JOURNAL
COMPUTER SCIENCES INSTITUTE

JCSI 16 (2020) 279–284
Received: 26 June 2020
Accepted: 6 July 2020

Comparison of MySQL, MSSQL, PostgreSQL, Oracle databases performance, including virtualization

Porównanie wydajności baz danych MySQL, MSSQL, PostgreSQL oraz Oracle z uwzględnieniem wirtualizacji

Rafał Kleweka*, Wojciech Truskowski*, Maria Skublewska-Paszkowska

Department of Computer Science, Lublin University of Technology, Nadbystrzycka 36B, 20-618 Lublin, Poland

Abstract
Oracle, MSSQL, MySQL and PostgreSQL are four of the most popular relational databases. They are often used in internet applications. This paper aims to compare the efficiency of these technologies in terms of speed using containerization with Docker. No publications that include that aspect were found among previous papers. After review of the literature, it was hypothesized that the Oracle engine would be the fastest. During the research, a series of experiments was carried out using the application, in which tests for measuring the time of instruction execution were implemented. Each query was measured 100 times and the first measurement was rejected. The obtained results confirmed the hypothesis about the superiority of the Oracle database. As in previous studies, it proved to be the fastest, also using containerization.

Keywords: virtualization; Docker; database performance

Streszczenie
Oracle, MSSQL, MySQL i PostgreSQL to cztery z najpopularniejszych relacyjnych baz danych. Są one często wykorzystywane w aplikacjach internetowych. Artykuł ma za cel porównanie efektywności tych technologii pod względem szybkości z wykorzystaniem konteneryzacji przy pomocy Docker. Wśród dotychczasowych publikacji nie znaleziono takich, które uwzględniałyby ten aspekt. Po przeglądzie literatury postawiono hipotezę, że silnik Oracle będzie najszybszy. Podczas badań przeprowadzono serię eksperymentów z użyciem aplikacji, w której zaimplementowane zostały testy do pomiaru czasu wykonania instrukcji. Każde zapytanie zostało zmierzone 100-krotnie, a pierwszy pomiar odrzucony. Uzyskane rezultaty potwierdziły hipotezę o przewadze bazy Oracle. Podobnie jak w dotychczasowych badaniach okazała się ona najszybsza, także z użyciem konteneryzacji.

Słowa kluczowe wirtualizacja; Docker; wydajność bazy danych

*Corresponding author
Email address: rafal.klewek@pollub.edu.pl (R. Klewek), wojciech.truskowski@pollub.edu.pl (W. Truskowski)

1. Wstęp

Bazy danych umożliwiają przechowywanie informacji wytworzonych przez działające systemy informatyczne w sposób ustandaryzowany i trwały. Stanowią nieodłączny element zdecydowanej większości wykorzystywanych na świecie aplikacji internetowych oraz programów komputerowych. Bez nich nie byłaby możliwa realizacja podstawowych zadań, których oczekuje się od systemów takich jak magazynowanie, przetwarzanie oraz zarządzanie danymi. Obecnie na rynku relacyjnych systemów zarządzania bazami danych jednymi z najpopularniejszych są cztery rozwiązania w następującej kolejności: Oracle, MySQL, Microsoft SQL Server oraz PostgreSQL [1].

Wirtualizacja jest pojęciem, które w dziedzinie technologii informatycznych odnosi się do procesu tworzenia wirtualnych zasobów, takich jak: całe platformy sprzętowe, pamięć operacyjna, nośniki danych, czy zasoby sieciowe. W roku 2013 na rynku pojawiło się oprogramowanie Docker, które umożliwia tworzenie izolowanych środowisk uruchomieniowych dla aplikacji występujących pod nazwą kontenerów, gdzie umieszczone są same aplikacje oraz biblioteki i zależności konieczne do ich poprawnego działania [2]. Technologię tę można wykorzystać do uruchomienia instancji środowisk bazodanowych na podstawie oficjalnych obrazów udostępnionych przez twórców. Po zainstalowaniu oprogramowania Docker instancjonowanie kolejnych baz danych sprowadza się do wskazania obrazu, który użytkownik chce użyć oraz uruchomienia nowego kontenera. Nie jest potrzebna konfiguracja komputera związana z konkretnym silnikiem bazodanowym, instalacja dodatkowego oprogramowania, sterowników oraz wymaganych bibliotek. Oprogramowanie to umożliwia także integrację z bardzo dynamicznie rozwijającymi się w ostatnich latach rozwiązaniami chmurowymi. Wszystkie te zalety sprawiają, że coraz więcej firm decyduje się na zastąpienie lokalnych instalacji środowisk bazodanowych na rzecz technologii Docker i wykorzystania kontenerów.

