SYSTEMY BAZY DANYCH Sprawozdanie Laboratorium 7-8



*Rozproszone bazy danych.*

*Projekt rozmów z klientem przez firmę telemarketingową*



**Autorzy: Grzegorz Parapura, Paweł Tarsała**

**Grupa: I8B3S4**

**Prowadzący: mgr inż. Maciej Szymczyk**

Spis treści

[1. Analiza problemu 3](#_Toc11665866)

[1.1. Model 3](#_Toc11665867)

[2. Zapytanie 4](#_Toc11665868)

[2.1. Opis zapytania 4](#_Toc11665869)

[2.2. Zapytanie SQL 4](#_Toc11665870)

[2.3. Zapytanie MongoDB 5](#_Toc11665871)

[2.4. Zapytanie dla 100 000 rekordów 6](#_Toc11665872)

[2.3.1 MSSQL 6](#_Toc11665873)

[2.3.2 MongoDB 6](#_Toc11665874)

[2.5. Zapytanie dla 250 000 rekordów 7](#_Toc11665875)

[2.4.1 MSSQL 7](#_Toc11665876)

[2.4.2 MongoDB 7](#_Toc11665877)

[2.6. Zapytanie dla 500 000 rekordów 8](#_Toc11665878)

[2.5.1 MSSQL 8](#_Toc11665879)

[2.5.2 MongoDB 8](#_Toc11665880)

[3. Repozytorum 8](#_Toc11665881)

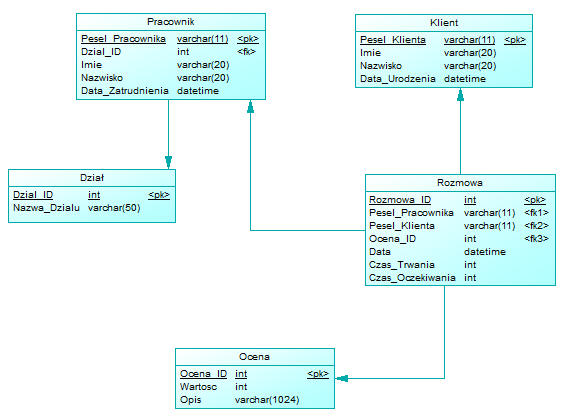
[4. Wnioski 9](#_Toc11665882)

# Analiza problemu

Celem realizacji projektu zawierającego rozproszone bazy danych jest stworzenie bazy danych zawierającej informacje na temat rozmów z klientami. Zapisywane rozmowy prowadzone są pomiędzy pracownikiem a klientem, każda rozmowa zostaje ocenione przez klienta pod względem jakość. Klient może ocenić rozmowa ocenami ze skali 1-5, gdzie to oznaczą że klient był zadowolony z przeprowadzonej rozmowy.

System został zaimplementowany przy użyciu MSSQL i MongoDB, schemat bazy danych został wykonany w Power Designer.

## Model



Projekt zawiera następujące tabele:

* Klient – przechowuje dane na temat klientów z którymi były prowadzone rozmowy
  + Atrybuty: Pesel klienta, Imię, Nazwisko, Data urodzenia
* Rozmowa – przechowuje dane na temat przeprowadzonej rozmowy pomiędzy klientem a pracownikiem
  + Atrybuty: Id rozmowy, Data, Czas trwania, Czas oczekiwania
* Ocena – przechowuje dane na temat oceny rozmowy
  + Atrybuty: Id oceny, Wartość, Opis
* Pracownik – przechowuje dane na temat pracowników którzy przeprowadzają rozmowy
  + Atrybuty: Id pracownika, Imię, Nazwisko, Data zatrudnienia
* Dział – przechowuje dane na temat działów w jakich pracują pracownicy przeprowadzający rozmowy
  + Atrybuty: Id działu, Nazwa działu

# Zapytanie

# Opis zapytania

Zapytanie zwracające imię i nazwisko pracownika oraz klienta, oraz liczbę oraz liczbę rozmów pomiędzy tymi osobami. Brani pod uwagę są tylko pracownicy z działów IT,HR i Marketing i rozmowy których ocena ma wartość większą niż średnia wszystkich ocen

