ZAAWANSOWANE TECHNOLOGIE BAZ DANYCH PROJEKT

Członkowie zespołu projektowego: Paweł Malec, Dominik Kulikowski

Temat projektu: Porównanie MongoDB oraz MySQL pod względem wydajności

1. Wprowadzenie

Technologia baz danych obejmuje dwa główne rodzaje: SQL (relacyjne bazy danych) oraz NoSQL (nierelacyjne bazy danych).

Główną cechą relacyjnych baz danych jest ich wysoce ustrukturyzowana natura, gdzie dane są przechowywane w wielu tabelach, w wierszach i kolumnach. Kolumny zawierają atrybuty, podczas gdy wiersze zawierają wpisy. Wzajemne powiązania między tabelami są określone przez schemat, który jest zaprojektowany w taki sposób, aby utrzymać tabele zsynchronizowane i zminimalizować redundancję danych. SQL jest używany do projektowania relacyjnych baz danych (RDBMS), a jego nazwa pochodzi od terminu 'Structured Query Language'. Jest jak skrypt po stronie serwera do wykonywania zapytań w celu pobierania danych lub ich edycji (aktualizacja, usuwanie lub tworzenie nowych rekordów). MySQL jest najczęściej używaną bazą danych open source, na której SQL wykonuje swoje zapytania.

Głównym atrybutem nierelacyjnych baz danych (NoSQL) jest elastyczność, którą oferują w zakresie szybkości i skalowalności. Osiąga się je poprzez przechowywanie danych w sposób zorientowany na dokumenty, gdzie wszelkiego rodzaju powiązane dane są przechowywane razem, bez podziału na kategorie. Jest to metoda pozbawiona schematów i jest znana jako 'Not Only SQL'. Ponieważ dane są przechowywane masowo, wymagają większej mocy obliczeniowej w porównaniu do danych ustrukturyzowanych. MongoDB jest najbardziej rozpowszechnioną nierelacyjną bazą danych typu open source (darmową). MongoDB wywodzi się swoją nazwą od słowa 'Mongo', które oznacza duży lub ogromny. Uzywa ona dokumentów typu JSON (Java Script Object Notation) do przechowywania danych. Ważną własnością MongoDB jest to, że przechowywanie danych nie wymaga odwzorowania obiektowo-relacyjnego. Ze względu na niestrukturalny format przechowywania danych, Mongo jest używany do manipulowania ogromnymi ilościami danych i jest szeroko stosowany w dziedzinie Big Data. Firmy, które przechodzą na przechowywanie danych w chmurze (dane rozproszone w wielu centrach danych) preferują MongoDB, ze względu na następujące cechy.

2. Przygotowanie i wykonywanie zapytań

- Tworzenie tabeli lub kolekcji

MySQL posiada zdefiniowany schemat bazy danych. W związku z tym każda kolumna w tabeli będzie miała określony typ danych, a wszystkie wiersze w tabeli będą miały określoną strukturę. Podczas tworzenia tabeli użytkownik powinien określić kolumny oraz typy danych zgodnie ze schematem. Klucz główny powinien być również określony domyślnie, a użytkownik powinien wprowadzić odpowiednie wartości. W przeciwieństwie do tego, MongoDB przechowuje pliki w formacie JSON, który jest w formie dokumentu mieszczącego w sobie różne struktury. Ten dokument składaujący się z danych jest określany jako 'Kolekcja' i może posiadać własną, unikalną strukturę z polami zawierającymi dowolny typ danych. Dane, które mają być przechowywane w tej kolekcji są w formie tablic. Kolekcja nie musi być koniecznie częścią jednego schematu. MongoDB obsługuje metodę dynamicznego schematu,

ułatwiając aktualizację danych w jej obrębie. Kolekcja może zostać utworzona za pomocą pierwszego polecenia ‘insertOne()’ lub ‘insertMany()’. Typy danych każdego dokumentu nie muszą być wstępnie określone. MongoDB dodaje klucz główny ‘_id’ automatycznie, jeśli użytkownik go nie określi.