W nawiązaniu do trendów opisanych powyżej niniejszy artykuł podejmuje tematykę porównania wydajności jednych z najpopularniejszych na rynku relacyjnych rozwiązań baz danych z uwzględnieniem konteneryzacji przy pomocy oprogramowania Docker.

2. Przegląd literatury

Autorzy w artykule Performance Evaluation MySQL InnoDB and Microsoft SQL Server 2012 for Decision Support Environments [3] porównują wydajność Microsoft SQL Server 2012 z MySQL InnoDB. Autorzy porównywali czas zapytań pobierających dane na zestawach danych o wielkości 1GB, 3GB, 6GB, 12GB i 24GB. Bazą danych do analizy była hurtowania składająca się z jednej tabeli faktu i czterech tabel wymiarów. Z artykułu wynika, że Microsoft SQL Server 2012 osiąga większą wydajność niż MySQL InnoDB. Wyniki badań pokazują, że wraz ze wzrostem liczby danych spadała będzie wydajność bazy. Można wywnioskować, że Microsoft SQL Server 2012 nadaje się do operacji na małych oraz średnich zestawach danych, a MySQL InnoDB na małych zestawach danych.

Autorzy w artykule Comparison of query performance in relational a non-relation databases [4] porównują wydajność relacyjnych i nierelacyjnych baz danych. Do oceny autorzy wybrali relacyjne bazy danych Oracle, MySQL oraz MSSQL. Jako nierelacyjne technologie zostały wybrane Redis, Mongo, GraphQl i Cassandra. Autorzy mierzyli czas wykonania operacji select, insert, update oraz delete. Testy były przeprowadzane na zestawach danych liczących 10 000 i 100 000 rekordów. Wyniki badań pokazują, że nierelacyjne bazy danych są bardziej wydajne od technologii relacyjnych. Stosunek wydajności rozwiązań nierelacyjnych do baz relacyjnych wyniósł 1:3 dla operacji select, 1:15 dla operacji insert, 1:9 dla instrukcji update i 1:6 dla operacji delete. Z poddanych analizie relacyjnych silników najwydajniejszy był MSSQL. MySQL miał najniższą wydajność. W porównaniu tylko nierelacyjnych baz danych najwydajniejszy był Mongo, najgorzej wypadł Redis i Cassandra.

Autorzy publikacji An Introduction to Docker and Analysis of its Performance [5] szczegółowo opisali elementy oprogramowania Docker tj. klient, serwer, obrazy, rejestr i kontenery. W artykule autorzy przedstawili porównanie technologii Docker oraz maszyny wirtualnej KVM. Analiza pokazuje, że w przypadku Dockera można łatwiej zarządzać zasobami, szybkość obliczeniowa jest większa, a uruchomienie kontenera jest znacznie szybsze niż uruchomienie maszyny wirtualnej. Maszyna wirtualna natomiast jest lepszym wyborem, gdy głównym kryterium jest poziom izolacji procesu.

Na konferencji ICACEA Ann Joy przedstawiła porównanie pomiędzy kontenerami linuksowymi [6], a maszynami wirtualnymi. Autorka wskazała, że kontenery są bardziej wydajne i lepiej przystosowane do skalowania. Z przeprowadzonych badań wynika, że kontenery skalują się 22 razy szybciej niż maszyny wirtualne. Ze względu na lepsze gospodarowanie zasobami i skalowalność rozwiązania wykorzystujące konteneryzację zmniejszają wykorzystanie zasobów. Autorka wskazuje, iż maszyny wirtualne posiadają lepsze zabezpieczenia, przez co lepiej nadają się do wykorzystania w przypadku serwisów wymagających dużej izolacji.

