ZESZYTY NAUKOWE UNIWERSYTETU SZCZECIŃSKIEGO

NR 651 EKONOMICZNE PROBLEMY USŁUG NR 68 2011

JĘDRZEJ MUSIAŁ
Wyższa Szkoła Bankowa w Poznaniu
JAKUB MARSZAŁKOWSKI
Politechnika Poznańska

PROPOZYCJA POPRAWY WYDAJNOŚCI BAZY DANYCH DLA NOWOCZESNYCH APLIKACJI INTERNETOWYCH

Wprowadzenie

Bardzo ważnym aspektem informatyki ostatnich lat jest rozwój sieci Internet. Liczba witryn internetowych bezustannie wzrasta. Powoduje to między innymi ciągły wzrost liczby osób posiadających dostęp do Internetu. Przez ostatnią dekadę wzrost wartości był stały i gwałtowny. Jak wynika z najnowszych danych statystycznych[1], 28,7% (1,966 mld) mieszkańców świata ma dostęp do sieci Internet. Największa penetracja usługi jest zauważalna w Ameryce Północnej i wynosi 77,4%. Kolejnymi regionami w kolejności malejącej są: Australia i Oceania (61,3%), Europa (58,4%), Ameryka Południowa (34,5%), Bliski Wschód (29,8%), Azja (21,5%) i Afryka (10,9%). Najważniejszy jest fakt bardzo dynamicznego wzrostu wskaźnika penetracji. W ostatnich 10 latach (2000–2010) wzrost liczby osób uzyskujących dostęp do Internetu wyniósł 444,8%, z 360 mln do 1,966 mld osób.

Główny Urząd Statystyczny[2] informuje, że poziom penetracji dostępu do sieci Internet w Polsce wynosi 57% (7,1 mln osób). Inne źródła wedle wyników swoich ankiet donoszą o wartości na poziomie 50%. Różnice mogą wynikać z innej metodologii lub szczegółowości przeprowadzonych badań.

---

[1] Internet World Stats, http://www.internetworldstats.com/stats.htm
[2] Główny Urząd Statystyczny, http://www.stat.gov.pl/gus

Tak ogromna liczba osób posiadających dostęp do Internetu powoduje codziennie gigantyczną liczbę odwołań do witryn internetowych. Zasadne okazuje się optymalizowanie architektury systemów internetowych, jak i procesów obsługujących żądania internautów. Bez wątpienia jednym z kluczowych elementów są systemy baz danych.

Warto również zauważyć, jak ogromną wartość przedstawia rynek *e-commerce'u*, który jest ściśle powiązany z liczbą użytkowników uzyskujących dostęp do Internetu, jak i witryn webowych. Badania przeprowadzone w Stanach Zjednoczonych Ameryki Północnej[3] pokazują, że wartość rynku *e-commerce'u* wzrosła z 7,4 mld dol. w połowie roku 2000 do 34,7 mld dol. w 2007.

1. Nowoczesne aplikacje internetowe

Dynamiczny wzrost liczby użytkowników Internetu i nowoczesne systemy gospodarki elektronicznej[4, 5, 6, 7] wpływają na wzrost wartości rynku *e-commerce'u*. Jednym z ważnych aspektów całego rynku gospodarki elektronicznej są zakupy internetowe. Z prowadzonych badań[8] wynika, że do roku 2013 prawie połowa mieszkańców Europy ma dokonywać zakupów w sklepach internetowych. W roku 2006 notowano odsetek klientów wynoszący 26% osób mających dostęp do sieci Internet.

Korzystanie z oferty sklepów internetowych/aukcji internetowych ma wiele zalet w porównaniu z tradycyjnymi sklepami. Najważniejszymi mogą być niższe ceny i możliwość zakupów z bardzo odległych miejsc. Należy jednak zauważyć, że zakupy związane są z dodatkowym kosztem wysyłki. Trzeba również wziąć pod uwagę fakt konsumpcji czasu, który jest niezbędny do znalezienia interesującej nas oferty. Zdarza się również, że przez stronę internetową nie można jednoznacznie określić, czy dwa dane produkty są identyczne. Szukanie najlepszej oferty na dany

---

[3] *On-line Shopping*, Pew Internet & American Life Project, http://www.pewinternet.org/Reports/2008/Online-Shopping.aspx

[4] J. Błażewicz, M.Y. Kovalyov, J. Musiał, A.P. Urbański, A. Wojciechowski: *Internet Shopping Optimization Problem*, „International Journal of Applied Mathematics and Computer Science", 20.02.2010, s. 385–390.