- Zmiana zapytania do tabeli

W MySQL, tabela może być zmieniona w formie strukturalnej za pomocą poleceń ‘ADD’ lub ‘DROP COLUMN’. Podczas dodawania kolumny, istotne jest określenie typu danych. W MongoDB nie jest wymagana żadna zmiana w strukturze na poziomie kolekcji, ponieważ składa się ona z danych dowolnego typu. Ale na poziomie dokumentu, pola mogą być dodawane lub usuwane z dokumentu za pomocą operacji ‘$set’ i ‘$unset’ odpowiednio wraz z funkcją ‘updateMany()’. Ponadto, dowolna ilość danych może być przechowywana w dowolnym dokumencie i nie jest konieczne, aby wszystkie dokumenty posiadały jeden format.

- Czytanie rekordów

W MySQL, użytkownik musi określić kolumny, które mają być za pomocą instrukcji ‘Select’ wybrane w zapytaniu. W MongoDB jest instrukcja ‘find()’. Metoda ta zawsze wypisuje id obiektu, bez względu na to, czy jest on określony w zapytaniu czy nie. Jeśli użytkownik chciałby go wykluczyć, musi on zostać określony w zapytaniu. Jest to również znane jako ‘proejction’.

- Indeksowanie

Obie bazy danych używają indeksów do szybkiego odnajdowania danych. Jeśli jednak indeksy nie są zdefiniowane, w MySQL silnik bazy danych analizuje całą tabelę w celu odnalezienia wszystkich odpowiednich wierszy. W MongoDB wszystkie dokumenty w kolekcji są badane, aby pobrać dokumenty, które są wywoływane przez zapytanie.

- Łączenie tabel

Największą zaletą MySQL jest możliwość łączenia rożnych tabel. Pomaga to w zapytaniach z różnych tabel. MongoDB nie obsługuje operacji JOIN, ale obsługuje wielowymiarowe typy daych, takie jak inne dokumenty i tablice. Ponieważ dane w MongoDB są zdenormalizowane, łączenie tabel może nie być wymagane.

- Atomicity – jedna z własności transakcji ACID

MySQL obsługuje atomowość, to znaczy może on wysyłać zapytania do pojedyńczych transakcji. Struktura relacyjnej bazy danych obsługuje ACID, które oznaczają niepodzielność, spójność oraz izolację transakcji, w przeciwieństwie do MongoDB. Pomaga to zaktualizować pojedyńczy wiersz w ciągu tabeli RDBMS szybciej i wydajniej. Jak sugeruje nazwa ACID, MySQL działa poprzez izolację danych do struktur atomowych, które można aktualizować i odpytywać niezależnie. Zachowuje spójność pomiędzy bazą danych przypisując konkretny schemat do każdej wprowadzonej wartości.

3. Wydajność

- Szybkość

Dane w MySQL mają normalną formę. Tak więc, aby kompilować dane z różnych źródeł, złożone zapytania i JOIN mogą być wymagane. MongoDB jest zdenormalizowane i w związku

z tym umożliwia użytkownikowi uzyskanie odpowiednich informacji za pomocą jednego zapytania. Może być więc szybszy w wykonywaniu zapytań.

- Skalowalność

MySQL ma problemy ze skalowaniem, ponieważ dane są przechowywane w określonych tabelach w zdefiniowanych formatach. Z drugiej strony MongoDB obsługuje 'auto-sharding'. Oznacza to, że skalowanie w górę można wykonać w wielu rozproszonych centrach danych wraz ze wzrostem ilości danych bez wpływu na aplikację i przestojami.

- Redundancja

W MySQL dane są znormalizowane. Oznacza to, że te same dane nie są duplikowane w wielu tabelach. Zamiast tego tworzona jest tabela nadrzędna z informacjami, a odniesienie (klucz obcy) jest pobierane od rodzica do drugiej tabeli (podrzędnej). W związku z tym wszelkie aktualizacje tabeli nadrzędnej można wykonać bez zakłóceń w innych tabelach. Jednak w MongoDB dokumenty są zdenormalizowane. Dlatego istnieje redundancja.

- Dostępność zasobów

W porównaniu do MySQL, MongoDB jest stosunkowo nowy i stąd społeczność nie jest tak silna jak w MySQL. Istnieje więcej kodów i projektów open-source dotyczących MySQL w porównaniu z Mongo. Nie ma narzędzi do raportowania dla Mongo, takich jak w MySQL.