W artykule Performance analysis of selected database systems: MySQL, MS SQL, PostgerSQL in the context of web applications [7] autor analizował wydajność trzech systemów bazodanowych MySQL, MSSQL i PostgreSQL. Do badań twórca pracy wykorzystał aplikację internetową. Eksperymenty zostały przeprowadzone przy pomocy oprogramowania Apache JMeter, które do połączeń z bazą danych używa interfejsu JDBC. Autor porównywał średni czas zapytań odczytu, dodawania, modyfikacji oraz usuwania rekordów. Wyliczona została także mediana oraz odchylenie standardowe dla każdego scenariusza badawczego. Z badań wynika, że najwydajniejszą bazą danych jest PostgreSQL. Najgorszą wydajność zaobserwowano na silniku MySQL. Z artykułu można wnioskować, że baza MSSQL w mniejszym stopniu korzysta z pamięci podręcznej niż MySQL oraz PostgreSQL.

W artykule Comparative analysis of databases working under the control of Windows system [8] autorzy analizowali wydajność oraz wykorzystywanie zasobów trzech systemów bazodanowych MySQL, PostgreSQL oraz Firebird na systemie operacyjnym Windows 10 Pro 64-bit. Analiza wydajności polegała na wykonaniu 10 powtórzeń każdego scenariusza, a wynikiem była średnia wartość z otrzymanych wyników. Z przeprowadzonych badań wynika, że spośród wybranych systemów bazodanowych na systemie Windows najwydajniejszy jest MySQL oraz najmniej obciąża on zasoby dyskowe. Baza danych Firebird najmniej obciąża procesor.

Przegląd artykułów naukowych pokazuje, że istnieje ciągłe zainteresowanie badaniami baz danych oraz wirtualizacją, również z wykorzystaniem oprogramowania Docker. Po przeglądzie literatury autorzy postawili hipotezę, iż wydajność relacyjnych baz danych na małych zbiorach danych jest bardzo zbliżona, ale wraz ze wzrostem liczby danych wydajność bazy Oracle będzie zauważalnie wyższa niż pozostałych. Jednocześnie nie napotkano badań, w których porównania dokonano przy uwzględnieniu wirtualizacji, w związku z czym w eksperymentach uwzględniony zostanie aspekt, którego nie obejmowała żadna z omawianych prac.

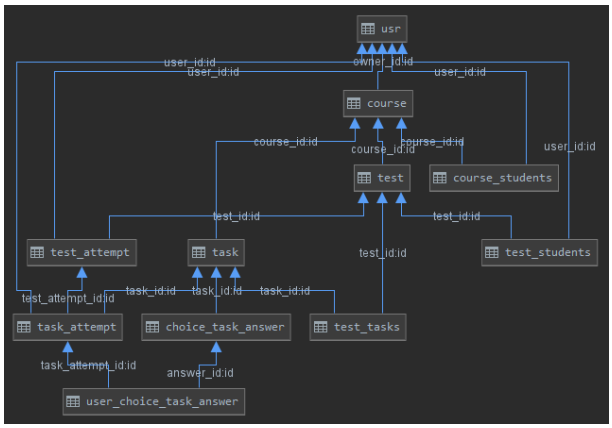
3. Aplikacja testowa

Do przeprowadzania badań stworzona została aplikacja webowa, której podstawowym założeniem jest możliwość tworzenia oraz rozwiązywania testów składających się z pytań wielokrotnego wyboru. Część serwerowa aplikacji została zaimplementowana z użyciem szkieletu aplikacyjnego Symfony [9] dla języka PHP, a do stworzenia interfejsu graficznego użytkownika posłużono się językiem JavaScript, wykorzystując technologię Vue.js [10]. Wśród głównych funkcjonalności należy wymienić:

- możliwość rejestracji oraz logowania do utworzonych kont,
- ograniczenie dostępu do wszystkich zakładek dla niezalogowanych użytkowników,
- trzy poziomy uprawnień w systemie, określone rolami: administrator, egzaminator, kursant,

- tworzenie kursów wewnątrz których można potem tworzyć zadania, gdzie dostępne do określenia są: tytuł, treść, maksymalna liczba punktów do uzyskania oraz lista odpowiedzi z zaznaczeniem, które są poprawne,
- definiowanie testów, w których skład wchodzą utworzone uprzednio zadania,
- wyszukiwanie, rozwiązywanie oraz podgląd wyników dla uzyskanych zestawów testowych.

Do przechowywania danych wygenerowanych podczas korzystania z aplikacji wykorzystywana jest baza danych o schemacie widocznym na rysunku 1.



