

Indeksy - Karta pracy nr 4

Imię i Nazwisko: Oscar Teeninga

W ramach czwartych ćwiczeń z indeksów w MS SQL, macie Państwo do wykonania miniprojekt, w którym przeprowadzicie analizę wybranych indeksów na przygotowanym przez siebie zestawie danych.

Zadanie

Należy zaprojektować tabelę w bazie danych, lub wybrać dowolny schemat danych (poza używanymi na zajęciach), a następnie wypełnić ją danymi w taki sposób, aby zrealizować poszczególne punkty w analizie indeksów.

Do analizy, proszę uwzględnić następujące rodzaje indeksów:

- klastrowany (nie po kluczu głównym)
- Filtered Index (Indeks warunkowy)
- Memory-optimized hash index (z przynajmniej dwoma różnymi wartościami kubełków)

Analiza

Proszę przygotować zestaw zapytań do danych, które:

- wykorzystują poszczególne indeksy
- które przy wymuszeniu indeksu działają gorzej, niż bez niego (lub pomimo założonego indeksu, tabela jest w pełni skanowana)

Raport

Raport z projektu powinien zawierać:


- Schemat tabeli
- Opis danych (ich rozmiar, zawartość, statystyki)
- Trzy indeksy:
 - Opis indeksu
 - Przygotowane zapytania, wraz z wynikami z execution planów (screeny)
 - Komentarze do zapytań, ich wyników
 - Sprawdzenie, co proponuje Database Engine Tuning Advisor (porównanie czy udało się Państwu znaleźć odpowiednie indeksy do zapytania)

Raport proszę zacząć od kolejnej strony tego dokumentu (czerwone ramki nie są konieczne).

Schemat tabeli

Wzorowana na tabelach z <https://www.w3resource.com/sql/sql-table.php>.

```
CREATE TABLE CUSTOMER
(
  ID INT NOT NULL PRIMARY KEY,
  NAME VARCHAR(40) NOT NULL,
  CITY CHAR(35),
  COUNTRY VARCHAR(20) NOT NULL,
  GRADE INT,
  EMAIL VARCHAR(17) NOT NULL,
  PHONE VARCHAR(17) NOT NULL
);
```

CUSTOMER	
	ID
	NAME
	CITY
	COUNTRY
	GRADE
	EMAIL
	PHONE

Dane

Stworzone za pomocą <https://www.mockaroo.com>. Liczba wierszy: 1000.


```
insert into CUSTOMER (ID, NAME, CITY, COUNTRY, GRADE, EMAIL, PHONE) values (995,
'Goddart', 'Sampangan', 'Indonesia', '09', 'gmccardrm@umich.edu', '6287694379');
```

Index klastrowany Usunąłem bazowy klaster na ID, i zastąpiłem go na NAME i COUNTRY posortowane.


```
CREATE CLUSTERED INDEX Index_Customer_Name_Country
ON CUSTOMER(NAME DESC, COUNTRY DESC)
```

Przykładowe zapytanie gorzej korzystające z indeksu





```
SELECT NAME FROM CUSTOMER
WHERE COUNTRY = 'Poland' AND GRADE IS NULL
```



SELECT
Cost: 0 %




Clustered Index Scan (Clustered)
[CUSTOMER].[Index_Customer_Name_Cou...
Cost: 100 %
0.000s
19 of
26 (73%)

<input checked="" type="checkbox"/>	Database Name ▾	Object Name ▾	Recommendation ▾	Target of Recommendation	Details	Partition Scheme ▾	Size (KB)
<input checked="" type="checkbox"/>	DATABASETEENINGA	[dbo].[CUSTOMER]	drop	 Index_Customer_Grade_Country			
<input checked="" type="checkbox"/>	DATABASETEENINGA	[dbo].[CUSTOMER]	create	 _dta_index_CUSTOMER_7_1173579219_K5			8
<input checked="" type="checkbox"/>	DATABASETEENINGA	[dbo].[Customer_10]	drop	 Customer_Hash_10			
<input checked="" type="checkbox"/>	DATABASETEENINGA	[dbo].[Customer_100]	drop	 Customer_Hash_100			

Widzimy, że indeks został wykorzystany, natomiast w tym zapytaniu lepiej sprawdziłby się inny (wg. Analizy). Zgadzałoby się, ponieważ w zapytaniu korzystamy z GRADE, a więc należałoby stworzyć taki właśnie indeks.


