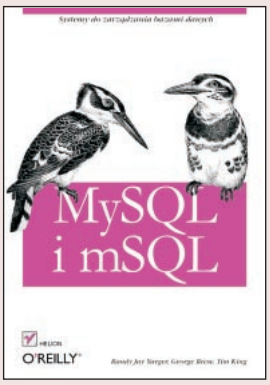
MySQL i mSQL



Autorzy: Randy Jay Yarger, George Reese, Tim King
Tłumaczenie: Łukasz Felsztukier, Aleksandra Jakubowska
ISBN: 83-7197-149-4
Tytuł oryginału: MySQL and mSQL
Format: B5, stron: 448

W przeciwieństwie do komercyjnych produktów, MySQL oraz mSQL są tanie i łatwe w użytkowaniu. Jeśli znasz podstawy C, Javy, Perla lub Pythona – możesz szybko pisać programy współdziałające z Twoją bazą. Co więcej, możesz osadzać zapytania bezpośrednio w pliku HTML tak, że strona www staje się jej własnym interfejsem do bazy danych.
Książka ta zawiera wszystko, czego potrzebujesz aby wykorzystać MySQL oraz mSQL. Przeprowadza Cię przez cały proces, od instalacji i konfiguracji po interfejsy programowania i podstawową administrację. Zawiera rozdziały z referencjami oraz pokaźny materiał szkoleniowy.

Wydawnictwo Helion
ul. Chopina 6
44-100 Gliwice
tel. (32)230-98-63
e-mail: helion@helion.pl

Spis treści

7

Pozostałe bazy danych klasy średniej

Kiedy MySQL pojawił się na rynku, był pierwszym serwerem klasy średniej obsługującym standard SQL. Nie cieszył się tym zaszczytem długo. Wszyscy dobrze wiemy dlaczego — narodził się MySQL. Od momentu powstania mSQL-a, na rynku pojawiło się kilka innych produktów. W tej książce skupiliśmy się na mSQL-u oraz MySQL-u z uwagi na ogromne podobieństwa oraz niezrównaną popularność. Jednak nie możemy nie wspomnieć o innych serwerach baz danych.

Baz danych używa się w tak wielu dziedzinach, że trudno jest w jednym pakiecie zmieścić wszystkie narzędzia do wszelkich możliwych zastosowań. Próbują tego dokonać producenci większych baz danych. Płacą za to zmniejszeniem wydajności, a ich odbiorcy zwiększeniem wydatków. Z drugiej strony serwery baz danych niższej klasy osiągnęły taką specjalizację, że nie nadają się dla małych firm lub niedochodowych organizacji czy dla kogoś o bardziej nieszablonowych wymaganiach. Serwery klasy średniej wypełniają lukę pomiędzy tymi dwoma skrajnymi grupami.

Dotychczas rozważaliśmy tylko dwa bardzo zbliżone sposoby na to, jak zaspokoić średnio zaawansowane wymagania bazodanowe. Bez wątpienia możliwe są też inne. Żadne prawo nie stwierdza, że ponieważ firma jest niewielka, więc nie będzie potrzebować przetwarzania transakcyjnego. Niektórzy użytkownicy w średnim sektorze mogą również potrzebować wyzwalaczy, zagnieżdżonych zapytań, procedur składowanych, obiektowości czy którejkolwiek z wielu możliwych własności — tyle że nie wszystkich naraz. Inne serwery bazodanowe średniej klasy mogą więc obsługiwać import w sposób niedostępny w MySQL-u czy w mSQL-u.

Co oznacza „bezpłatny”?

Słyszy się czasem, że MySQL i mSQL są „bezpłatne”. Można także spotkać się z opinią, że MySQL jest „bardziej bezpłatny” niż mSQL. Jakkolwiek zwrot „bardziej bezpłatny” kłóci się ze zdrowym rozsądkiem, to właśnie w świecie oprogramowania wynaleziono „stopnie bezpłatności”.

Do tej pory świadomie unikaliśmy mówienia o MySQL-u i mSQL-u jako o „bezpłatnych” motorach bazodanowych z uwagi na wielość znaczeń, którą termin „bezpłatny” ma dla wytwórców opro-

gramowania. Oba produkty mogą w rzeczywistości wymagać opłaty licencyjnej. Opłata zależy od tego, kim jest użytkownik. Według licencji dostępnych w trakcie drukowania tej książki uczelnie nie musiały wnosić opłat licencyjnych za żaden z dwóch motorów bazodanowych. Komercyjny użytkownik mSQL-a musiał natomiast za licencję zapłacić. Stwierdzenie, że MySQL jest bardziej bezpłatny od mSQL-a oznacza, iż jest on bezpłatny dla większej liczby osób.