Rysunek 1: Schemat bazy danych

Oprócz funkcjonalności związanych z interfejsem użytkownika zostały zaimplementowane specjalne klasy operacji służące do wykonywania pojedynczych operacji wymagających połączenia z bazą danych. Klasy te napisane zostały w taki sposób, aby po ich użyciu dostępny był czas wykonania realizowanej instrukcji bazodanowej. Zostały one wykorzystane do badań, gdzie przy użyciu testów jednostkowych zostały wywołane 100-krotnie, a czasy zapytań zapisywane były do plików w formacie csv.

4. Metoda badawcza

Eksperyment polegał na uruchomieniu testów jednostkowych udostępnianych przez dedykowaną aplikację, która realizowała operacje wymagające komunikacji z bazą danych. Do badań użyto baz:

- MySQL ver. 8.0.17,
- Microsoft SQL Server ver. 15.00.4033,
- PostgreSQL ver. 12.3,
- Oracle Database 18c Express Edition Release 18.0.0.0.0 – Production.

Testy były uruchamiane na systemie Windows 10 Education x64 kompilacja 18362 z użyciem Hyper-V i kontenerów linuksowych, a uruchomienie środowisk bazodanowych w kontenerach zostało uzyskane przy pomocy platformy Docker w wersji 19.03.8. Do zachowania stanu bazy katalogi robocze zostały podmontowane z systemu plików systemu macierzystego. Wszystkie obrazy baz danych uruchomione zostały z domyślnymi ustawieniami dostarczonymi przez twórców. Port na którym działała baza w kontenerze odpowiadał portowi na maszynie macierzystej. Podczas każdej próby uruchomiona była tylko jedna instancja kontenera. Schemat bazy danych był tworzony przy zastosowaniu *collation* z alfabetem polskim, niewrażliwy na wielkość znaków oraz niewrażliwy na akcent. W tabeli 1 została przedstawiona specyfikacja techniczna maszyny, która posłużyła do wykonania badań wydajności. Na czas prowadzonych badań środowisko platformy Docker otrzymało zasoby widoczne w tabeli 2.

Tabela 1: Specyfikacja urządzenia testowego

Procesor	Procesor Intel(R) Core(TM) i7-8750H CPU @ 2.20GHz, 2201 MHz, Rdzenie: 6, Procesory logiczne: 12
Dysk	Model SPCC Solid State Disk 500GB
RAM	16GB 2400MHz

Tabela 2: Zasoby platformy Docker

Liczba rdzeni procesora	2
RAM	3,5 GB
Swap	1 GB

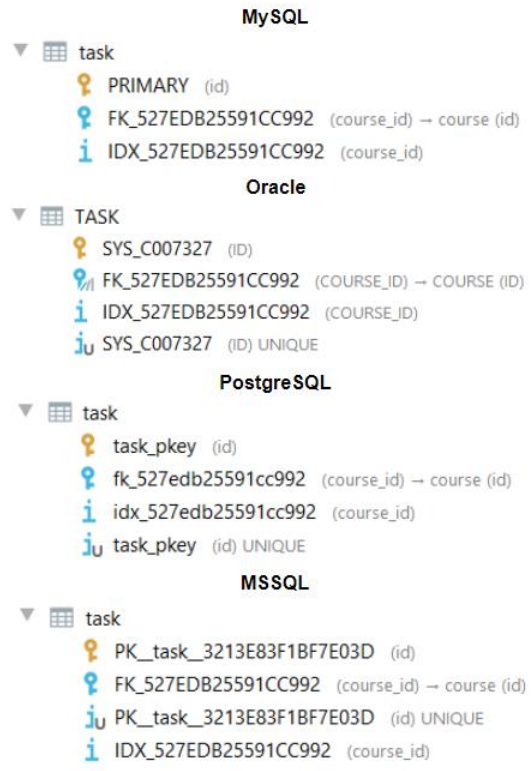
Każda operacja wykonana została 100-krotnie w celu uzyskania jak najbardziej rzetelnych rezultatów. Przy porównywaniu wyników odrzucany był pierwszy rezultat, który ze względu na brak zbudowanego planu zapytania trwał zauważalnie dłużej niż kolejne. Zmierzono czasy wykonywanych instrukcji bazodanowych dla pojedynczych operacji wyszukiwania danych. Podczas pobierania rekordów badany był wpływ różnych warunków zawężających oraz funkcji. Te same pomiary zostały przeprowadzone na każdym z porównywanych rozwiązań relacyjnych baz danych: MySQL, MSSQL, PostgreSQL oraz Oracle. Każdy scenariusz został wykonany dla trzech zestawów danych różniących się liczbą rekordów w poszczególnych tabelach. Dla wszystkich technologii bazodanowych wygenerowane automatycznie informacje, którymi zostały wypełnione, były identyczne. Tabela 3 zawiera zestawienie liczby rekordów we wszystkich zestawach testowych:

