# Wyszukiwanie pełnotekstowe
# (Full tekst serach)

## Informacje wstępne: przykłady są stworzone w oparciu o bazę danych przesłaną w folderze

Jak sama nazwa wskazuje będziemy dziś zajmować się głównie bazami danych z danymi tekstowymi i tylko takie będą nas interesować.

Zacznijmy od krótkiego wstępu, zapewne każdy z nas zna funkcję LIKE, służy do odnajdywania danego słowa w klauzuli WHERE, to słowo może być poprzedzone dowolną ilością znaków oraz mogę te znaki wystąpić to tym słowie, ale zadajmy sobie pytanie czy to zapytanie jest efektywne? Otóż nie, stosując LIKE przeglądamy każdy wiersz z danymi szukając określonego wzorca, więc jak nie trudno wywnioskować w bazie danych która posiada setki tysięcy wierszy takie wyszukanie może nam zająć chwile czasu.

Czym są indeksy? Najprościej  mówiąc możemy je porównać do spisu treści w książce. Wiec szukając jakiegoś określonego elementu nie musimy już przeglądać całej tabeli bo budując indeks na przeszukiwanej tabeli nasze wyszukiwanie przebiega znacznie szybciej. Więc możemy zadać sobie pytanie czy powinniśmy tworzyć indeksy na każdej tabeli by przyspieszyć ewentualne wyszukiwanie wyników? Otóż nie, stworzony indeks zabiera nam przestrzeń na dysku więc zalecane jest stosowanie indeksów tylko na tabelach często przeszukiwanych.

Czym więc jest wyszukiwanie pełnotekstowe? Wyszukiwanie to pozwala wyszukiwać nam dane tesktowe zapisane w tabelach oraz co czasem może być pomocne FTS możmey stosować do przeszukiwania plików binarnych. A więc co mozmey zrobić za pomocą FTS? Możemy zastąpić funkcję LIKE dużo szybszymi zapytaniami jest to podstawowa funkcjnalność. Możemy wyszukać kilka słów, rózne formy gramatyczne danych słów, słowa bliskoznaczne oraz synonimy. Możemy równiesz szukać słów rozdzielonych określoną ilością innych słów, można też zwracać wagę wyszukanych słów/fraz tak aby zostały nam zwrócone najbardziej trafne wyszukania razem z ich warosćią/rankingiem.

Teraz omówię jak takie indeksy stworzyć, na samym początku musimy stworzyć katalog grupujący indeksy pełnotekstowe, potem należy stworzyć unikalny indeks na danaje kolumnie w tabeli którą chcemy przystosować do wyszukiwania pełnotekstowego. Indeksy pełnotekstowe nie maja swojej nazwy gdyż w jednej tabeli może istnieć tylko jeden taki indeks. Może być on utworzony na jednej jak i na kilku kolumnach. Indeksy takie możemy włączać jak i wyłączać. Kompletny przykład jak to zrobić znajduje się w pliku Przykłady sql.txt. Do zastosowania wyszukiwania będziemy wykorzystywać dwie funkcje stosowane w klauzuli WHERE: freetext oraz contains. W pliku z przykładami zamieściłem sporo przykładów działania tych funkcji. Ogólne działanie freetext jest takie że bez żadnych dodatkowych obostrzeń zawsze wyszukuje nas wszystkie formy fleksyjne danego wyrazu wraz z synonimami, ale nie możemy stosować operatorów logicznych, prefiksów, nie mamy wagi dla wyrazów oraz nie mozemy szukac wyrazow położonych blisko siebie. Contains za

to w podstawowej formie zwraca nam tylko wystapienia danego słowa ale odpowiednio modyfikując dane zapytnie również może nam odnalezc różne formy fleksyjne, synonimy, mozmey stosować prefiksy, słowa mają swoje wagi, dopuszcza się stosowanie operatorów logicznych. Podsumowując Contains jest zapytaniem dużo szybszym i bardziej precyzyjnym, zwracającym mniejszą liczbę wyników ale jednocześnie jest to zapytanie jest bardziej skomplikowane od freetext. W przykładach stworzyłem tabele w której umieściłem synonimy słowa dwa, zapytanie freetext zwróci mi całą tabele przy wyszukania jakiegokolwiek słowa zawartego w tej tabeli a zwykłe contains zwróci tylko to jedno szukane słowo.