# Zapytanie SQL

select sel1.DANE\_KLIENTA, sel1.DANE\_PRACOWNIKA, count(\*) as LICZBA\_ROZMOW

from

(select

concat (k.Imie,' ', k.Nazwisko) as DANE\_KLIENTA,

concat (p.Imie,' ', p.Nazwisko) as DANE\_PRACOWNIKA,

o.WARTOSC as OCENA,

r.DATA as DATA\_ROZMOWY

from Rozmowa r

inner join Ocena o on r.Ocena\_ID = o.Ocena\_ID

inner join Klient k on r.Pesel\_Klient = k.Pesel\_Klient

inner join Pracownik p on r.Pesel\_Pracownik = p.Pesel\_Pracownik

inner join Dział d on d.Dzial\_ID = p.Dzial\_ID

where

d.Nazwa\_Dzialu IN ('IT','HR','Marketing') and

o.WARTOSC >

(select avg(o1.WARTOSC)

from Rozmowa r1

inner join Ocena o1 on r1.Ocena\_ID = o1.Ocena\_ID)

) as sel1

group by DANE\_KLIENTA, DANE\_PRACOWNIKA

order by "LICZBA\_ROZMOW" desc;

# Zapytanie MongoDB

db.ROZMOWA.aggregate([{

"$match": {

"$and": [{

"Pracownik.Dzial.Nazwa\_Dzialu": {

$in: [ "IT","HR","Marketing" ]

}

},

{

"Ocena.Wartosc": {

"$gt": db.ROZMOWA.aggregate([{

"$group": {

"\_id": null,

"srednia": {

"$avg": "$Ocena.Wartosc"

}

}

}]).map(function(u) { return u.srednia })[0]

}

}

]}

},

{

"$group": {

"\_id": {

"Imie pracownika": "$Pracownik.Imie",

"Nazwisko pracownika": "$Pracownik.Nazwisko",

"Imie klienta": "$Klient.Imie",

"Nazwisko klienta": "$Klient.Nazwisko"

},

"Rozmowa": {

"$sum": 1

}

}

},

{

"$sort": {

"Rozmowa": -1

}

},

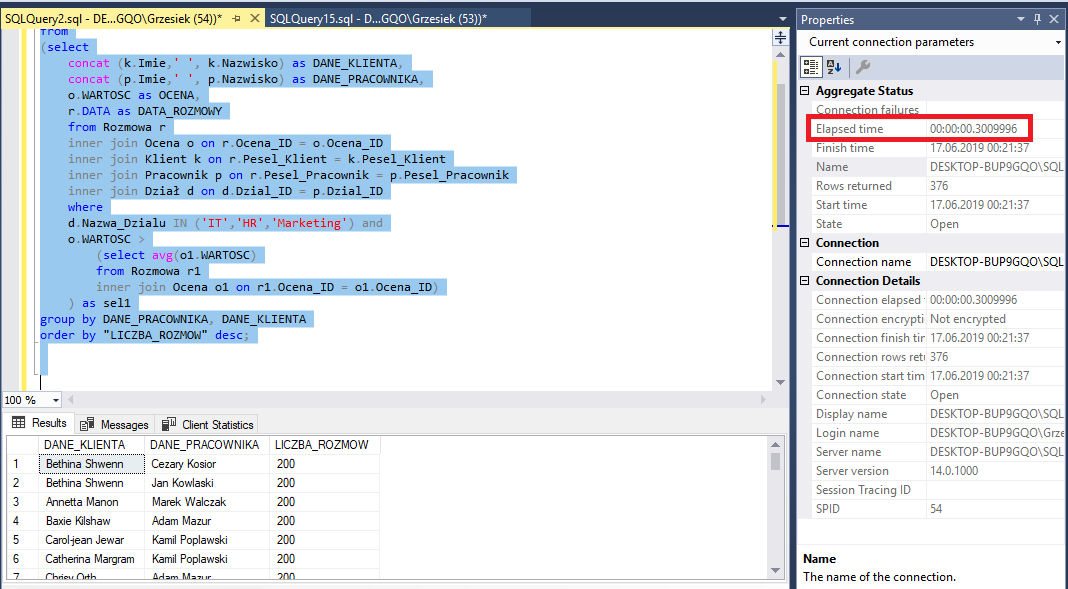
{

"$limit": 100

}])