Tabela 3: Zestawianie liczby rekordów dla zestawów testowych

Tabela/Nr zestawu	1	2	3
course	5 000	25 000	100 000
task	50 000	250 000	1 000 000
usr	30 000	150 000	600 000
test	10 000	50 000	200 000
course_student	25 000	125 000	500 000

Obrazy baz MySQL, MSSQL i PostgreSQL zostały uruchomione przy użyciu obrazów udostępnionych przez producenta z publicznego repozytorium *DockerHub*. Do budowy obrazu bazy Oracle wykorzystana została instrukcja z artykułu *Oracle magazine* [12]. Schemat bazy danych dla każdego silnika był analogiczny. Utworzona została taka sama liczba kluczy obcych oraz indeksów dla odpowiadających kolumn. Efekt uzyskano dzięki zastosowaniu do tworzenia struktur mechanizmów udostępnianych przez szkielet aplika-

cyjny Symfony. Struktura bazodanowa jest generowana, zgodnie ze wzorcem definiowanym przy pomocy klas zwanych *encjami*. Są to pliki zawierające pojedyncze klasy PHP z dodatkowymi adnotacjami opisującymi atrybuty dla tworzonej tabeli. Na rysunku 2 zostały zaprezentowane indeksy i klucze utworzone dla tabeli *Task*.



Rysunek 2: Indeksy oraz klucze dla tabeli *Task*

5. Scenariusze badawcze

Pierwszy scenariusz polegał na wykonaniu pojedynczego zapytania wybierającego dane bez żadnych zawężeń zbioru wynikowego. Zapytanie badane w czasie uruchamiania tego testu jest widoczne na listingu 1.

Listing 1: Kod instrukcji SQL wykonywanej w pierwszym teście

```
SELECT t0_.id               AS id_0,
       t0_.name             AS name_1,
       t0_.number_of_points AS number_of_points_2,
       t0_.content          AS content_3,
       t0_.creation_date    AS creation_date_4,
       t0_.enabled          AS enabled_5,
       t0_.type             AS type_6,
       t0_.image_name       AS image_name_7,
       t0_.image_size       AS image_size_8,
       t0_.updated_at       AS updated_at_9
FROM task t0_
ORDER BY t0_.name ASC
LIMIT 20
```

Dla kolejnego scenariusza pod uwagę brana była szybkość zapytania wyszukującego korzystającego z klauzuli zawężającej *WHERE*. Treść polecenia realizowanego przez bazę służy do pobrania rekordów z tabeli *Task* zawężonych o wartość z kolumny *name*. Wygenerowane zapytanie zostało przedstawione na listingu 2.

Listing 2: Zapytanie używane dla scenariusza drugiego

```
SELECT t0_.id               AS id_0,
       t0_.name             AS name_1,
       t0_.number_of_points AS number_of_points_2,
       t0_.content          AS content_3,
       t0_.creation_date    AS creation_date_4,
       t0_.enabled          AS enabled_5,
       t0_.type             AS type_6,
       t0_.image_name       AS image_name_7,
       t0_.image_size       AS image_size_8,
       t0_.updated_at       AS updated_at_9
FROM task t0_
WHERE t0_.id = 1
ORDER BY t0_.id ASC
```

Scenariusz trzeci użyty został do sprawdzenia wydajności porównywanych rozwiązań relacyjnych baz danych w przypadku wyszukiwania danych z wykorzystaniem podzapytania skorelowanego. Wyszukiwane są kursy oraz nazwy użytkowników tworzących, gdzie dla drugiej kolumny do wybrania wartości użyte zostało podzapytanie skorelowane. Treść wynikowej instrukcji SQL została zamieszczona na listingu 3.