Druga sprawa dotycząca pojęcia bezpłatnego oprogramowania nie ma nic wspólnego z ceną. Chodzi o możliwość zobaczenia i zmiany kodu źródłowego bez żadnej dodatkowej opłaty. W tym świetle zarówno MySQL, jak i mSQL są zupełnie darmowymi serwerami baz danych. Wystarczy odwiedzić ich strony internetowe, aby uzyskać kod źródłowy. Darmowy dostęp do postaci źródłowej mają również ci użytkownicy mSQL-a i MySQL-a, którzy muszą płacić za jego użytkowanie.

Świat oprogramowania wprowadził nowy termin, która pozwala uniknąć wieloznaczności słowa „bezpłatny". Chodzi o Open Source. Faktycznie Open Source to obecnie znak towarowy, oznaczający oprogramowanie, którego kod źródłowy jest otwarty bez względu na opłaty związane z użytkowaniem tego oprogramowania. Linux, Netscape, FreeBSD, Perl, Apache, wszystkie produktu z rodziny GNU, jak również wiele innych produktów wspomnianych w tej książce (np. MySQL, mSQL, mm.mysql.jdbc czy mSQL-JDBC) są produktami Open Source.

Wszystkie produkty, o których mowa w tym rozdziale, również są produktami Open Source. Otwarty kod źródłowy jest bardzo istotny dla średniego sektora rynku, ponieważ gigantom wydaje się, że jest on za mały, by zasługiwać na uwagę, a wytwórcom w niższym sektorze, że jest on zbyt skomplikowany.

Czego brak w MySQL-u oraz mSQL-u?

Piszemy o braku, bo nie potrafimy znaleźć lepszego słowa. Jak wspomnieliśmy, twórcy mSQL-a oraz MySQL-a świadomie wykluczyli kilka cech, które miałyby negatywny wpływ na wydajność. Innymi słowy priorytetem w tworzeniu oraz rozwijaniu mSQL-a i MySQL-a była wydajność. Niektórzy użytkownicy serwerów klasy średniej gotowi są jednak poświęcić wydajność w zamian za inne własności. W celu pełniejszego zrozumienia cech udostępnianych przez pozostałe produkty klasy średniej warto wymienić rzeczy, które w MySQL-u i mSQL-u pominięto.

MySQL ma wprowadzić wiele tych cech z możliwością wyłączenia ich, gdy komuś nie będą potrzebne. W trakcie powstawania książki Monty rozważał wprowadzenie mechanizmu procedur składowanych, zagnieżdżonych zapytań, a być może nawet transakcji.

Transakcje

Transakcje umożliwiają grupowanie wielu zapytań SQL jako jednostkę. Grupowanie poleceń daje pewność, że nikt niepowołany nie zobaczy częściowo zmienionej zawartości bazy danych. Poza tym w przypadku, gdy jedno z zapytań zawiedzie, zawiedzie również cała jednostka pracy. Sposobem na zrozumienie transakcji jest wyobrażenie sobie zatłoczonej drogi. Jednowątkowy system kolejkowania, np. mSQL, to odpowiednik równorzędnego skrzyżowania. Samochody jadą jeden za drugim według reguły pierwszeństwa. Jeśli dwa pojazdy jadą tą sama drogą, istnieje prawdopodobieństwo, że się rozdzielą na skrzyżowaniu.

Wielowątkowy system blokowania, występujący np. w MySQL-u, przypomina bardziej sytuację, kiedy ruch na skrzyżowaniu nie jest regulowany przez sygnalizację świetlną lecz przez policjanta. Ruch na takim skrzyżowaniu może przebiegać w dowolnym kierunku, przy dowolnej prędkości, pod warunkiem, że policjant inteligentnie kieruje ruchem. Jeśli w tym samym momencie nadjadą dwa pojazdy z różnych stron, policjant przepuści jeden, poczeka, aż opuści on skrzyżowanie, a następnie pozwoli przejechać drugiemu.

