STUDIA INFORMATICA 2013
Volume 34 Number 2A (111)

Marcin GRZESIAK, Aleksander CIANCIARA, Adam PIÓRKOWSKI
AGH Akademia Górniczo-Hutnicza, Wydział Geologii, Geofizyki i Ochrony Środowiska,
Katedra Geoinformatyki i Informatyki Stosowanej

SKALOWALNOŚĆ WYDAJNOŚCI OPERACJI BAZODANOWYCH W SYSTEMACH Z PROCESORAMI WIELORDZENIOWYMI

Streszczenie. Artykuł opisuje testy wydajności trzech popularnych wolno dostępnych systemów baz danych: PostgreSQL, MySQL oraz Firebird. Testami objęto wpływ liczby rdzeni w procesorze na wydajność pracy każdej bazy. Testy Przeprowadzono przez pomiar czasu wykonania pojedynczych zapytań DML o różnym stopniu skomplikowania. Dodatkowo przetestowano obciążenia przez wielokrotne zadanie tego samego zapytania.

Słowa kluczowe: wydajność, systemy wielordzeniowe, darmowe bazy danych

PERFORMANCE SCALABILITY OF DATABASE OPERATIONS ON SYSTEMS WITH MULTI-CORE PROCESSORS

Summary. This article presents tests of three popular, public domain, database systems: PostgreSQL, MySQL and Firebird. We are examine the impact on performance of each database the number of cores in the processor. The tests were performed by measuring the execution time of individual DML queries of varying complexity. In addition, the load test was performed by the single query executed multiple times.

Keywords: performance, multiple core processor, public database

1. Wstęp

1.1. Wprowadzenie

Rozwój technologii informatycznych we wszystkich dziedzinach życia (np. medycyna, finanse, handel, nauka) generuje ogromne zapotrzebowanie na gromadzenie, przechowywanie i analizowanie danych. Naprzeciw tym potrzebom wychodzą bazy danych, które stały się jednym z najważniejszych elementów systemów informatycznych. W połączeniu z wydajnymi

serwerami opartymi na architekturze wieloprocesorowej system jest w stanie wykonać więcej operacji w tej samej jednostce czasu.

Coraz większe ilości przechowywanych danych powodują, że czas odpowiedzi bazy danych na zadane pytanie również wzrasta. Wraz ze wzrostem ilości danych rośnie też poziom komplikacji zadawanych zapytań, tak aby otrzymać odpowiedź jak najbardziej precyzyjną i dokładną. To także spowalnia działanie bazy danych. Zatem coraz istotniejsze jest badanie wydajności systemów baz danych i wpływu, jaki mają na jej zwiększenie różne aspekty sprzętowe i programowe. Podejmowane są próby wykorzystania mocy obliczeniowych kart graficznych [1] w systemach baz danych czy optymalizacji struktur [2]. Kolejnym sposobem na skrócenie czasu obliczeń jest takie zaplanowanie przetwarzania, aby dane w całości umieszczać w pamięci RAM [3]. Jak widać, kwestia mierzenia wydajności jest badana pod wieloma względami. Bez problemu można znaleźć w Internecie bezpośrednie porównania wydajności pomiędzy różnymi bazami [10], jednak nie udało nam się znaleźć testów pokazujących wpływ zastosowania procesorów wielordzeniowych. W niniejszej pracy poddamy testom wpływ na wydajność samego procesora, a dokładniej wypływ na wydajność bazy danych liczby dostępnych dla niej rdzeni.

1.2. Cel pracy

Celem artykułu jest porównanie wydajności wybranych silników baz danych w systemach z procesorami wielordzeniowymi.

Testami objęto trzy najbardziej popularne wolno dostępne silniki relacyjnych baz danych, czyli PostgreSQL, MySQL oraz Firebird. Głównym celem było zbadanie zachowania bazy danych w zależności od dostępnej liczby rdzeni procesora przez wykonanie zapytania DML (Data Manipulation Language) języka SQL i zmierzenie czasu odpowiedzi serwera. Zbadano również wpływ obciążenia w tzw. load test, który polega na symulowaniu pracy serwera pod dużym obciążeniem.

Aby zmniejszyć wpływ innych czynników, w konfiguracji stanowiska testowego zastosowano szybki dysk SSD oraz wystarczającą ilość pamięci RAM.

2. Narzędzia wykorzystane w projekcie

2.1. Platforma testowa

2.1.1. Procesor

Z racji tego, że projekt dotyczy wydajności baz danych w systemach z procesorami wielordzeniowymi, w testach wykorzystano czterordzeniowy procesor Intel i5-3330. Układ ten

został oparty na architekturze SMP [4], w której każdy z rdzeni może zostać przypisany do dowolnego zadania. Rdzenie współdzielą zasoby wejścia/wyjścia oraz pamięć główną. Dodatkowo wewnątrz procesora współdzielona jest również pamięć podręczna trzeciego poziomu (L3 Cache – 6 MB). Każdy rdzeń ma własną pamięć pierwszego (L1 – 64 KB, w tym 32 KB dane, 32 KB instrukcje) i drugiego poziomu (L2 Cache – 254 KB). Procesor taktowany jest z częstotliwością 3,0 GHz (do 3,2 GHz w trybie Turbo Boost).