[5] W. Chu, B. Choi, M.R. Song: *The Role of On-line Retailer Brand and Infomediary Reputation in Increasing Consumer Purchase Intention*, „International Journal of Electronic Commerce", 9.03.2005, s. 115–127.

[6] C. Holsapple et. al.: *Decision Support Applications in Electronic Commerce*, w: M. Shaw et al. (eds.): *Handbook on Electronic Commerce*, Springer, Berlin Heidelberg 2000.

[7] A. Wojciechowski, J. Musiał: *Towards Optimal Multi-item Shopping Basket Management: Heuristic Approach*, w: R. Meersman et al. (eds.): OTM 2010 Workshops, LNCS, 6428, Springer, Heidelberg 2010, s. 349–357.

[8] *E-commerce across Europe – Progress and prospects*, The Future Foundation 2008.

produkt znacznie ułatwiają serwisy oferujące porównywanie cen, tzw. porównywarki cen. Aplikacje prezentują oferty sklepów na dany produkt. Klient od razu wie, gdzie można najtaniej kupić żądany towar. Według rankingu Alexa[9] wiele najpopularniejszych porównywarek cen znajduje się w pierwszym tysiącu najpopularniejszych (posiadających największą liczbę odwiedzin) ston internetowych na świecie. We wspomnianych programach bardzo popularnym system baz danych, na którym operują aplikacje, jest MySQL.

Kolejnymi ważnymi aplikacjami środowiska *e-commerce'owego* są systemy zarządzania treścia (*Content Management System*– CMS). Są to aplikacje, które umożliwiają redagowanie i publikowanie różnego rodzaju treści, jak: artykuły, wiadomości, pliki audio, pliki wideo, prezentacje, informacje o produktach, zdjęcia i wiele innych. Najczęściej używanymi są: Wordpress, Joomla!, Mambo, MODx, PHP-Nuke[10]. Domyślnym systemem baz danych proponowanym w aplikacjach CMS jest MySQL.

Serwisy społecznościowe (*Social Networking System* – SNS) szturmem zyskały ogromną liczbę zwolenników i użytkowników. Są to aplikacje pozwalające na kontakt ze znajomymi osobami oraz wymianę informacji, plików. Serwisy społecznościowe po części zastąpiły kontakty rzeczywiste kontaktami przy użyciu komputera, monitora i klawiatury. Najpopularniejszymi przedstawicielami serwisów SNS są bez wątpienia Facebook i Myspace. Facebook posiada ponad 500 mln aktywnych użytkowników[9]. Technologie PHP i MySQL są popularnie używane wśród programistów serwisów społecznościowych.

Najnowszym trendem są gry przeglądarkowe, w języku angielskim nazywane *social games*. Raport agencji NPD[11] podaje, że w Stanach Zjednoczonych 20% populacji przyznaje się do grania w takie gry, to 56,8 mln ludzi – potężny rynek. Brak jest analogicznych badań dla Polski, jednakże można wskazać, że najpopularniejsze tytuły mają przeszło po 100 tys. graczy każdy[12]. W grze przeglądarkowej internauta spędza czas, wielokrotnie wyświetlając jej strony i generując znaczne obciążenia dla serwera, w tym bazy danych – większość korzysta z MySQL.

---

[9] Alexa Rank, www.alexa.com

[10] R. Shreves: *Open Source CMS market Share*, Water & Stone 2008.

[11] *Social Network Gaming*, NPD Group 2010.

[12] J. Marszałkowski: *Problematyka pomiaru popularności gier przeglądarkowych*, Homo Ludens, 1.02.2010, s. 97–106.

2. Eksperymentalne środowisko bazodanowe

Wszystkie testy były przeprowadzane na komputerze IBM S50, wyposażonym w procesor Intel Pentium IV 2,8 GHz oraz 1 GB RAM, pracującym pod kontrolą systemu FreeBSD 8.1-RELEASE. Serwerem web dla tego badania był Apache 1.3.42, używający PHP 5.3.3_2 z APC 3.1.4. Testowana była wersja 5.1.51_1 MySQL, w czasie przeprowadzania badań ostatnia stabilna wersja z gałęzi 5.1. Wszystkie elementy oprogramowania były wybierane ze względu ich wysoką popularność w rozwiązaniach serwerowych.

Na potrzeby eksperymentu pomiarowego została stworzona testowa baza danych, starannie odwzorowująca, jak aplikacje internetowe zazwyczaj komunikują się z bazą danych. Aby oddać to w sposób reprezentatywny, autorzy poddali obserwacji wiele aplikacji, takich jak: porównywarki cen, serwisy społecznościowe, e-sklepy, gry przeglądarkowe oraz CMS-y.

