# Ćwiczenia 10 - Analiza wydajności schematów normalizowanych i denormalizowanych oraz indeksów

## 1.Wprowadzenie

Celem ćwiczeń jest analiza wydajności tabel w postaci zdenormalizowanej i znormalizowanej oraz wpływ nałożenia indeksów na kolumny. Testy wydajności dla złączeń i zagnieżdżeń będą przeprowadzane na systemach zarządzania bazami danych SQL Servel Managment Studio oraz PostgreSQL. Eksperyment został otworzony z pracy „Wydajność złączeń i zagnieżdżeń dla schematów znormalizowanych i zdenormalizowanych” Łukasz JAJEŚNICA, Adam PIÓRKOWSKI. (7. Źródla 1)

Normalizacja baz danych ma na celu usunięcie powtarzającyh się danych z bazy. Dane nie są usuwanie ale zmienia się schemat bazy, układ danych i relacje pomiędzy nimi.

## 2.Utworzenie bazy danych

Pierwszym krokiem było utworzenie bazy danych GeoChronology. Znalazły się w niej tabele znormalizowanej wersji tabeli geochronologicznej (GeoEon, GeoEra, GeoOkres, GeoEpoka oraz GeoPiętro), zdenormalizowana wersja tabeli geochronologicznej (GeoTabela) oraz pomocnicze tabele (Dziesiec i Milion). Tabela Milion zawiera liczby od 0 do 999 999, które będą użyte w testach wydajnościowych. Następnie tabele geochronologiczne zostały wypełnione danymi wzgodnie z *Tabelą 2.1*. GeoPiętra zostały uzupełnione według rozszerzonej wersji tabeli dostępnej w Internecie (7.Źródła 2). Kod wykonania tych czynności w SQL Server Management Studio oraz PostgreSQL znajduje się w *Tabeli 2.2* .