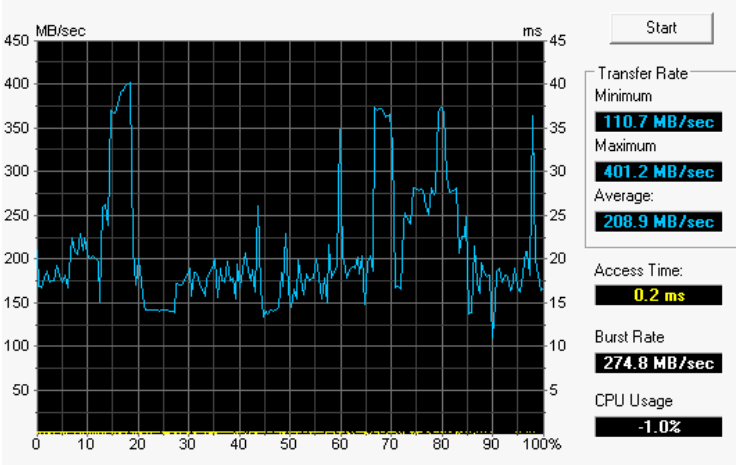
2.1.2. Pamięć RAM

Wykorzystano dwa czterogigabajtowe układy pamięci Patriot działające w trybie Dual Channel oraz pracujące z maksymalną częstotliwością 1600 MHz. Łącznie do dyspozycji było 8 GB. Taka ilość pamięci powinna w zupełności zmniejszyć narzut czasowy dzięki buforowaniu danych potrzebnych do wykonania zapytania i zmniejszyć liczbę operacji I/O.

2.1.3. Dysk SSD

Rys. 1. Odczyt i zapis danych o rozmiarze od 0,5 KB do 8192 KB. Test przeprowadzono w ATTO Disk Benchmark

Fig. 1. Read and write data of the size from 0,5 KB to 8192 KB. Test results from ATTO Disk Benchmark



Rys. 2. Rzeczywista charakterystyka odczytu. Test przeprowadzono w programie HD Tune.

Fig. 2. The actual characteristics of the reading. The test assay was performed in the HD Tune

W testach zastosowano dysk SSD marki OCZ Solid 3 o pojemności 120 GB i deklarowanym przez producenta maksymalnym odczycie 500 MB/s (rys. 1 oraz rys. 2 – rzeczywista charakterystyka odczytu) oraz zapisie 400 MB/s, wykorzystujący interfejs SATA 3. Dzięki zastosowaniu pamięci flash w miejsce tradycyjnych nośników magnetycznych można zauważyć co najmniej dwukrotne zwiększenie szybkości odczytu i zapisu (średnia szybkość odczytu i zapisu dla współczesnych dysków HDD to 50-150 MB/s).

2.2. Platforma systemowa

Testy wykonano w 64-bitowym systemie Windows 7 Professional.

2.3. Systemy zarządzania bazami danych

2.3.1. PostgreSQL 9.2

PostgreSQL to silnik bazy danych rozwijany od ponad 15 lat. Implementuje większość standardu SQL: 2008, m.in. typy danych (INTEGER, NUMERIC, BOOLEAN, CHAR, VARCHAR, DATE, INTERVAL, TIMESTAMP). Wspiera przechowywanie dużych obiektów binarnych jako elementów tabeli. Silnik spełnia zbiór właściwości ACID zapewniających poprawne przetwarzanie transakcji dzięki MVCC[1], który polega na tym, że każda transakcja dostaje własny obraz bazy danych i przeprowadzone zmiany nie są widoczne dla transakcji wykonywanych w tym samym momencie.

2.3.2. MySQL Server 5.5

Serwer MySQL jest rozwijany przez firmę Oracle, która w 2010 roku kupiła dotychczas zajmującą się bazą MySQL firmę Sun Microsystems. Baza ta zawiera wiele mechanizmów składowania danych, m.in. MyISAM, InnoDB (wykorzystywany w tym projekcie), CSV, BerkeleyDB. Od wersji 5. dostępne są procedury składowe, wyzwalacze, kursory, partycjonowanie tabel [6].

2.3.3. FireBird 2.5.2

FireBird 2.5.2 to relacyjna baza danych oferująca większość właściwości standardu ANSI SQL. Udostępnia kilka typów silników [8]:

- SuperServer – przeznaczony dla małych baz, wykorzystujący mechanizm pamięci dzielonej, ale mający problem ze skalowalnością,
- ClassicServer – umiejący wykorzystywać systemy wieloprocesorowe, ponieważ każde nowe połączenie to nowy proces z własnym buforem strony,

[1] MVCC – Multiversion Concurrency Control [7].

- SuperClassic (wykorzystany w projekcie) – ma te same właściwości co ClassicServer z tą różnicą, że każdy nowy klient to nowy wątek, dzięki czemu zmniejszono obciążenie systemu, ponieważ utworzenie nowego wątku jest mniej kosztowną operacją niż stworzenie procesu.

2.4. Oprogramownie testujące

2.4.1. Apache JMeter

Apache JMeter to narzędzie służące do testowania wydajności różnego rodzaju serwerów (baz danych, HTTP, HTTPS, LDAP, SOAP), napisane w języku Java. Pozwala na tworzenie rozbudowanych scenariuszy za pomocą takich elementów, jak: timer, preprocesory, postprocesory, asercje, słuchacze. Ma wiele rozszerzeń służących np. do wizualizacji danych.

2.4.2. Process Explorer

Process Explorer to program umożliwiający wyświetlanie szczegółowych informacji na temat uruchomionych procesów w systemie. Umożliwia zakończenie, uśpienie, ponowne uruchomienie, ustawienie koligacji, priorytetu wybranego procesu. Ponadto wyświetla użycie mocy procesora, liczbę operacji I/O, monitor sieci.

