Natalia Abramowicz 410635

24.05.2020 r.

Sprawozdanie Bazy Danych ćwiczenie 9

„Analiz wydajności złączeń i zagnieżdżeń dla

schematów znormalizowanych i zdenormalizowanych”

1. Wstęp

Celem ćwiczenia było przeprowadzenie analizy wydajności dla złączeń i zagnieżdżeń dla schematów znormalizowanych i zdenormalizowanych. Do przeprowadzenia testu stworzono dwie tabele opierające się o uproszczony schemat tabeli geochronologicznej. Znalazły się w nich takie jednostki wymiaru czasowego jak: eon, era, okres, epoka i piętro oraz odpowiadające mi jednostki stratygraficzne. Pierwsza tabela opierała się o schemat znormalizowany (płatka śniegu), zaś druga tabela miała formę zdenormalizowaną. Kwerendy bazujące na danych tabelach zostały wykonane w systemach zarządzania bazami danych takimi jak: PostgreSQL i MySQL.

2. Konfiguracja sprzętowa

Testy przeprowadzono na laptopie o następującej specyfikacji:

- Procesor: AMD Ryzen 7 4700U with Radeon Graphics 2.00 GHz
- Pamięć RAM: 32 GB
- Dysk SSD: SSDPR-PX500-01T-80; pojemność: 952 GB
- System Operacyjny:
    - Typ: 64-bitowy system operacyjny, procesor x64
    - Wersja: Windows 11 Home

Specyfikacja serwerów baz danych:

- PostgreSQL 15.2
- MySQL 8.0.33

Testy wykonano w środowiskach:

- MySQL Workbench 8.0 CE
- pgAdmin 4 v.6.15

3. Tabela geochronologiczna

Uproszczoną tabelę geochronologiczną opierającą się o schemat płatka śniegu zbudowano na podstawie informacji dostępnych na *Wikipedii*. Fragment kodu wskazujący na budowę tabeli *GeoEon* (*PostgreSQL*):

```
CREATE TABLE Tabela_stratygraficzna.GeoEon(
id_eon INTEGER PRIMARY KEY,
nazwa_eon CHAR(10) NOT NULL
);
```

oraz wypełnienie tabeli:

```
--eon(id_eon, nazwa)
INSERT INTO Tabela_stratygraficzna.GeoEon VALUES
(1, 'Fanerozoik');
```

Tabela zdenormalizowaną (*GeoTab*) zostało stworzona poprzez złączenie wewnętrzne wszystkich tabel tworzących hierarchię:

```
SELECT
pi.id_pietro AS ID_pietro,
pi.nazwa_pietro AS Nazwa_pietro,
ep.id_epoka AS ID_epoka,
ep.nazwa_epoka AS Nazwa_epoka,
o.id_okres AS ID_okres,
o.nazwa_okres AS Nazwa_okres,
er.id_era AS ID_era,
er.nazwa_era AS Nazwa_era,
eo.id_eon AS ID_eon,
eo.nazwa_eon AS Nazwa_eon
INTO
Tabela_stratygraficzna.GeoTabela
FROM
Tabela_stratygraficzna.GeoPietro pi
INNER JOIN
Tabela_stratygraficzna.GeoEpoka ep ON pi.id_epoka=ep.id_epoka
INNER JOIN
Tabela_stratygraficzna.GeoOkres o ON ep.id_okres=o.id_okres
INNER JOIN
Tabela_stratygraficzna.GeoEra er ON o.id_era=er.id_era
INNER JOIN
Tabela_stratygraficzna.GeoEon eo ON er.id_eon=eo.id_eon;

ALTER TABLE tabela_stratygraficzna.GeoTabela ADD PRIMARY KEY(ID_pietro);
```

4. Testy wydajności

W celu przeprowadzenia testów stworzono tabelę *Milion,* którą wypełniono liczbami naturalnymi od 0 do 999999. Tabela wypełniono przy pomocy wyrażenia *Recursive Common Table Expression*:

```
WITH RECURSIVE ID(number)
AS
(
SELECT 0 AS number
UNION ALL
```

```
SELECT number + 1
FROM ID
WHERE number < 999999
)
INSERT INTO Tabela_stratygraficzna.Milion SELECT number FROM ID;
```

