Zuzanna Słobodzian
nr albumu: 412204

Kraków, 25.05.2023

Akademia Górniczo-Hutnicza im. Stanisława Staszica w Krakowie

Wydajność złączeń i zagnieżdżeń dla schematów znormalizowanych i zdenormalizowanych.

1. Wstęp

Celem projektu jest porównanie prędkości działania dwóch systemów zarządzania relacyjnymi bazami danych: MySQL i PostgreSQL. Analizę przeprowadzono rejestrując czas wykonania czterech zapytań na bazie danych Geochronologia, zawierającej informacje dotyczące nazw eonów, er, okresów, epok, progów oraz ich rozłożenia w czasie.

Parametry komputera:
CPU: AMD Ryzen 7 5825U with Radeon Graphics
RAM: 32,0 GB
SSD: M.2 PCIe
S.O.: Windows 11

Systemy zarządzania bazami danych:
MySQL Community Server 8.0.33
PostgreSQL 15.3

2. Opis testów

Po utworzeniu bazy danych, kluczy obcych oraz tabeli z informacjami na temat eonów, er, okresów, epok i progów, utworzono tabelę GeoTabela, która zawiera wszystkie dane z pozostałych tabel, w celu osiągnięcia formy zdenormalizowanej. (rys. 2.1)

```
CREATE TABLE GeoTabela AS (SELECT * FROM GeoPietro NATURAL JOIN GeoEpoka
NATURAL JOIN GeoOkres NATURAL JOIN GeoEra NATURAL JOIN GeoEon );
ALTER TABLE GeoTabela ADD PRIMARY KEY (IDPietro);
```

rys. 2.1

W kolejnym kroku wykorzystując autozłączenie z tabelą Dziesiec (rys. 2.2) utworzono tabelę Milion (rys 2.3) z syntetycznymi danymi, zawierającą liczby od 0 do 999 999.

```
CREATE TABLE dziesiec(
Cyfra INT,
Bit INT
);

INSERT INTO dziesiec VALUES(0, 1);
INSERT INTO dziesiec VALUES(1, 1);
INSERT INTO dziesiec VALUES(2, 1);
INSERT INTO dziesiec VALUES(3, 1);
INSERT INTO dziesiec VALUES(4, 1);
INSERT INTO dziesiec VALUES(5, 1);
INSERT INTO dziesiec VALUES(6, 1);
INSERT INTO dziesiec VALUES(7, 1);
INSERT INTO dziesiec VALUES(8, 1);
INSERT INTO dziesiec VALUES(9, 1);
```

rys. 2.2

```
CREATE TABLE Milion(
Liczba INT,
Cyfra INT,
Bit INT
);

INSERT INTO  Milion
SELECT a1.Cyfra + 10 * a2.Cyfra + 100 * a3.Cyfra
+ 1000 * a4.Cyfra + 10000 * a5.Cyfra + 10000 * a6.Cyfra AS liczba,
a1.Cyfra AS Cyfra, a1.Bit AS Bit
FROM Dziesiec a1, Dziesiec a2, Dziesiec a3,
Dziesiec a4, Dziesiec a5, Dziesiec a6;
```

rys. 2.3

Celem testów było określenie wydajności złączeń oraz zapytań zagnieżdżonych poprzez zmierzenie czasu wykonywania obliczeń, opierających się na łączeniu danych z tabeli geochronologicznej z danymi z tabeli Milion.

