duplikacja danych jest nieefektywna, każda zmiana w pożyczce będzie wymagała zmian w każdej „osobnej" pożyczce, co finalnie doprowadzić może do niespójności danych. Kolejną niepożądaną cechą modelu, w którym występować mogą duplikaty jest brak możliwości korzystania z prostych zapytań. I tak w przypadku pożyczek, nie będzie możliwym obliczeniem sumy długów zaciągniętych w konkretnym banku, bowiem do sumy zaliczać się będą nie tylko unikalne kwoty, ale także duplikaty, co będzie tworzyć niezgodność w analizie[4]. Remedium dla części tych problemów okazał się być model sieciowy.

W przypadku modelu sieciowego, nazwa pochodzi od połączeń między fragmentami danych, które tworzą sieć. Informacje pobierane są z danych oraz z budowy połączeń w sieci między konkretnymi dokumentami. Model sieciowy pozwala pozbyć się części ograniczeń wynikających z budowy modelu hierarchicznego – w jego przypadku jeden

[4] D. Sullivan NoSQL Przyjazny przewodnik, Helion, 2016, s.35

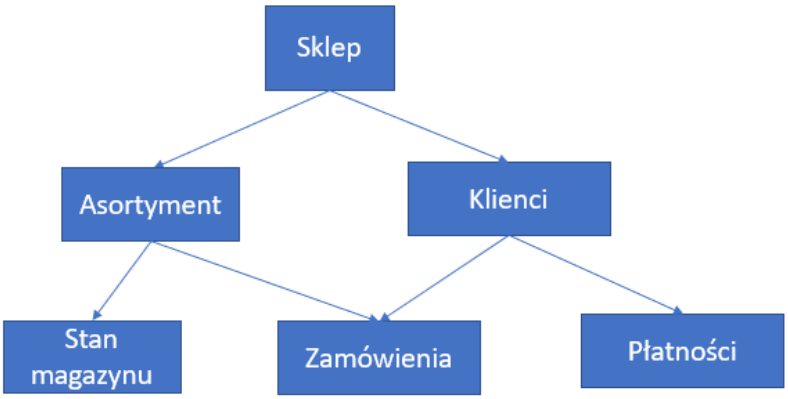
potomek mógł posiadać więcej przodków niż jedynie jednego jak było to w modelu hierarchicznym[5].

Modele sieciowe cechują się jednak bardzo dużą złożonością – mogą być trudne do projektowania, implementacji oraz późniejszego zarządzania. Konieczność nawigowania pomiędzy gałęziami powoduje wzrost czasu oczekiwania na operację. Im bardziej baza jest skomplikowana i im więcej poziomów danych zawiera, tym więcej ścieżek należy przejść by dostać się do szukanej informacji.



Rysunek 3 Hierarchiczny model danych

Źródło: Opracowanie własne



Rysunek 4 Sieciowy model danych

Źródło: Opracowanie własne

[5] D. Karpisz, Nie tylko relacyjny model danych, „Mechanika – czasopismo techniczne", Wydawnictwo Politechniki Krakowskiej, 2011, Zeszyt 7

Systemy hierarchiczne i sieciowe były używane w zdecydowanej większości aplikacji oraz komputerów do końca lat 70' XX wieku. Szybko stały się one jednak niewystarczające dla ciągle rozwijającego się świata technologii oraz coraz większych zasobów danych. Problemy, takie jak przymus przewidywania już w początkowych stopniach projektu zapytań, które wykorzystywane byłyby w późniejszych fazach działania bazy, a także utrudnione dodawanie nowych elementów do istniejącego już systemu zaczynały coraz bardziej przeszkadzać użytkownikom oraz projektantom baz[6].

1.2 Druga rewolucja bazodanowa

Wspomniany wcześniej dr Edgar F. Codd jest bez wątpienia głównym działaczem drugiej rewolucji bazodanowej. Ojciec relacyjnych baz danych przedstawił swój koncept w dziele pod tytułem „A Relational Model of Data for Large Shared Databanks" w 1970 roku.[7] Główną zaletą relacyjnego modelu są jego matematyczne oraz algebraiczne podwaliny.

Podstawową zasadą budowy relacyjnej bazy danych jest przechowywanie relacji za pomocą tabel składających się z krotek oraz atrybutów. Atrybut odpowiada wartości kolumny, a krotka wiersza. Każda krotka jest możliwa do zidentyfikowania za pomocą unikalnej wartości – klucza głównego (ang. Primary key), co stanowi dużą różnicę pomiędzy relacyjną bazą danych, a nawigacyjnymi modelami bazodanowymi, w przypadku których dostanie się do konkretnego rekordu wymagało znajomości struktury bazy. [8]

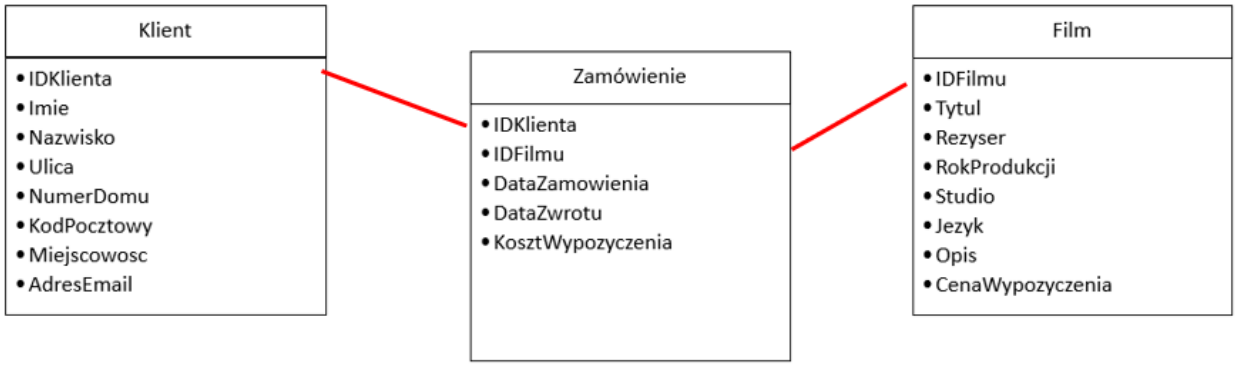
Poniżej rysunek prezentujący prosty relacyjny model danych, składający się z trzech tabel – Klienta, Zamówienia oraz Filmu. Kluczami głównymi w tabeli Klient oraz Film są odpowiednio ID Klienta oraz ID Filmu, tabela Zamówienie korzysta zaś z kluczy

[6] G. Harrison, NoSQL, New SQL I Big Data, Bazy danych następnej generacji, Helion 2019, s.22

[7] Edgar F. Codd, A Relational Model of Data for Large Shared Databanks, „Communications of the ACM", 1970, s. 377 – 387.

[8] M. J. Hernandez, Projektowanie baz danych dla każdego, przewodnik krok po kroku, Helion 2014, s. 43

obcych. Kolejne przedstawione są dwie tabele prezentujące relacje Pracowników pracujących w poszczególnych firmach. oraz tabele zawierające relacje.



Rysunek 5 Przykład relacyjnego modelu danych
Źródło: Opracowanie własne

Tabela 1 Relacja Pracownik

IdPracownika	ImiePracownika	NazwiskoPracownika	DataUrodzenia
1	Ambroży	Sas	20.04.1950
2	Michalina	Brożek	09.12.1995
3	Aleksy	Mikołaj	16.09.1976
4	Małgorzata	Nowak	04.10.1998

Źródło: Opracowanie własne

Tabela 1 Relacja Firma

IdFirmy	Nazwa	Branza	DataZalozenia
1	Farby i pędzle	remonty	02.08.1988
2	Mlecze i konwalie	ogrodnictwo	21.03.2000
3	Zakąska i przegryzka	gastronomia	24.12.2004

Źródło: Opracowanie własne

Tabela 2 Relacja Pracownik-Firma zawierająca związek wiele do wielu

IdPracownika	IdFirmy	StazPracy
1	1	5
1	3	7
2	1	3
3	2	10
4	2	1

Źródło: Opracowanie własne

Koncept relacyjnych baz danych nieprzerwanie króluje w branży od 40 lat. Biorąc pod uwagę stale rozwijające się technologie i mnogość nowych rozwiązań, którymi zasypuje rynek branża jest to niezwykle długi okres. Główne przyczyny stojące za popularnością relacyjnych baz danych:

- koncept rozwijany oraz użytkowany był przez dwa ogromne koncerny informatyczne: IBM oraz Oracle,
- rozwiązuje wiele problemów klientów – może używany być do budowy bardzo dużych systemów informatycznych
- zapytania, które obsługiwane są przez relacyjne bazy danych są proste i intuicyjne (w przeciwieństwie do późniejszych baz NoSQL, gdzie wymagana jest wyższa znajomość programowania)[9]

Jednakże, rozwój Internetu w ostatnich latach sprawił, że technologia napotkała nowy problem leżący w relacyjnych bazach danych. Sprawdzały się one dla klientów biznesowych, nawet w sytuacjach, gdy użytkownikami bazy było kilkanaście tysięcy pracowników danej firmy. Przetrzymywanie i korzystanie z zapytań stało się jednak niezwykle trudne dla firm zbierających dane o użytkownikach Internetu, których już nie można liczyć w tysiącach, tylko w milionach.

Główną potrzebą internautów był jak najkrótszy czas reakcji, jednak dla Google, który posiada w swoich bazach ogromną ilość danych korzystanie z relacyjnego modelu

[9] A. Wójcik, Nierelacyjne bazy danych, „Zeszyty Naukowe WSEI seria: Transport i Informatyka”, 4(1/2014), s. 83-96

bazy okazywało się niewystarczające. Z podobnym problemem mierzyły się inne ogromne korporacje, chociażby lider sprzedaży internetowej – Amazon[10], czy międzynarodowy serwis społecznościowy dla pracowników biznesowych LinkedIn[11]. Techniki oraz sposoby, które wystarczające były w przeszłości w momencie coraz większego rozwoju Internetu przestały być wystarczające.