3. Metodyka testów

```
CREATE TABLE test11
(
imie VARCHAR,
nazwisko VARCHAR,
wiek INTEGER
miasto VARCHAR,
wojewodztwo VARCHAR,
ulica VARCHAR
)
```

Rys. 3. Schemat tabeli stworzonej w każdym silniku bazy danych
Fig. 3. Scheme of the table created for each database engine

Na każdej bazie danych stworzono tabele (o schemacie przedstawionym na rys. 3) w przestrzeni tabel znajdującej się na partycji dysku SSD. Na początku stworzono 4000 krotek, a później za pomocą składni CREATE TABLE test AS SELECT… lub SELECT … INTO FROM … wygenerowano 16 mln wpisów. Po utworzeniu każda baza danych zajmowała na dysku ponad 1 GB.

3.1. Test pojedynczego zapytania

Do bazy danych wysyłano zapytania z tabeli 1. Każde zapytanie było wykonywane cztery razy przy różnej wartość koligacji (od 1 do 4). Do zmiany liczby dostępnych rdzeni dla procesu odpowiedzialnego za wykonanie żądania wykorzystano program Process Explorer. Czas odpowiedzi mierzono w programie Apache JMeter.

Tabela 1

Zapytania DML wysyłane do baz danych

Z1	`SELECT imie, nazwisko, wiek FROM test11 WHERE wiek = (SELECT MAX(wiek) FROM test11) AND imie LIKE '_P%' ORDER BY imie;`

Z2	`SELECT imie, nazwisko, wiek FROM test11 WHERE wiek = (SELECT MAX(wiek) FROM test11) AND imie LIKE '_P%'`
	`UNION`
	`SELECT imie, nazwisko, wiek FROM test11 WHERE wiek = (SELECT MIN(wiek) FROM test11) AND imie LIKE '_T%' ORDER BY imie;`

Z3	`SELECT imie, nazwisko, wiek FROM test11 WHERE imie LIKE '_P%' ORDER BY imie;`
Z4	`SELECT imie, nazwisko, wiek FROM test11 WHERE wiek = 20;`
Z5	`SELECT DISTINCT imie, nazwisko, wiek FROM test11 GROUP BY imie, nazwisko, wiek;`

3.2. Test obciążenia

Za pomocą Apache JMeter do każdej z baz danych wysyłano stukrotnie to samo zapytanie (tabela 2) w ciągu 1 sekundy. Następnie zmierzono minimalny, średni i maksymalny czas wykonania zapytania. Procedurę powtórzono przy różnej liczbie dostępnych rdzeni dla procesu odpowiedzialnego za wykonanie zapytania. W celu zmniejszenia obciążenia procesora serwera żądania były wysyłane ze zdalnego komputera znajdującego się w tej samej sieci opartej na standardzie Ethernet.

Tabela 2

Zapytanie DML, które było wielokrotnie wysyłane do baz danych

Z	`SELECT DISTINICT imie, nazwisko, wiek FROM test11 GROUP BY wojewodztwo, imie, wiek nazwisko HAVING imie LIKE 'P%';`

4. Wyniki

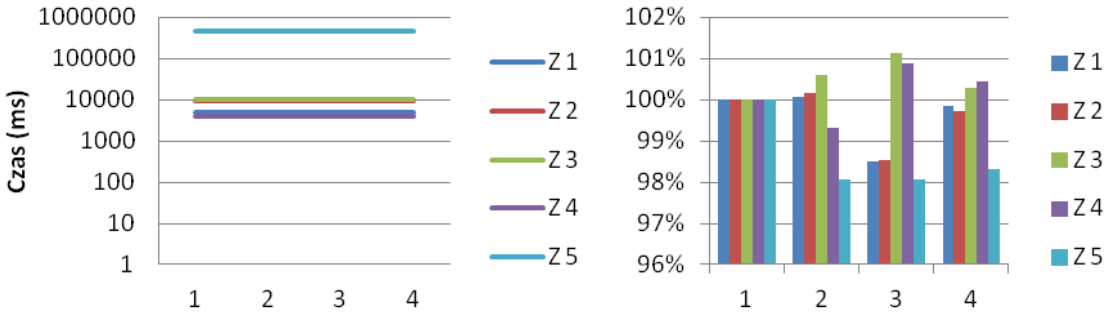
4.1. Test pojedynczego zapytania

4.1.1. PostgreSQL 9.2

Na podstawie uzyskanych wyników można stwierdzić, że silnik PostgreSQL 9.2, nie potrafi skorzystać z dodatkowych rdzeni niezależnie od skomplikowania zapytania. Świadczy o tym wykres na rys. 4, przedstawiający procentowy zysk wynikający z użycia dodatkowych rdzeni. Jest to najprawdopodobniej spowodowane tym, że każda sesja jest obsługiwana przez

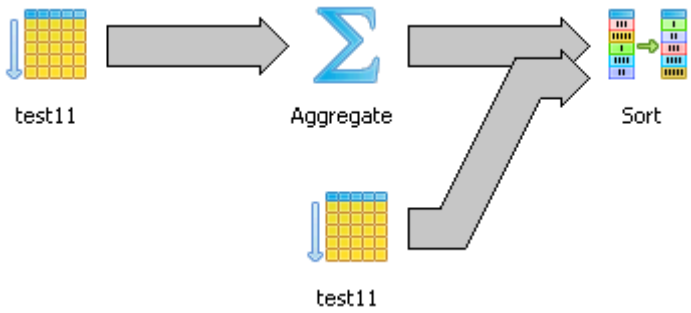
osobny proces, oraz tym, że kod PostgreSQL nie jest bezpieczny wątkowo [5], dlatego twórcy nie zdecydowali się na optymalizację zapytań za pomocą wątków.

