Porównanie wydajności złączeń i zagnieżdżeń dla schematów znormalizowanych i zdenormalizowanych w MySql i PostgreSql.

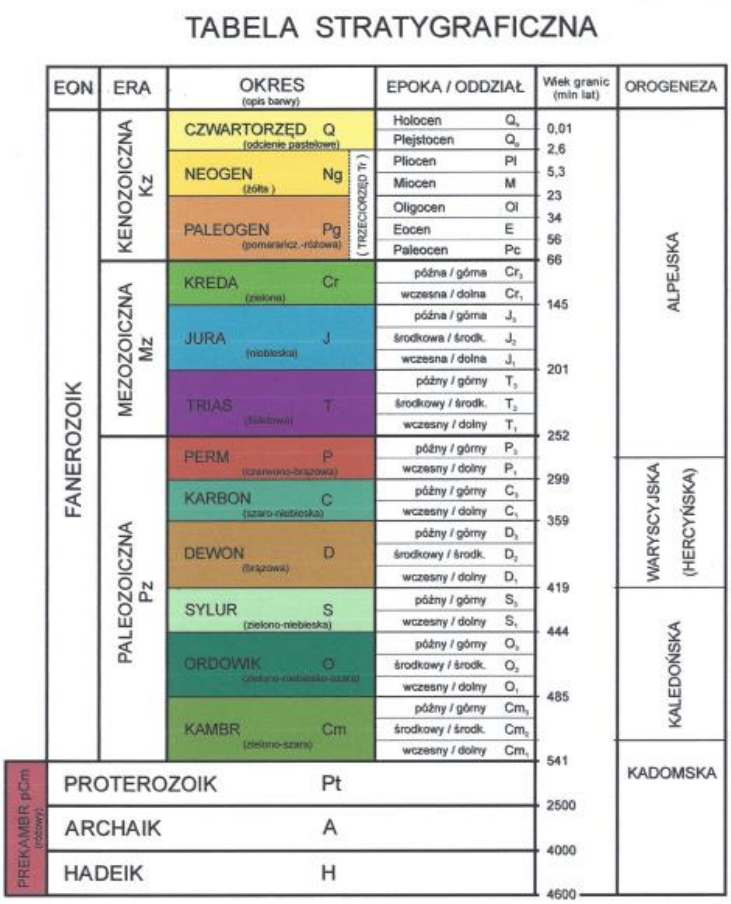
Piotr Powroźnik

1. *Wstęp*

Celem niniejszej pracy jest porównanie wydajności zapytań dla schematów znormalizowanych i zdenormalizowanych dla *odwzorowując pracę* Łukasza Jaśnica i Adama Piórkowskiego *z 2010r.*

2. *Tabela geochronologiczna*

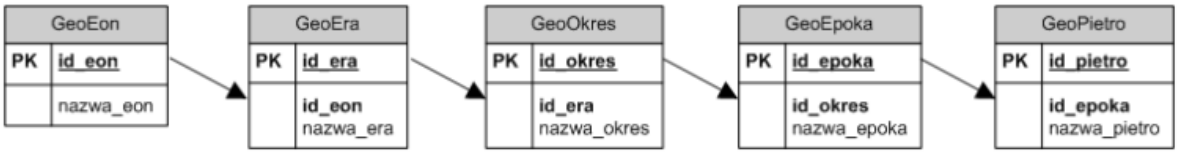
Tabela geochronologiczna przedstawia schemat przebiegu historii Ziemi na podstawie następstwa procesów geologicznych i układu warstw skalnych. W pracy przyjęto tabele stratygraficzną pochodzącą z CBDG.