*Tabela 2.1*

Obraz zawierający tekst, numer, Czcionka, zrzut ekranu

Opis wygenerowany automatycznie

*Tabela 2.2*

|  |  |
| --- | --- |
| **Utworzenie tabel i wypełnienie ich danymi** | |
| Wersja SQL Server Management Studio | Wersja PostgreSQL |
| CREATE DATABASE GeoChronology;  GO  USE GeoChronology;  --Znormalizowany schemat tabeli geochronologicznej  CREATE TABLE GeoEon (  id\_eon INT PRIMARY KEY,  nazwa\_eon VARCHAR(100) NOT NULL  );  CREATE TABLE GeoEra (  id\_era INT PRIMARY KEY,  id\_eon INT NOT NULL,  nazwa\_era VARCHAR(100) NOT NULL,  );  CREATE TABLE GeoOkres (  id\_okres INT PRIMARY KEY,  id\_era INT NOT NULL,  nazwa\_okres VARCHAR(100) NOT NULL,  );  CREATE TABLE GeoEpoka (  id\_epoka INT PRIMARY KEY,  id\_okres INT NOT NULL,  nazwa\_epoka VARCHAR(100) NOT NULL,  );  CREATE TABLE GeoPietro (  id\_pietro INT PRIMARY KEY,  id\_epoka INT NOT NULL,  nazwa\_pietro VARCHAR(100) NOT NULL,  );  --Dodanie kluczy obcych za pomocą ALTER TABLE  ALTER TABLE GeoEra  ADD CONSTRAINT FK\_GeoEra\_GeoEon FOREIGN KEY (id\_eon) REFERENCES GeoEon(id\_eon);  GO  ALTER TABLE GeoOkres  ADD CONSTRAINT FK\_GeoOkres\_GeoEra FOREIGN KEY (id\_era) REFERENCES GeoEra(id\_era);  GO  ALTER TABLE GeoEpoka  ADD CONSTRAINT FK\_GeoEpoka\_GeoOkres FOREIGN KEY (id\_okres) REFERENCES GeoOkres(id\_okres);  GO  ALTER TABLE GeoPietro  ADD CONSTRAINT FK\_GeoPietro\_GeoEpoka FOREIGN KEY (id\_epoka) REFERENCES GeoEpoka(id\_epoka);  GO  --Zdenormalizowany schemat tabeli geochronologicznej  CREATE TABLE GeoTabela (  id\_pietro INT PRIMARY KEY,  nazwa\_pietro VARCHAR(100) NOT NULL,  id\_epoka INT NOT NULL,  nazwa\_epoka VARCHAR(100) NOT NULL,  id\_okres INT NOT NULL,  nazwa\_okres VARCHAR(100) NOT NULL,  id\_era INT NOT NULL,  nazwa\_era VARCHAR(100) NOT NULL,  id\_eon INT NOT NULL,  nazwa\_eon VARCHAR(100) NOT NULL  );  GO  --Wypełnienie tabeli danymi  INSERT INTO GeoEon (id\_eon, nazwa\_eon)  VALUES  (1, 'FANEROZOIK');  GO  INSERT INTO GeoEra (id\_era, id\_eon, nazwa\_era)  VALUES  (1, 1,'Kenzoik'),  (2, 1,'Mezozoik'),  (3, 1,'Paleozoik');  GO  INSERT INTO GeoOkres(id\_okres, id\_era, nazwa\_okres)  VALUES  (1, 1,'Czwartorząd'),  (2, 1,'Trzeciorząd Neogen'),  (3, 1,'Trzeciorząd Paleogen'),  (4, 2,'Kreda'),  (5, 2,'Jura'),  (6, 2,'Trias'),  (7, 3,'Perm'),  (8, 3,'Karbon'),  (9, 3,'Dewon');  GO  INSERT INTO GeoEpoka(id\_epoka, id\_okres, nazwa\_epoka)  VALUES  (1, 1,'Halocen'),  (2, 1,'Plejstocen'),  (3, 2,'Pliocen'),  (4, 2,'Miocen'),  (5, 3,'Oligocen'),  (6, 3,'Eocen'),  (7, 3,'Paleocen'),  (8, 4,'Górna'),  (9, 4,'Dolna'),  (10, 5,'Górna'),  (11, 5,'Środkowa'),  (12, 5,'Dolna'),  (13, 6,'Górna'),  (14, 6,'Środkowa'),  (15, 6,'Dolna'),  (16, 7,'Górny'),  (17, 7,'Dolny'),  (18, 8,'Górny'),  (19, 8,'Dolny'),  (20, 9,'Górny'),  (21, 9,'Środkowy'),  (22, 9,'Dolny');  GO  INSERT INTO GeoPietro (id\_pietro, nazwa\_pietro, id\_epoka)  VALUES  (1, 'Gelas', 3),  (2, 'Piacent', 3),  (3, 'Zankl', 3),  (4, 'Mesyn', 