Najczęściej powtarzanym przez nie zapytaniem jest prosty *SELECT*, pobierający informacje o jednej krotce danych. Aplikacje powszechnie używają od kilkunastu do kilkudziesięciu takich zapytań pozwalających na przeczytanie informacji niezbędnych do wygenerowania strony. Typy danych w pobieranych w ten sposób krotkach są zazwyczaj numeryczne oraz tekstowe, od najkrótszych do dość długich.

Schemat przygotowanej bazy danych kształtował się następująco:
$a = 32$ – liczba kolumn (pól) w wierszu (krotce),
$t \in \{t_1, t_2, t_3, t_4\}$ – pola zawierające dane typu: t_1 – tinyint (1 bajt), t_2 – int (4 bajty), t_3 – char (32 bajty), t_4 – text (512 bajtów),
$n \in \{1,64\}$ – dla każdego typu danych t dwie zawierające go tabele o nazwach długości n i liczbie kolumn a.

Wybrane typy danych odwzorowują najczęstsze potrzeby serwisów internetowych. Do przechowywania flag wyboru jakiejś opcji czy informacji, takich jak płeć, język itp., posiadających do 256 różnych wartości – zazwyczaj służy typ *tinyint*. Zmienne typu *int* (integer 4-bajtowy) powszechnie używane są do przechowywania wszelkiej informacji liczbowej, jak: ceny, identyfikatory pól, klucze, zmienne daty i czasu. Krótkie zmienne znakowe *char* ograniczone do 32 znaków przechowują nazwy, loginy, hasła, sumy kontrolne, identyfikatory sesji i podobne dane. I wreszcie typ *text* to dłuższe pola tekstowe, które mogą przechowywać opisy produktów osób lub każdy inny potrzebny ciąg znaków.

3. Eksperyment pomiarowy

Dla bazy danych opisanej w poprzednim rozdziale został przeprowadzony następujący eksperyment pomiarowy. Generowane jest zapytanie, które pobiera dane typu $t \in \{t_1, t_2, t_3, t_4\}$ z $c \in \{1, 2,... 32\}$ kolumn tabeli. Zapytanie jest kiero-

wane do tabel o nazwach długości $n \in \{1,64\}$. Dodatkowo dla $n = 64$ kierowane jest jeszcze jedno zapytanie, gdzie nazwa tabeli otrzymuje krótki, jednoznakowy skrót symboliczny (*alias*), np. *SELECT fields FROM table_name AS a*, oznaczany dalej jako $n = 64a$. Każde zapytanie z przedstawionego zestawu powtarzane jest 100 razy, aby wyeliminować przypadkowe pomiary odstające. Łącznie wykonano 38 400 pomiarów. Czasy wykonania zapytań zostały zmierzone za pomocą programu profilującego (*profiler*) MySQL[13], nowego narzędzia dostępnego od wersji 5.0.37. Wyniki pomiarów dla liczby kolumn $c \in \{1, 2, 4, 8, 16, 32\}$ – tabela 1.

Tabela 1

Czasy wykonania zapytań

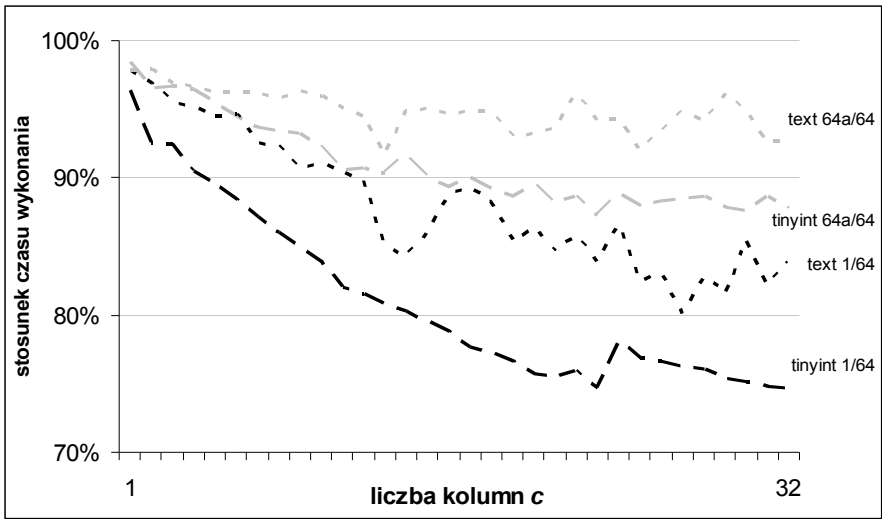
	tinyint (1)			int (4)			char (32)			text (512)		
n / c	1	64a	64	1	64a	64	1	64a	64	1	64a	64
1	259	265	269	262	265	271	262	267	269	310	310	317
2	268	280	290	270	280	285	273	285	285	319	322	329
4	281	299	310	282	297	307	285	298	306	335	340	352
8	309	335	358	311	335	348	310	328	346	370	384	401
16	354	401	449	356	392	425	356	389	422	442	471	498
32	450	529	602	449	522	591	449	522	584	586	649	699