4. Integralność danych

Dokładność, wiarygodność i spójność danych przechowywanych w bazie danych jest określana mianem 'integralności danych', która idzie w parze z 'jakością danych'. Mówi się, że dane są wysokiej jakości, jeśli odpowiadają one swojemu przeznaczeniu, a ich wydajność jest wysoka. Dane muszą być również chronione przed nieautoryzowanym dostępem oraz modyfikacją. Aby zapewnić integralność i bezpieczeństwo danych, specjaliści ds. Bezpieczeństwa baz danych stosują różne środki, takie jak ograniczenia integralności, szyfrowanie, tworzenie kopii zapasowych, kontrola dostępu, walidacja danych itp. Ograniczenia integralności definiowane podczas wprowadzania danych dają pewność, że dane są zgodne z odpowiednimi regułami.

Integralność danych w MySQL jest wymuszana za pomocą różnych ograniczeń. Składają się one z następujących typów:

- Integralność wiersza

  Klucz podstawowy zapewnia, że wszystkie wiersze mają unikalny identyfikator. Taki indentyfikator zapewnia, że poszczególnym wierszom w danej kolumnie przypisywane są unikalne wartości.

- Integralność kolumny

  Wszystkie dane w kolumnie mają ten sam format, strukturę i definicję. To ograniczenie definiuje typ danych, długość i inne atrybuty specyficzne dla kolumny.

- Integralność referencyjna

Klucze obce zapewniają integralność danych podczas aktualizacji danych poprzez tworzenie relacji między tabelami.

- Integralność zdefiniowana przez użytkownika

  Specyficzne wymagania biznesowe, których nie można zdefiniować przez powyżej wymienione ograniczenia integralności, są obsługiwane przez ograniczenia zdefiniowane przez użytkownika. Definiuje to zdefiniowane przez użytkownika ograniczenia dla danych w kolumnie.

Integralność danych w MongoDB:

- Na poziomie dokumentu operacje zapisu są zgodne z ACID (Atomicity, Consistency, Isolation and Durability) – zapewniając izolację, gdy dokument jest modyfikowany podczas transakcji.
- Brak wbudowanych funkcji zapewniających spójność w MongoDB. Obowiązkiem inżyniera jest nie zapisywanie niespójnych danych do bazy danych. Trudno jest zagwarantować integralność danych w Mongo ze względu na brak relacji kluczy obcych po stronie serwera.

5. Bezpieczeństwo

- MySQL – wysokie standardy bezpieczeństwa danych MySQL pochodzą z jego systemu uprawnień. Podstawową funkcją systemu uprawnień MySQL jest uwierzytelnianie użytkownika i powiązanie go z uprawnieniami w określonej bazie danych. Baza danych jest chroniona hasłem, dzięki czemu ma dobry mechanizm uwierzytelniania. System uprawnień MySQL zapewnia, że wszyscy użytkownicy mogą wykonywać tylko dozwolone operacje. W ten sposób autoryzacja jest kontrolowana, dając dostęp do odpowiednich użytkowników i baz danych tylko w razie potrzeby.
- MongoDB wykorzystuje kontrolę dostępu opartą na rolach z elastycznym zestawem uprawnień. Brak obsługi szyfrowania danych w spoczunku i słabe uwierzytelnianie na poziomie bazy danych może powodować duże zagrożenia bezpieczeństwa. Aby zrekompensować słabość bezpieczeństwa na poziomie bazy danych, Mongo tworzy kopie baz danych i udostępnia je na wielu serwerach. Ponadto może używać protokołu TLS/SSL – (Transport Layer Security / Secure Sockets Layer) do szyfrowania całego ruchu sieciowego MongoDB, aby upewnić się, że jest on czytelny tylko dla zamierzonego klienta.