W teście wykonano następujące kwerendy (fragmenty kodu wykonanego w programie *pgAdmin*):

- Zapytanie 1 (ZL1): złączenie syntetycznej tabeli *Milion* z tabelą geochronologiczną w postaci zdenormalizowanej. Do warunku złączenia dodano operacje modulo, która dopasowuje zakresy wartości złączanych kolumn:

```
SELECT COUNT(*) AS ZL1
FROM Tabela_stratygraficzna.Milion m
JOIN Tabela_stratygraficzna.GeoTabela gt ON (m.liczba % 75) =
gt.id_pietro;
```

- Zapytanie 2 (ZL2): złączenie syntetycznej tabeli *Milion* z tabelą geochronologiczną w postaci znormalizowanej, reprezentowaną przez złączenie pięciu tabel:

```
SELECT COUNT(*) AS ZL2
FROM Tabela_stratygraficzna.Milion mi
INNER JOIN Tabela_stratygraficzna.GeoPietro pi ON (mi.liczba % 75) =
pi.id_pietro
INNER JOIN Tabela_stratygraficzna.GeoEpoka ep ON ep.id_epoka =
pi.id_epoka
INNER JOIN Tabela_stratygraficzna.GeoOkres o ON o.id_okres =
ep.id_okres
INNER JOIN Tabela_stratygraficzna.GeoEra er ON er.id_era = o.id_era
INNER JOIN Tabela_stratygraficzna.GeoEon eo ON eo.id_eon = eo.id_eon;
```

- Zapytanie 3 (ZG3): złączenie syntetycznej tabeli *Milion* z tabelą geochronologiczną w postaci zdenormalizowanej, gdzie złączenie jest wykonywane poprzez zagnieżdżenie skorelowane:

```
SELECT COUNT(*) AS ZG3
FROM Tabela_stratygraficzna.Milion m
WHERE (m.liczba % 75) =
(SELECT id_pietro
FROM Tabela_stratygraficzna.GeoTabela
WHERE (m.liczba % 75) = id_pietro);
```

- Zapytanie 4 (ZG4): złączenie syntetycznej tabeli *Milion* z tabelą geochronologiczną w postaci znormalizowanej, gdzie złączenie wykonywane jest poprzez zagnieżdżenie skorelowane, a zapytanie wewnętrzne jest złączeniem tabel poszczególnych jednostek geochronologicznych:

```
SELECT COUNT(*) AS ZG4
FROM Tabela_stratygraficzna.Milion m
WHERE (m.liczba % 75) IN
(SELECT pi.id_pietro
```

```
FROM Tabela_stratygraficzna.GeoPietro pi
INNER JOIN Tabela_stratygraficzna.GeoEpoka ep ON ep.id_epoka =
pi.id_epoka
INNER JOIN Tabela_stratygraficzna.GeoOkres o ON o.id_okres =
ep.id_okres
INNER JOIN Tabela_stratygraficzna.GeoEra er ON er.id_era = o.id_era
INNER JOIN Tabela_stratygraficzna.GeoEon eo ON eo.id_eon = er.id_eon);
```

Każde z zapytań zostało wykonane zarówno na danych baz założonych indeksów, jak i z założonymi indeksami na wszystkie kolumny.

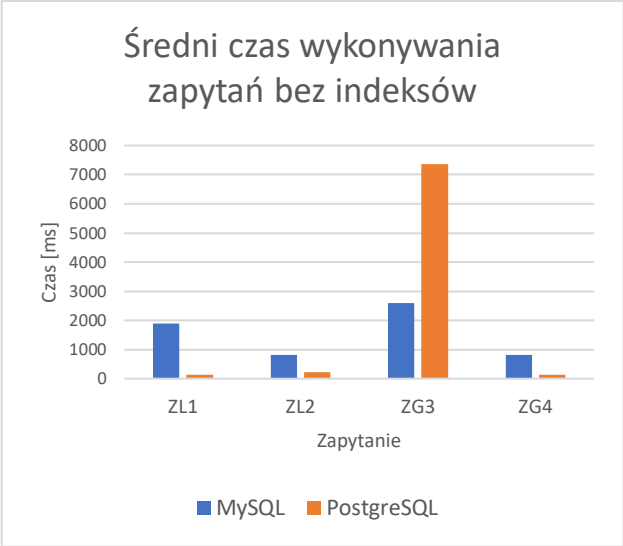
5. Wyniki testów

Każdy z testów został powtórzony 10-krotnie. Na podstawie zgromadzonych wyników obliczono średni czas wykonywania zapytania, który wraz z najmniejszym pomiarem przedstawiono w *Tabeli 1*.