Jednocześnie z pomiarami czasu sprawdzana była liczba bajtów przesyłanych przez bazę, za pomocą zapytań typu *SHOW SESSION STATUS*. Stwierdzono, że przy niezmiennych c i t różnice między wersjami zapytań ze względu na $n \in \{1, 64, 64a\}$ są stałe. Różnica między $n = 1$ a $n = 64$ wynosiła 126 bajtów, a między $n = 64$, a $n = 64a$ tylko 63 bajty.

Pozwala to wysunąć wniosek, że z każdą kolumną pobieraną przez zapytanie z bazy posyłany jest wiersz nagłówkowy, który zawiera powtórzoną dla każdej kolumny nazwę tabeli oraz użyty alias. Jeżeli zaś nie użyto aliasu, zawiera dwukrotnie nazwę tabeli. Można też stwierdzić, że to właśnie te nagłówki kolumn generują badane narzuty czasu wykonania – są długie na kilkadziesiąt czy ponad sto bajtów, podczas gdy dane zawarte w kolumnie często liczą tylko kilka bajtów.

Na rysunku 1 przedstawiono stosunek między czasem wykonania zapytania dla $n = 64$ a $n = 1$ w kolorze czarnym oraz $n = 64a$ w kolorze szarym. W celu zachowania czytelności wykresu wybrano tylko dwa skrajne typy danych *tinyint* (1) – oznaczony linią kreskowaną, o największym zysku wydajności, oraz text (255) – oznaczony linią kropkowaną, o najmniejszym zysku wydajności.

[13] MySQL AB, MySQL: *The World's Most Popular Open Source Database*, http://www.mysql.org



Rys. 1. Zysk wydajności na krótkich nazwach tabel lub przy zastosowaniu aliasów

4. Eksperyment praktyczny

Dodatkowy eksperyment pomiarowy został przeprowadzony na modelu aplikacji internetowej oraz dwóch rzeczywistych aplikacjach. Wszystkie pomiary powtarzano 100 razy.

Model został skonstruowany tak, by odzwierciedlać rzeczywiste zapytania dokonywane przez prostą aplikację, takie jak: czytanie danych antiflood, konfiguracji, danych użytkownika, newsów, artykułów itd. Łącznie składał się z 15 tabel zawierających od 5 do 40 kolumn, z typami danych stanowiącymi rozsądną mieszankę tych użytych w punkcie 2. Do każdej z tabel wywoływane było jedno zapytanie wybierające jedną krotkę identyfikowaną kluczem głównym. Testy obejmowały porównanie czasów wykonania aplikacji dla zapytań do tabel o nazwach 2-znakowych oraz dla 34-znakowych, także w wersji z aliasem. Dwa znaki pozwalają na dostateczną liczbę nazw tabel niemal dla każdego projektu, z kolei 34 były średnią obserwowaną w obu badanych rzeczywistych aplikacjach. Średni czas wykonania dla tabel o nazwach 2-znakowych zmierzono jako 5,63 ms, dla 34-znakowych 6,23 ms, a z aliasem 5,91 ms. Zysk wydajności z zastosowania aliasu wynosi więc 5,2%, a z krótkiej nazwy tabeli 9,6%.

Do testów na rzeczywistych aplikacjach wybrano dwa wiodące pakiety CMS[11]: PHP Nuke oraz MODx. Oba wykonują kilkadziesiąt zapytań do bazy danych, by wyświetlić pojedynczą stronę WWW. Aplikacjom zmierzono czasy wykonania w ich oryginalnych formach z tabelami o nazwach długości 21-47 znaków.

Następnie skrócono nazwy tabel do dwuznakowych i ponowiono pomiary. W przypadku PHP Nuke skróciło to czas pojedynczego wykonania z 38,26 ms do 35,91 ms – a więc o 6,1%. Dla MODx: z 473,53 ms do 464,63 ms, tu już tylko o 1,9%.