Listing 3: Select z użyciem podzapytania

```
SELECT c0_.name AS name_0,
       (
           SELECT u1_.username
           FROM usr u1_
           WHERE u1_.id = c0_.owner_id
       ) AS sclr_1
FROM course c0_
ORDER BY c0_.name ASC
LIMIT 20
```

Kolejny test pozwolił na mierzenie szybkości realizacji operacji wybierania danych z wielokrotnym łączeniem tabel oraz klauzulą grupującą. Zliczana była liczba zadań dla poszczególnych testów w systemie, a pełna wersja zapytania przedstawiona jest na listingu 4.

Listing 4: Ciało zapytania z wielokrotnym łączeniem tabel oraz klauzulą grupującą

```
SELECT t0_.id AS id_0, t0_.name AS name_1, count(t1_.id) AS sclr_2
FROM test t0_
     INNER JOIN test_tasks t2_ ON (t0_.id = t2_.test_id)
     INNER JOIN task t1_ ON (t1_.id = t2_.task_id)
     INNER JOIN course c3_ ON (c3_.id = t0_.course_id)
GROUP BY t0_.id, t0_.name
LIMIT 20
```

6. Analiza wyników

W pierwszym badanym scenariuszu sprawdzany był czas wykonania instrukcji *SELECT* z pojedynczej tabeli *Task* bez żadnych dodatkowych warunków ograniczających. Rezultat miał być posortowany alfabetycznie według wartości kolumny *name*. Przyglądając się wynikom prób zauważyć można, że zdecydowanie najlepiej

pod względem wydajności dla instrukcji wybierania danych z pojedynczej tabeli wypadło rozwiązanie firmy Oracle. Czas wykonania zapytania był bliski jednej milisekundzie dla wszystkich zbiorów danych, gdzie pozostałe rozwiązania bazodanowe uzyskały wyniki na poziomie kilkudziesięciu lub kilkuset ms. Na podstawie minimalnych czasów zauważyć można, iż najgorzej w tej próbie wypadły technologie MySQL oraz MSSQL, które dla największego zestawu danych uzyskały średnie czasy około 700 ms, gdzie dla identycznego zestawu rekordów system PostgreSQL osiągnął wynik 289 ms, a Oracle zaledwie 1,19 ms. Ogromna różnica przemawiająca na korzyść ostatniego badanego rozwiązania, co zostało zobrazowane na tabeli 4.

Tabela 4: Średni czas wykonania zapytań dla scenariusza drugiego

Baza/Nr zestawu	Średni czas dla 1 zestawu [ms]	Średni czas dla 2 zestawu [ms]	Średni czas dla 3 zestawu [ms]
MySSQL	32,5106	156,774	694,385
MSSQL	35,3669	165,584	712,929
PostgrSQL	26,033	73,3064	289,373
Oracle	1,2326	1,2168	1,1903

W następnym scenariuszu była sprawdzana wydajność baz danych podczas wykonania zapytania wybierającego dane z pojedynczej tabeli *Task* przy ograniczeniu na kolumnie, dla której nie było założonego indeksu. Do tego celu wykorzystana została instrukcją języka SQL reprezentowana przez klauzulę *WHERE*. Rozpatrując wyniki zaprezentowane na tabeli 5 można zauważyć, że najgorsze wyniki uzyskał MySQL. Rozmiar bazy danych nie wpłynął na wydajność technologii firmy Oracle. Średni czas zapytania na każdym zestawie rekordów dla bazy danych Oracle wynosił ok. 1,3 ms. MSSQL i PostgreSQL uzyskały zbliżoną wydajność na poziomie 80-90 milisekund. Wraz ze wzrostem liczby rekordów różnica wydajności pomiędzy MSSQL i PostgreSQL zauważalnie się zwiększała. Przy trzecim zestawie danych PostgreSQL osiągnął lepszy czas o 16,799 ms od bazy firmy Microsoft.

Tabela 5: Średni czas wykonania zapytań dla scenariusza drugiego

Baza/Nr zestawu	Średni czas dla 1 zestawu [ms]	Średni czas dla 2 zestawu [ms]	Średni czas dla 3 zestawu [ms]
MySSQL	8,931	163,688	734,137
MSSQL	10,303	27,668	92,925
PostgrSQL	8,9309	24,867	76,126
Oracle	1,282	1,279	1,316