Problemem stała się również ogromna liczba zapytań, które w jednej sekundzie baza musiała obsłużyć. Również sposoby w jakie radzono sobie z tymi problemami wcześniej nie spełniały warunków teraźniejszości, dokładanie większej ilości pamięci czy zmiana procesorów na lepsze było tylko chwilowym rozwiązaniem, który nie dość, że był ogromnie problematyczny to jeszcze kosztowny. Rozwiązaniem mogłoby być przeprojektowanie bazy, aby dopasować ją do rosnących wymagań, jednak skutkuje to bardzo często anomaliami danych, które są jednak nieakceptowalne z perspektywy analizy danych. Innym pomysłem na rozwiązanie tych problemów mogłoby być wykorzystanie wielu serwerów dla potrzeb jednej bazy danych, wprowadza to jednak kolejne niepotrzebne niedogodności. Nie dość, że sama operacja implementacji jednego relacyjnego modelu danych na wielu serwerach jest skomplikowana, to zarządzanie tak złożoną bazą również nie należy do najprostszych, a wykonywanie wielu operacji, wykorzystujących dane z kilku różnych serwerów nie jest wydajne. Z technicznego punktu widzenia, głównym zarzutem stawianym przeciwko relacyjnym bazom danych okazał się brak możliwości deklarowania typów złożonych. W przypadku zapisu adresu, który występować mógłby jako typ złożony, należy rozbić go na zestaw atrybutów i każdy analizować z osobna. Kolejnym był przymus tworzenia klucza sztucznego w sytuacjach, gdy atrybuty danych są niewystarczające do identyfikacji krotki[12]. Wszystkie te cechy doprowadziły w końcu do kolejnej innowacji w dziedzinie baz danych.

---

[10] J. Clement, Amazon - Statistics & Facts, Opublikowny: 03.02.2020, https://www.statista.com/topics/846/amazon/, dostęp: 29.07.2020

[11] A. Auradkar, Introducing Espresso - LinkedIn's hot new distributed document store, Opublikowany: 21.01.2015, https://engineering.linkedin.com/espresso/introducing-espresso-linkedins-hot-new-distributed-document-store, dostęp: 29.07.2020

[12] A. Wójcik, Nierelacyjne bazy danych, „Zeszyty Naukowe WSEI seria: Transport i Informatyka”, 4(1/2014),
s. 83-96

1.3 Trzecia rewolucja bazodanowa

W rynku bazodanowym wyszczególniono cztery główne problemy, a zarazem potrzeby, z którymi mierzą się relacyjne bazy danych. Są to: skalowalność, elastyczność, dostępność i koszt. [13]

- Skalowalność: zagadnienie odpowiada na pytanie: „Czy baza danych jest gotowa na zwiększające się obciążenie?". Rozwiązaniem dla tego problemu jest możliwość skalowania w pionie bądź w poziomie. Skalowanie w pionie polega na modyfikacji serwerów za pomocą dodawania do nich kolejnych kości pamięci czy lepszych procesorów – czyli wzrost bazy w obrębie jednej maszyny. Może zostać wykorzystane w przypadku, gdy baza nie będzie rosła w nieskończoność, bowiem staje się coraz droższa ze wzrostem ilości danych. Koszty dodatkowych komponentów rosną wykładniczo w miarę zwiększania wydajności bazy. Z kolei skalowanie poziome polega na dodawaniu kolejnych serwerów baz danych do sytemu – rozwiązanie to sprawdza się, gdy ilość rekordów jest bardzo duża. Jest to sposób bardziej elastyczny od skalowania pionowego, bowiem w przypadku, gdy tak duża ilość serwerów nie będzie już potrzebna jest możliwość wyłączenia ich. W przypadku baz NoSql gdy serwery są dodawane lub usuwane, automatycznie dostosowują bazę do istniejącego zbioru serwerów.
- Elastyczność: różnorodność wymagań rynku wymusza na bazach dopasowywanie się do indywidualnych potrzeb każdego z projektów. Projektanci baz danych podczas fazy projektowania muszą zaplanować jakie tabele powstaną w czasie zakładania bazy, jakie atrybuty będą posiadać i jakie relacje będą je łączyć. Rekordy w tabeli powinny również zawierać wartości dla wszystkich atrybutów, co jest jasnym rozwiązaniem dla większości branż korzystających z dobrodziejstw relacyjnych baz danych. Jednakże na przykład w przypadku aplikacji porównującej ze sobą filmy, książki i audycje radiowe, a także inne dzieła kultury, liczba atrybutów jest bardzo ciężka do określenia.

[13] G. Harrison, NoSQL, New SQL I Big Data, Bazy danych następnej generacji, Helion 2019, s.46

W przypadku książki będzie to między innymi autor, liczba stron, rok wydania, tłumacz, dla filmu reżyser, dźwiękowiec, wytwórnia czy liczba przyznanych Oscarów. Każdy z tych rodzajów dzieł będzie musiał występować albo jako osobna tabela albo jako jedna zbiorcza tabela z niezliczoną liczbą kolumn – co powoduje nieczytelność zbioru. Rozwiązaniem dla takich problemów są niektóre rodzaje baz NoSQL, które nie wymuszają na projektancie skonkretyzowania budowy i zawartości tabel w etapie projektowania.

- Dostępność: każdy z użytkowników Internetu przynajmniej raz natknął się na błąd związany z brakiem dostępu do strony internetowej z powodu błędu serwerów, częste prace systemowe na przykład w aplikacjach bankowych również mogą stawiać klientów tych banków w niewygodnych sytuacjach. W wygodnym świecie do jakiego przywykli internauci nie ma miejsca na podobne błędu. Bazy NoSQL zostały przystosowane do pracy na kilku serwerach – w przypadku awarii jednego z nich, jego zadania przejmują pozostałe. Serwery pomocnicze posiadają także back-up danych serwerów głównych, więc usterka nie doprowadzi do braków w bazie. Wpływa to oczywiście na wydajność pracy, czas oczekiwania na odpowiedź może być dłuższy, jednak serwis wciąż działa.[14]
- Koszt: kwoty, które zapłacić trzeba za licencję programów bazodanowych są bardzo wysokie, tym bardziej, gdy w jednej firmie użytkowników bazy jest bardzo dużo. Licencje często nadawane są na bazy o określonych parametrach – każda większa baza, czy większy serwer to kolejny ogromny koszt. Większość baz NoSQL oparta jest na licencji open-source, w przypadku której dostęp jest darmowy, a twórcy tych oprogramowani zapewniają stałe wsparcie dla użytkowników.[15]

Bazy NoSQL, których początek ekspansji na rynku przypada na początek XX wieku są odpowiedzią programistów na problemy projektantów baz danych wynikających z tych problematycznych obszarów.

[14] D. Sullivan NoSQL Przyjazny przewodnik, Helion, 2016, s.46-48

[15] Ibidem, s.46-48

Termin NoSQL jest hasłem zbiorczym dla wszystkich rozwiązań, w których zrezygnowano z używania modelu relacyjnego, nie jest to nazwa wyłącznie jednej, specyficznej technologii.[16] Rozwinięcie nazwy powoduje czasem pewne problemy. Oficjalnie skrót NOSQL oznacza Not Only SQL (z ang. Nie tylko SQL), a nie tak jak na pierwszy rzut oka może się wydawać – No SQL (Nie SQL). Wręcz przeciwnie, w niektórych rodzajach nierelacyjnych baz danych język bardzo podobny w składni i działaniu do SQL jest z powodzeniem używany.

Przez kilkanaście ostatnich lat, kiedy nierelacyjne bazy danych przeżywają ogromy wzrost popularności wykształciły się spośród nich cztery główne typy, których szczegółowe opisy zawarte są w rozdziale drugim tej pracy.

[16] A. Wójcik, Nierelacyjne bazy danych, „Zeszyty Naukowe WSEI seria: Transport i Informatyka", 4(1/2014), s. 83-96

2. Charakterystyka nierelacyjnych baz danych

Odpowiedzią na potrzeby inżynierów baz danych stało się wiele systemów, które wykorzystują różne modele danych. Jeden z podstawowych zaproponowanych podziałów wyszczególnia cztery główne typy rozwiązań:

- bazy danych klucz – wartość,
- bazy danych dokumentów,
- bazy danych rodziny kolumn,
- bazy z grafowym modelem danych.

Istnieją również modele łączące poszczególne zarówno technologie NOSQL jak i relacyjne modele danych, jednakże nie są one obiektem badań tej pracy.[17]

2.1 Bazy danych klucz-wartość

Jest to jeden z podstawowych i zarazem najprostszych nierelacyjnych modeli danych. Główne wymaganie stojące przed projektantem korzystającym z bazy danych klucz – wartość jest ustanowienie dla każdej wartości w bazie unikatowego klucza będącego jej identyfikatorem. Taka architektura kojarzyć może się ze znanymi z relacyjnych baz danych tabelami. Jednakże, w przypadku baz danych klucz – wartość nie mamy do czynienia z danymi w postaci tabel, klucz jest jedynym sposobem dotarcia do każdej z wartości. Dlatego bardzo ważnym krokiem w projektowaniu tego typu bazy danych jest zadbanie o odpowiednią konstrukcję klucza.

Poprawną konstrukcję klucza opisać można na przykładzie bazy danych używanej przez bibliotekę publiczną w podkrakowskiej gminie. Każdy czytelnik korzystający z biblioteki posiada elektroniczną kartę biblioteczną – posiada zatem swój numer klienta. Biblioteka zbiera także informacje personalne: imię, nazwisko, adres oraz ilość wypożyczonych książek. Najprostszym sposobem ustanowienia klucza byłoby numerowanie czytelników po kolei za pomocą liczb porządkowych.

