**1. Konfiguracja sprzętowa i programowa**

Wszystkie testy omówione w niniejszym artykule wykonano na komputerze o następujących parametrach:

CPU: Intel® Core ™ i5-8265U CPU @ 1,6 GHz

RAM: Pamięć 8,00 GB

HDD: WDC PC SN520 SDAPNUW-512G-1014

S.O.: Windows 10 Home

Jako systemy zarządzania bazami danych wybrano oprogramowanie wolno dostępne:

MySQL, wersja Community Server 5.1.42,

PostgreSQL, wersja 13.

Testy wykonywano wielokrotnie na komputerze dla każdego systemu zarządzania bazą danych, przy czym w trakcie testów na danym komputerze zainstalowany był tylko jeden z nich, kolejność instalacji była krzyżowa.

**1.2. Kryteria testów**

W teście wykonano szereg zapytań sprawdzających wydajność złączeń i zagnieżdżeń z tabelą geochronologiczną w wersji zdenormalizowanej i znormalizowanej.

* Zapytanie 1 (1 ZL), którego celem jest złączenie syntetycznej tablicy miliona wyników z tabelą geochronologiczną w postaci zdenormalizowanej, przy czym do warunku złącze-nia dodano operację modulo, dopasowującą zakresy wartości złączanych kolumn:

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

* Zapytanie 2 (2 ZL), którego celem jest złączenie syntetycznej tablicy miliona wyników 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;

* Zapytanie 3 (3 ZG), którego celem jest złączenie syntetycznej tablicy miliona wyników z tabelą geochronologiczną w postaci zdenormalizowanej, przy czym złączenie jest wy-konywane poprzez zagnieżdżenie skorelowane:

SELECT COUNT(\*) FROM Milion WHERE mod(Milion.liczba,68)=

(SELECT id\_pietro FROM GeoTabela WHERE mod(Milion.liczba,68)=(id\_pietro));

* Zapytanie 4 (4 ZG), którego celem jest złączenie syntetycznej tablicy miliona wyników z tabelą geochronologiczną w postaci znormalizowanej, przy czym złączenie jest wyko-nywane poprzez zagnieżdżenie skorelowane, a zapytanie wewnętrzne jest złączeniem ta-bel poszczególnych jednostek geochronologicznych:

SELECT COUNT(\*) FROM Milion WHERE mod(Milion.liczba,68)=

(SELECT GeoPietro.id\_pietro FROM GeoPietro NATURAL JOIN GeoEpoka NATURAL JOIN GeoOkres NATURAL JOIN GeoEra NATURAL JOIN GeoEon;

**2. Wyniki testów**

Czasy wykonania zapytań 1 ZL, 2 ZL, 3 ZG i 4 ZG [s]

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 1 ZL | | 2 ZL | | 3 ZG | | 4 ZG | |
|  | MIN | AVG | MIN | AVG | MIN | AVG | MIN | AVG |
| BEZ INDEKSÓW |  |  |  |  |  |  |  |  |
| PostgreSQL | 0,26 | 0,2922 | 0,346 | 0,382 | 13,23 | 13,694 | 0,208 | 0,2432 |
| MySQL | 0,12204 | 0,14221 | 0,07841 | 0,11137 | 3,82551 | 4,13567 | 0,10862 | 0,1797 |
| Z INDEKSAMI |  |  |  |  |  |  |  |  |
| PostgreSQL | 0,221 | 0,2446 | 0,282 | 0,317 | 13,049 | 13,79 | 0,203 | 0,2294 |
| MySQL | 0,05782 | 0,06333 | 0,05385 | 0,06487 | 1,56753 | 1,71702 | 0,06314 | 0,07919 |

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Wykres 1. Porównanie czasów dla PostgeSQL i MySQL bez indeksów.  Wykres 2. Porównanie czasów PostgreSQL z My SQL z indeksami.  Wykres 3. Porównanie czasów dla samego PostrgeSQLa.  Wykres 4. Porównanie czasów dla samego MySQL. | | | | | | | | | | | |
|  | | |  | | |  | | |  | | |
|  |  |  | |  |  | |  |  | |  |  |
|  |  |  | |  |  | |  |  | |  |  |
|  |  |  | |  |  | |  |  | |  |  |
|  | | | | | | | | | | | |
|  |  |  | |  |  | |  |  | |  |  |
|  |  |  | |  |  | |  |  | |  |  |

**3. Wnioski.**

Na podstawie przeprowadzonej analizy możemy stwierdzić, że wersja z indeksami jest wydajniejsza zarówno dla PostgreSQL jak i dla MySQL.

W przypadku opcji bez indeksów jak i z indeksami MySQL osiąga dużo lepsze wyniki czasowe. Jest szybszy we wszystkich zapytaniach, jakie zostały uruchomione. Szczególnie jest to zauważalne podczas wykonywania się zapytania 3 ZG, gdzie równica czasowa jest dosyć duża, równa około 12 sekund.

PostgreSQL w obu przypadkach (dla z i bez indeksów) osiągnął porównywalne wyniki czasowe, które różnią się od siebie mniej więcej 2 ms.

MySQL osiągnął najbardziej zróżnicowane wyniki czasowe. Czas wykonywania się zapytań dla opcji bez indeksów jest ponad dwa razy dłuższy niż dla opcji z indeksami.