4),  (5, 'Torton', 4),  (6, 'Serravall', 4),  (7, 'Lang', 4),  (8, 'Burdigal', 4),  (9, 'Akwitan', 4),  (10, 'Szat', 5),  (11, 'Rupel', 5),  (12, 'Priabon', 6),  (13, 'Bartoni', 6),  (14, 'Lutet', 6),  (15, 'Iperz', 6),  (16, 'Tanet', 7),  (17, 'Zeland', 7),  (18, 'Dan', 7),  (19, 'Mastrycht', 8),  (20, 'Kampan', 8),  (21, 'Santon', 8),  (22, 'Kon', 8),  (23, 'Turon', 8),  (24, 'Cenoman', 8),  (25, 'Alb', 9),  (26, 'Apt', 9),  (27, 'Barrem', 9),  (28, 'Hoteryw', 9),  (29, 'Walanzyn', 9),  (30, 'Berias', 9),  (31, 'Tyton', 10),  (32, 'Kimeridż', 10),  (33, 'Oksford', 10),  (34, 'Kelowej', 11),  (35, 'Baton', 11),  (36, 'Bajos', 11),  (37, 'Aalen', 11),  (38, 'Toark', 12),  (39, 'Pliensbach', 12),  (40, 'Synemur', 12),  (41, 'Hetang', 12),  (42, 'Retyk', 13),  (43, 'Noryk', 13),  (44, 'Karnik', 13),  (45, 'Ladin', 14),  (46, 'Anizyk', 14),  (47, 'Olenek', 15),  (48, 'Ind', 15),  (49, 'Tatar', 16),  (50, 'Kazań', 16),  (51, 'Ufa', 16),  (52, 'Kungur', 17),  (53, 'Artinsk', 17),  (54, 'Sakmar', 17),  (55, 'Assel', 17),  (56, 'Gżel', 18),  (57, 'Kasimow', 18),  (58, 'Moskow', 18),  (59, 'Baszkir', 18),  (60, 'Serpuchow', 19),  (61, 'Wizen', 19),  (62, 'Turnej', 19),  (63, 'Famen', 20),  (64, 'Fran', 20),  (65, 'Żywet', 21),  (66, 'Eifel', 21),  (67, 'Ems', 22),  (68, 'Prag', 22),  (69, 'Lochkow', 22);  --Wypełnienie GeoTabeli danymi  INSERT INTO GeoTabela (id\_pietro, nazwa\_pietro, id\_epoka, nazwa\_epoka, id\_okres, nazwa\_okres, id\_era, nazwa\_era, id\_eon, nazwa\_eon)  SELECT  GeoPietro.id\_pietro,  GeoPietro.nazwa\_pietro,  GeoEpoka.id\_epoka,  GeoEpoka.nazwa\_epoka,  GeoOkres.id\_okres,  GeoOkres.nazwa\_okres,  GeoEra.id\_era,  GeoEra.nazwa\_era,  GeoEon.id\_eon,  GeoEon.nazwa\_eon  FROM  GeoPietro  JOIN  GeoEpoka ON GeoPietro.id\_epoka = GeoEpoka.id\_epoka  JOIN  GeoOkres ON GeoEpoka.id\_okres = GeoOkres.id\_okres  JOIN  GeoEra ON GeoOkres.id\_era = GeoEra.id\_era  JOIN  GeoEon ON GeoEra.id\_eon = GeoEon.id\_eon;  GO  -- Utworzenie tabel Dziesiec i Milion oraz wypełnienie ich danymi  CREATE TABLE Dziesiec (  cyfra INT,  bit INT  );  INSERT INTO Dziesiec (cyfra, bit) VALUES  (0, 0),  (1, 1),  (2, 0),  (3, 1),  (4, 0),  (5, 1),  (6, 0),  (7, 1),  (8, 0),  (9, 1);  CREATE TABLE Milion (  liczba INT,  cyfra INT,  bit INT  );  INSERT INTO Milion (liczba, cyfra, bit)  SELECT  a1.cyfra + 10 \* a2.cyfra + 100 \* a3.cyfra + 1000 \* a4.cyfra + 10000 \* a5.cyfra + 100000 \* 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; | --Znormalizowany schemat tabeli geochronologicznej  CREATE TABLE GeoEon (  id\_eon SERIAL PRIMARY KEY,  nazwa\_eon VARCHAR(100) NOT NULL  );  CREATE TABLE GeoEra (  id\_era SERIAL PRIMARY KEY,  id\_eon INT NOT NULL,  nazwa\_era VARCHAR(100) NOT NULL,  FOREIGN KEY (id\_eon) REFERENCES GeoEon(id\_eon)  );  CREATE TABLE GeoOkres (  id\_okres SERIAL PRIMARY KEY,  id\_era INT NOT NULL,  nazwa\_okres VARCHAR(100) NOT NULL,  FOREIGN KEY (id\_era) REFERENCES