Rys 0 International stratigraphic chart 2015/1

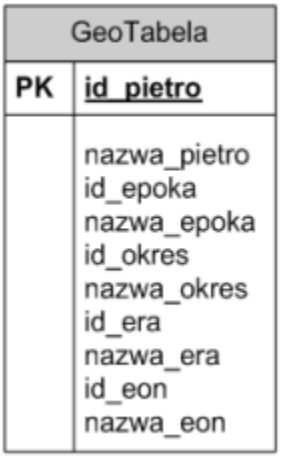
Tabele geochronologiczną przedstawiona na 2 sposoby:

- W postaci znormalizowanej (schemat płatka śniegu):



Rys 1 Tabela geochronologiczna znormalizowana

- W postaci zdenormalizowanej (schemat gwiazdy):



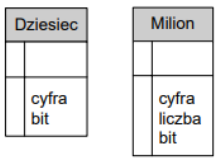
Rys 2 Tabela geochronologiczna zdenormalizowana

3. *Konfiguracja sprzętowa i programowa*

Testy wykonano na komputerze o następujących parametrach:
CPU: Intel Core i5-8400, 2.8 GHz
GPU: Nvidia Geforce Gtx 1050 Ti
RAM: DDR 4, 8Gb, 2400 MHz
SSD: SSDPR-CX400-256
S.O. Windows 10
MySql, wersja Community Server 8.0
PostgreSql, wersja 15.3

4. *Testy wydajności*

W celu sprawdzenia wydajności wykorzystano 4 zapytania w których łączono dane z tabeli geochronologicznej z syntetycznymi danymi z tabeli Milion, wypełnionej liczbami naturalnymi od 0 do 999 999. Tabela Milion została utworzona na podstawie auto złączenia tabeli Dziesięć wypełnionej liczbami od 0 do 9;



Rys 3 Schemat tabeli Milion oraz tabeli Dziesiec

Wszystkie testy złączeń przeprowadzono najpierw bez nałożonych indeksów na kolumny danych, a następnie po nałożeniu indeksów na wszystkie kolumny biorące udział w złączeniu.

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

```
SELECT COUNT(*) FROM geo.Milion
INNER JOIN geo.GeoTabela ON (mod(geo.Milion.Liczba,68) = (geo.GeoTabela.IdPietro));
```

- 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 geo.Milion
INNER JOIN geo.GeoPietro ON (mod(Milion.liczba,68) = GeoPietro.IdPietro)
NATURAL JOIN geo.GeoEpoka
NATURAL JOIN geo.GeoOkres
NATURAL JOIN geo.GeoEra
NATURAL JOIN geo.GeoEon;
```

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

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

- Zapytanie 4 (4 ZL), którego celem jest złączenie syntetycznej tablicy miliona wyników 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 geo.Milion
WHERE mod(Milion.Liczba,68) IN
(SELECT geo.GeoPietro.IdPietro FROM geo.GeoPietro
NATURAL JOIN geo.GeoEpoka
NATURAL JOIN geo.GeoOkres
NATURAL JOIN geo.GeoEra
NATURAL JOIN geo.GeoEon);
```

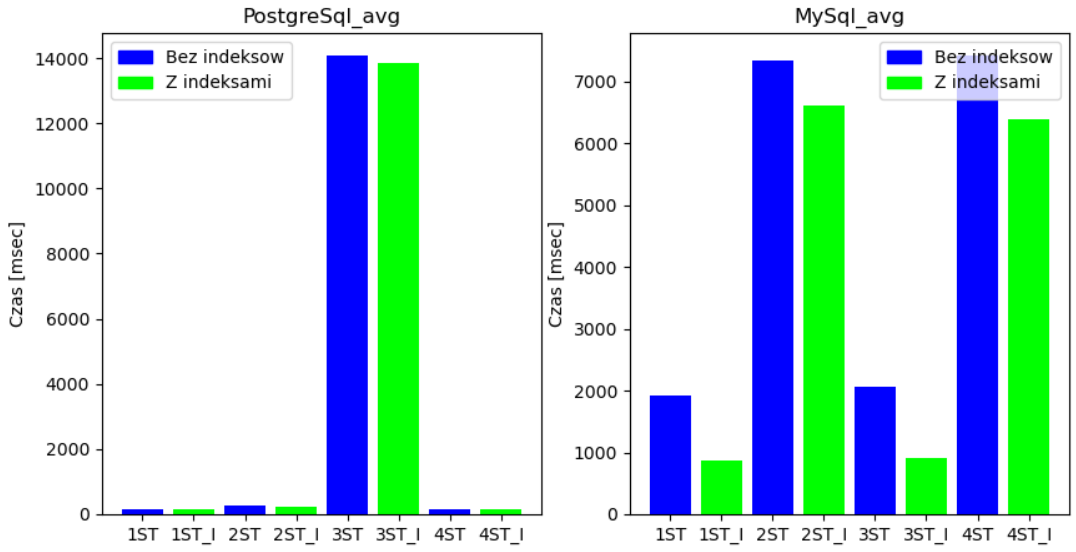
5. *Wyniki*

Każde zapytanie wykonano 12 krotnie dla wersji bez indeksów oraz z indeksami.
Z otrzymanych danych usunięto po 1 wartości odstającej, a następnie przedstawiono w postaci wykresów.

