WYDZIAŁ ELEKTRONIKI
I TECHNIK INFORMACYJNYCH

Rozproszone systemy operacyjne

Projekt – RSO NoSQL

Autorzy:

Tomasz Adamiec
Piotr Cebulski
Marek Kowalski
Mateusz Rosiewicz
Paweł Sokołowski
Marcin Wnuk

Warszawa, 2013

1. Wstęp ... 3
1.1 Potrzeba baz rozproszonych ... 3
1.2 Czym jest ruch NoSQL ? ... 3
1.3 Pierwszze projekty ... 3
2. Model relacyjny vs. model zorientowany na dokumenty ... 4
2.1 Model relacyjny ... 4
2.2 Model zorientowany na dokumenty ... 4
3. Ogólna koncepcja ... 5
3.1 Spójność ... 5
3.2 Partycjonowanie ... 5
4. Wnioski ... 6

1. Wstęp

1.1 Potrzeba baz rozproszonych

Bazy danych oparte na modelu relacyjnym zostały opracowane w latach 60/70 na założeniu, że jeden odpowiednio silny serwer zarządzany przez doświadczonego administratora jest w stanie zapewnić wydajne działanie systemu.

W związku z rosnącą ilością użytkowników sieci Internet można zaobserwować wzrost popytu na usługi realizowane przez rozmaite serwisy internetowe. Relacyjne bazy już w latach 90 zaczęły być dostosowywane do tego, by pracować na wielu komputerach, co spowodowało wzrost złożoności tych baz danych.

1.2 Czym jest ruch NoSQL ?

NoSQL (akronim od "Not Only SQL") jest alternatywnym podejściem do problemu bazy danych. wynika z poszukiwania alternatywy wobec tradycyjnych baz danych, zaś jej celem jest osiąganie wysokich przepustowości, ograniczenie kosztów ludzkich utrzymywania systemu i łatwa skalowalność. Jednym z założeń NoSQL jest częściowe poświęcenie integralności danych i rezygnacja z cech ACID na rzecz poprawy wydajności.

1.3 Pierwszze projekty

Ruch NoSQL jest szeroko dyskutowany od 2009, od kiedy Johan Oskarsson rozpoczął debatę na temat otwarto-źródłowych rozwiązań bazodanowych. Oskarsson jest współtwórcą serwisu Last.fm. Serwis ten pozwala na porównywanie gustów muzycznych swoich użytkowników. Informacje o słuchanej muzyce zbierane są od użytkowników za pomocą wtyczek (scrobbler) w odtwarzaczach audio (Winamp, Windows Media Player, Foobar itp.). Pojedyncza informacja zawiera tytuł utworu, wykonawcę i czas odtworzenia. Informacje te nie będą nigdy modyfikowane, ale jest ich bardzo dużo, ponieważ użytkownicy mogą mieć dziesiątki czy nawet setki tysięcy odsłuchanych kawałków. W przypadku tego serwisu niezbędne było wykorzystanie baz zorientowanych dokumentowo, gdyż wykonywanie złączeń przy takiej ilości rekordów byłoby zbyt kosztowne. Zamiast tego tworzy się dodatkowe dokumenty zliczające ilość odtworzeni kawałków danego artysty przez danego użytkownika, ilość odtworzeni kawałków danego artysty przez wszystkich użytkowników serwisu, ilość odtworzeni danego kawałka przez danego użytkownika itd.

W bazach nierelacyjnych został odkryty spory potencjał przez takie firmy jak Facebook (Cassandra), Google (BigTable), bądź Amazon (SimpleDB). Wykorzystywane są w wielu masowych serwisach internetowych operujących zbiorami danych sięgającym wartościom nawet rzędu petabajtów.

Jednym z głównych przedstawicieli NoSQL jest Apache Cassandra – silnik stworzony na potrzeby Facebooka. Twórcy tego serwisu położyli duży nacisk na niezawodność i na jego nieprzerwane działanie, dlatego Cassandra potrafi przetwarzać duże ilości danych rozmieszczone na wielu węzłach. Apache Cassandra jest projektem open source, w odróżnieniu od baz Google'a, czy Amazona.

Rozwinięciem bazy BigTable jest projekt Hypertable sponsorowany przez chińskiego potentata wyszukiwarkowego Baidu, Inc. Baza ta jest darmowa i współpracuje z systemami rozproszonymi Apache Hadoop DFS, GlusterFS i Kosmos File System (KFS).