[17] V. Abramova, Experimental evaluation of NoSQL Databases, „International Journal of Database Management Systems" Vol.6, No.3, June 2014A.

W takim wypadku dla pierwszego czytelnika otrzymano klucze: 1.numerKarty, 1.imieNazwisko, 1.adres.

Wiadomo jednak, że baza danych biblioteki oprócz danych o czytelnikach zbiera również dane chociażby o pozycjach, jakie znajdują się na jej półkach, czy o filiach biblioteki w innych miejscowościach. Korzystając z takiego systemu kluczy pierwsza filia biblioteki swój adres przechowywać będzie jako 1.adres – tak samo jak klient, co tworzy wyraźny problem. Dobrym rozwiązaniem w takich przypadkach będzie wykorzystanie do tworzenia klucza typu encji – i tak dla czytelnika dodano przed numerem prefiks „czytelnik", dla filii biblioteki „filia", a dla książki – „książka". [18] W ten sposób zapewniono spełnienie głównego kryterium przy tworzeniu kluczy – każdy z nich jest unikatowy w swojej przestrzeni nazw.

Wartości w bazach klucz – wartość mogą przyjmować różnorakie formy, począwszy od prostych słów – ciągów znaków, przez liczby, aż do obrazów czy nawet list – ten typ bazy NOSQL nie wymaga definiowania typów zmiennych pojawiających się w bazie. Wartości mogą mieć jednak ograniczenia pod względem rozmiarowym – problemy wydajnościowe są częstym problemem programistów korzystających z baz klucz – wartość.

Tabela 4 Przykład tworzenia kluczy dla bazy filii biblioteki

Klucz	Wartość
filia1.numer	1
filia1.adres	ul. Św. Jakuba 19, 32-091 Michałowice
filia2.numer	2
filia2.adres	ul. Długa 89, 32-091 Michałowice

Źródło: Opracowanie własne

[18] D. Sullivan, NoSQL Przyjazny przewodnik, Helion, 2016, s.69-71

Tabela 5 Przykład tworzenia kluczy dla bazy czytelników

Klucz	Wartość
czytelnik1.numerKarty	12009
czytelik1.imieINazwisko	Małgorzata Dyląg
czytelnik1.adres	ul. Główna 1, 32-091 Michałowice
czytelnik1.iloscWypozyczen	4
czytelnik2.numerKarty	12010
czytelnik2.imieINazwisko	Karol Nowak
czytelnik2.adres	ul. Koronkowa 13, 32-091 Michałowice

Źródło: Opracowanie własne

Porównanie baz klucz – wartość oraz baz relacyjnych

Główną różnicą pomiędzy systemami relacyjnymi oraz modelem klucz – wartość jest niewykorzystywanie danych w formacie tabel, dlatego też nie występują w bazach klucz – wartość kolumny. Nie spotka się tutaj również klucza obcego. Klucz obcy powstaje w bazach relacyjnych w momencie, gdy klucz główny wykorzystywany jest w tabeli innej niż ta, którą identyfikuje, jednakże w bazach klucz – wartość z powodu braku złączeń między tabelami taka instytucja nie istniej. Dla projektanta baz zaznajomionego z systemem relacyjnym konwencja nazywania kluczy w bazach klucz – wartość będzie znajoma. Tak jak w przywoływanym wcześniej przykładzie biblioteki – klucz czytelnik1.numerKarty w bazie relacyjnym znalazłby swój odpowiednik w tabeli o nazwie czytelnik, w kolumnie o nazwie numer karty oraz w wierszu, którego identyfikowałby unikalny klucz główny o wartości 1. [19]

Podstawowy model bazy klucz – wartość nie wspiera również żadnego języka zapytań, który mógłby być porównywany w swojej funkcjonalności do SQL wykorzystywanego w bazach relacyjnych. Istnieją, jednakże rozwiązania pozwalające

[19] D. Sullivan, NoSQL Przyjazny przewodnik, Helion, 2016, s.72

programistom na implementację potrzebnych funkcjonalności, które pozwalają na analizę czy pobieranie danych podobne jak SQL.

2.2 Bazy danych dokumentów

Bazy danych dokumentów są typem nierelacyjnych baz danych wykorzystującym dane w postaci dokumentów, najczęściej wykorzystywane formaty to XML (ang. Extensible Markup Language) oraz JSON (ang. JavaScript Object Notation), dużo rzadziej wykorzystywany format to BSON (ang. Binary JSON). Termin „dokument" nie oznacza w tym przypadku pliku tekstowego, a strukturę danych przechowywanych w postaci ciągów znaków bądź binarnej reprezentacji ciągów znakowych. [20] W dokumentach zawarte są zestawy kluczy i odpowiadających im wartości.

Występuje duże podobieństwo pomiędzy bazami danych dokumentów, a bazami klucz – wartość. W obu przypadkach zasady tworzenia kluczy, a także ograniczenia wartości są takie same, mogą być one reprezentowane zarówno przez liczby czy ciągi znaków jak i obiekty takie jak zdjęcia czy pliki audio. W bazach klucz – wartość możliwym jest także przechowywanie listy jako atrybutu, jednakże dostanie się do niego w tym modelu bazy może być wykonane jedynie za pomocą klucza. Programiści baz danych dokumentów mają jednak dostęp do bardziej nowoczesnych sposobów wyszukiwania na podstawie atrybutów. Możliwe jest zaimplementowanie interfejsów programistycznych bądź prostych języków zapytań. Dlatego baza danych dokumentów przez wielu uważana jest za złoty środek między bazami relacyjnymi, a bazami NOSQL – pozwalają na dużą elastyczność przy zarządzaniu bardziej złożonymi strukturami niż ma to miejsce w bazach klucz – wartość.

Format danych w zależności od wyboru pomiędzy XML a JSON może znacząco się różnić. XML był podstawą dla pierwszych baz dokumentów, co pozwala nazwać go prekursorem bardziej nowoczesnego typu – JSON. Format XML na początku XX wieku wykorzystywany był jako podstawowy format dokumentów,

[20] D. Sullivan, NoSQL Przyjazny przewodnik, Helion, 2016, s.73

nie tylko w standardzie Webowym, ale również dla dokumentów pochodzących z wszelakich edytorów teksów czy arkuszy kalkulacyjnych. Jednakże wraz z rozwojem technologii zauważono kilka znaczących wad standardu XML. Znaczniki, które wykorzystywane są w tym języku są bardzo rozbudowane, co stanowi problem przy przetwarzaniu danych – potrzebna jest bardzo duża moc obliczeniowa maszyny.[21] Bazy danych formatu JSON popularność zdobyły po 2008 roku i w tym momencie są wykorzystywane znacznie częściej niż format XML w bazach danych dokumentów. Jednakże, oba typy danych zaprojektowane zostały do bycia wykorzystywanymi w zupełnie różnych przypadkach. Systemy wykorzystujące XML wykorzystywane są głównie do obsługi treści – gdzie głównym zadaniem jest porządkowanie, zarządzanie i utrzymywanie zbiorów plików tekstowych, natomiast JSON najlepiej sprawdza się w aplikacjach internetowych. W takich aplikacjach niezwykle ważnym jest możliwość zarządzania treściami dynamicznymi.

Poniżej zaprezentowano porównanie dokumentów w formacie XML oraz JSON. Oba zapisy opisują książkę znajdującą się w bibliotece.

a. XML

```
<ksiazka>
        <idKsiazki> 1000 </idKsiazki>
        <tytul> "Harry Potter i Komnata Tajemnic" </tytul>
        <autor> "J. K. Rowling" </autor>
        <rokWydania> 1998 </rokWydania>
        <wydawnictwo> "Media Rodzina" </wydawnictwo>
        <gatunek> "fantastyka" </gatunek>
        <liczbaStron> 389 </liczbaStron>
</ksiazka>
```

[21] G. Harrison, NoSQL, New SQL I Big Data, Bazy danych następnej generacji, Helion 2019, s.70

b. JSON

```
{
    "idKsiazki": 1000,
    "tytul": "Harry Potter i Komnata Tajemnic",
    "autor": "J. K. Rowling",
    "rokWydania": 1998,
    "wydawnictwo": "Media Rodzina",
    "gatunek": "fantastyka",
    "liczbaStron": 389
}
```

Już w tak prostym przypadku zauważalna jest różnica w długości zapisu, która stanowi duży problem dla projektantów baz danych. Dla bazy, gdy podobnych dokumentów jest tysiące, a w samych dokumentach zagnieżdżone są kolejne dokumenty wykonywanie zapytań staje się dużo wolniejsze i bardziej wymagające dla maszyny.

Porównanie baz dokumentów oraz baz relacyjnych

Głównym podobieństwem pomiędzy bazami dokumentów oraz bazami relacyjnymi jest możliwość wykorzystania języka zapytań, jednakże te pierwsze oferują równocześnie większą swobodę programiście – poszczególne obiekty mogą różnić się między sobą atrybutami. Z tego wynika duża różnica pomiędzy tymi modelami danych – bazy dokumentów nie muszą być przedstawione za pomocą sztywnego, zdefiniowanego schematu, a ich struktura może być zmieniona dosłownie w każdej chwili. Dokumenty w nierelacyjnym typie danych mogą zawierać w sobie inne dokumenty, co stanowi pewne podobieństwo do relacji, które łączą tabele w modelu relacyjnym.