W pierwszej części testów indeksy zostały nałożone jedynie na klucze główne tabeli. W MySQL jest to częścią automatycznego procesu przy tworzeniu tabeli, natomiast w PostgreSQL ta operacja została wykonana ręcznie (rys. 2.4).

```
CREATE INDEX EonInd ON GeoEon(IDEon);
CREATE INDEX EpokaInd ON GeoEpoka(IDEpoka);
CREATE INDEX EraInd ON GeoEra(IDEra);
CREATE INDEX OkresInd ON GeoOkres(IDOkres);
CREATE INDEX PietroInd ON GeoPietro(IDPietro);
CREATE INDEX PietroTabInd ON GeoTabela(IDPietro);
```

rys. 2.4

Następnie powtórzono te same zapytania nakładając uprzednio indeksy na wszystkie kolumny biorące udział w złączeniu. (rys. 2.5, 2.6)

```
CREATE INDEX TabelaInd ON GeoTabela(IDEon, IDEra, IDOkres, IDEpoka);
CREATE INDEX MilionLInd ON Milion(Liczba);
```

rys. 2.5

```
CREATE INDEX TabelaInd ON GeoTabela(IDEon, IDEra, IDOkres, IDEpoka);
CREATE INDEX MilionLInd ON Milion(Liczba);
```

rys. 2.6

Zapytanie nr 1 (1 ZL) polegało na złączeniu zdenormalizowanej tabeli geochronologicznej (GeoTabela) z tabelą Milion, przy warunku mającym na celu dopasować zakresy wartości złączanych kolumn. (rys. 2.7)

```
SELECT COUNT(*) FROM Milion INNER JOIN GeoTabela ON (mod(Milion.liczba, 77) = (GeoTabela.IDPietro));
```

rys. 2.7

Zapytanie nr 2 (2 ZL) polegało na złączeniu tabeli Milion ze znormalizowaną tabelą geochronologiczną, uzyskaną poprzez złączenie pojedynczych tabel z jednostkami geochronologicznymi. (rys. 2.8)

```
SELECT COUNT(*) FROM Milion INNER JOIN  GeoPietro  ON (mod(Milion.liczba, 77) = GeoPietro.IDPietro)
NATURAL JOIN GeoEpoka NATURAL JOIN GeoOkres NATURAL JOIN GeoEra NATURAL JOIN GeoEon;
```

rys. 2.8

Zapytanie nr 3 (3 ZG) polegało na złączeniu zdenormalizowanej tabeli geochronologicznej (GeoTabela) z tabelą Milion, poprzez zagnieżdżenie skorelowane. (rys. 2.9)

```
SELECT COUNT(*) FROM Milion WHERE mod(Milion.liczba, 77)
IN (SELECT IDPietro FROM GeoTabela WHERE mod(Milion.liczba, 77) = (IDPietro));
```

rys. 2.9

Zapytanie nr 4 (4 ZG) polegało na złączeniu, poprzez zagnieżdżenie skorelowane, tabeli Milion ze znormalizowaną tabelą geochronologiczną. (rys. 2.10)

```
SELECT COUNT(*) FROM Milion WHERE mod(Milion.liczba, 77)
IN (SELECT GeoPietro.IDPietro FROM GeoPietro NATURAL JOIN GeoEpoka
NATURAL JOIN   GeoOkres NATURAL JOIN GeoEra NATURAL JOIN GeoEon);
```