SimpleDB jest bazą napisaną w języku funkcyjnym Erlang przez firmę Amazon.com. Amazon oferuje swoją bazę w formie usługi, którą można wykupić, a cena zależna jest od ilości transferu danych wychodzącego z bazy.

2. Model relacyjny vs. model zorientowany na dokumenty

Stosowanie baz NoSQL niesie ze sobą wiele korzyści, jednak deweloperzy przyzwyczajeni do baz relacyjnych będą zmuszeni do zmiany dotychczasowego podejścia. Przestawienie się na model zorientowany dokumentowo może z początku nastręczać problemów, gdyż wiąże się to z zerwaniem z praktykami, które dla bazodanowców są naturalne i weszły w nawyk. Główną motywacją do podjęcia takiego trudu jest potrzeba elastyczności w skalowaniu systemu i w modelu danych.

Technologia baz relacyjnych to technologia skalująca się w górę (*scale up*) – jeśli zachodzi potrzeba zwiększenia przepustowości lub pojemności, należy kupić większy serwer. Dzisiaj preferuje się skalowalność horyzontalną (*scale out*) – zamiast większego serwera, kupuje się więcej serwerów. Zmniejsza to koszty, gdyż umożliwia użycie sprzętu łatwo dostępnego (*commodity servers*) oraz korzystanie z usług serwerowych oferowanych w różnych innych formach. Skalowanie horyzontalne, które jest standardem na poziomie logiki biznesowej, wchodzi dopiero w dziedzinę baz danych.

Równie ważne co skalowalność horyzontalna, jest podejście do zarządzania danymi. Bazy relacyjne opierają się na schemacie. Aby dodać rekord do takiej bazy musi on być zgodny z tym schematem, narzucona jest z góry np. szerokość kolumn czy typ danych. Zmiana schematu w istniejącej bazie jest trudna, w szczególności gdy baza jest podzielona i znajduje się na wielu serwerach. Bazy NoSQL nie wymagają żadnego schematu, ani przy dodawaniu rekordów, ani przy dalszym zarządzaniu danymi, co daje im istotną przewagę w pewnych zastosowaniach.

2.1 Model relacyjny

Każdy rekord w bazie relacyjnej musi być zgodny ze schematem – musi mieć stałą liczbę kolumn, każda o określonym znaczeniu i typie danych. Każdy rekord jest taki sam. Jeśli chce się umieścić w bazie rekord innego typu, należy zmienić schemat. Dodatkowo, model relacyjny podlega normalizacji, co oznacza, że duże tabele są dzielone na mniejsze będące ze sobą w relacji.

Efekt normalizacji jest taki, że kawałki rekordów są porozrzucane pomiędzy tabelami, a rekordy z jednej tabeli mogą być wspólne dla wielu rekordów z innej tabeli. Zaletą tego jest oczywiście brak duplikacji danych. Wadą natomiast, jest blokowanie wielu tabel przy zmianie rekordu w jednej tabeli. Przetwarzanie transakcji z wykorzystaniem paradygmatu zawartego w ACID w bazach relacyjnych jest więc skomplikowane i jest przyczyną ograniczonej wydajności rozproszonych baz tego typu.

Teraz, gdy cena przestrzeni dyskowej jest niewielka, coraz rzadziej pojawia się realna potrzeba ograniczenia wielkości bazy. Zużycie większej ilości przestrzeni w zamian za lepszą wydajność i możliwość rozłożenia obciążenia pomiędzy maszyny jest lepszym rozwiązaniem przy tworzeniu wielu współczesnych aplikacji.

2.2 Model zorientowany na dokumenty

Dokument w bazach NoSQL oznacza rekord, który sam dostarcza informacji o sobie – o tym jakie pola się w nim znajdują i jakie są ich wartości. Przeważnie są to dokumenty w formacie XML, HTML lub JSON. Dane w tych dokumentach są zdenormalizowane. Dokument zawiera więc wszystkie właściwe dla niego informacje, nie odwołuje się do innych dokumentów. Nie ma problemów z zachowaniem spójności danych w obrębie pojedynczego rekordu, a więc także zachowanie poprawności transakcji jest o wiele prostsze. Wydajność zarówno modyfikacji, jak i odczytu rekordów jest w tym przypadku dużo wyższa niż w bazach relacyjnych.