Przykładowe zapytanie dobrze korzystające z indeksu

```
SELECT COUNTRY, NAME FROM CUSTOMER
WHERE COUNTRY = 'Poland' AND NAME = 'Terra'
```



SELECT

Cost: 0 %



Clustered Index Scan (Clustered)

[CUSTOMER].[Index_Customer_Name_Cou...

Cost: 100 %

0.000s




37 of

37 (100%)

Estimated improvement: 0%

Partition Recommendations

Index Recommendations

<input checked="" type="checkbox"/>	Database Name	Object Name	Recommendation	Target of Recommendation	Details	Partition Sch
<input checked="" type="checkbox"/>	DATABASETEENINGA	[dbo].[CUSTOMER]	drop	 Index_Customer_Grade_Country		
<input checked="" type="checkbox"/>	DATABASETEENINGA	[dbo].[Customer_10]	drop	 Customer_Hash_10		
<input checked="" type="checkbox"/>	DATABASETEENINGA	[dbo].[Customer_100]	drop	 Customer_Hash_100		

Analiza wykazała, że indeks jest ok i należy jedynie usunąć te nadmiarowe.

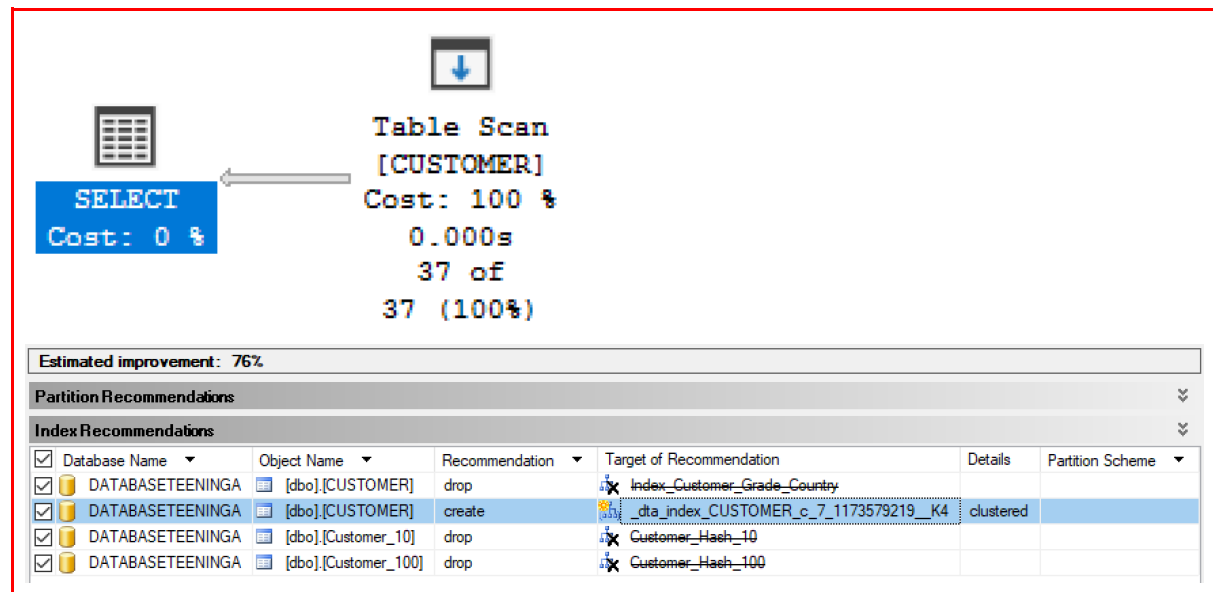
Indeks warunkowy

Jedno pole może być NULL i jest to GRADE. Dlatego taki postawiłem warunek:

```
CREATE INDEX Index_Customer_Grade_Country
ON CUSTOMER(GRADE, Country)
WHERE GRADE IS NOT NULL
```