Na przykładzie dokumentu w formacie JSON możemy opisać poprzednie stanowiska pracownika podkrakowskiej biblioteki.[22]

[22] D. Sullivan, NoSQL Przyjazny przewodnik, Helion, 2016, s.73

```
{       „imie”: „Małgorzata”,
        „nazwisko”: „Dyląg”,
        „adresFilii”: „ul. Św. Jakuba 19, 32-091 Michałowice”,
        „stanowisko”: „Starszy Stażysta”,
        „dataZatrudnienia”: „13 czerwca 2018”,
        „adresEmail”: „malgorzata.dylag@gmail.com”
        „PoprzedniaFilia”: [
          {
                „adresFilii”: „ul. Św. Jakuba 19, 32-091 Michałowice”,
                „stanowisko”: „Stażysta”,
                „dataRozpoczecia”: „13 czerwca 2018”,
                „dataZakonczenia”: „29 września 2018”
          } ] }
```

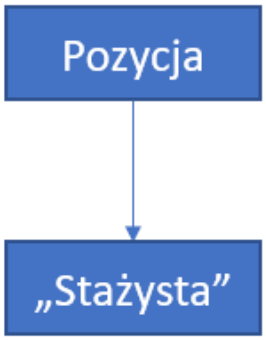
2.3 Bazy danych rodziny kolumn

Bazy danych dokumentów są jednym z trudniejszych do zrozumienia rodzajem nierelacyjnych baz danych. Głównym powodem jest terminologia, która w dużym stopniu przypomina tę, która wykorzystywana jest w relacyjnych bazach danych. Występuje tutaj zarówno kolumna, jak i wiersz, jednakże oba te terminy określają coś innego niż w bazach relacyjnych.

Kolumna to w bazach danych rodziny kolumn podstawowa jednostka w jakiej przechowywane są dane, odpowiada nazwie oraz wartości. Wiersz jest określeniem grupy kolumn. Rodzina kolumn to termin określający kolekcję powiązanych ze sobą kolumn. W przypadku biblioteki, informacje o autorze, tytule czy wydawnictwie są często wykorzystywane wspólnie, dlatego są przykładem prostej rodziny kolumn. Ten rodzaj baz danych przystosowany jest to przetrzymywania bardzo dużej ilości kolumn, dlatego bardzo często znajduje użytek w sektorze VLDB (ang. Very Large DataBase).[23]

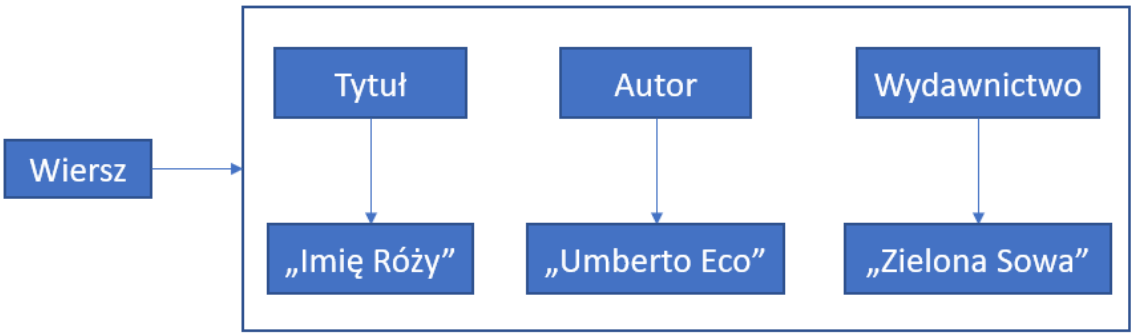
[23] A. Pasik, Związek bez relacji – nierelacyjne bazy danych (NoSQL). Część 2 – poszczególne typy nierelacyjnych baz danych, https://b2bnetwork.pl/zwiazek-bez-relacji-nierelacyjne-bazy-danych-cz-2/undefined, dostęp: 08.08.2020

Poniżej przedstawiony został przykład kolumny z określeniem stanowiska pracownika biblioteki oraz wiersza opisującego pozycję książkową w bibliotece.



Rysunek 6 Graficzne przedstawienie kolumny w kolumnowych bazach danych

Źródło: Opracowanie własne



Rysunek 7 Graficzne przedstawienie wiersza w kolumnowej bazie danych

Źródło: Opracowanie własne

Porównanie baz dokumentów oraz baz relacyjnych

Głównym wspomnianym już wcześniej w tej pracy podobieństwem pomiędzy bazami kolumnowymi, a bazami relacyjnymi jest wspólna nomenklatura. Jednakże zrozumienie terminów wiersz oraz kolumna są różne. W przypadku bazy dokumentów, podobnie jak w bazach relacyjnych występuje język zapytań przypominający język SQL. W bazach danych dokumentów nie korzysta się z tabel,

rozumianych jak w przypadku modeli relacyjnych. Bazy danych dokumentów nie mają sztywnego schematu encji, dlatego rodziny kolumn mogą różnić się pomiędzy wierszami. Wpływa to na znacznie większą elastyczność baz danych dokumentów w porównaniu
z relacyjnymi bazami danych. Operacja dodawania kolumny w bazie relacyjnej wymusza na programiście zmianę schematu systemu, gdzie w bazie rodziny kolumn jest to o wiele prostsza operacja. Tam, gdzie w relacyjnych bazach danych wykorzystana zostałaby operacja złączenia tabel, w bazie kolumnowej wszystkie dane o obiekcie znajdują się w jednym wierszu. [24] Bazy rodziny kolumn podobnie do baz relacyjnych używają kluczy indeksujących wiersze, które wykorzystywane są później do szybkiego wyszukiwania obiektów. W bazach relacyjnych są to klucze główne, w bazach rodziny kolumn zaś, klucze wierszy.

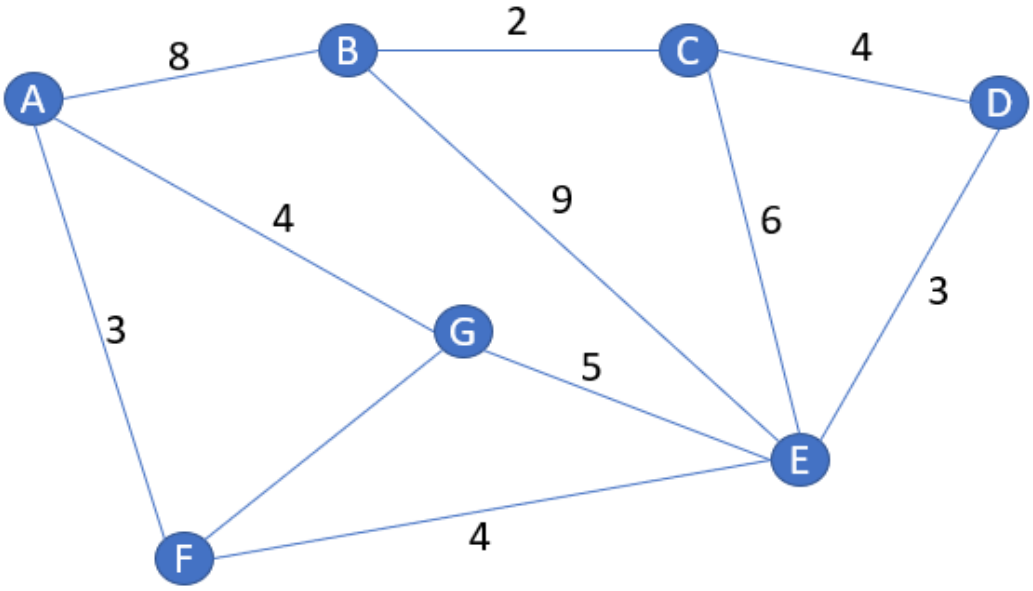
2.4 Grafowe bazy danych

Grafowe bazy danych są ostatnim z wyszczególnionych w tej pracy rodzajem baz. Są one najbardziej profesjonalne i wszechstronne. W budowie, zamiast znanych nam kolumn i wierszy wykorzystywane są tutaj węzły (inaczej wierzchołki) oraz relacje (inaczej krawędzie).

Warto wytłumaczyć najpierw czym jest graf. Jako podstawowa struktura matematyczna wykorzystywana w rozważaniach teorii grafów, służy ona do przedstawiania oraz badania relacji zachodzących między obiektami.[25] Zatem wierzchołki, które reprezentują obiekty mogą być przedstawieniem dosłownie każdego rodzaju przedmiotu, czy myśli abstrakcyjnej. Jednym z najbardziej popularnych zastosowań teorii grafów jest znany w metodach optymalizacji problem Komiwojażera, polegający na znalezieniu najkrótszej drogi pomiędzy dwoma punktami na mapie (grafie). Poniżej zaprezentowano przykładowy problem.