GeoEra(id\_era)  );  CREATE TABLE GeoEpoka (  id\_epoka SERIAL PRIMARY KEY,  id\_okres INT NOT NULL,  nazwa\_epoka VARCHAR(100) NOT NULL,  FOREIGN KEY (id\_okres) REFERENCES GeoOkres(id\_okres)  );  CREATE TABLE GeoPietro (  id\_pietro SERIAL PRIMARY KEY,  id\_epoka INT NOT NULL,  nazwa\_pietro VARCHAR(100) NOT NULL,  FOREIGN KEY (id\_epoka) REFERENCES GeoEpoka(id\_epoka)  );  --Wypełnienie tabeli danymi  INSERT INTO GeoEon (nazwa\_eon)  VALUES  ('FANEROZOIK');  INSERT INTO GeoEra (id\_eon, nazwa\_era)  VALUES  (1, 'Kenzoik'),  (1, 'Mezozoik'),  (1, 'Paleozoik');  INSERT INTO GeoOkres(id\_era, nazwa\_okres)  VALUES  (1, 'Czwartorząd'),  (1, 'Trzeciorząd Neogen'),  (1, 'Trzeciorząd Paleogen'),  (2, 'Kreda'),  (2, 'Jura'),  (2, 'Trias'),  (3, 'Perm'),  (3, 'Karbon'),  (3, 'Dewon');  INSERT INTO GeoEpoka(id\_okres, nazwa\_epoka)  VALUES  (1, 'Halocen'),  (1, 'Plejstocen'),  (2, 'Pliocen'),  (2, 'Miocen'),  (3, 'Oligocen'),  (3, 'Eocen'),  (3, 'Paleocen'),  (4, 'Górna'),  (4, 'Dolna'),  (5, 'Górna'),  (5, 'Środkowa'),  (5, 'Dolna'),  (6, 'Górna'),  (6, 'Środkowa'),  (6, 'Dolna'),  (7, 'Górny'),  (7, 'Dolny'),  (8, 'Górny'),  (8, 'Dolny'),  (9, 'Górny'),  (9, 'Środkowy'),  (9, 'Dolny');  INSERT INTO GeoPietro (nazwa\_pietro, id\_epoka)  VALUES  ('Gelas', 3),  ('Piacent', 3),  ('Zankl', 3),  ('Mesyn', 4),  ('Torton', 4),  ('Serravall', 4),  ('Lang', 4),  ('Burdigal', 4),  ('Akwitan', 4),  ('Szat', 5),  ('Rupel', 5),  ('Priabon', 6),  ('Bartoni', 6),  ('Lutet', 6),  ('Iperz', 6),  ('Tanet', 7),  ('Zeland', 7),  ('Dan', 7),  ('Mastrycht', 8),  ('Kampan', 8),  ('Santon', 8),  ('Kon', 8),  ('Turon', 8),  ('Cenoman', 8),  ('Alb', 9),  ('Apt', 9),  ('Barrem', 9),  ('Hoteryw', 9),  ('Walanzyn', 9),  ('Berias', 9),  ('Tyton', 10),  ('Kimeridż', 10),  ('Oksford', 10),  ('Kelowej', 11),  ('Baton', 11),  ('Bajos', 11),  ('Aalen', 11),  ('Toark', 12),  ('Pliensbach', 12),  ('Synemur', 12),  ('Hetang', 12),  ('Retyk', 13),  ('Noryk', 13),  ('Karnik', 13),  ('Ladin', 14),  ('Anizyk', 14),  ('Olenek', 15),  ('Ind', 15),  ('Tatar', 16),  ('Kazań', 16),  ('Ufa', 16),  ('Kungur', 17),  ('Artinsk', 17),  ('Sakmar', 17),  ('Assel', 17),  ('Gżel', 18),  ('Kasimow', 18),  ('Moskow', 18),  ('Baszkir', 18),  ('Serpuchow', 19),  ('Wizen', 19),  ('Turnej', 19),  ('Famen', 20),  ('Fran', 20),  ('Żywet', 21),  ('Eifel', 21),  ('Ems', 22),  ('Prag', 22),  ('Lochkow', 22);  --Zdenormalizowany schemat tabeli geochronologicznej  CREATE TABLE GeoTabela AS (SELECT \* FROM GeoPietro NATURAL JOIN GeoEpoka NATURAL  JOIN GeoOkres NATURAL JOIN GeoEra NATURAL JOIN GeoEon );  CREATE TABLE Dziesiec (  cyfra INT,  bit INT  );  -- Utworzenie tabel Dziesiec i Milion oraz wypełnienie ich danymi  CREATE TABLE Dziesiec (  cyfra INT,  bit INT  );  INSERT INTO Dziesiec (cyfra, bit) VALUES  (0, 0),  (1, 1),  (2, 0),  (3, 1),  (4, 0),  (5, 1),  (6, 0),  (7, 1),  (8, 0),  (9, 1);  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 ; |