Transakcje przypominają bardziej skrzyżowanie ze światłami. Samochody, które mają czerwone światło, zatrzymują się, by przepuścić te, które mają zielone. Praktycznym przykładem zastosowania transakcji jest aplikacja bankowa, w której transfer z rachunku oszczędnościowego na rachunek bieżący („czekowy") wiąże się ze zmianą salda na rachunku oszczędnościowym, a następnie
— zmianą salda na rachunku bieżącym. Poniższe dwa zapytania SQL mogą być wykorzystywane w takiej aplikacji:

```
#Odejmij 1000 PLN od 1100 PLN na rachunku oszczędnościowym.
UPDATE konto
SET saldo = 100.00
WHERE id = 1234
#Dodaj 1000 PLN do 550 PLN na rachunku bieżącym.
UPDATE konto
SET saldo = 1550.00
WHERE id = 5678
```

Pomiędzy dwoma uaktualnieniami mogłaby zdarzyć się transakcja, w której inny klient sprawdzałby saldo na rachunku bieżącym i oszczędnościowym, by ustalić, czy jest dość pieniędzy, żeby wystawić czek. Jeśli ta transakcja miałaby miejsce, czek byłby bez pokrycia. Co gorsza, jeśliby serwer padł pomiędzy uaktualnieniami, klient „utopiłby" 1000 złotych w pliku zrzutów z pamięci.

Ujmując te dwa zapytania w jednej transakcji, stwierdzamy, że oba zarazem muszą kończyć się bądź sukcesem, bądź porażką. Jeśli jedno z nich się powiedzie, a drugie zawiedzie, można wydać polecenie tzw. zwinięcia transakcji *(rollback*), które przywraca bazę do jej stanu sprzed transakcji. Podobnie nikt inny nie może uzyskać dostępu do zmienianych plików, dopóki nie zakończymy swoich zmian.[1] MySQL pozwala częściowo emulować transakcje za pomocą zamków LOCK TABLES. Zamki mogą skutecznie zapobiec uszkodzeniom spójności danych, ale nie pozwalają na zwijanie operacji. mSQL w żaden sposób nie wspiera transakcji.

Wyzwalacze

Wyzwalacze są blisko związane z transakcjami. Prowadząc analogię do ruchu ulicznego o krok dalej, wyobraźmy sobie policjanta obserwującego skrzyżowanie ze wzgórza. Gdy tylko któryś z samochodów złamie przepisy, policjant włącza się do ruchu i rozpoczyna pościg.

Wyzwalacz to jedno lub więcej zapytań SQL przechowanych w bazie danych, które są wykonywane po zajściu pewnych wcześniej określonych zdarzeń. Wyzwalacze są metodą automatyzacji nadzoru. W przypadku kiedy pewien warunek jest spełniony, wyzwalacze mogą wykonać działania na danych lub donieść po prostu o zajściu wyzwalającego zdarzenia.

[1] Własność ta ma specjalny niuans zwany wartością „izolacji transakcji". Czasem nie dbamy o to, czy użytkownicy mają dostęp tylko do odczytu do niespójnych danych. Pozwalając na to, przyspieszamy bazę danych, nie zmuszając ich do oczekiwania na zakończenie transakcji.

Procedury składowane

Najprostsze procedury składowane to jedno lub kilka zapytań SQL przechowywanych w bazie danych pod jakąś prostą nazwą, służących utrwaleniu jakichś zachowań. W przykładzie z przelewem pieniędzy pomiędzy kontami oba zapytania SQL można by przechować po prostu jako pojedynczą procedurę składowaną o nazwie „przelew". Aplikacja przekazuje do procedury dwa numery kont oraz kwotę i procedura wykonuje te dwa zapytania SQL w jednej transakcji.

Na bardziej zaawansowanym poziomie procedury składowane mogą uzupełnić podstawową składnię SQL-a tak, że zacznie on bardziej przypominać tradycyjne języki programowania. Język PL/SQL Oracle i TransactSQL z Microsoft i Sybase są dwoma przykładami takich rozszerzeń SQL-a. Często słyszy się od ludzi, którzy używają tych procedur składowanych, o „przenoszeniu logiki biznesowej do bazy danych".