Przykładowe zapytanie niekorzystające z indeksu

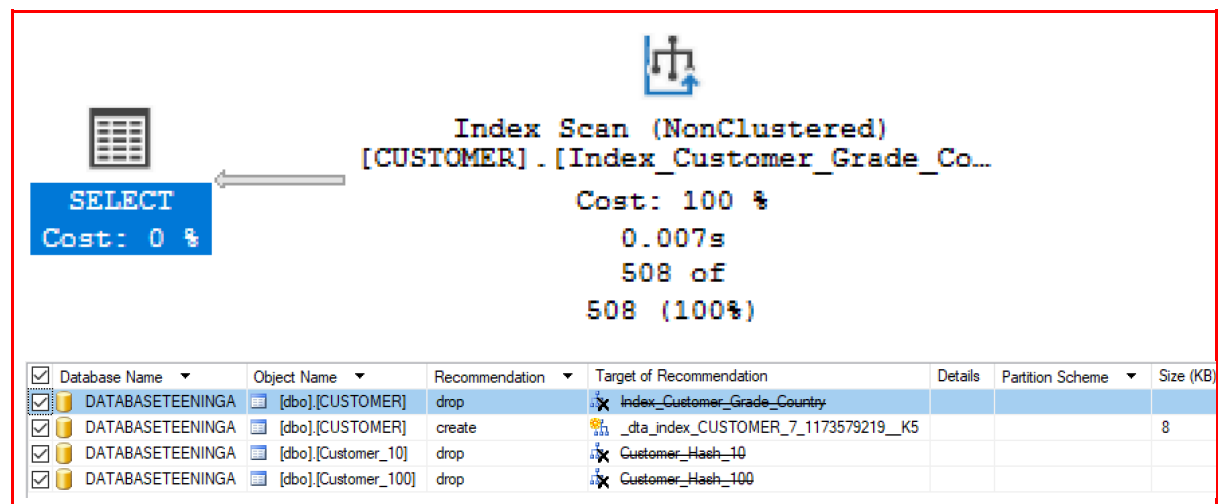
```
SELECT GRADE FROM CUSTOMER
WHERE COUNTRY = 'Poland'
```



Zwracane wartości GRADE mogą być NULL, więc nie można skorzystać z indeksu.

Przykładowe zapytanie korzystające z indeksu

```
SELECT GRADE FROM CUSTOMER
WHERE GRADE IS NOT NULL
```



Wybieramy tylko te wartości, których GRADE nie jest NULL, więc indeks uwzględnia wszystkie możliwe zwracane wartości. Analiza znalazła lepszy indeks, natomiast wynika to z tego, że indeks zawiera niepotrzebne z punktu widzenia zapytania pole Country, którego można by było się pozbyć.

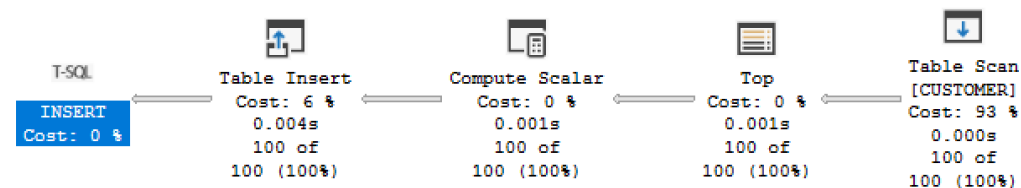
Memory-optimized hash index

Stworzyłem dwa kubelki: Customer_10 i Customer_100. Różnią się jednym polem – EMAIL/GRADE.