[24] D. Sullivan, NoSQL Przyjazny przewodnik, Helion, 2016, s.261

[25] Graf (matematyka), https://pl.wikipedia.org/wiki/Graf_(matematyka), dostęp: 10.08.2020



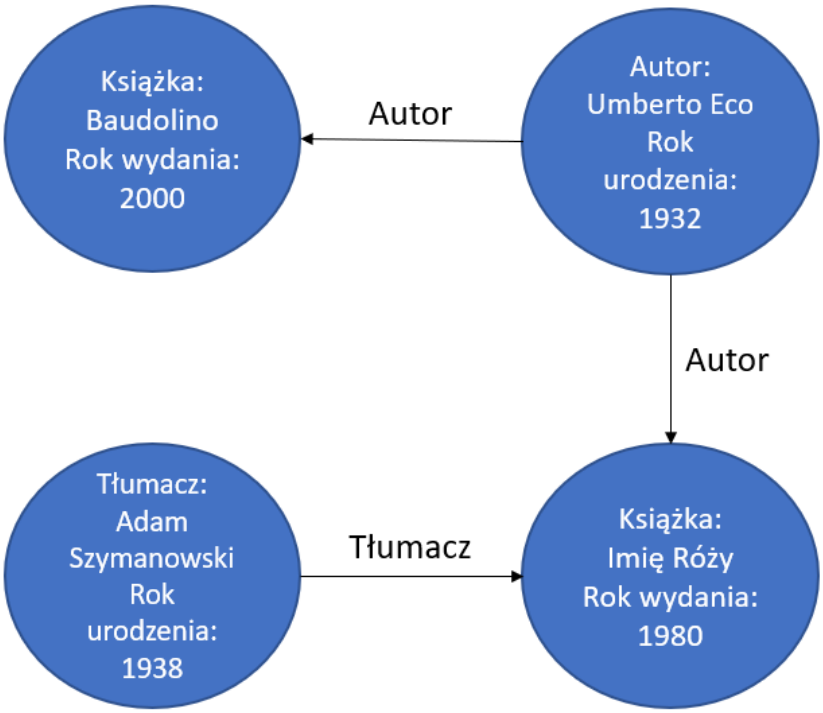
Rysunek 8 Przykładowy problem Komiwojażera dla 7 miast

Źródło: Opracowanie własne

Węzeł to obiekt, w którym wyszczególnić można identyfikator i zestaw atrybutów. Relacja jest obiektem łączącym dwa węzły, określa ona również atrybuty relacji. Dla dwóch miast, które reprezentowane są przez wierzchołki, relacją będzie trasa między nimi. W firmie pracowników łączą relacje w strukturze organizacyjnej, w rodzinie, relacją będzie wspólne pochodzenie. Niezwykła elastyczność jaką zapewniają grafy, pozwala przedstawiać w ten sposób zarówno połączenia niematerialne – tak jak te w przypadku rodziny, jak i materialne – tak jak drogi pomiędzy miastami. Drogi są również przykładem powiązań długoterminowych, w pracy zaś, powiązania będą krótkoterminowe, z uwagi na szczeble kariery, czy częste zmiany miejsca pracy przez zatrudnionych. [26]

Na grafie poniżej zaprezentowano prosty graf składający się z czterech węzłów oraz trzech relacji.

[26] D. Sullivan, NoSQL Przyjazny przewodnik, Helion, 2016, s.303



Rysunek 9 Graf przedstawiający prostą bazę danych opisującą książki Umberto Eco

Źródło: Opracowanie własne

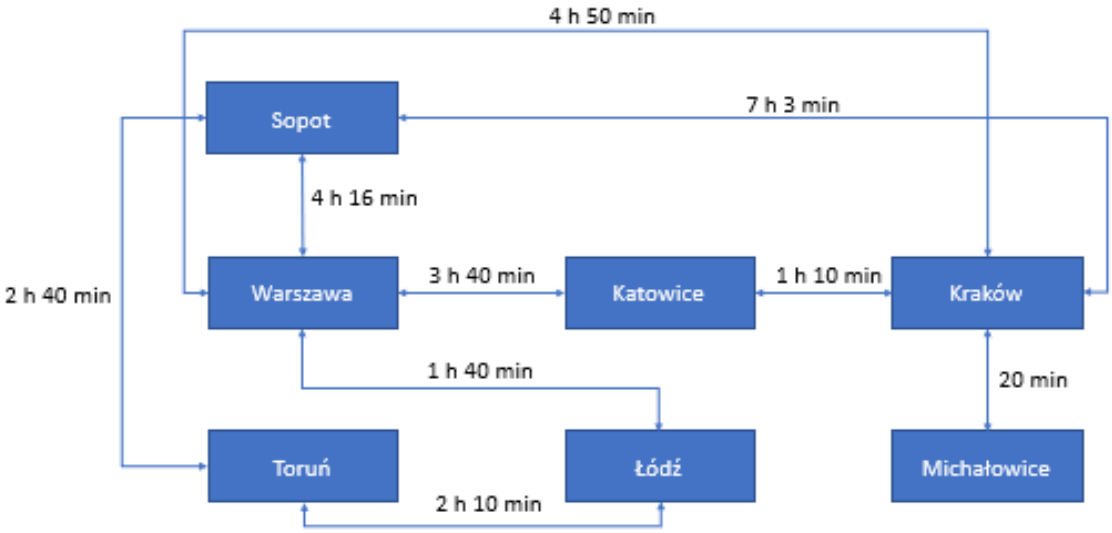
W obecnych czasach warto również wspomnieć o możliwości wykorzystania grafu jako graficzne przedstawienie ścieżki rozwoju choroby. Będzie to przykład grafu przedstawiającego relacje krótkoterminowe – każda interakcja międzyludzka, pomiędzy osobą zdrową a zakażoną będzie wpływała na zmianę stanu grafu. Każda z osób wchodząca w graf będzie posiadała swoje atrybuty – ich stan zdrowia, czy są zakażeni, czy zdrowi, możliwą wartością może być też bycie uodpornionym na wirusa. Ważne cechy to również wiek, waga czy posiadanie chorób współistniejących. [27]

[27] G. Harrison, NoSQL, New SQL I Big Data, Bazy danych następnej generacji, Helion 2019, s.80

Porównanie baz grafowych oraz baz relacyjnych

Pomiędzy relacyjnymi bazami danych oraz grafowymi bazami istnieje jedno duże podobieństwo. Relacje między obiektami, czyli więzami są równie ważne w obu typach baz. Modele relacyjne bardzo łatwo przedstawić za pomocą grafu, jednak nie są tak elastyczne jak grafowe bazy danych. [28]

Na podstawie przybliżonego w tym rozdziale wcześniej problemu Komiwojażera porównać można analizę poruszania się między miastami na podstawie grafu oraz tabeli.



Rysunek 10 Graf prezentujący połączenia drogowe między miastami polski

Źródło: Opracowanie własne

Tabela 3 Tabela przedstawiająca połączenia między miastami Polski

Miasto A	Miasto B	Czas podróży
Sopot	Kraków	7 h 3 min
Sopot	Warszawa	4 h 16 min
Sopot	Toruń	2 h 40 min
Warszawa	Katowice	3 h 40 min
Warszawa	Łódź	1 h 40 min

[28] G. Harrison, NoSQL, New SQL I Big Data, Bazy danych następnej generacji, Helion 2019, s.81

Warszawa	Kraków	4 h 50 min
Toruń	Łódź	2 h 10 min
Katowice	Kraków	1 h 10 min
Kraków	Michałowice	20 min

Źródło: Opracowanie własne (na podstawie https://maps.google.pl)

Analizując czas podróży między miastami na grafie znaleźć można najkrótszą trasę pomiędzy Sopotem a Michałowicami. Bardzo łatwo zauważyć można, że trasę można wykonać z jedynie jednym przystankiem. Z Sopotu wyruszyć można do Krakowa skąd już bardzo krótka droga dzieli podróżnika od celu. Na podstawie tabeli będącej zapisem relacji łączących miasta, wyszukanie tej trasy jest znacznie trudniejsze. Dlatego korzystanie z baz grafowych pozwala na znacznie efektywniejsze wyszukiwanie połączeń pomiędzy rozważanymi obiektami.

Różnorodność nierelacyjnych baz danych jest dużym plusem dla programistów i projektantów, daje im to możliwość dopasowania bazy pod swoje wymagania na wielu różnych płaszczyznach. Jednakże wybór jednego, najlepszego rodzaju dla wielu pozostaje trudnym zadaniem. W kolejnym rozdziale przybliżone zostały przypadku użycia każdej z baz danych oraz przykładowe oprogramowania pracujące na każdym z czterech głównych modeli opisanych w tym rozdziale.

3. Analiza wykorzystania poszczególnych systemów nierelacyjnych baz danych

Nierelacyjne bazy danych wraz z rozwojem technologii Internetu i zapotrzebowaniem na obróbkę danych nie tylko coraz większych rozmiarów, ale również coraz bardziej różnorodnych powoli zastępują bazy relacyjne. Trwającą rewolucję ciężko jednak porównać do sytuacji wejścia na rynek baz relacyjnych, które prawie całkowicie wyparły struktury używane przez programistów wcześniej, czyli opisane w rozdziale pierwszym struktury baz hierarchicznych i sieciowych. Projektant baz danych stojący przez zadaniem wyboru bazy danych odpowiedniej do implementowanego projektu, oprócz jednego z rodzajów, przybliżonych w rozdziale drugim może wybierać z pośród wielu oprogramowań.