6. Wymagania do uruchomienia projektu:

- Instalacja Mysql community 8.0.28 z hasłem root „password”
- Instalacja MongoDB 5.0.5
- Node.JS
- Sklonowanie repozytorium https://github.com/pmcmal/MySQL_vs_MongoDB
- W katalogu projektu „PROJEKT” shift+prawy przycisk myszki -> Otwórz tutaj okno programu Powershell
- npm install
- OPCJA1 (Logi w konsoli) : npm start
- OPCJA2 (Logi do pliku txt) : npm start > log.txt

7. Kod projektu

- Główny plik benchmark.js

  Inicjuje program, przedstawia się i pokazuje aktualnie wykonywany test a następnie informuje o zakończeniu pracy. Do wykonania testu na danej bazie istnieje odwołanie do danego pliku (mongo lub mysql.js)

benchmark.js

```
var mongo = require('./mongo');
var mysql = require('./mysql');
var seconds= 10;

let main = () =>{
console.log("MONGODB VS MYSQL PROJEKT ZTBD 2022");
console.log("TESTUJE MONGODB");
mongo.runBechmark(seconds).then((resolve) => {
    console.log(resolve);
    console.log("TESTUJE MYSQL - BAZA NIEINDEKSOWANA");
    mysql.runBechmark(seconds, 1).then((resolve) => {
        console.log(resolve);
        console.log("TESTUJE MYSQL - BAZA INDEKSOWANA Z PRIMARY KEY");
        mysql.runBechmark(seconds, 2).then((resolve) => {
            console.log(resolve);
            console.log("UKOŃCZONO WSZYSTKO :)");
            return;
        });
    });
});

};

main();
```

- Plik mongo.js

  Łączymy się tutaj z bazą pod url ‘mongodb://localhost:27017/test’. Maksymalny czas wykonania zapytań to 1s. Wobec tego testujemy tutaj ile dana baza wykona ilościowo zapytań w tym przedziale czasowym.

benchmark.js | mongo.js | mysql.js

```
var mongo = require('mongoose');
const process = require('process');
var url = "mongodb://localhost:27017/test";

var test_a = mongo.Schema({
    id: Number,
    value_1: Number,
    value_2: Number,
    value_3: Number,
    value_4: Number,
    seq: Number,
    created: {
        type: Date,
        default: Date.now
    }
});

var Data_A = mongo.model('Data_A', test_a, 'test_a');
var dv = 1000000000; //default value
var ev = dv; //end value
var nv = 9182506147; //update value
const MAX_TIME_MS = 999000000;

let insert = async (seconds) => {
    console.log("SEKCJA INSERT");
    var iterations = 1;
    var count = 0;
    var value = dv;
    var total_count = 0;
    var start = process.hrtime()
    while (true) {
        var new_obj = new Data_A({
            value_1: value,
            value_2: value,
            value_3: value,
            value_4: value
        });
        var result = await new_obj.save();
        value++;
        count++;

        var end = process.hrtime(start);
        if (end[1] > MAX_TIME_MS || end[0] > 0) {
            iterations++;
```

- Plik mysql.js

  Baza ‘test’ z domyślnym hasłem ‘password’. Jeden insert posiada ‘value1’,’value2’,’value3’ oraz ‘value4’. Wykonywane są po kolei zapytania w formie typowej dla SQL tylko, że w porównaniu do bazy MongoDB istnieje tu podział na bazę nieindeksowaną jak i indeksowaną.

benchmark.js | mongo.js | mysql.js

```
var mysql = require('mysql');
const process = require('process');

var connect = mysql.createConnection({
    host: 'localhost',
    user: 'root',
    password: 'password',
    database: 'test'
});

connect.connect();
var dv = 1000000000; //default value
var ev = dv; //end value
var nv = 9182506147; //update value

let query = (str) => {
    return new Promise((resolve, reject) => {
        connect.query(str, (err, rows, fields) => {
            if (err) {
                return reject(err);
            }
            resolve(rows);
        });
    });
}

let insert = async (seconds) => {
    console.log("SEKCJA INSERT");
    var iterations = 1;
    var count = 0;
    var value = dv;
    var total_count = 0;
    var start = process.hrtime()

    while (true) {

        var insert_A = 'INSERT INTO `test_a`(`value_1`, `value_2`, `value_3`, `value_4`) \
        VALUES (' + value + ',' + value + ',' + value + ',' + value + ');';

        var result = await query(insert_A);

        value++;
        count++;
```