Na rysunkach od 5 do 9, przedstawiono plany wykonania zapytań.



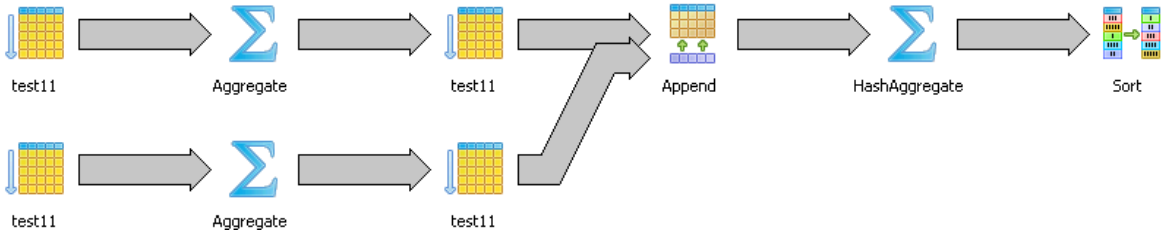
Rys. 4. Po lewej – czasy wykonania zapytań w bazie PostgreSQL w skali logarytmicznej, po prawej – procentowy zysk z dodatkowych rdzeni

Fig. 4. On the left – query execution time by PostgreSQL database in a logarithmic scale, on the right – gain percentage from the additional cores



Rys. 5. Plan wykonania zapytania Z1 w bazie PostgreSQL

Fig. 5. Z1 query execution plan in database PostgreSQL

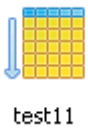


Rys. 6. Plan wykonania zapytania Z2 w bazie PostgreSQL

Fig. 6. Z2 query execution plan in database PostgreSQL

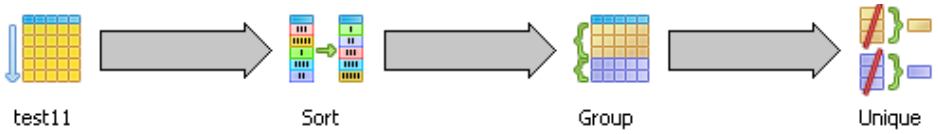
Rys. 7. Plan wykonania zapytania Z3 w bazie PostgreSQL

Fig. 7. Z3 query execution plan in database PostgreSQL



Rys. 8. Plan wykonania zapytania Z4 w bazie PostgreSQL

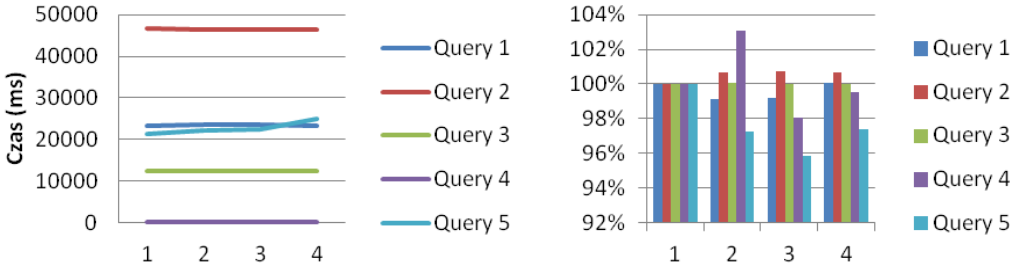
Fig. 8. Z4 query execution plan in database PostgreSQL



Rys. 9. Plan wykonania zapytania Z5 w bazie PostgreSQL
Fig. 9. Z5 query execution plan in database PostgreSQL

4.1.2. FireBird 2.5.2

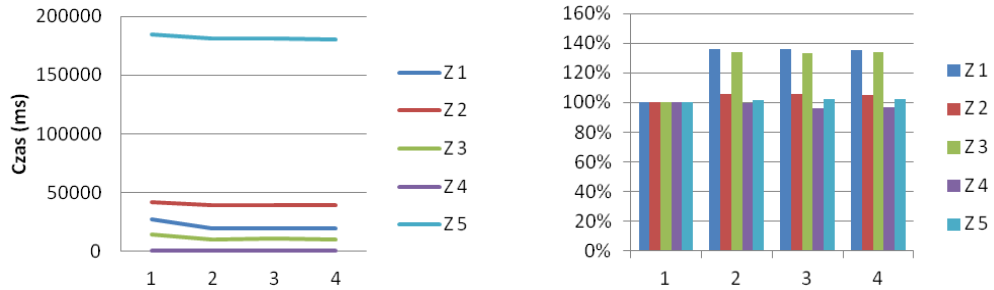
System FireBird 2.5.2, tak jak poprzednik, nie został zaprojektowany do wykorzystania w pojedynczym zapytaniu procesorów wielordzeniowych. Świadczą o tym wyniki przedstawione na rys. 10. Występują pewne różnice w wydajności (np. czas wykonania zapytania Z4 wzrósł o 3% przy dwóch rdzeniach, a przy trzech zmniejszył się o 2%), ale na ich podstawie nie można stwierdzić, że FireBird potrafi zrównoleglić przetwarzanie pojedynczego zapytania, niezależnie od jego złożoności. Różnice te należy traktować jako błędy pomiarowe.