Zagnieżdżanie zapytań

Standardowe polecenie `SELECT` SQL umożliwia dostęp do wszystkich danych zawartych w tablicy, jeśli wiadomo, czego szukać. W zastosowaniach innych niż wypisanie zawartości całej tablicy nawet najprostsze formy zapytania z użyciem `SELECT` wymagają podania przynajmniej cząstki danych, które chcemy uzyskać. Na przykład użycie `SELECT imie FROM przyjaciele WHERE imie LIKE 'B%'` wymaga podania chociaż jednej z liter imienia, którego szukamy. Jeśli chcielibyśmy sprawdzić, kto miał w tym miesiącu pensję wyższą od przeciętnej, zapytanie prawdopodobnie wyglądałoby następująco:

```
SELECT imie FROM ludzie WHERE pensja > ???
```

Większa niż ile? Nie dowiemy się, jaka jest średnia pensja bez polecenia `SELECT`! Musimy wziąć wartość `SELECT AVG(pensja) FROM ludzie` i wstawić ją do poprzedniego zapytania. Pozwala na to zagnieżdżanie:

```
SELECT imie
FROM ludzie
WHERE pensja > (SELECT AVG(pensja) FROM ludzie)
```

Obiekty

Świat baz danych nie kończy się na ich relacyjnych odmianach. Na rynku znajduje się też dużo zorientowanych obiektowo oraz obiektowo-relacyjnych baz danych. Wśród produktów wysokiej klasy powoli zanika idea czysto relacyjnych baz danych. Standard SQL3 wprowadzi wiele zmian dla cech związanych ze wsparciem obiektowym.

W przypadku systemów zarządzania relacyjnymi bazami danych (RDBMS) wszystkie informacje przechowywane są w tablicach, będących zbiorem rekordów, czyli zestawów fragmentów informacji przedstawiających tekst, liczby lub inne typy danych. W przypadku obiektowo zorientowanych systemów zarządzania bazami danych (OODBMS) podstawową jednostką informacji jest obiekt. Obiekt może zawierać zarówno różne typy danych znane z relacyjnych systemów, jak także wielowymiarowe obiekty czy wielowymiarowe typy danych, jak tablice lub nawet wykonywane funkcje, znane w świecie obiektowym jako metody.

PostgreSQL

Obecna wersja systemu zarządzania obiektowo zorientowaną bazą danych znana jest jako PostgreSQL, a także jako Postgres 6. Sam system ma ponad 10 lat, chociaż obsługę SQL-a posiada zaledwie od 5 lat. Dr Michael Stonebreaker z Kalifornijskiego Uniwersytetu w Berkeley we wczesnych latach 80. ubiegłego wieku zaprojektował system baz danych, który zainicjował wiele koncepcji, jakie można odnaleźć w dzisiejszych relacyjnych bazach danych. Motor początkowo został nazwany Ingres. Później jego nazwa zmieniła się na University Ingres. Ingres był projektem finansowanym ze środków uniwersyteckich, który w krótkim czasie zainteresował innych komputerowych naukowców z całego świata.

Pewna firma dostrzegła potencjał rynkowy w tym akademickim produkcie i przekształciła Ingres w produkt komercyjny. Oryginalna, bezpłatna wersja Ingresa została przemianowana na University Ingres i jej rozwój był kontynuowany niezależnie od wersji komercyjnej. W trakcie udoskonalania programu dr Michael Stonebreaker odstąpił od początkowych założeń. Zdecydował, że nadszedł najwyższy czas na zaprojektowanie kompletnie nowego systemu baz danych, który rozszerzałby koncepcje Ingresa i wkraczał na nowe pola. System został nazwany Postgres (od Post-Ingres).

Postgres, podobnie jak Ingres, był projektem uniwersyteckim. W krótkim czasie zainteresowały się nim firmy i tak powstał komercyjny produkt Illustra.[2] Darmowa wersja Postgres była nadal rozwijana i obecnie jest równie popularna, jak MySQL czy mSQL w średniej klasie serwerów baz danych.

W roku 1995 dwie rzeczy wpłynęły na dalszy rozwój Postgresa. Pierwszą z nich była działalność dwóch studentów doktora Stonebreakera — Andrew Yu oraz Jolly Chen, którzy zaprojektowali interfejs SQL do tego systemu. Kilka lat po tym, jak David Hughes rozwinął MiniSQL jako interfejs SQL Postgresa, Postgres osiągnął stopień rozwoju, w którym posiadał własny, prawdziwy interfejs SQL. Wraz z pojawieniem się SQL-a zwiększyła się momentalnie popularność tego produktu. Tak jak w przypadku MySQL-a oraz mSQL-a zwiększenie popularności spowodowało zwiększone zapotrzebowanie na nowe własności. W rezultacie powstał zorientowany obiektowo-relacyjny motor baz danych klasy średniej, który wspierał transakcje, wyzwalacze i zagnieżdżanie zapytań. Więcej informacji na temat PostgreSQL można znaleźć pod adresem *http://www.postgresql.org*.