rys. 2.10

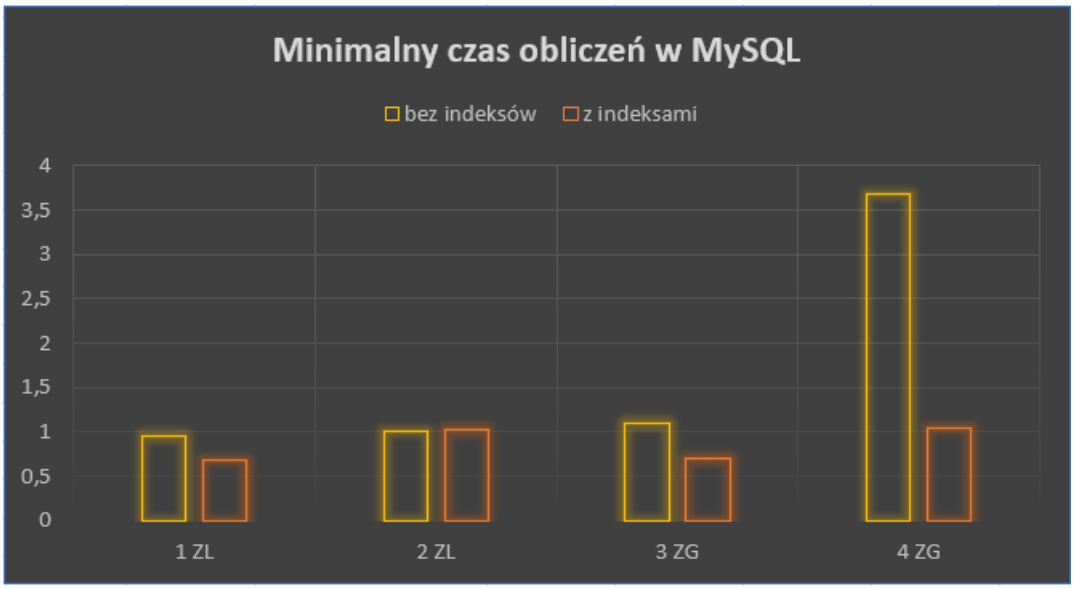
3. Zestawienie wyników

W poniższej tabeli zestawiono wartości minimalne oraz średnie czasu wykonywania zapytań dla kolumn z indeksami lub bez, w obu systemach bazodanowych. Czas został podany w sekundach. (rys. 3.1)

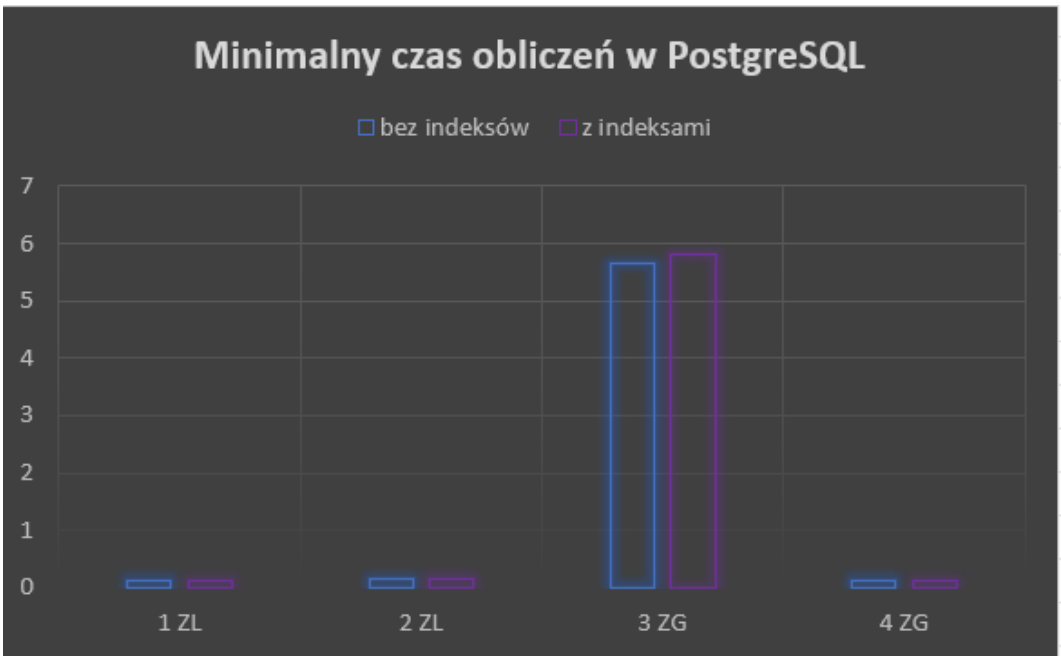
	1 ZL		2 ZL		3 ZG		4 ZG	
	MIN [s]	SR [s]	MIN [s]	SR [s]	MIN [s]	SR [s]	MIN [s]	SR [s]
BEZ INDEKSÓW								
MySQL	0,953	0,958	1,016	1,026	1,094	1,115	3,687	3,739
PostgreSQL	0,116	0,139	0,157	0,175	5,64	5,829	0,112	0,121
Z INDEKSAMI								
MySQL	0,687	0,692	1,031	1,036	0,703	0,718	1,047	1,073
PostgreSQL	0,106	0,108	0,157	0,172	5,82	6,008	0,12	0,128

rys. 3.1

Wykonano również wykresy przedstawiające minimalny czas wykonania każdego z zapytań w zależności od indeksowania kolumn w MySQL oraz PostgreSQL. (rys. 3.2, 3.3)