W trzecim scenariuszu badaniu poddana została instrukcja języka SQL wykorzystująca podzapytanie skorelowane. Wykorzystane one zostało do uzyskania listy kursów w systemie wraz z nazwą użytkownika tworzącego. Z przeprowadzonych testów wynika, iż dla pierwszego zestawu danych o najmniejszej liczbie rekordów w tabelach średnie czasy były zbliżone dla wszystkich baz, z wyjątkiem Oracle, które już w tej próbie uzyskało wynik ponad dwukrotnie gorszy niż pozostałe. Dla kolejnych zbiorów testowych tendencja utrzymała się i najdłużej zapytanie wykonywane było dla bazy danych Oracle. W trzeciej próbie średni czas wyniósł 199,53 ms. Różnica pomiędzy pozostałymi rozwiązaniami relacyjnych baz danych najlepiej widoczna była podczas testów dla najliczniejszego zestawu danych testowych. Z analizy wyników dla bazy wypełnionej największą liczbą rekordów wynika, że najlepiej poradził sobie MSSQL ze średnim czasem realizacji zapytania równym 29,2807 ms, następnie PostgreSQL z wynikiem 50,2726 ms. Trzeci był MySQL z rezultatem na poziomie 72,3806 ms. Średnie czasy uzyskane podczas wykonywania instrukcji z podzapytaniem skorelowanym dla poszczególnych silników bazodanowych zostały zaprezentowane na tabeli 6.

Tabela 6: Średni czas wykonania zapytań dla scenariusza trzeciego

Baza/Nr zestawu	Średni czas dla 1 zestawu [ms]	Średni czas dla 2 zestawu [ms]	Średni czas dla 3 zestawu [ms]
MySSQL	5,1094	19,4097	72,3806
MSSQL	5,522	14,4066	29,2807
PostgrSQL	5,3445	14,5357	50,2726
Oracle	12,8493	50,1	199,527

Scenariusz czwarty dotyczył porównania baz w przypadku wykonywania zapytania korzystającego z wielu złączeń pomiędzy tabelami, grupowania oraz zliczania. Klauzula *GROUP BY* pozwala na grupowanie zbioru wynikowego na podstawie identycznej wartości we wskazanej kolumnie. Testowane zapytania pozwala uzyskać liczbę zadań w poszczególnych testach. Z analizy danych wynika, że zdecydowanie najgorzej poradził sobie MySQL Jest to widoczne na tabeli 7, gdzie zastosowano skalę logarytmiczną. Dla największego zestawu danych średni czas wykonania zapytania wyniósł blisko 6 sekund, gdzie dla identycznego zbioru testowego Oracle osiągnął czas równy 245,7870 ms. Widać tutaj, że silnik MySQL jest złym wyborem, jeśli chodzi o operacje grupowania danych przy wielokrotnych złączeniach pomiędzy tabelami. Dla baz PostgreSQL oraz MSSQL wyniki były zbliżone, niezależnie od liczby rekordów w systemie. Najlepszym rozwiązaniem bazodanowym podczas tego badania okazał się MSSQL.

Tabela 7: Średni czas wykonania zapytań dla scenariusza czwartego

Baza/Nr zestawu	Średni czas dla 1 zestawu [ms]	Średni czas dla 2 zestawu [ms]	Średni czas dla 3 zestawu [ms]
MySSQL	147,634	1191,55	5702,37
MSSQL	1,5111	1,495	1,5836
PostgrSQL	3,2147	3,2906	3,2823
Oracle	13,8178	60,9568	245,787

7. Wnioski

W artykule przeprowadzone zostały badania wydajności czterech najpopularniejszych na rynku relacyjnych baz danych, czyli Oracle, MSSQL, MySQL oraz PostgreSQL. Wszystkie technologie bazodanowe uruchomione zostały w środowisku wirtualnym z zastosowaniem oprogramowanie Docker. Po przeglądzie literatury podejmującej temat porównania wydajności baz danych uruchamianych w wyniku standardowej instalacji postawiona została teza, iż podczas zastosowania konteneryzacji przy pomocy oprogramowania Docker najlepiej poradzą sobie silniki Oracle oraz MSSQL.