Krótko opowiem jeszcze o zwracaniu wagi dla szukanych fraz, słuzy do tego funkcja Freetextable umieszczamy ją w klauzuli from potem w nawiasie określany tabale gdzie ma szukać, kolumny które mają zostać przeszukane i słowo/fraz które chcemy znalezc, wyniki są zwracane jedynie w postaci id danego wiersza oraz jego rankingu, by wyglądało to normalnie musimy troszkę zmodyfikować nasze zapytanie, przykład zawarty w pliku.

Zbudujmy takie oto zapytanie:
Select * from sys.dm_fts_parser('"Wlazł kotek na płotek i mruga, ładna to piosenka nie długa"', 1045, 0,0)
gdzie 1045 oznacza język polski o liczba 0 oznacza że korzystamy z systemowego słowanika wyrazów szuków.
Po skompilowaniu widzimy, że częśt słów typu i, na w tabeli special_term określona jest jako noise word, są to tak zwane słowa szumy, słowa które są pomijane , nie są brane pod uwagę przy liczeniu bliskości słów oraz ich wadze. Słowa te są zdefiniowane przez SQL Server, możemy je sprawdzic ( w przykadzie pokazane dla języka polskiego i angielskiego), możemy też sami zdefiniować taka listę, po co to robić? Przykładowo przeglądając jakas dokumentację programu w javie często można tam znalezc słowo array, które nie będzie dla nas istotne więc dodajemy go do naszej listy. Przykład zawarty w pliku z przykładami.

Omówimy teraz słownik tezaurus, to zwykły plik xml który  u mnie znajduje się w takiej lokazliacji: C:\Program Files\Microsoft SQL Server\MSSQL13.SQLEXPRESS\MSSQL\FTData
mamy tam pliki dostępne dla wielu języków, nas interesuje plik tsplk → słownik dla języka polskiego. Tezaurus jest słownikiem synonimów które sami możemy określać, freetext zawsze z tego korzysta a contains przy odpowiednio zmodyfikowanym zapytaniu.
Domyślnie każdy plik jest zakomentowany, spójrzmy na linijki które zmieniłem:

```
<expansion>
        <sub>pole</sub>
        <sub>dwor</sub>
        <sub>zewnatrz</sub>
 </expansion>
```

Chodzi nam o to iż jeśli ktoś będzie szukał słowa pole to wyszuka również dwor oraz zewnątrz, zapytania można dowolnie permutować. Wpisujemy tam nasze własne synonim, ludzie z Krakowa chodzą na pole, z Warszawy na dwór a jeszcze inni mogą chodzić na zewnatrz chodz to wszystko znaczy tak naprawdę to samo.
Kolejna klauzula:

```
<replacement>
        <pat>Konwersatorium</pat>
```

<sub>Wyklad</sub>
</replacement>
Przy szukaniu słowa Konwersatorium zostanie nam odnalezione słowo Wykład, ale Konwersatorium nie zostanie już znalezione, jak widać zastępujemy wyszukanie jednego słowa drugim. Po dokonaniu tych zmian, plik należy odkomentować, przeladować słowinik w sql serverze i działać. Kilka przykładów zamieściłem w pliku z przykładmi.
O plikach binarncyh wspomniałem już na wstępie więc nie ma się co powtarzać, możemy na nich robić dokladnie wszystko to samo co na danych tekstowych w tabelach.
Dziękuję za wysłuchanie i mam nadzieje że wiedza którą przedstawilem przyda się Państwu w pracy z SQL Serverem.

Sławomir Guzik