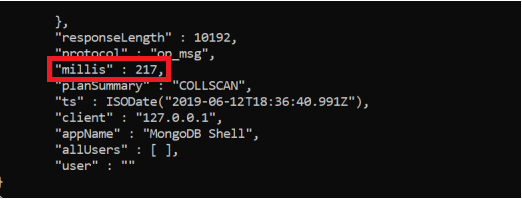
# Zapytanie dla 100 000 rekordów

# MSSQL



**Czas: 301 ms**

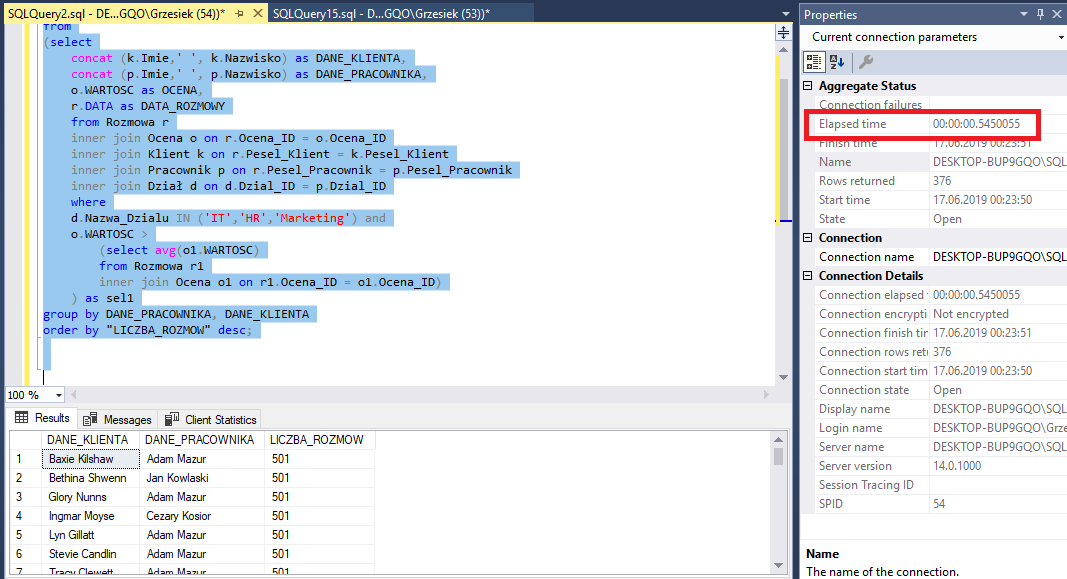
# MongoDB



**Czas: 217 ms**

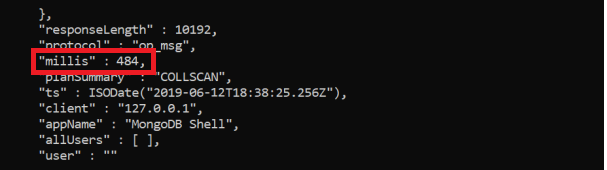
# Zapytanie dla 250 000 rekordów

# MSSQL



**Czas: 545 ms**

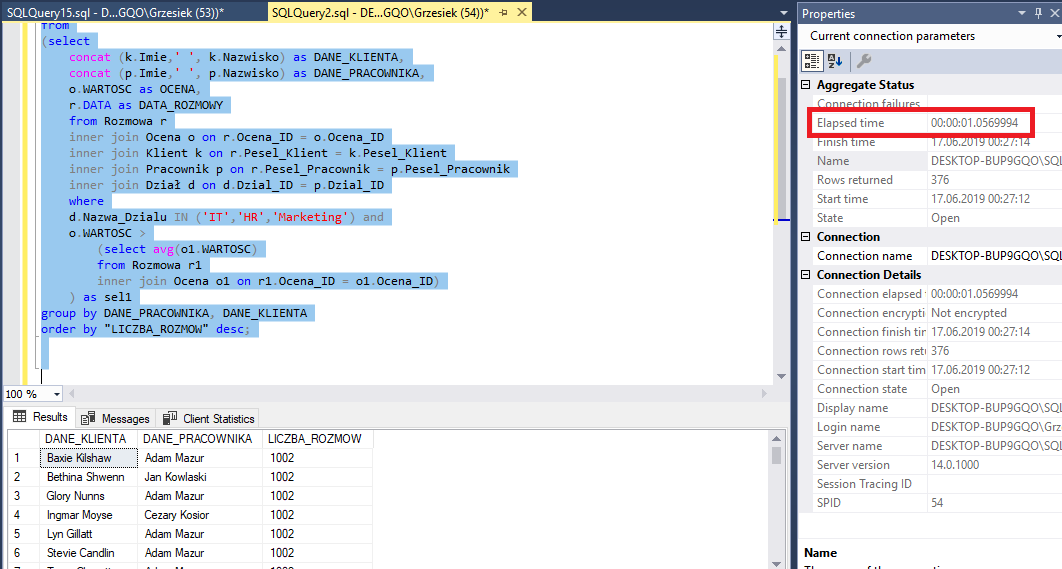
# MongoDB



**Czas: 484ms**

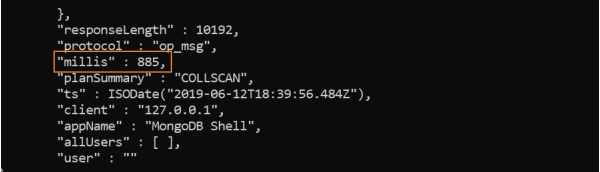
# Zapytanie dla 500 000 rekordów

# MSSQL



**Czas: 1057 ms**

# MongoDB



**Czas: 885 ms**

# Repozytorum

Dane zostały wygenerowane przy użyciu generatora online.

Projekt jest dostępny na repozytorium GitHub - <https://github.com/grzeniu/SBD_RBD>

# Wnioski

**Napotkane problemy i ich rozwiązania**:

Największym problemem okazało się napisanie zapytania dla bazy NoSQL, aby działało w sposób identyczny jak zapytanie SQL. Problem ten wynikał na pewno z faktu, że nie korzystaliśmy nigdy z baz danych typu MongoDB. Lepsza znajomość składni i zaznajomienie z bazą mogłoby skutkować krótszym czasem zapytania. Wpływ na otrzymany czas ma również struktura dokumentów umieszczonych w naszej kolekcji zawierająca zagnieżdżone pola.

Problemem okazał się też import danych, w linii poleceń było to praktycznie niemożliwe, dopiero skorzystanie z IntelijIDEA pomogło wstawić dużą liczbę rekordów w szybkim czasie.

**Uwagi** :

• Pierwsze zapytanie na bazie MS SQL trwa dłużej ponieważ jest ono wtedy kompilowane i generowany jest plan dla zapytania. Zapytanie następnie trafia do cache do znacznie przyśpiesz późniejsze kolejne wywołania tego zapytania. Każde zapytanie wymaga planu zapytania, zanim zostanie wykonane. Jest on przechowywany w pamięci podręcznej planu kwerend SQL Server. W ten sposób po ponownym uruchomieniu danego zapytania program SQL Server nie musi tworzyć kolejnego planu. Czas