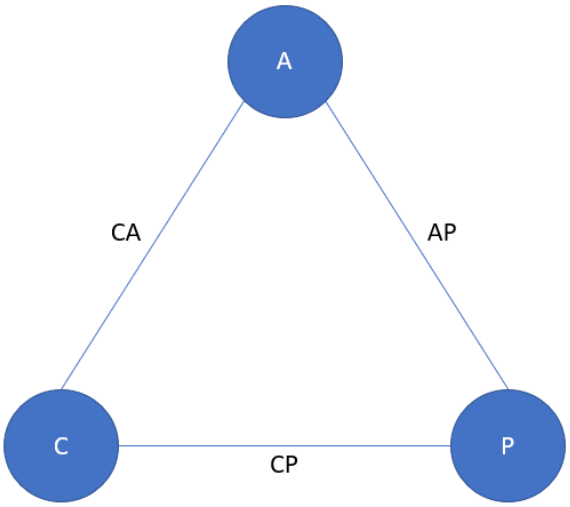
Jednym z kryteriów, które pozwoli wybrać programiście odpowiedni typ bazy danych może być trójkąt znany z teorii CAP.[29] Litera C w trójkącie pochodzi od angielskiego słowa Consistency, co oznacza spójność. Każdy z serwerów wykorzystywanych do działania bazy danych posiada jednakowe dane, każdy klient korzystający z bazy ma więc taki sam wgląd do danych. Litera A oznacza dostępność (ang. Availability) – oznacza to, że serwis dostępny jest ciągle i o każdej porze, klient korzystający z bazy ma gwarancję, że jego zapytanie zostanie rozwiązane pomyślnie niezależnie od tego, kiedy zostało wysłane. Odporność na partycjonowanie to litera P (ang. Partition Tolerance), oznacza to gwarancję, że baza po spartycjonowaniu również będzie działać. [30]

Twierdzenie CAP głosi, że rozproszona baza danych może posiadać jedynie dwie z trzech cech. Podział jaki występuje pomiędzy bazami, na grupę CA, CP oraz AP występuje niezależnie od modelu bazy. Na przykład bazy kolumnowe występują w każdej z trzech grup, dlatego dobór bazy na tym poziomie będzie wykorzystywał konkretnie oprogramowania. W ten sposób dla baz, które cechować powinny się CA – dostępnością

[29] Julian Browne, Brewer's CAP Theorem, January 2009, http://www.julianbrowne.com/article/brewers-cap-theorem, dostęp 10.08.2020

[30] Ibidem, dostęp 10.08.2020

oraz spójnością – wykorzystać można tradycyjne relacyjne bazy danych lub między innymi oprogramowanie Vertica, korzystające z modelu kolumnowego. Dla baz CP polecane oprogramowanie działające na modelu dokumentowym to MongoDB, dla modelu klucz – wartość jest to między innymi Redis lub Scalaris, a z baz kolumnowych BigTable, Hypertable lub HBase. Grupa AP, gdzie najbardziej rozwinięta jest dostępność oraz odporność na partycjonowanie zawiera w sobie oprogramowania takie jak Dynamo, Voldemort lub KAI – wszystkie wykorzystują model klucz – wartość, Cassandra, działająca na modelu kolumnowym lub bazy dokumentowe takie jak CouchDB lub SimpleDB.[31]



Rysunek 11 Trójkąt teorii CAP [32]

Źródło: Opracowanie własne, na podstawie: https://blog.nahurst.com/visual-guide-to-nosql-systems

Każdy z modeli baz nierelacyjnych sprawdzi się do nieco innych zastosowań rynkowych. Przeanalizować należy potrzeby, jakie dana baza ma spełniać oraz dobrać jak najlepszy do tego system. Poniżej przedstawiono praktyczne zastosowania każdego z nierelacyjnych systemów bazodanowych.

[31] N. Hurst, Visual Guide to NoSQL Systems, https://blog.nahurst.com/visual-guide-to-nosql-systems, dostęp 25.08.2020

[32] Ibidem, dostęp 25.08.2020

3.1 Projekty wykorzystujące model bazy danych klucz – wartość

Bazy danych klucz wartość przystosowane są do operowania na wielu różnych rodzajach danych, od typów prostych, takich jak liczby całkowite po typy strukturalne, takie jak listy czy dokumenty JSON. Ten typ baz dobrze sprawdzi się w przypadku, gdy zapis oraz odczyt wykonywane są często na niewielkich i mało skomplikowanych zbiorach danych. Oprogramowania korzystające z tego modelu posiadają najczęściej mechanizmy wspierające proste zapytania, dzięki którym poprzez klucz można dostać się do wartości.

Główne potrzeby, które spełnić może model bazy klucz - wartość to między innymi przechowywanie obrazów lub plików audio, przechowywanie i praca na kontach użytkowników w aplikacjach, na przykład mobilnych. Wykorzystany być może również jako wsparcie dla bazy relacyjnej – w takim przypadku baza klucz – wartość przechowywać może w pamięci podręcznej dane z bazy relacyjnej, aby poprawić jej szybkość działania i wydajność. Bazy te odnajdą swoje zastosowanie również do przechowywania danych o koszykach w sklepach internetowych. W przypadku ogromnych firm sprzedażowych, gdzie w sekundę pojawiają się miliony nowych koszyków i w konsekwencji zakupów. Dla bazy klucz – wartość możliwym jest zapis ogromnych ilości danych o bardzo wysokiej zmienności od wielu różnych użytkowników. W aplikacjach, które zbierają dane o użytkowniku i jego wyborach w trybie sesyjnym – od rozpoczęcia sesji, czyli zalogowania aż do zakończenia – wylogowania baza danych przechowywać będzie wszystkie informacje związane z daną sesją. Informacje te zapisane będą w pamięci lub w bazie danych. Każda z sesji jest unikalnym identyfikatorem. Dane związane z sesją mogą zawierać informacje o profilu, wiadomości, a nawet personalizowane reklamy czy promocje. [33]

[33] What is a Key-Value Database, https://aws.amazon.com/nosql/key-value/, dostęp: 11.08.2020

3.2 Projekty wykorzystujące model bazy danych dokumentów

Główną zaletą charakteryzującą bazy danych dokumentów jest ich elastyczność, będą więc odpowiednim wyborem dla przypadków, w których projektant potrzebuje posiadać możliwość przechowywania dużych ilości danych często ze zmiennymi atrybutami. W bazach dokumentów istnieje możliwość przechowywania dokumentów osadzonych, dlatego wszystkie informacje o danym obiekcie zwarte są w jednej lokalizacji, a nie tak jak w przypadku baz relacyjnych – w wielu różnych tabelach. Podobnie jak w bazach klucz – wartość istnieje tutaj możliwość korzystania z zapytań – dodatkowo programista może filtrować oraz indeksować dokumenty na podstawie jego atrybutów. Wiele baz danych dokumentów ma również możliwość zapisu danych w chmurze, dlatego rzadko występuje tutaj problem ze zbyt małą ilością miejsca na zapis informacji. [34]

Bazy danych dokumentów są wykorzystywane najczęściej ze wszystkich typów baz NoSQL. Często używane są jako wsparcie dla stron internetowych, dla których charakterystyczna jest duża ilość operacji zapisu oraz odczytu danych. Dzięki swojej elastyczności, sprawdzą się świetnie dla aplikacji korzystających ze zmiennych typów metadanych bądź do zarządzania informacjami o zmiennych atrybutach. Dane produktów w sklepach internetowych najczęściej posiadają różne atrybuty, zarządzanie tysiącami atrybutów w bazie relacyjnej jest nieintuicyjne i nieoptymalne, a czas zapisu lub odczytu niezwykle duży, również zmiana atrybutu w jednym produkcie nie wpłynie w żaden sposób na inne. Format zapisu pozwala wspierać wszelkie oprogramowania lub aplikacje korzystające z danych w formacie JSON lub BSON. Dokumentowa baza danych będzie odpowiednim wyborem również dla twórców internetowych, zarządzających treściami na blogach lub kanałach filmowych. Każda jednostka, która następnie będzie analizowana zapisana będzie jako jeden osobny dokument. [35]

[34] D. Sullivan, NoSQL Przyjazny przewodnik, Helion, 2016, s. 354

[35] What is a Document Database?, https://aws.amazon.com/nosql/document/, dostęp: 10.08.2020

3.3 Projekty wykorzystujące model bazy danych rodziny kolumn

Projektanci baz danych, dla których ważna jest możliwość zapisu dużej ilości danych, a także duża dostępność wraz z wysoką wydajnością operacji zapisu i odczytu danych powinni zdecydować się na bazę rodziny kolumn. Jeżeli planowane jest korzystanie z większej ilości serwerów, bazy rodziny kolumn wspierają takie formy modeli, dla przypadków, gdy wystarczający będzie jeden serwer, lepszym wyborem będzie baza klucz – wartość lub dokumentowa.

Aplikacje, w których potrzeba jest ciągłej aktualizacji danych, lub wykorzystujące pola dynamiczne najlepiej sprawdzą się działając na bazach rodziny kolumn. W przypadkach, gdy przewidywana jest niespójność w replikach rekordów, baza kolumnowa będzie najbardziej odporna oraz tolerancyjna. Dla aplikacji pracujących na ogromnych bazach danych, dla danych składających się z tysięcy terabajtów, na przykład sieci społecznościowych i ich skrzynki odbiorcze, czy wyszukiwarek również będą odpowiednim wyborem. Kolejnym z przykładów będą aplikacje analizujące rynki giełdowe, które charakteryzują się ogromną zmiennością wartości. Dla naukowców, badających dane genetyczne czy wytrzymałościowych – gdzie wymagane są możliwości przetwarzania bardzo dużej ilości danych. [36]

3.4 Projekty wykorzystujące grafowy model bazy danych

Bazy danych korzystające z grafów najlepszym wyborem będą dla przypadków, w których rozwiązywane problemy mogą zostać zaprezentowane jako obiekty połączone relacjami. Należy jednak ocenić, czy dane relacje pomiędzy obiektami są użyteczne. Dla przykładu w sklepie internetowym dwa różne zamówienia nie są ze sobą połączone – jedyną cechą wspólną między nimi może być użytkownik zamawiający produkty – jest to jednak jedynie wspólny atrybut, a nie konkretna relacja. Podobnie w przypadku gier komputerowych niekorzystających z form gier on – line, postęp gry jednego użytkownika nie wchodzi w żadną relację ze stanami gry innych

[36] D. Sullivan, NoSQL Przyjazny przewodnik, Helion, 2016, s. 355

graczy. Dla takich zastosowań lepiej sprawdzą się omówione wcześniej bazy klucz - wartość, dokumentów lub klasyczne bazy relacyjne.

Bazy danych są najlepszym wyborem, gdy problemy w aplikacji rozwiązywane mogą być dzięki modelowaniu relacji i wyznaczania ścieżek między nimi. Tak, jak opisano w rozdziale drugim – bazy grafowe sprawdzą się w przypadku wyznaczania optymalnych połączeń między miastami lub określania relacji między współpracownikami. Innymi przypadkami użytkowania baz grafowych może być zarządzanie infrastrukturą sieci, dostępami, a także analiza problemów i procesów biznesowych. Bazy te sprawdzą się również jako mechanizm wspierający dla sieci społecznościowych, jednakże w przypadkach, gdy wymagane jest przetwarzanie grafów na o wiele większą skalę w sieciach społecznościowych do przetwarzania lepiej wykorzystać bazę rodziny kolumn.