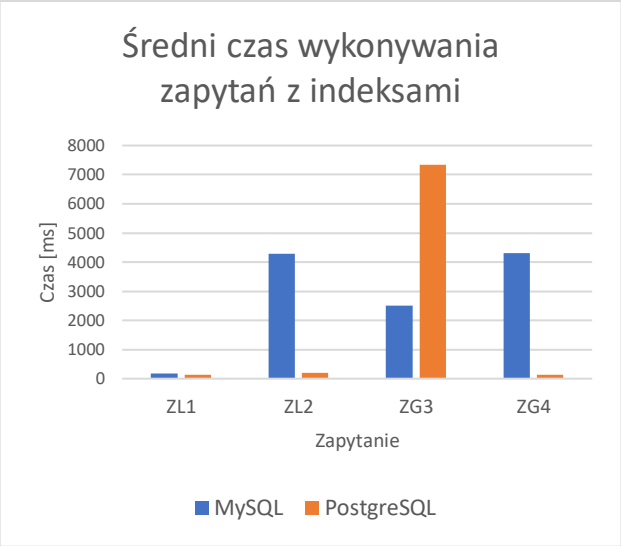
*Tabela 1. Czasy wykonywania zapytań ZL1, ZL2, ZG3, ZG4 [ms]*

| | ZL1 | | ZL2 | | ZG3 | | ZG4 | |
|---|---|---|---|---|---|---|---|---|
| Bez indeksów | Min | Średnia | Min | Średnia | Min | Średnia | Min | Średnia |
| MySQL | 1865 | 1882,2 | 797 | 813,6 | 2578 | 2591,1 | 812 | 815,2 |
| PostgreSQL | 130 | 142,2 | 210 | 216 | 7301 | 7359,5 | 131 | 138,7 |
| Z indeksami | | | | | | | | |
| MySQL | 173 | 181,5 | 4234 | 4286,2 | 2421 | 2511,2 | 4250 | 4299,2 |
| PostgreSQL | 130 | 135,2 | 204 | 208,6 | 7260 | 7330,3 | 129 | 137,8 |

Wyniki średnich pomiarów czasów z Tabeli 1 w formie graficznej przedstawiono za pomocą *Wykresów 1-2*.




*Wykres 1. Średni czas wykonywania zapytań bez indeksów na podstawie danych z Tabeli 1.*




*Wykres 2. Średni czas wykonywania zapytań z indeksami na podstawie danych z Tabeli 1.*

## 6. Wnioski

Na podstawie powyższych danych można stwierdzić, że:

- zastosowanie indeksów w przypadku tabeli w postaci znormalizowanej składającej się ze złączeń kilku tabel przeważnie wpłynęło negatywnie na wydajność przetwarzania zapytań.
- Normalizacja danych w większości przypadków prowadzi do spadku wydajności wykonywania zapytań.
- W większości przypadków zapytania zostały wykonane szybciej przy wykorzystaniu systemu *PostgreSQL* niż w przypadku *MySQL*. Nie należy jednak ignorować faktu, że w przypadku zapytania do formy zdenormalizowanej tabeli, gdzie złączenie było wykonywane poprzez zagnieżdżenie skorelowane zapytanie w systemie *PostgreSQL* wykonywało się blisko 3 razy dłużej niż w *MySQL*. Nie można zatem jednoznacznie porównać wymienionych systemów zarządzania bazami danych.

## 7. Bibliografia

- Praca bazuje na artykule P. mgr inż. Łukasza Jajeśnicy oraz P. dr hab. Inż. Adama Piórkowskiego „Wydajność złączeń i zagnieżdżeń dla schematów znormalizowanych i zdenormalizowanych”

- https://pl.wikipedia.org/wiki/Tabela_stratygraficzna