8. Działanie programu

Program działa w konsoli i za pomocą ‘console.log’ informuje użytkownika o wykonywanych akcjach. Możliwy także zapis strumienia logów do pliku np. z rozszerzeniem txt.

```
PS C:\Users\PMC-PC\Desktop\ZTBD\PROJEKT> npm start

> mongo-vs-mysql@1.0.0 start
> node benchmark.js

MONGODB VS MYSQL PROJEKT ZTBD 2022
TESTUJE MONGODB
SEKCJA INSERT
Wiersz :        1163   Czas wykonania: 0s 999.5172ms
Wiersz :        1322   Czas wykonania: 0s 999.2242ms
Wiersz :        1383   Czas wykonania: 0s 999.3974ms
Wiersz :        1562   Czas wykonania: 0s 999.3167ms
Wiersz :        2060   Czas wykonania: 0s 999.2144ms
Wiersz :        2083   Czas wykonania: 0s 999.37ms
Wiersz :        2085   Czas wykonania: 0s 999.1789ms
Wiersz :        2070   Czas wykonania: 0s 999.402ms
Wiersz :        2072   Czas wykonania: 0s 999.1881ms
Wiersz :        2080   Czas wykonania: 0s 999.2029ms
Wiersz :        17880 INSERTED

SEKCJA SELECT
Wiersz :        86     Czas wykonania: 1s 0.2504ms
Wiersz :        87     Czas wykonania: 1s 1.9633ms
Wiersz :        88     Czas wykonania: 1s 5.9921ms
Wiersz :        87     Czas wykonania: 1s 10.4623ms
```

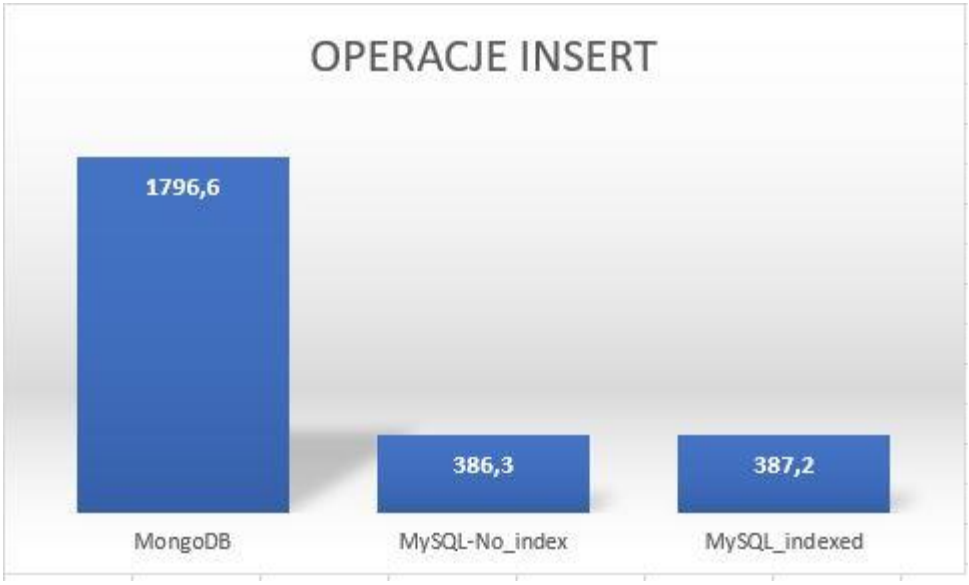
Metodologia testowania podjęta w tym projekcie polega na zebraniu danych dla każdego z operacji CRUD (Insert,Select,Update oraz Delete) na bazach MongoDB, MySQL z indeksowaniem oraz MySQL bez indeksowania. Każde z operacji jest wykonane przez 1s a lepiej radząca sobie baza będzie miała większą ilość wykonanych zapytań.

9. Wykresy porównawcze

Zebrane dane za pomocą przekazania logów do pliku log.txt dostępnego w katalogu DOCS projektu zostały eksportowane do arkusza MS Excel a następnie obliczono średnie. Wykonano wykresy dla każdej z operacji CRUD po kolei. Zebrane dane można także dalej analizować w zależności od informacji jaką chcemy uzyskać. W tym przypadku interesowała nas liczba wykonanych zapytań w okresie czasu 1 sekundy. Mimo, że taki test jest dosyć prostą metodologią testowania to wyniki już dają nam dużo do myślenia.