Rys. 10. Po – lewej czasy wykonania zapytań w bazie Firebird, po prawej – procentowy zysk z dodatkowych rdzeni
Fig. 10. On the left – query execution time by Firebird database, on the right – gain percentage from the additional cores

4.1.3. MySQL Sever 5.5

Wyniki (rys. 11) jednoznacznie pokazują, że wydajność bazy danych zwiększa się tylko przy zmianie koligacji z jednego rdzenia na dwa, w dodatku dla mniej złożonych obliczeniowo zapytań (wzrost o ok. 34% dla Z1 i Z3, Z2 +5%, Z4 -0,58%). Większa liczba dostępnych rdzeni nie prowadzi do wzrostu wydajności, a czasami nawet ją zmniejsza (Z4 dla 4 rdzeni – 3,26% względem jednego rdzenia). Jest to spowodowane narzutem czasowym generowanym przez procesor w czasie zmiany rdzenia, na którym ma zostać wykonany wątek.



Rys. 11. Po lewej – czasy wykonania zapytań w bazie MySQL, po prawej – procentowy zysk z dodatkowych rdzeni
Fig. 11. On the left – query execution time by MySQL database, on the right – gain percentage from the additional cores

Na rysunkach od 12 do 16 przedstawiono plany wykonania zapytań.

id	select_type	table	type	possible_keys	key	key_len	ref	rows	Extra
1	PRIMARY	test11	ALL	NULL	NULL	NULL	NULL	16001166	Using where; Using filesort
2	SUBQUERY	test11	ALL	NULL	NULL	NULL	NULL	16001166	

Rys. 12. Plan wykonania zapytania Z1 w bazie MySQL
Fig. 12. Z1 query execution plan in database MySQL

id	select_type	table	type	possible_keys	key	key_len	ref	rows	Extra
1	PRIMARY	test11	ALL	NULL	NULL	NULL	NULL	16001166	Using where
2	SUBQUERY	test11	ALL	NULL	NULL	NULL	NULL	16001166	
3	UNION	test11	ALL	NULL	NULL	NULL	NULL	16001166	Using where
4	SUBQUERY	test11	ALL	NULL	NULL	NULL	NULL	16001166	
NULL	UNION RESU...	<union1,3>	ALL	NULL	NULL	NULL	NULL	NULL	Using filesort

Rys. 13. Plan wykonania zapytania Z2 w bazie MySQL
Fig. 13. Z2 query execution plan in database MySQL

id	select_type	table	type	possible_keys	key	key_len	ref	rows	Extra
1	SIMPLE	test11	ALL	NULL	NULL	NULL	NULL	16001166	Using where; Using filesort

Rys. 14. Plan wykonania zapytania Z3 w bazie MySQL
Fig. 14. Z3 query execution plan in database MySQL

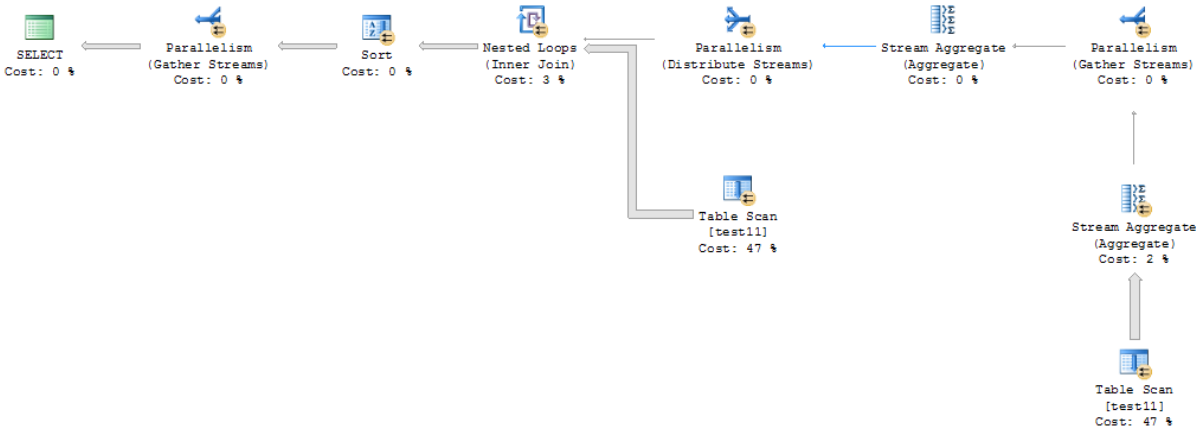
id	select_type	table	type	possible_keys	key	key_len	ref	rows	Extra
1	SIMPLE	test11	ALL	NULL	NULL	NULL	NULL	16001166	Using where

Rys. 15. Plan wykonania zapytania Z4 w bazie MySQL
Fig. 15. Z4 query execution plan in database MySQL

id	select_type	table	type	possible_keys	key	key_len	ref	rows	Extra
1	SIMPLE	test11	ALL	NULL	NULL	NULL	NULL	16001166	Using temporary; Using filesort

Rys. 16. Plan wykonania zapytania Z5 w bazie MySQL
Fig. 16. Z5 query execution plan in database MySQL