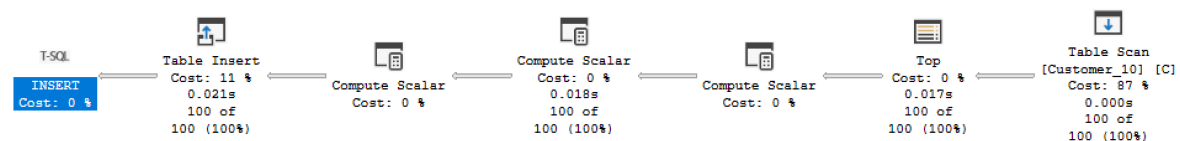
```
CREATE TABLE Customer_10(  
    ID INT NOT NULL PRIMARY KEY NONCLUSTERED IDENTITY(1,1),  
    NAME VARCHAR(40) NOT NULL,  
    CITY CHAR(35),  
    COUNTRY VARCHAR(20) NOT NULL,  
    GRADE INT,  
    PHONE VARCHAR(17) NOT NULL,  
    INDEX Customer_Hash_10 HASH ([NAME]) WITH (BUCKET_COUNT = 10)  
) WITH (  
    MEMORY_OPTIMIZED = ON,  
    DURABILITY = SCHEMA_AND_DATA  
)  
  
CREATE TABLE Customer_100(  
    ID INT NOT NULL PRIMARY KEY NONCLUSTERED IDENTITY(1,1),  
    NAME VARCHAR(40) NOT NULL,  
    CITY CHAR(35),  
    COUNTRY VARCHAR(20) NOT NULL,  
    EMAIL VARCHAR(50) NOT NULL,  
    PHONE VARCHAR(17) NOT NULL,  
    INDEX Customer_Hash_100 HASH ([NAME]) WITH (BUCKET_COUNT = 100)  
) WITH (  
    MEMORY_OPTIMIZED = ON,  
    DURABILITY = SCHEMA_AND_DATA  
)
```

Wstawiłem w Customer_10 200 wierszy z Customer oraz 100 wierszy do Customer_100 z Customer_10.

```
INSERT INTO Customer_10  
(NAME, CITY, COUNTRY, GRADE, PHONE)  
SELECT TOP(100) CUSTOMER.NAME, CUSTOMER.CITY, CUSTOMER.COUNTRY, CUSTOMER.GRADE, CUS-  
TOMER.PHONE  
FROM CUSTOMER  
GO 2
```



```
INSERT INTO Customer_100  
(NAME, CITY, COUNTRY, EMAIL, PHONE)  
SELECT TOP(100) C.NAME, C.CITY, C.COUNTRY, 'email@email.com', C.PHONE  
FROM Customer_10 C  
GO
```



Estimated improvement: 9%				
Partition Recommendations				
Index Recommendations				
<input checked="" type="checkbox"/>	Database Name	Object Name	Recommendation	Target of Recommendation
<input checked="" type="checkbox"/>	DATABASETEENINGA	[dbo].[CUSTOMER]	drop	Index_Customer_Grade_Country
<input checked="" type="checkbox"/>	DATABASETEENINGA	[dbo].[CUSTOMER]	create	_dta_index_CUSTOMER_7_1173579219__K2_3_4_5_7
<input checked="" type="checkbox"/>	DATABASETEENINGA	[dbo].[Customer_10]	drop	Customer_Hash_10
<input checked="" type="checkbox"/>	DATABASETEENINGA	[dbo].[Customer_100]	drop	Customer_Hash_100

Sprawdziłem zawartość kubełków.

	object name	index name	total_bucket_count	empty_bucket_count	empty_bucket_percent	avg_chain_length	max_chain_length
1	Customer_10	Customer_Hash_10	16	0	0	12	24
2	Customer_100	Customer_Hash_10	128	68	53	1	4

Zgodnie z oczekiwaniami, brakło nam miejsca w bucket'cie podczas kopiowania. W przypadku większego bucketu zostało nad 68 wolnego miejsca, co odpowiada 53% jego rozmiaru (w przybliżeniu).

Problemy

Musiałem zwiększyć pamięć podczas analizy Database Engine Tuning Advisor.

Advanced Tuning Options

☒ Define max. space for recommendations (MB): 30

☐ Include plan cache events from all databases

 Max. columns per index: 1023

 Online index recommendations

☒ All recommendations are offline

☐ Generate online recommendations where possible

☐ Generate only online recommendations

OK Cancel Help

Ustawienie stałej wartości 30MB rozwiązywał problem.