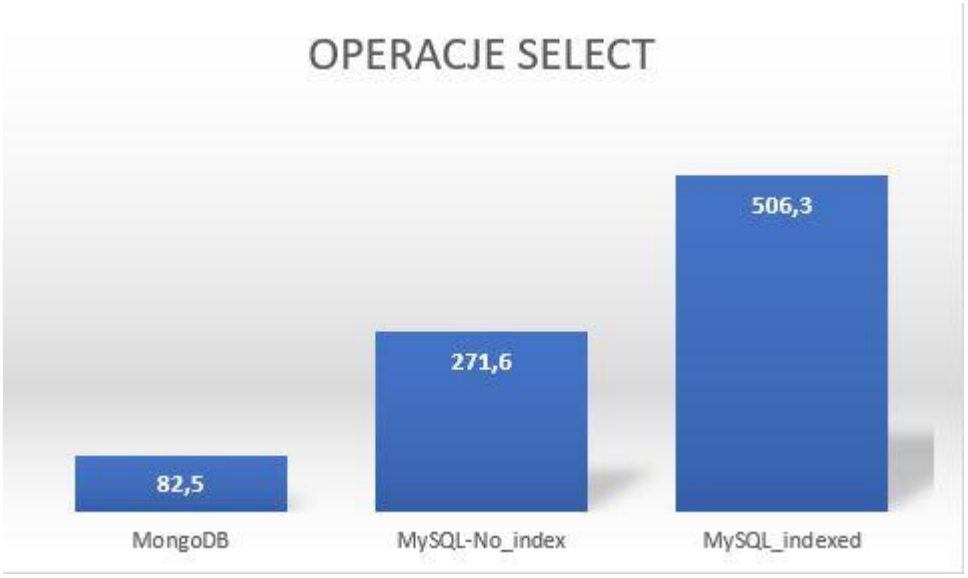
- Operacje Insert

  Jak można zauważyć na wykresie poniżej MongoDB wygrywa zdecydowaną większością wykonanych zapytań pozostawiając obydwie bazy danych MySQL daleko za sobą. Zatem jeśli firmie zależy na dużej ilości wprowadzeń danych w jak najmniejszym czasie to MongoDB jest świetnym wyborem.

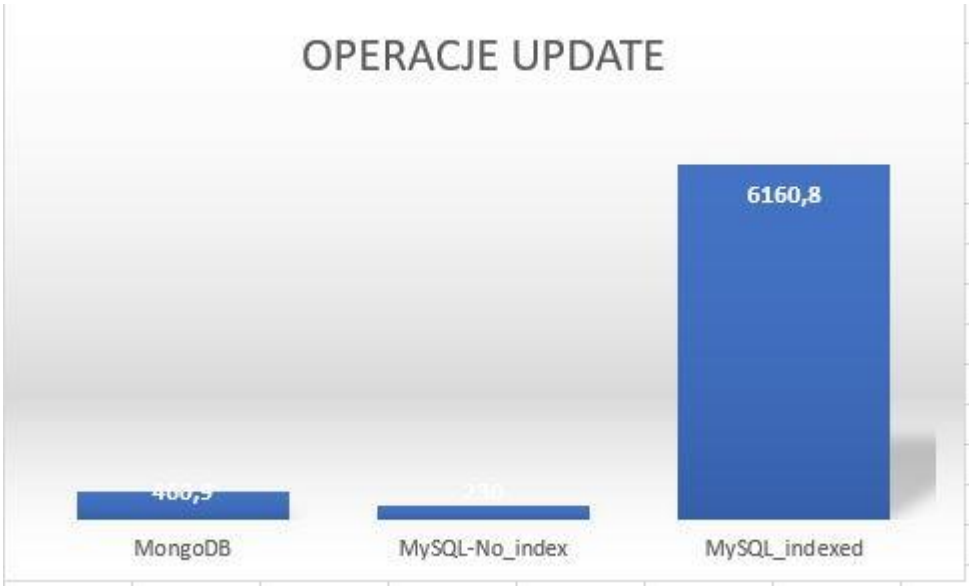


- Operacje Select

  Jak zauważymy na wykresie poniżej operacja Select najlepszy wynik osiągnęła w bazie danych MySQL z indeksowaniem co skolei przedstawia zarazem jej największą zaletę. Jeśli więc firma wykonuje dużą ilość zapytań tego typu to jest to dla niej dobry wybór.