## 3.Konfiguracja sprzętowa i programowa

Testy wydajnościowe zostały przeprowadzone na komputerze o następujących parametrach:

CPU: 12th Gen Intel® Core™ i5-12450H, 2 GHz

RAM: Pamięć DDR5 16GB (DRAM Frequency 2394.1 MHz)

SSD: SOLIDGM SSDPFINW512GZL

S.O.: Windows 11 Home

SQL Server Management Studio: 20

PostgreSQL: 16.3-2

## 4.Testy

W testach wykonano szereg zapytań sprawdzających wydajność złączeń oraz zapytań zagnieżdżonych. Test wykonano w dwóch wariantach

1.Bez nałożonych indeksów na kolumny (jedyne indeksowane kolumny to klucze główne poszczególnych tabel)

2.Z nałożonymi indeksami na wszytskie kolumny biorące udział w zapytaniach

Zapytania użyte do testów:

1 ZL – Złączenie tabeli Milion z tabela geochronologiczną w postaci zdenormalizowanej (GeoTabela). W każdym zapytaniu w warunku złączenia dodano operacje modulo ,która dopasowuje zakresy wartości łączonych kolumn.

2 ZL – Złączenie tabeli Milion z tabela geochronologiczną w postaci znormalizowanej reprezentowanej przez 5 tabel ( GeoPietro, GeoEpoka, GeoOkres, GeoEra, GeoEon).

3 ZG - Złączenie tabeli Milion z tabela geochronologiczną w postaci zdenormalizowanej (GeoTabela). Złączenie jest wykonywane poprzez zapytanie zagnieżdzone.

4 ZG - Złączenie tabeli Milion z tabela geochronologiczną w postaci znormalizowanej reprezentowanej przez 5 tabel ( GeoPietro, GeoEpoka, GeoOkres, GeoEra, GeoEon).

Kod przeprowadzonych testów wydajności znajduje się w *Tabeli 4.1*, nałożenie indeksów na kolumny biorące udział w zaptaniach znajdują się w *Tabeli 4.2*.