4.1.4. Porównanie planów zapytań dla MS SQL Serwer w wersji Enterprise

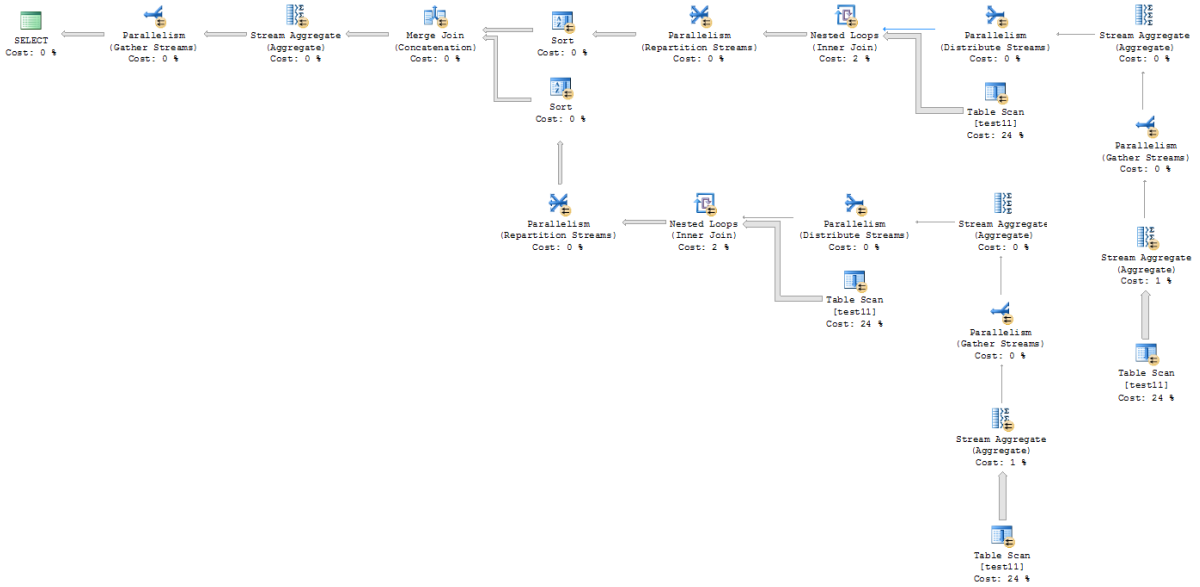


Rys. 17. Plan wykonania zapytania Z1 w bazie MS SQL Enterprise Edition
Fig. 17. Z1 query execution plan in database MS SQL Enterprise Edition

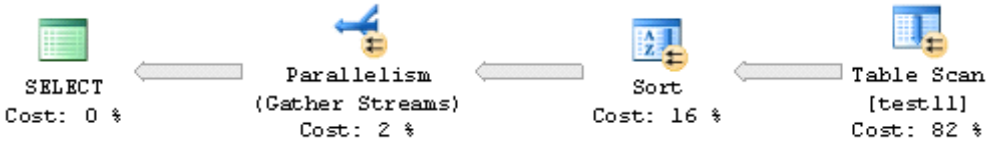
Na zakończenie przedstawiamy plany przetwarzania zapytań, jakie oferuje baza MS SQL Serwer Enterprise (rys. od 17 do 21). Miejsca, gdzie mechanizm bazy danych wykorzystuje

zrównoleglenie przetwarzania na wielu procesorach (rdzeniach), są oznaczane jako „Parallelism”.

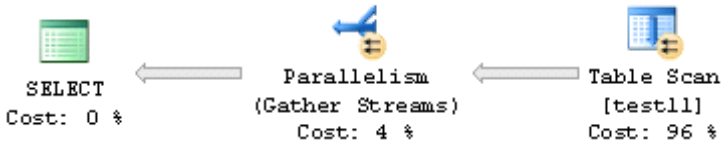
Widać zatem wyraźnie, że nawet tak proste zapytania, jakie zostały zastosowane w niniejszych testach, mogą z powodzeniem wykorzystywać wydajność oferowaną przez dodatkowe rdzenie procesora.



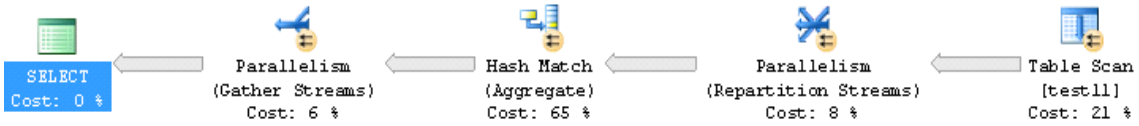
Rys. 18. Plan wykonania zapytania Z2 w bazie MS SQL Enterprise Edition
Fig. 18. Z2 query execution plan in database MS SQL Enterprise Edition



Rys. 19. Plan wykonania zapytania Z3 w bazie MS SQL Enterprise Edition
Fig. 19. Z3 query execution plan in database MS SQL Enterprise Edition



Rys. 20. Plan wykonania zapytania Z4 w bazie MS SQL Enterprise Edition
Fig. 20. Z4 query execution plan in database MS SQL Enterprise Edition



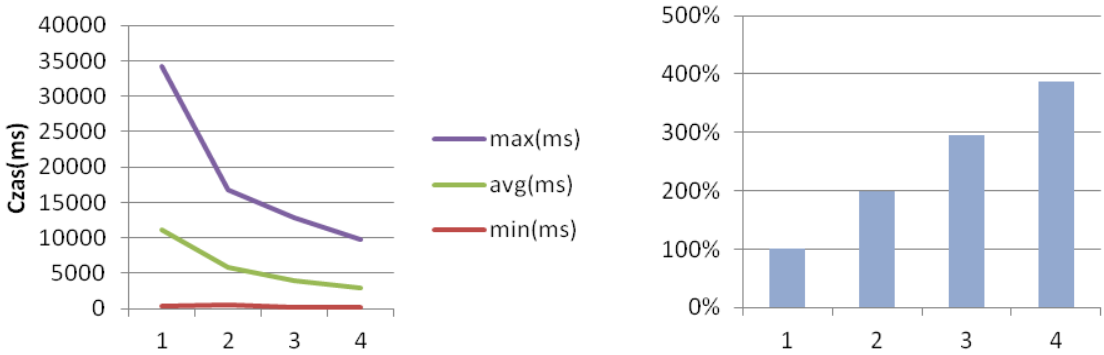
Rys. 21. Plan wykonania zapytania Z5 w bazie MS SQL Enterprise Edition
Fig. 21. Z5 query execution plan in database MS SQL Enterprise Edition

Widać zatem wyraźnie, że MS SQL wykorzystuje mechanizmy pozwalające na zwiększenie wydajności, jednak test tej bazy wykraczał poza ramy przyjętego eksperymentu.

4.2. Test obciążenia

4.2.1. PostgreSQL 9.2

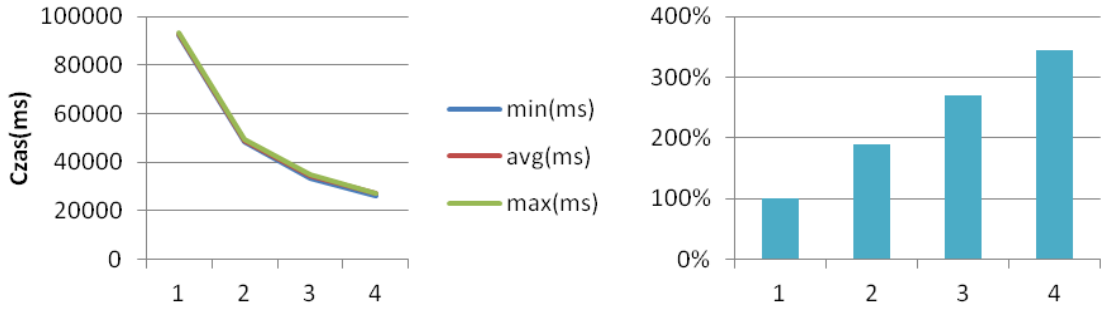
Przedstawione na wykresach rys. 22 wyniki jednoznacznie pokazują, że PostgreSQL nie ma problemów ze skalowalnością. Silnik ten zawdzięcza takie wyniki tworzeniu nowego procesu dla każdego połączenia [9] i umiejętnemu dzieleniu zasobów. Duży wpływ ma też zastosowany model transakcyjny MVCC. Każdy nowy klient dostaje „snapshot" bazy danych, który jest niewidoczny dla innych użytkowników do momentu zatwierdzenia transakcji. Dzięki temu baza danych spełnia wymogi ACID. Należy też zauważyć bardzo dużą rozpiętość czasu odpowiedzi. Sugeruje to, że baza PostgreSQL przechowuje w pamięci podręcznej wyniki już przetworzonych zapytań.



Rys. 22. Po lewej – minimalny, średni i maksymalny czas wykonania zapytania w bazie PostgreSQL, po prawej – procentowy zysk z dodatkowych rdzeni

Fig. 22. On the left – minimum, average and maximum execution time for query in a PostgreSQL database, on the right – gain percentage from the additional cores

4.2.2. MySQL Sever 5.5



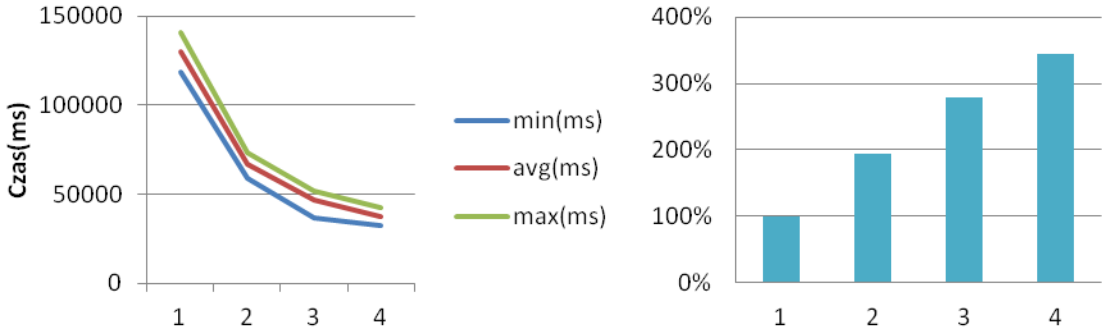
Rys. 23. Po lewej – minimalny, średni i maksymalny czas wykonania zapytania w bazie MySQL, po prawej – procentowy zysk z dodatkowych rdzeni

Fig. 23. On the left – minimum, average and maximum execution time for query in a MySQL database, on the right – gain percentage from the additional cores

Pomimo, że MySQL nie potrafi zrównoleglić procesu wykonania jednego zapytania, jest w stanie wykorzystać dodatkową moc obliczeniową pochodzącą z dodatkowych rdzeni. Każde zwiększenie wartości koligacji dla tego silnika prowadzi do wzrostu wydajności średnio o ok. 80%. Patrząc na lewy wykres rys. 23, można stwierdzić, że każde żądanie wykonywane jest w takim samym czasie.