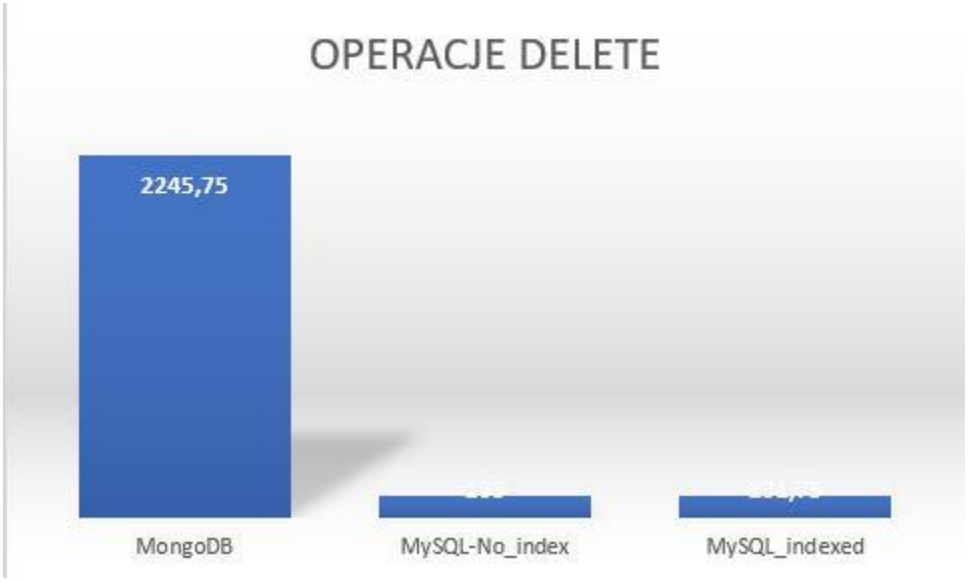
- Operacje Update



Powyżej widzimy, że baza MySQL z indeksowaniem jeszcze bardziej zachęca nas do jej używania i stosowania jeśli nie tylko wykonujemy dużo zapytań Select ale także Update. Co ciekawe różnica między nieideksowaną bazą a indeksowaną jest ogromna.

- Operacje Delete

MongoDB pokazało nam, że najlepiej radzi sobie w wykonaniu operacji typu Insert, zatem można było się spodziewać, że operacja odwrotna będzie również deklasowała obydwie bazy danych MySQL.



10. Podsumowanie projektu

W czasie implementacji oraz eksperymentów na bazach danych typu MySQL oraz MongoDB można od razu zauważyć, że MySQL posiada znacznie bardziej oddaną społeczność. Zarówno pod względem dostępnych projektów na github jak i ilości poradników. Może to wynikać z tego, że SQL jest już bardzo długo z nami i na pewno każdy informatyk miał z nim doczynienia. Już sam fakt, że SQL jako strukturalny język zapytań jest używany nie tylko w MySQL zwiększa jego popularność.

Jest on także dłużej na rynku gdyż został opracowany w latach 70tych w firmie IBM. MongoDB jest także otwartym systemem typu open-source tak samo jak MySQL. Natomiast jego pierwsza wersja powstała w 2007 roku a pierwsza stabilna wersja w lutym 2009 roku. Największą różnicą między obydwoma bazami jest forma w jakiej działają. MongoDB jest kojarzone z dokumentami w stylu JSON zaś MySQL z modelem relacyjnym czyli dane przedstawione w postaci relacyjnej.

Po wykonaniu testów zauważono, że MongoDB najlepiej wykorzystać jeśli wprowadzamy dużą ilość danych oraz często są one usuwane. Natomiast MySQL zdecydowanie polecany jest gdy wykonujemy skomplikowane operacje Select oraz modyfikujemy dane. Język SQL jest już na tyle rozwinięty, że pozwala nie tylko na pisanie skomplikowanych operacji Select na wielu tabelach i z wieloma cechami, ale także na tworzenie zapytań poprzez graficzne wybieranie np w SQLServer.