*Tabela 4.1*

|  |  |
| --- | --- |
| **Kod przeprowadzonych testów wydajności** | |
| Wersja SQL Server Management Studio | Wersja PostgreSQL |
| SET STATISTICS TIME ON;  --Zapytanie 1 (1 ZL)  SELECT COUNT(\*) FROM Milion INNER JOIN GeoTabela ON  (Milion.liczba % 68 =  GeoTabela.id\_pietro);  -- Zapytanie 2 (2 ZL)  SELECT COUNT(\*) FROM Milion  INNER JOIN GeoPietro ON (Milion.liczba % 68) = GeoPietro.id\_pietro  INNER JOIN GeoEpoka ON GeoPietro.id\_epoka = GeoEpoka.id\_epoka  INNER JOIN GeoOkres ON GeoEpoka.id\_okres = GeoOkres.id\_okres  INNER JOIN GeoEra ON GeoOkres.id\_era = GeoEra.id\_era  INNER JOIN GeoEon ON GeoEra.id\_eon = GeoEon.id\_eon;  -- Zapytanie 3 (3 ZG)  SELECT COUNT(\*) FROM Milion  WHERE (Milion.liczba % 68) = (SELECT id\_pietro FROM GeoTabela WHERE (Milion.liczba % 68) = id\_pietro);  -- Zapytanie 4 (4 ZG)  SELECT COUNT(\*)  FROM Milion  WHERE (Milion.liczba % 68) =  (  SELECT TOP 1 GeoPietro.id\_pietro  FROM GeoPietro  INNER JOIN GeoEpoka ON GeoPietro.id\_epoka = GeoEpoka.id\_epoka  INNER JOIN GeoOkres ON GeoEpoka.id\_okres = GeoOkres.id\_okres  INNER JOIN GeoEra ON GeoOkres.id\_era = GeoEra.id\_era  INNER JOIN GeoEon ON GeoEra.id\_eon = GeoEon.id\_eon  WHERE (Milion.liczba % 68) = GeoPietro.id\_pietro  );  GO  SET STATISTICS TIME OFF; | --Zapytanie 1 (1 ZL)  SELECT COUNT(\*) FROM Milion INNER JOIN GeoTabela ON  (mod(Milion.liczba,68)=(GeoTabela.id\_pietro));  -- Zapytanie 2 (2 ZL)  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)  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)  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 LIMIT 1); |

*Tabela 4.2*

|  |  |
| --- | --- |
| **Nałożenie indeksów na wszytskie kolumny biorące udział w złączeniu** | |
| Wersja SQL Server Management Studio | Wersja PostgreSQL |
| CREATE INDEX idx\_liczba ON Milion(liczba);  CREATE INDEX idx\_id\_eon ON GeoEon(id\_eon);  CREATE INDEX idx\_id\_era ON GeoEra(id\_era);  CREATE INDEX idx\_id\_okres ON GeoOkres(id\_okres);  CREATE INDEX idx\_id\_epoka ON GeoEpoka(id\_epoka);  CREATE INDEX idx\_id\_pietro ON GeoPietro(id\_pietro);  CREATE INDEX idx\_id\_pietro ON GeoTabela(id\_pietro);  CREATE INDEX idx\_id\_epoka ON GeoTabela(id\_epoka);  CREATE INDEX idx\_id\_okres ON GeoTabela(id\_okres);  CREATE INDEX idx\_id\_era ON GeoTabela(id\_era);  CREATE INDEX idx\_id\_eon ON GeoTabela(id\_eon); | CREATE INDEX inx\_liczba ON Milion(liczba);  CREATE INDEX idx\_id\_eon ON GeoEon(id\_eon);  CREATE INDEX idx\_id\_era ON GeoEra(id\_era);  CREATE INDEX idx\_id\_okres ON GeoOkres(id\_okres);  CREATE INDEX idx\_id\_epoka ON GeoEpoka(id\_epoka);  CREATE INDEX idx\_id\_pietro ON GeoPietro(id\_pietro);  CREATE INDEX idx\_id\_pietro\_tabela ON GeoTabela(id\_pietro);  CREATE INDEX idx\_id\_epoka\_tabela ON GeoTabela(id\_epoka);  CREATE INDEX idx\_id\_okres\_tabela ON GeoTabela(id\_okres);  CREATE INDEX idx\_id\_era\_tabela ON GeoTabela(id\_era);  CREATE INDEX idx\_id\_eon\_tabela ON GeoTabela(id\_eon); |

## 5.Wyniki testów

Każde zapytanie zostało powtórzone 20-krotnie, skrajne odstające wyniki zostały odrzucone. Czasy zapytań zostały zebrane w *Tabeli 5.1.*