Grafowe bazy danych dają możliwość również wyspecjalizowanej ochrony przed fałszerstwami. Relacje wykorzystać można do procesowania transakcji pieniężnych i zakupowych w czasie rzeczywistym. Wyszukiwanie podejrzanych transakcji, wykorzystujących adresy mailowe lub numery kart, które zostały użyte w przeszłości do oszustw może zostać bardzo szybko przetworzone i zablokowane. Bazy te mogą być również pomocne do wyszukiwania wzorców związków takich jak wielu użytkowników skojarzonych z jednym adresem mailowym, czy korzystanie z jednego numeru IP przy różniących się adresach zamieszkania.

Aplikacje wykorzystywane do rekomendacji oraz wspierania wyborów również oparte mogą być na bazach grafowych. Połączenie pomiędzy zainteresowaniami klienta, znajomymi i historią zakupów mogą być przechowywane jako obiekt i relacje. Baza grafowa może polecać kolejne produkty do zakupu korzystając z listy poprzednich transakcji użytkownika lub list transakcji użytkowników zbliżonych do niego zainteresowaniami. Z aplikacji społecznościowych znany jest mechanizm polecania znajomych. W takich przypadkach baza grafowa wyszukuje użytkowników,

z którymi korzystający z aplikacji ma wspólnych znajomych oraz dodaje go do listy znajomych, których może on znać. [37]

3.5 Projekt nierelacyjnej bazy danych dla sklepu internetowego

Aby wykorzystać ideę nierelacyjnych baz danych w praktyce opracowano projekt bazy danych dla sklepu internetowego w którym klienci kupić mogą utwory kultury – płyty czy książki. Sklep internetowy będzie w przyszłości poszerzany o kolejne produkty. Dla klienta ważna jest zatem możliwość dopasowania bazy nie tylko pod teraźniejsze potrzeby, ale również pod zmiany, które w strukturze sklepu zajdą w przyszłości. Produkty sprzedawane w sklepie charakteryzują się różnorodnością cech – dlatego baza powinna wspierać dodawanie obiektów posiadających zmienną strukturę.

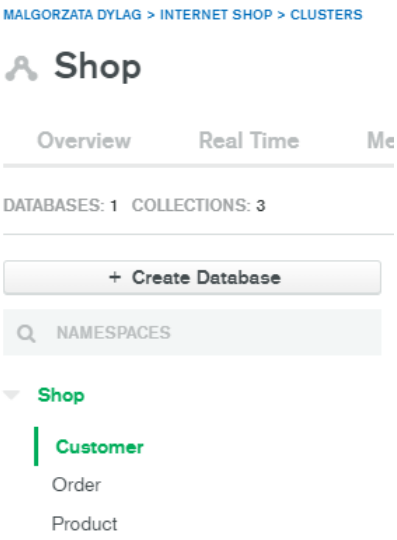
Na podstawie tych wymagań stwierdzono, że najlepszym wyborem dla klienta będzie skorzystanie z dokumentowej bazy danych, która nie tylko poradzi sobie z szerokim asortymentem sklepu, ale także zapewni wystarczającą elastyczność dopasowaną do planu rozwoju biznesu. Do wykonania projektu bazy wykorzystano oprogramowanie MongoDB Atlas, które działa na modelu dokumentowym oraz do zapisu danych korzysta z chmury. [38] W tym konkretnym projekcie skorzystano z wersji MongoDB 4.2, która pochodzi z 2019 roku.[39]

Na potrzeby klienta stworzono trzy kolekcje: Customer, Order oraz Product. Zgodnie z nazwami – kolekcja Customer zawiera listę klientów sklepu, Order jest listą zamówień, Product zaś to lista produktów.

[37] D. Sullivan, NoSQL Przyjazny przewodnik, Helion, 2016, s. 356

[38] https://cloud.mongodb.com/, dostęp: 26.08.2020

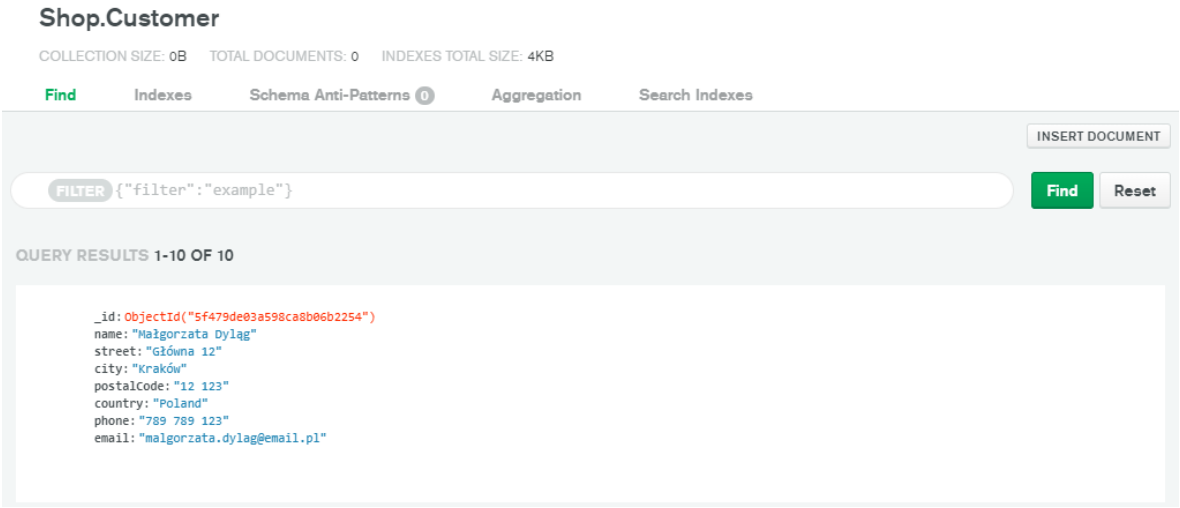
[39] https://docs.mongodb.com/manual/release-notes/4.2/, dostęp: 26.08.2020



Rysunek 12 Lista kolekcji wykorzystanych w projekcie bazy dla sklepu internetowego

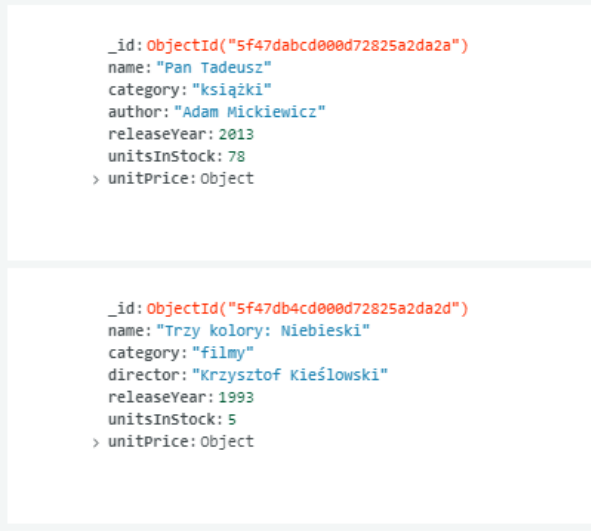
Źródło: Opracowanie własne przy użyciu https://cloud.mongodb.com/

Do kolekcji Customer dodane zostało 10 obiektów. Wszystkie składają się z informacji podawanych przez użytkowników przy rejestracji. Id obiektów jest ustalane automatycznie przez oprogramowanie. Informacje adresowe oraz kontaktowe wpisane są do systemu ręcznie.



Rysunek 13 Przykładowy obiekt wykorzystywany w kolekcji Customer

Źródło: Opracowanie własne przy użyciu https://cloud.mongodb.com/

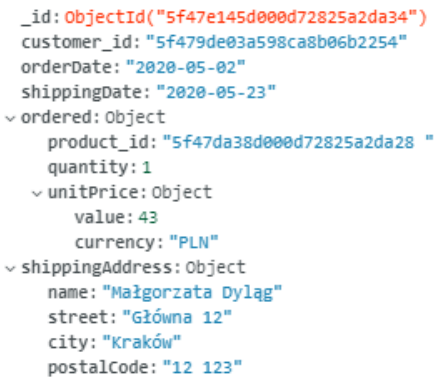


Rysunek 14 Przykładowe obiekty z kolekcji Products

Źródło: Opracowanie własne przy użyciu https://cloud.mongodb.com/

Kolekcja Product składa się z 10 obiektów, cztery z nich to książki, 3 to filmy na płytach DVD, a trzy to płyty muzyczne na nośnikach CD. W bazie relacyjnej na każdy z trzech rodzajów produktów zostałyby zaprojektowane osobne tabele, tutaj – wszystkie rekordy znajdują się w jednej kolekcji. Każdy z produktów posiada informację o kategorii, ilości dostępnych produktów oraz cenie wraz z walutą. Cechy zmienne takie jak reżyser, autor bądź wykonawca występują jedynie w odpowiadającej im kategorii produktów.