GNU SQL

Projekt GNU SQL jest symbolem wolności dla wielu ludzi w przemyśle komputerowym. Oficjalny projekt GNU jest dostępny za darmo z pełną możliwością ingerencji w kod źródłowy. Można odnaleźć wersje GNU wielu poleceń środowiska Unix, np. edytora (Emacs), powłoki (bash) lub jądra systemu operacyjnego (Hurd). Do niedawna jedynym wyjątkiem od tej reguły był system zarządzania bazą danych.

Programiści w Instytucie Programowania Systemowego Rosyjskiej Akademii Nauk ciężko pracują, aby zmienić ten stan rzeczy. Kilka lat temu została wypuszczona pierwsza publiczna wersja beta GNU SQL — w pełni funkcjonalnego systemu zarządzania relacyjnymi bazami danych SQL wydanego z licencją GNU Public (GPL). W trakcie pisania tej książki produkt GNU SQL był w wersji 0.7beta.

---

[2] Illustra została kupiona przez Informix w 1995 roku i obecnie jest częścią ich produktu o nazwie Universal Server (Faktycznie, jest taka część serwera Illustra, która jest częścią Informix Dynamic Server — *przyp. red.*).

Kiedy powstał GNU SQL, nie było jeszcze specyfikacji SQL2. Z tego względu początkowa wersja GNU SQL udostępniała jedynie funkcjonalność standardu SQL89. Cechy SQL2 były sukcesywnie dodawane wraz z ich pojawianiem się na rynku.

GNU SQL obsługuje obecnie wiele zaawansowanych cech, takich jak transakcje, zapytania czy kursory. Ze względu na to, że produkt jest w wersji beta, nie zaleca się jego używania w systemie produkcyjnym. Wraz z rozwojem staje się systemem wartym wykorzystywania. Więcej informacji na temat GNU SQL można znaleźć pod adresem: *http://www.ispras.ru/~kml/gss/index.htm*.

Beagle

Beagle to darmowy silnik SQL zaprojektowany i stworzony przez Roberta Kleina. Podobnie jak w przypadku GNU SQL, Beagle ma w planach pełną zgodność ze standardem SQL, ze wszystkimi jego cechami, włącznie z obiektowo-relacyjnymi rozszerzeniami, wprowadzonymi przez PostgreSQL. Beagle również jest nadal rozwijany. Obecnie osiągnął poziom rozwoju wystarczający do testowania i wykorzystywania w środowisku testowym lub rozwojowym. Nie powinien jednak jeszcze być używany w środowisku produkcyjnym.

Jednym z bardziej interesujących aspektów Beagle jest to, że autor utrzymuje od samego początku dziennik rozwoju produktu. Poprzez studiowanie dziennika można mieć wgląd w rozwój serwera SQL — począwszy od prostego systemu klient-serwer TCP, aż do w pełni rozwiniętego funkcjonalnie serwera SQL, którym Beagle jest dzisiaj. Strona główna projektu Beagle znajduje się pod adresem *http://www.beaglesql.org*.

Porównania

Tak jak wiele aplikacji, MySQL również posiada zestaw testów, które weryfikują, czy system rzeczywiście obsługuje te cechy, które powinien. MySQL nazwał własny zestaw testów crash-me (zniszcz mnie), ponieważ jednym z zadań tego zestawu testowego było unicestwienie działającego serwera baz danych MySQL.

W trakcie rozwoju produktu zauważono, że crash-me jest programem przenośnym. Nie tylko może działać z różnymi systemami operacyjnymi, ale również może być wykorzystywany do testowania różnych silników baz danych. Począwszy od tego odkrycia, crash-me przemienił się z prostego pakietu testowego w program porównawczy. Pakiet sprawdza cały zestaw funkcjonalności standardu SQL, jak również rozszerzenia oferowane przez wiele serwerów. Dodatkowo program sprawdza wydajność serwera pod dużym obciążeniem. Po wykonaniu całościowego testu program udostępnia kompletny obraz zdolności danego silnika bazy danych.

Można wykorzystać pakiet crash-me do porównania dwóch lub więcej silników baz danych online. Strona crash-me znajduje się pod adresem *http://www.mysql.com/crash-me-choose.htmy*.