Udało się wykonać wszystkie zaplanowane scenariusze testowe, a uzyskane rezultaty pozwoliły na wyciągnięcie następujących wniosków:

- podczas operacji wyszukiwania danych bez użycia żadnych warunków zawężających najszybsza okazała się baza Oracle. Niezależnie od rozmiaru tabeli, dla której wykonywane było zapytanie średni czas wykonania wyniósł około 1,2 ms, co zostało przedstawione na tabeli 4. Analiza wyników pozwoliła zaobserwować, że ta sama operacja na innych silnikach trwała w przypadku trzeciego zestawu testowego kilkaset razy dłużej. Najgorzej w tej próbie wypadł system bazodanowy MSSQL
- przy teście zapytania SQL korzystającego z klauzuli WHERE ponownie bezkonkurencyjna okazała się baza Oracle, niezależnie od liczby rekordów. Dla pozostałych technologii zwiększony rozmiar danych powodował wydłużenie czasu wykonania. Najgorzej poradziła sobie baza MySQL, co widać na tabeli 5,
- dla testowanych zapytań bazodanowych, system Oracle najgorzej poradził sobie w przypadku instrukcji SQL, w której użyto podzapytania skorelowanego, co pokazuje tabela 6. Średni czas wykonania był najgorszy dla wszystkich badanych zbiorów danych. W teście tym największą efektywność osiągnął MSSQL, co jest widoczne zwłaszcza dla ostatniego zestawu testowego, gdzie średni wynik wyniósł 29,28 ms,
- analiza rezultatów badań dla złożonego zapytania wykorzystującego wielokrotne złączenia, grupowanie oraz zliczanie rekordów pozwoliła wykazać, że w przypadku tego typu operacji najlepszym okazał się silnik MSSQL. Co ciekawe, średni czas wykonania był zbliżony niezależnie od liczby rekordów w tabelach. Badanie to pozwoliło także zaobserwować, że baza MySQL źle radzi sobie w przypadku zapytań, gdzie zastosowano wielokrotne złączenia. Minimalny czas dla najliczniejszego zbioru danych wyniósł ponad 5 sekund, gdzie najszybsza baza osiągnęła rezultat na poziomie 1,5836 ms, co widać na tabeli 7.

Uzyskane wyniki pokazują, że podczas korzystania z relacyjnych baz danych w środowisku wirtualnym uruchomionym przy pomocy oprogramowania Docker, w większości przypadków najbardziej wydajnym okazała się baza Oracle. Warto również zauważyć, że do testów wykorzystana została darmowa wersja Express, gdzie dla edycji komercyjnej rezultaty mogłyby być jeszcze lepsze. Uzyskane rezultaty pokazały, że hipoteza postawiona po przeglądzie literatury była zasadna. Potwierdziło się, że czas wyszukiwania danych rośnie wraz z wielkością przeszukiwanego zbioru rekordów w bazie, a Oracle radzi sobie najlepiej w większości sprawdzanych scenariuszy.

Literatura

[1] G. Eason, B. Noble, I. N. Sneddon, On certain integrals of Lipschitz-Hankel type involving products of Bessel functions, Phil. Trans. Roy. Soc. London A247 (1955) 529–551.

[2] Ranking najpopularniejszych baz danych, https://pypl.github.io/DB.html, [24.06.2020]

[3] Czym jest Docker, https://docs.docker.com/get-started/overview/, [24.06.2020]

[4] R. Almeida, P. Furtado, J. Bernardino, Performance Evaluation MySQL InnoDB and Microsoft SQL Server 2012 for Decision Support Environments, Proceedings of the Eighth International C* Conference on Computer Science & Software Engineering - C3S2E '15. (2008).

[5] R. Čerešňák, M. Kvet, Comparison of query performance in relational a non-relation databases, Transportation Research Procedia. 40 (2019) 170–177.

[6] M. Yasir, A Review on Introduction to Docker and its Features, International Journal of Advanced Research in Computer Science and Software Engineering. 8 (2018) 12.

[7] A.M. Joy, Performance comparison between Linux containers and virtual machines, 2015 International Conference on Advances in Computer Engineering and Applications. (2015).

[8] K. Lachewicz, Performance analysis of selected database systems: MySQL, MS SQL, PostgerSQL in the context of web applications. Journal of Computer Sciences Institute. 14, (Mar. 2020), 94-100.

[9] S. Stets, G. Kozieł, Comparative analysis of databases working under the control of Windows system, Journal of Computer Sciences Institute. 13 (2019) 298–301.

[10] Salehi, S. 2016. Mastering symfony. Packt Publishing Limited.

[11] Czym jest Vue.js, https://vuejs.org/v2/guide/, [24.06.2020].

[12] Tworzenie kontenera dla bazy danych Oracle, https://blogs.oracle.com/oraclemagazine/deliver-oracle-database-18c-express-edition-in-containers [24.06.2020].