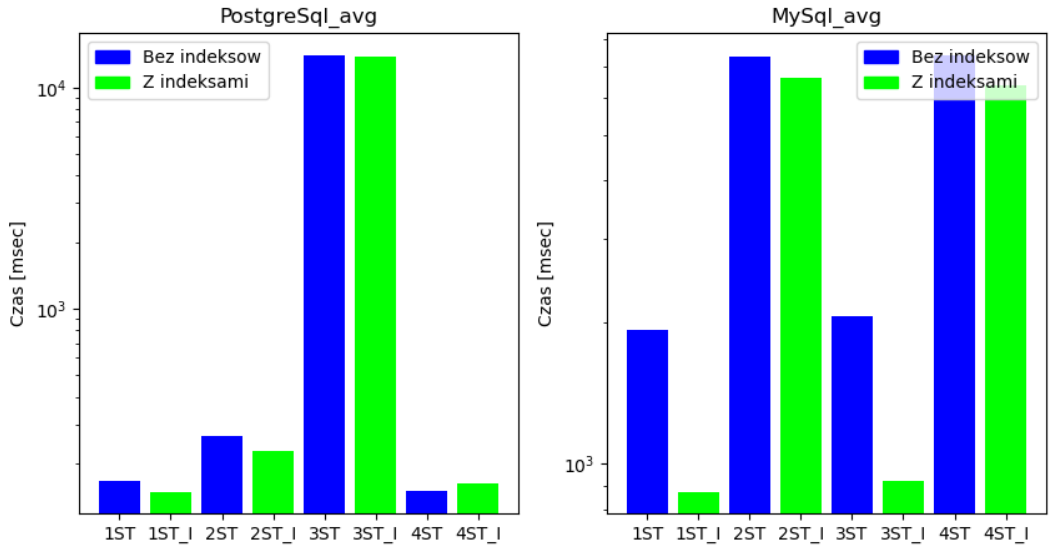
Czasy wykonania zapytań 1ZL, 2ZL, 3ZL, 4ZL [ms]

czas[ms]	1ZL		2ZL		3ZL		4ZL	
Bez indeksów	Min	Śr	Min	Śr	Min	Śr	Min	Śr
MySql	1890	1925	7234	7325	1953	2056	7328	7408
PostgreSql	143	168	250	267	13596	14067	147	151
Z indeksami								
MySql	859	874	6500	6597	906	917	6328	6383
PostgreSql	140	149	202	228	13564	13851	144	162

Wyniki analizy przedstawiono za pomocą wykresów ze skalą liniową oraz logarytmiczną.



Rys 4 Wyniki testów, skala liniowa



Rys 5 Wyniki testów, skala logarytmiczna

6. *Wnioski*

Z otrzymanych wyników można wyciągnąć następujące wnioski:

- *Czas wykonywania 3 zapytania (wykorzystującego zagnieżdżenie skorelowane) czas był znacząco mniejszy w przypadku MySql, natomiast w przypadku pozostałych 3 zapytań czasy wykonywania były dużo dłuższe niż w PostgreSql.*
- *Utworzenie indeksów nie poprawiło w znaczący sposób wydajności czasów zapytań w przypadku PostgreSql.*
- *W przypadku MySql dodanie indeksów znacznie poprawiło czas wykonywania 1 i 3 zapytania (wykorzystujących postać znormalizowaną), natomiast w przypadku 2 i 4 zapytania (wykorzystujących postać znormalizowaną) poprawa jest minimalna.*

Podsumowując normalizacja w większości przypadków powoduje spadek wydajności zapytań, ale ułatwia przechowywanie danych redukując szanse na powstanie niespójności danych, ułatwia utrzymanie i modyfikacje w bazie danych oraz poprawia przejrzystość przechowywanych danych.