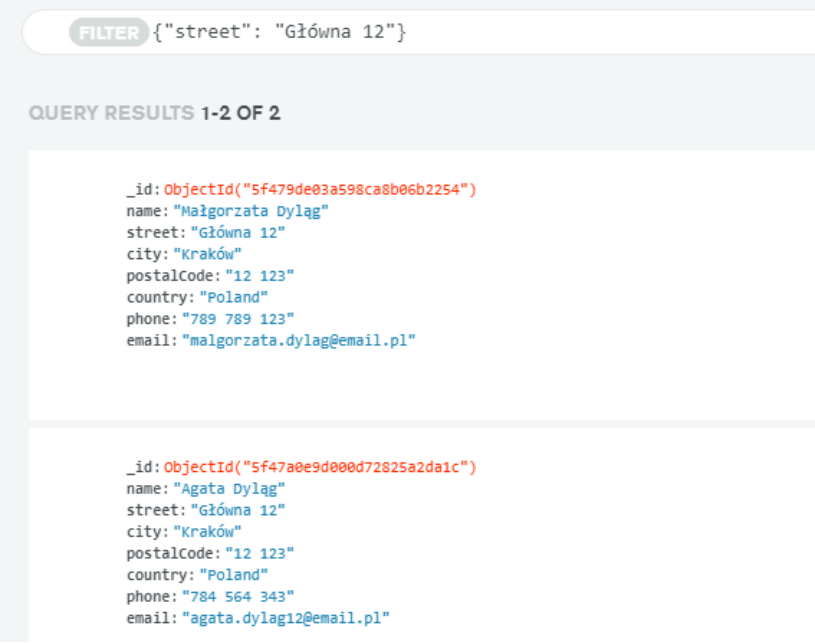
Do kolekcji Order dodano trzy przykłady zakupów wykonanych w sklepie. Cechami zamówień jest data złożenie zamówienia, data dostawy – dla zamówień, które nie zostały jeszcze zrealizowane wartość ta będzie pusta, adres dostawy, lista zamówionych produktów oraz cena. Adres zamówienia występuje dwa razy – w kolekcji Customer odnosi się on do adresu zamieszkania danego klienta, w kolekcji Order może być on oczywiście inny, jeżeli klient wyrazi chęć dostawy na dowolnie wybrany przez niego adres – na przykład do siedziby firmy, w której pracuje.



Rysunek 15 Przykładowy obiekt z kolekcji Order

Źródło: Opracowanie własne przy użyciu https://cloud.mongodb.com/

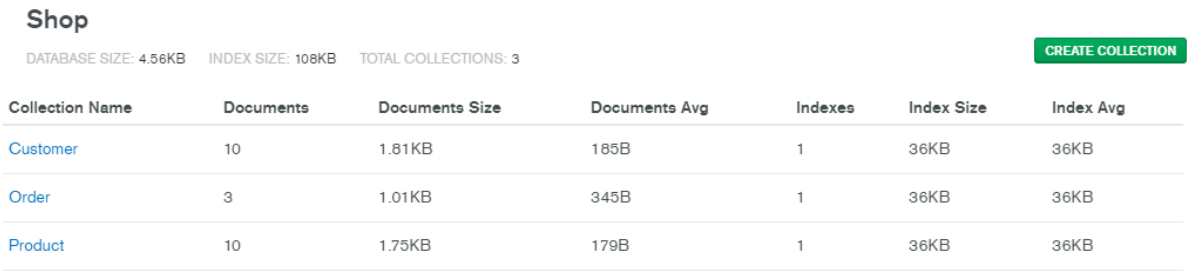
W oprogramowaniu MongoDB Atlas możliwe jest proste filtrowanie wyników. Dla przykładu, aby znaleźć w kolekcji Customer wszystkich klientów, którzy jako adres zamieszkania podali ulicę Główną 12 wykorzystamy funkcji: {„street": „Główna 12"}. Pierwsza wartość to cecha, której szukamy, druga jest jej wartością.



Rysunek 16 Wynik przeszukiwania kolekcji Customer za pomocą filtra ulicy

Źródło: Opracowanie własne przy użyciu https://cloud.mongodb.com/

Tak prosta baza danych zajmuje w chmurze 45,8 KB – w wersji w pełni darmowej MongoDB Atlas umożliwiają w swoich serwerach przetrzymywanie 512 MB danych. Uśredniając wagę dokumentów, z których korzysta przykładowa baza taka pojemność wystarczy na około 2,5 miliona podobnych dokumentów. Wydaje się być to dużą ilością, jednak dla dobrze prosperującego sklepu, będzie to stanowczo za mało – dla takich potrzeb istnieją jednak wersje płatne, które zapewniają również większa szybkość wykonywania zapytań.



Rysunek 17 Podstawowe statystyki bazy danych dla sklepu internetowego

Źródło: Opracowanie własne przy użyciu https://cloud.mongodb.com/

Powyższy przykład ukazuje prostotę z jaką działają nierelacyjne bazy danych dokumentów. Udowadnia to również, że przejście przez przedsiębiorcę na nowy rodzaj bazy nie będzie tak dużym wyzwaniem jak mogłoby się to wydawać. Uproszczenia w strukturze, biorąc pod uwagę fakt, ile tabel należało by stworzyć, jeżeli baza dla sklepu miałaby zostać stworzona w strukturach relacyjnych to tylko jedna z wielu zalet stojących za tym rozwiązaniem. Najtrudniejszym punktem wdrażania nowych technologii jest wybór modelu spośród opisanych w powyższych rozdziałach. Ilość dróg jakie wybrać może klient decydując się na skorzystanie z nierelacyjnych modeli danych jest niezliczona, a każdy kolejny pomysł na swój sposób wyjątkowy.

Zakończenie

Pomimo faktu, że rozwiązania NoSQL ewoluują coraz bardziej, dopasowując się do potrzeb rynku wciąż w porównaniu z bazami relacyjnymi liczba ich użytkowników jest znacznie mniejsza. Nadal większość projektantów baz wybiera bliższe im rozwiązania – czyli bazy relacyjne i wielu przypadkach wciąż są to rozwiązania optymalne. Nierelacyjne bazy danych są w tym momencie najlepszą możliwą opcją dla klientów pracujących z gospodarczymi danymi – pozwalają one na największą elastyczność i dopasowanie do konkretnych zastosowań.

Omówione przykłady użycia każdego z nierelacyjnych modeli baz danych ukazują jak wiele zastosowań dla takiego rozwiązania może być znalezione. Projekt bazy danych dla sklepu internetowego jest dobrym przykładem na ukazanie prostoty działania i elastyczności baz dokumentowych, które są najpopularniejszym rodzajem baz nierelacyjnych.

W czasach, kiedy projektantów baz danych ogranicza jedynie wyobraźnia, możemy spodziewać się rozwoju kolejnych rozwiązań i nowych technologii. Bardzo możliwe jest również, że któryś z czterech głównych typów nierelacyjnych baz danych przestanie być za kilka lat tak znaczący. Nie przewiduje się jednak popularyzacji jednego rodzaju nierelacyjnych baz danych do takiego stopnia jak było to w przypadku relacyjnego modelu bazy, który swój monopol na rynku posiadał przez kilkadziesiąt lat.

Bibliografia

Publikacje

[1] Abramova A.: Experimental evaluation of NoSQL Databases, „International Journal of Database Management Systems" Vol.6, No.3, June 2014A.

[2] Codd E. F.: A Relational Model of Data for Large Shared Databanks, „Communications of the ACM", 1970

[3] Harrison G.: NoSQL, New SQL I Big Data, Bazy danych następnej generacji, Helion 2019

[4] Hernandez M. J.: Projektowanie baz danych dla każdego, przewodnik krok po kroku, Helion 2014

[5] Karpisz D.: Nie tylko relacyjny model danych, „Mechanika – czasopismo techniczne", Wydawnictwo Politechniki Krakowskiej, 2011, Zeszyt 7

[6] Pasik A.: Związek bez relacji – nierelacyjne bazy danych (NoSQL). Część 2 – poszczególne typy nierelacyjnych baz danych, https://b2bnetwork.pl/zwiazek-bez-relacji-nierelacyjne-bazy-danych-cz-2/undefined, dostęp: 08.08.2020

[7] Rozmus S.: Projektowanie baz danych z pełną historią zmian danych – model bitemporalnej bazy danych i operacje zapisu, „Biuletyn WAT", 2016, Vol. LXV, Nr. 1

[8] Sullivan D.: NoSQL Przyjazny przewodnik, Helion, 2016

[9] Wójcik A.: Nierelacyjne bazy danych, „Zeszyty Naukowe WSEI seria: Transport i Informatyka", 4(1/2014)

Źródła internetowe

[1] Auradkar A.: Introducing Espresso - LinkedIn's hot new distributed document store, Opublikowany: 21.01.2015, https://engineering.linkedin.com/espresso/introducing-espresso-linkedins-hot-new-distributed-document-store, dostęp: 29.07.2020

[2] Clement J.: Amazon - Statistics & Facts, Opublikowany: 03.02.2020, https://www.statista.com/topics/846/amazon/, dostęp: 29.07.2020

[3] Graf (matematyka), https://pl.wikipedia.org/wiki/Graf_(matematyka), dostęp: 10.08.2020

[4] Hurst N.: Visual Guide to NoSQL Systems, https://blog.nahurst.com/visual-guide-to-nosql-systems, dostęp 25.08.2020

[5] NoSQL, https://pl.wikipedia.org/wiki/NoSQL, dostęp w dniu 04.08.2020

[6] Słownik Języka Polskiego PWN, hasło: baza danych, https://sjp.pwn.pl/sjp/baza-danych;2451201.html, dostęp w dniu 28.07.2020

[7] What is a Key-Value Database, https://aws.amazon.com/nosql/key-value/, dostęp: 11.08.2020

[8] What is a Document Database?, https://aws.amazon.com/nosql/document/, dostęp: 10.08.2020

[9] https://trends.google.com/trends/explore?date=all&q=MongoDB, dostęp: 24.08.2020 https://cloud.mongodb.com/, dostęp: 26.08.2020

[10] https://docs.mongodb.com/manual/release-notes/4.2/, dostęp: 26.08.2020

[11] https://cloud.mongodb.com/, dostęp 25.08.2020

Spis Tabel

Spis Rysunków: