Paweł Dela, Student Geoinformatyki

Akademia Górniczo-Hutnicza w Krakowie

10.06.2022

Wydajność złączeń i zagnieżdżeń dla schematów znormalizowanych i zdenormalizowanych

W tej pracy przedstawię wyniki badania wydajności złączeń i zagnieżdżeń dla schematów znormalizowanych i zdenormalizowanych. Jako przykładu użyję tabeli geochronologicznej zawierającej nazwy kolejno: eonów, er, okresów, epok oraz pięter dla danych okresów geologicznych na Ziemi. Każda tabela zawiera identyfikator własny będący kluczem głównym, własną nazwę oraz identyfikator jednostki geochronologicznej w której się zawiera (z wyjątkiem eonów, z racji na to że jest to pierwsza tabela).

Konfiguracja sprzętowa:

- CPU: Intel Core i5-9600k 3.70GHz

- RAM: 16 GB DDR4

- SSD: SAMSUNG 870 QVO 1TB

- System operacyjny: Windows 10 Pro

Systemy zarządzania bazami danych:

- PostgreSQL wersja 14.2

-MySQL wersja Community 8.0.29.0

Wykonanie testów:

Do przetestowania użyto czterech zapytań sprawdzających wydajność oraz dodatkowej tabeli *Milion* złożonej z miliona rekordów (wartości od 0 do 999 999). Do uzyskania właściwych wyników każde zapytanie jest uruchamiane 5 krotnie a wartości skrajne odrzucane. Z wyników przyjętych wylicza się średnią oraz minimum.

Zapytania prezentują się następująco:

1) Celem zapytania nr 1 jest złączenie tabeli *Milion* z tabelą geochronologiczną w postaci zdenormalizowanej, przy czym do warunku złączenia dodano operacje modulo aby dopasować wartości złączanych kolumn:

SELECT COUNT(*) FROM Milion INNER JOIN GeoTabela ON (mod(Milion.liczba,68)=(GeoTabela.id_pietro));

2) Celem zapytania nr 2 jest złączenie tabeli *Milion* z tabelą geochronologiczną w postaci znormalizowanej, reprezentowaną przez złączenia pięciu tabel:

SELECT COUNT(*) FROM Milion INNER JOIN GeoPietro ON (mod(Milion.liczba,68)=GeoPietro.id_pietro) NATURAL JOIN GeoEpoka NATURAL JOIN GeoOkres NATURAL JOIN GeoEra NATURAL JOIN GeoEon;

3) Celem zapytania nr 3 jest złączenie tabeli *Milion* z tabela geochronologiczną w postaci zdenormalizowanej, przy czym złączenie jest wykonywane poprzez zanieżdżenie skorelowane:

SELECT COUNT(*) FROM Milion WHERE mod(Milion.liczba,68) IN (SELECT id_pietro FROM GeoTabela WHERE mod(Milion.liczba,68)=(id_pietro));

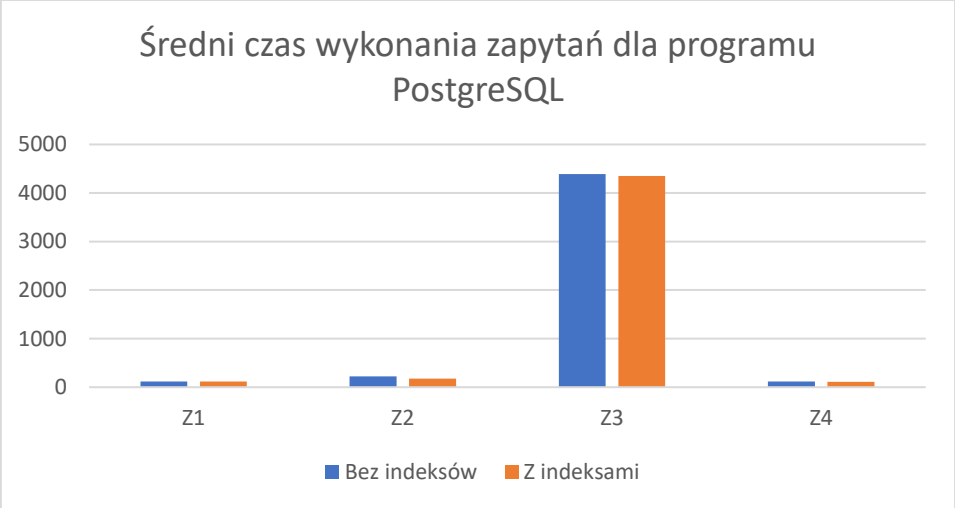
4) Celem zapytania nr 4 jest złącznie tabeli *Milion* z tabelą geochronologiczną w postaci znormalizowanej, przy czym złączenie jest wykonywane poprzez zagnieżdżenie skorelowane, a zapytanie wewnętrzne jest złączeniem tabel poszczególnych jednostek geochronologicznych:

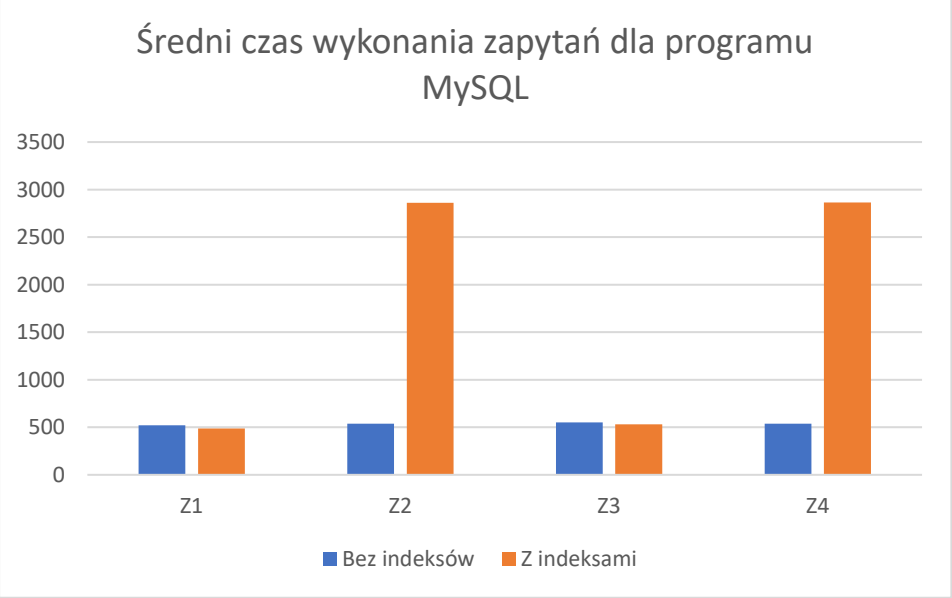
SELECT COUNT(*) FROM Milion WHERE mod(Milion.liczba,68) IN (SELECT GeoPietro.id_pietro FROM GeoPietro NATURAL JOIN GeoEpoka NATURAL JOIN GeoOkres NATURAL JOIN GeoEra NATURAL JOIN GeoEon);

Wyniki testów:

Wyniki liczone były w milisekundach dla obu programów i wszystkich zapytań. Każde zapytanie zostało wykonane 5 razy a z wyników zostało wyznaczone minimum oraz wyliczona średnia.

BEZ INDEKSÓW				Z INDEKSAMI				NR ZAPYTANIA
MySQL		PostgreSQL		MySQL		PostgreSQL		
MIN	AVG	MIN	AVG	MIN	AVG	MIN	AVG	
516	522	117	120	484	487	114	115	Z1
531	537	214	220	2844	2862	169	177	Z2
546	553	4380	4390	531	532	4320	4353	Z3
531	538	115	119	2844	2866	103	110	Z4





Wnioski:

- Postać zdenormalizowana jest w większości przypadków wydajniejsza (z wyjątkiem zapytania 3 i 4 w PostgreSQL)
- Do wykonania zagnieżdżenia potrzeba więcej czasu
- Program PostgreSQL pokazuje drobne poprawienie się czasu wykonywania zapytań z ideksem w porównaniu do tych samych zapytań bez indeksów co wskazuje na opymalizacje złączeń w tym programie
- Program MySQL wykonuje zapytania w postaci znormalizowanej dużo dłużej od tych zdenormalizowanych gdy są dodane indeksy. Bez indeksów różnica jest jedynie zauważalna