MODx dla komunikacji z bazą danych używa dodatkowego obiektowego wrappera (czyli dodatkowych funkcji opakowujących, których zadaniem jest wywołanie szeregu innych funkcji), który spowodował, że zmiana nazw tabel w kodzie jest bardzo prosta – każdą tabelę trzeba było podmienić w jednym pliku definicyjnym. Obudowuje on jednak zapytania dodatkowymi elementami, sam z siebie używa wielu aliasów i nader często łączenia tabel. Uważa się, że wiąże się to ze znacznym narzutem na wydajność, co zapewne przykryło zyski z krótkich nazw tabel.

PHP Nuke używa dość prostych zapytań wysyłanych bezpośrednio do bazy danych. Zmiana nazw tabel wymagała wyszukania i zamiany ciągów w kilkuset zapytaniach – można to było jednak przeprowadzić półautomatycznie. Tutaj zysk z krótkich nazw tabel jest zupełnie zauważalny.

Podsumowanie

Jak zostało wykazane, tak prosty element jak długość nazw tabel w bazie danych może mieć znaczny wpływ na wydajność aplikacji internetowych. Należy zalecić używanie rozsądnie krótkich nazw tabel w nowo projektowanych aplikacjach. Dla aplikacji istniejących zaś przede wszystkim unikanie prefiksów nazw tabel, do których zachęcają niemalże wszystkie programy instalacyjne takich aplikacji. Jednocześnie, jeżeli potrzeba jest dodatkowych kilku procent wydajności, skrócenie nazw tabel wiąże się niemalże z zerowym kosztem i jest możliwe do przeprowadzenia w istniejącym oprogramowaniu.

Do każdego zapytania do bazy danych winien być stosowany alias dla nazwy tabeli, możliwie krótki, najlepiej jednoznakowy. Nie będzie to w żaden sposób utrudniać pracy z pobranymi zmiennymi, a zapytania będą wykonane szybciej.

Przepisanie jednego z popularnych wrapperów bazy danych, tak by korzystał z krótkich nazw tabel i krótkich aliasów, pozwoliłoby aplikacjom z niego korzystającym na zyski w wydajności. Jednocześnie wrapper taki może całkowicie zasłaniać nazwy tabel od strony programisty, oferując interfejs z wygodnymi nazwami zmiennych.

Literatura

1. Alexa Rank, www.alexa.com
2. Błażewicz J., Kovalyov M.Y., Musiał J., Urbański A.P., Wojciechowski A.: *Internet Shopping Optimization Problem*, „International Journal of Applied Mathematics and Computer Science", 20.02.2010.

3. Chu W., Choi B., Song MR: *The Role of On-line Retailer Brand and Infomediary Reputation in Increasing Consumer Purchase Intention*, „International Journal of Electronic Commerce", 9.03.2005.
4. *E-commerce across Europe. Progress and prospects*, The Future Foundation 2008.
5. Główny Urząd Statystyczny, http://www.stat.gov.pl/gus
6. Holsapple C. et. al.: *Decision Support Applications in Electronic Commerce* w: Shaw M. et al. (eds.): *Handbook on Electronic Commerce*, Springer, Berlin Heidelberg 2000.
7. Internet World Stats, http://www.internetworldstats.com/stats.htm
8. Marszałkowski J.: *Problematyka pomiaru popularności gier przeglądarkowych*, Homo Ludens, 1.02.2010.
9. MySQL AB, MySQL: *The World's Most Popular Open Source Database*, http://www.mysql.org
10. *On-line Shopping*, Pew Internet & American Life Project, http://www.pewinternet.org/Reports/2008/Online-Shopping.aspx
11. Shreves, R.: *Open Source CMS market Share*, Water & Stone 2008.
12. *Social Network Gaming*, NPD Group 2010.
13. Wojciechowski A., Musiał J.: *Towards Optimal Multi-item Shopping Basket Management: Heuristic Approach*, w: R. Meersman et al. (eds.): OTM 2010 Workshops, LNCS, 6428, Springer, Heidelberg 2010.

DATABASE EFFICIENCY IMPROVEMENT PROPOSITION FOR MODERN WEB-BASED APPLICATIONS

Summary

With rapid growth of Internet usage e-commerce projects, such as content management systems, on-line shops, social networking pages and online web-based gaming, those applications meet new performance challenges. In this note we enclose information on database systems using MySQL working in the background of web-based applications. Effect of table name length in simple queries is measured in experimental environment. Conclusions over execution time and amounts of data sent are made. Also comparison performance tests for long and short table names are made on a model, and two real web-based applications.

Translated by Jędrzej Musiał, Jakub Marszałkowski