trwania planu kwerendy w pamięci podręcznej zależy od tego, jak często zapytanie jest wykonywane. Plany, które są używane częściej, pozostają w pamięci podręcznej na dłuższy czas i odwrotnie. W związku z tym czas przetwarzania zapytania nie odzwierciedlał rzeczywistego czasu, który musi zostać poświęcony na jego wykonanie. Ręczne usunięcie cache również nie przynosiło oczekiwanych rezultatów.

• MongoDB nie tylko szybciej zwracało wynik zapytania ale też szybciej wykonywało wstawianie nowych rekordów.

**Pomysły:**

Uważam że pomocnym narzędziem dla osób uczących się obsługi bazy MongoDB byłby konwerter zapytań SQL, tak aby były one wykonywane na bazie NoSQL.

**Ocena środowisk**

• Microsoft SQL Server Management Studio 2017 SQL Server Management Studio to darmowe zintegrowane środowisko do zarządzania dowolną infrastrukturą SQL. W tym przypadku jest to SQL Server. Zapewnia narzędzia do konfigurowania, monitorowania i administrowania bazami danych. Środowiska używa się aby mieć możliwość wdrożenia, monitorowania i aktualizowania komponentów warstwy danych używanych przez aplikacje, a także tworzenia zapytań i skryptów. W przypadku realizacji tego zadania, środowisko to nie przysparzało żadnych problemów. Wstawienie dużej liczby danych przebiegło pomyślnie, dzięki czemu możliwe było szybkie przedstawienie wyników zapytań dla bardzo dużej liczby rekordów. Ewentualne błędy były dobrze opisane i możliwa była szybka ich naprawa.

• MongoDB jest nierelacyjnym systemem zarządzania bazami danych. Charakteryzuje się dużą skalowalnością wydajnością, brakiem konkretnej struktury obsługiwanych baz danych i szybkością. Te zalety MongoDB zawdzięcza między innymi dużemu zużyciu pamięci RAM. Wśród wad można też wymienić słabą dokumentacje i brak wsparcia dla operacji join.