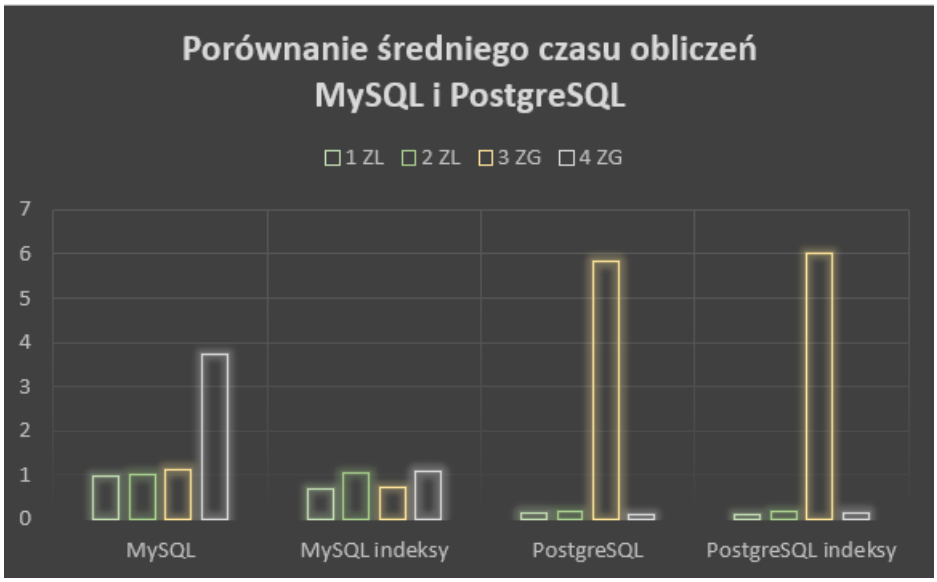


rys. 3.2



rys. 3.3

Na poniższym wykresie zobrazowano średni czas wykonywania każdego zapytania dla obu systemów bazodanowych w zależności od obecności indeksów kolumn. (rys. 3.4)



rys. 3.4

4. Wnioski

Na podstawie przeprowadzonych testów można wyciągnąć następujące wnioski:

1. Po nadaniu indeksów wszystkim kolumnom czas wykonywania zapytań w MySQL zmniejszył się, zarówno dla złączeń jak i zagnieżdżeń skorelowanych. W przypadku czwartego zapytania czas zmniejszył się nawet trzykrotnie (z 3,7 s do 0,11 s).

2. Nadanie indeksów wszystkim kolumnom nie wpłynęło na czas obliczeń zapytań w PostgreSQL.

3. Większość zapytań wykonuje się szybciej w PostgreSQL, z wyjątkiem zapytania trzeciego, gdzie można zaobserwować duży wzrost czasu wykonania (prawie 6 s) w stosunku do pozostałych zapytań (od 0,16 s do 0,17 s).

4. Użycie zagnieżdżeń skorelowanych wydłuża czas wykonywania się algorytmu w obu analizowanych systemach bazodanowych.

5. Zapytania wykorzystujące postać zdenormalizowaną wykonują się szybciej (z wyjątkiem zapytania trzeciego w PostgreSQL).

5. Bibliografia

1. Łukasz Jajeśnica, Adam Piórkowski „Wydajność złączeń i zagnieżdżeń dla schematów znormalizowanych i zdenormalizowanych”

2. https://pl.wikipedia.org/wiki/Tabela_stratygraficzna (tabela stratygraficzna)