4.2.3. FireBird 2.5.2

Zainstalowany typ serwera FireBird – SuperClassic – zgodnie z informacjami producenta w pełni korzysta z systemu wieloprocesorowego. Nietrudno zauważyć zależność czasu wykonania testu od liczby dostępnych rdzeni. Dzieje się tak za sprawą modelu zaimplementowanego w serwerze SuperClassic. Każde nowe połączenie to nowy wątek (stwierdzono to po wzroście liczby uchwytów w procesie serwera). Warto także zwrócić uwagę na niewielką różnicę pomiędzy maksymalnym a minimalnym czasem odpowiedzi.



Rys. 24. Po lewej – minimalny, średni i maksymalny czas wykonania zapytania w bazie Firebird, po prawej – procentowy zysk z dodatkowych rdzeni

Fig. 24. On the left – minimum, average and maximum execution time for query in a Firebird database, on the right – gain percentage from the additional cores

5. Podsumowanie

Przedstawione systemy baz danych nie wykorzystują w pełni możliwości współczesnych procesorów wielordzeniowych do przetwarzania pojedynczego zapytania. Jedynie baza MySQL potrafiła skorzystać z dwu rdzeni, jednak też w ograniczonym zakresie. Dla kontrastu, komercyjna baza danych, jaką jest MS SQL Serwer Enterprise, generowała plany wykonania zapytania, jawnie wykorzystując możliwości zrównoleglenia procesów obliczeniowych. Nie możemy jednak stwierdzić, na ile to zrównoleglenie miało wpływ na wydajność, ponieważ wykonanie takich testów znacznie wykraczało poza ramy tego porównania.

Zupełnie inaczej wyglądają testy obciążeniowe. Przedstawione wyniki pokazują, że systemy wieloprocesorowe potrafią zwiększyć wydajność wszystkich testowanych baz danych. Podstawowe miejsce, w którym odczuwalne jest zwiększenie wydajności przez dodatkowe rdzenie w procesorze, to przetwarzanie większej liczby żądań. Przeciętny przyrost wydajności to średnio 80% na każdy dodatkowy rdzeń. Dzieje się tak, ponieważ zaprezentowane bazy danych zostały zaprojektowane do pracy w systemach wielodostępnych, gdzie w tym samym czasie muszą przetwarzać wiele zapytań od różnych użytkowników.

BIBLIOGRAFIA

1. Aptekorz M., Szostek K., Młynarczuk M.: Możliwości akceleracji przestrzennych baz danych na podstawie procesorów kart graficznych oraz funkcji. Studia Informatica, Vol. 33, No. 2B (106), Gliwice 2012.
2. Bach M., Werner A., Duszeńko A.: Dobór struktur danych pod kątem optymalizacji przetwarzania analitycznego. Studia Informatica, Vol. 33, No. 2A (105), Gliwice 2012.
3. Jajeśnica Ł., Piórkowski A.: Wydajność złączeń i zagnieżdżeń dla schematów znormalizowanych i zdenormalizowanych. Studia Informatica, Vol. 31, No. 2A (89), Gliwice 2010.
4. http://en.wikipedia.org/wiki/File:SMP_-_Symmetric_Multiprocessor_System.svg.
5. http://www.p2d2.cz/files/p2d2-parquery.pdf.
6. http://dev.mysql.com/doc/refman/5.5/en/index.html (MySQL 5.5 Reference Manual).
7. http://www.postgresql.org/docs/9.2/interactive/index.html.
8. http://www.firebirdsql.org/en/reference-manuals/.
9. Smith G.: Wysoko wydajny PostgreSQL 9.0. Helion, Gliwice 2011.
10. http://www.slideshare.net/tomneko/firebird25-benchmarksenglish20091031.

Wpłynęło do Redakcji 16 stycznia 2013 r.

Abstract

The article describes the influence of the number of processor cores on the database performance.

Tests included the three most popular public domain database systems: PostgreSQL, MySQL and Firebird. Each database was tested by a series of varying complexity DML queries. Performance assessment was by measure the execution time of each DML query with a different number of available processor cores.

The first part of the test consisted of measuring the efficiency of processing a single query depending on the number of cores. In the case of PostgreSQL and MySQL databases there were shown queries executed plans. For comparison were also presented plans for the implementation of the same querires by commercial database Microsoft SQL Server Enterprise.

In the second part all tested databases were tested in simulated multiple end-users environment. A single query was repeated several times. As in first part, performance was rated by measure the execution time.

All tests were performed on a computer equipped with a Intel Core i5 quad-core processor running a 64-bit MS Windows 7 Professional.

Issues of control and allocation of processor cores available for each database engine was performed by Process Explorer software from Sysinternals.

To test scores were as much as possible dependant on number of running cores in processor the computer was equipped with 8 GB of RAM and a fast SSD.

Adresy

Marcin GRZESIAK: AGH Akademia Górniczo-Hutnicza, Wydział Geologii, Geofizyki i Ochrony Środowiska, Katedra Geoinformatyki i Informatyki Stosowanej, al. Mickiewicza 30, 30-059 Kraków, Polska, marcin.grzesiak@gmail.com.

Aleksander CIANCIARA: AGH Akademia Górniczo-Hutnicza, Wydział Geologii, Geofizyki i Ochrony Środowiska, Katedra Geoinformatyki i Informatyki Stosowanej, al. Mickiewicza 30, 30-059 Kraków, Polska, alexc@geol.agh.edu.pl.

Adam PIÓRKOWSKI: Akademia Górniczo-Hutnicza, Wydział Geologii, Geofizyki i Ochrony Środowiska, Katedra Geoinformatyki i Informatyki Stosowanej, al. Mickiewicza 30, 30-059, Kraków, Polska, pioro@agh.edu.pl.