*Tabela 5.1*

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Czasy wykonania zapytań 1 ZL 2ZL 3ZG 4ZG** | | | | | | | | | | | | |
|  | 1 ZL | | | 2 ZL | | | 3 ZL | | | 4 ZL | | |
| Bez indeksów | MIN | ŚR | MAX | MIN | ŚR | MAX | MIN | ŚR | MAX | MIN | ŚR | MAX |
| SQL SSMS | 82 | 106,85 | 137 | 78 | 95,15 | 148 | 82 | 110,25 | 154 | 2001 | 2046,4 | 2104 |
| PostgreSQL | 128 | 169,50 | 204 | 252 | 272,55 | 319 | 20344 | 21276,65 | 22111 | 102 | 124,4 | 211 |
| Z indeksami |  | | | | | | | | | | | |
| SQL SSMS | 43 | 95,6 | 459 | 72 | 91,95 | 119 | 64 | 85,15 | 119 | 1904 | 1968,45 | 2041 |
| PostgreSQL | 145 | 174,05 | 217 | 196 | 221,75 | 266 | 21345 | 21989,9 | 23524 | 98 | 139,45 | 227 |

W wersji wykorzystującej SQL Server Managment Studio w Pierwszym zapytaniu 1ZL I 2 ZL czyli nie zagnieżdżonym schemat tabeli znormalizowany jest szybszy. W zapytaniu zagnieżdzonym 3ZL i 4ZL schemat zdenormalizowany jest szybszy w przypadku braku indeksów 18,6 raza , z indeksami 23 razy. We wszytskich przypadkach zapytania bez nałożonych indeksów na kolumny biorące w nich udział są wolniejsze od tych z indeksami. Dodatkowo zapytania , w których biorą udział tabele w postaci znormalizowanej po nałożeniu indeksów stają się szybsze niż te wykorzystujące tabele zdenormalizowaną. Zapytanie 1ZL po nałożeniu indeksów staje się o 10,52% szybsze a 3ZL o 22,76%. Za to zapytania 2ZL I 4ZG stają się szybsze o około 3%.

W wersji wykorzystującej PostgreSQL w Pierwszym zapytaniu 1ZL I 2 ZL czyli nie zagnieżdżonym schemat tabeli zdenormalizowany jest szybszy. W zapytaniu zagnieżdzonym 3ZL i 4ZL schemat znormalizowany jest szybszy podnad 150 razy. We większości przypadków zapytania bez nałożonych indeksów na kolumny biorące w nich udział są szybsze od tych z indeksami.

Porównując zapytania w SQL Server Managment Studio i PostgreSQL jedynie w przypadku 4 zapyatnia zagnieżdżonego na tabeli znormalizowanej szybciej wykonuje się one w PostgreSQL, reszta zapytań szybsza jest w SQL Server Managment Studio. Największa różnica jest w zapytaniu 3ZG na tabeli zdenormalizowanej , zapytania w PostgreSQL wykonują się ponad 190 razy dłużej.

## 6.Podsumowanie

We większości przypadków tabela w postaci znormalizowanej jest szybsza.

We większości przypadków zagnieżdżenia skorelowane są wolniejsze w wykonaniu niż złączenia.

W systemie SQL Server Managment Studio we wszystkich przypadkach nałożenie indeksów przyspiesza zapytanie a w PostgreSQL nałożenie indeksów na kolumny spowalnia zapytania w większości wypadków.

## 7.Źródła

1. Łukasz JAJEŚNICA, Adam PIÓRKOWSKI Akademia Górniczo – Hutnicza, Katedra Geoinformatyki i Informatyki Stosowanej WYDAJNOŚĆ ZŁĄCZEŃ I ZAGNIEŻDŻEŃ DLA SCHEMATÓW ZNORMALIZOWANYCH I ZDENORMALIZOWANYCH1

2.http://stareaneksy.pwn.pl/historia\_ziemi/przyklady/?pokaz=tabela