Dokumenty w bazie NoSQL posiadają identyfikatory, które stanowią odpowiednik klucza głównego w bazie relacyjnej. Zazwyczaj identyfikator występuje w bazie tylko raz. Dane mogą być sortowane po identyfikatorze w taki sposób, aby dokumenty z podobnymi identyfikatorami były bliżej siebie. W ten sposób dane, dla których jest największe prawdopodobieństwo zażądania, będą dostępne w najkrótszym czasie.

Przykładem wykorzystania takiego podejścia, może być tworzenie oddzielnych dokumentów na komentarze pod artykułem. Komentarze nie powinny być umieszczone w tym samym dokumencie co artykuł, ponieważ jednoczesna edycja przez wielu komentatorów, byłaby utrudniona – blokowali by siebie nawzajem. W zamian, należy dla każdego komentarza utworzyć nowy dokument o identyfikatorze bliskim ID artykułu. Przykład ten pokazuje przeciwieństwo baz relacyjnych i NoSQL – w pierwszych wszystko podlega normalizacji, a w drugich odwrotnie.

3. Ogólna koncepcja

3.1 Spójność

Podstawową cechą systemu rozproszonego jest gwarancja spójności danych. Systemy NoSQL umożliwiają zachowanie spójności danych rozumianą w sensie Eventual Consistency - czyli takiej, że po upłynięciu pewnego skończonego czasu każdy węzeł systemu będzie miał dostęp do najbardziej aktualnej wersji danych.

Trudność tego zagadnienia polega na ustaleniu metody wersjonowania. Zcentralizowane bazy danych wykorzystują pojedynczy licznik, dzięki któremu wiadomo które dane są nowsze, a które nie. W zdecentralizowanych systemach każdy węzeł posiada licznik. Każdy dokument jest wersjonowany przez cały wektor, przechowujący informację o stanie wszystkich liczników w sieci. Wykorzystywane rozwiązanie opiera się na wykorzystaniu wektora liczników, w którym przechowuje się wartości licznika dla każdego węzła systemu.

3.2 Partycjonowanie

Partycjonowanie danych służy podziałowi danych pomiędzy maszyny, tak by podzielić obciążenie na węzłach w możliwie uczciwy sposób, tj. w zależności od mocy przerobowej każdego węzła.

Identyfikator każdego obiektu przepuszcza się przez funkcję hashującą, otrzymując wartość mieszczącą się w pewnym skończonym zakresie. Każdy węzeł sieci wybiera sobie pewien podzakres z tego zakresu, i następnie przechowuje wszystkie obiekty, których hash mieści się w tym podzakresie. Obciążenie węzłów może być dynamicznie zmieniane poprzez zmniejszanie bądź zwiększanie podzakresu danych danego węzła.

Zaleta tego rozwiązania polega na tym, że w celu odnalezienia obiektu nie potrzeba żadnego scentralizowanego rejestru, tylko wystarczy znać identyfikator żądanego obiektu, wiedzieć jakie podzakresy przechowują węzły w sieci i mieć dostęp do funkcji hashującej.

Rozwiązanie te przewiduje też możliwość replikacji danych w celu zabezpieczenia się przed awariami węzłów. Replikacja polega na tym, że podzakresy wybierane przez węzły sieci mogą się na siebie nakładać. Dzięki temu, każda informacja może znajdować się na wielu węzłach.

4. Wnioski

Rosnąca w ogromnym tempie ilość danych do przetworzenia, a także zmiany w charakterze świadczonych usług, zmuszają twórców oprogramowania do poszukiwania rozwiązań alternatywnych w stosunku do tradycyjnych baz danych. NoSQL daje tą alternatywę, rozwiązując kwestie skalowalności oraz usuwając problemy związane z dostosowywaniem bazy do nowych danych. Narzut potrzebnej pamięci spowodowany denormalizacją danych nie jest dzisiaj przeszkodą i stanowi niewielki koszt, w stosunku do zalet które z tego podejścia płyną. Duża wydajność, skalowalność i elastyczność są kluczowe w wielu bazach tworzonych na potrzeby współczesnych serwisów internetowych.

Bazy NoSQL nie mają jednak na celu zastąpienia baz relacyjnych. Bazy te prezentują odmienne podejścia co do normalizacji danych niż bazy relacyjne, ale należy pamiętać, że w praktyce właściwy model danych zależny jest od przypadku ich użycia. NoSQL stanowi więc dodatkowe narzędzie, na którego użycie mogą zdecydować się projektanci baz danych.