308. Commmon Table Expression oraz tabele tymczasowe - ćwiczenia

CTE (Common Table Expression) oraz tabele tymczasowe to narzędzia, które pozwalają na tymczasowe przechowywanie danych i wykonywanie zapytań na tych danych. CTE to wyrażenie, które definiuje zapytanie, które może być użyte wewnątrz innych zapytań.

Tabele tymczasowe natomiast są tworzone w bazie danych na czas wykonywania zapytania, a po zakończeniu działania zapytania są usuwane automatycznie. Dzięki tym narzędziom jest łatwiej i bardziej czytelnie pisać zapytania SQL oraz unikać powielania kodu.

CTE

Załóżmy, że mamy bazę danych z tabelą "Pracownicy", która zawiera informacje o pracownikach w firmie. Chcemy wyświetlić listę pracowników, którzy zarabiają więcej niż średnia pensja w firmie. Możemy to zrobić za pomocą CTE w następujący sposób:

```
WITH PracownicyFirma AS (
SELECT * FROM Pracownicy
)
SELECT * FROM PracownicyFirma
WHERE Pensja > (SELECT AVG(Pensja) FROM PracownicyFirma)
```

W tym przykładzie tworzymy CTE o nazwie "PracownicyFirma", która zawiera wszystkie rekordy z tabeli "Pracownicy". Następnie wykonujemy zapytanie SQL, które filtruje tylko te rekordy, których pensja jest większa niż średnia pensja wśród wszystkich pracowników. Dzięki CTE kod jest bardziej czytelny i łatwiejszy do zrozumienia. Warto zauważyć, że w tym przypadku CTE nie jest konieczne i można to zrobić bezpośrednio na tabeli "Pracownicy". Jednak w przypadku bardziej skomplikowanych zapytań CTE może znacznie ułatwić pracę i zwiększyć czytelność kodu.

Tabele tymczasowe

Tabele tymczasowe można podzielić na dwie kategorie: lokalne i globalne. Lokalne tabele tymczasowe są dostępne tylko dla sesji, w której zostały utworzone, natomiast globalne tabele tymczasowe są dostępne dla wszystkich sesji.

Poniżej przedstawiam przykład użycia tabeli tymczasowej w języku SQL Server. Załóżmy, że mamy bazę danych z tabelą "Zamówienia", która zawiera informacje o zamówieniach w sklepie internetowym. Chcemy wyświetlić sumę sprzedaży z ostatnich trzech miesięcy. Możemy to zrobić za pomocą tabeli tymczasowej w następujący sposób:

```
CREATE TABLE #Tymczasowa (
   Miesiąc DATETIME,
```

```
SumaSprzedaży DECIMAL(18,2)
)

INSERT INTO #Tymczasowa

SELECT

DATEADD(month, DATEDIFF(month, 0, DataZamówienia), 0) AS Miesiąc,

SUM(Cena) AS SumaSprzedaży

FROM Zamówienia

WHERE DataZamówienia >= DATEADD(month, -3, GETDATE())

GROUP BY DATEADD(month, DATEDIFF(month, 0, DataZamówienia), 0)

SELECT * FROM #Tymczasowa
```

W tym przykładzie tworzymy lokalną tabelę tymczasową o nazwie "#Tymczasowa", która zawiera dwie kolumny: "Miesiąc" i "SumaSprzedaży". Następnie wstawiamy dane do tabeli tymczasowej za pomocą zapytania SQL, które grupuje zamówienia według miesiąca i sumuje wartość zamówień z ostatnich trzech miesięcy. W końcu wyświetlamy dane z tabeli tymczasowej za pomocą zapytania SELECT.

Warto zauważyć, że lokalna tabela tymczasowa "#Tymczasowa" zostanie usunięta automatycznie po zakończeniu działania zapytania, więc nie będzie wpływać na strukturę bazy danych.

Kiedy stosować podzapytania, CTE oraz tabele tymczasowe?

• Subquery:

- Gdy potrzebujemy tylko jednej wartości zwracanej z zapytania, np. wartości maksymalnej, minimalnej, sumy, liczby rekordów.
- o Gdy zapytanie jest stosunkowo proste i nie wymaga definiowania tymczasowej tabeli.
- Gdy zapytanie jest wykonywane tylko raz i nie jest potrzebne wielokrotne użycie.

• CTE (Common Table Expression):

- Gdy potrzebujemy wielokrotnie użyć tymczasowej tabeli w zapytaniach.
- Gdy zapytanie jest złożone i trudne do zrozumienia, a dzięki podziałowi na mniejsze kroki kod staje się bardziej czytelny.
- o Gdy chcemy rozbić zapytanie na mniejsze, łatwiejsze do zrozumienia części.

Tabele tymczasowe:

- Gdy potrzebujemy przechować tymczasowo duże ilości danych, na których będziemy wykonywać złożone operacje.
- Gdy zapytanie wymaga wielu operacji, które są niepraktyczne lub niemożliwe do wykonania w jednym zapytaniu.
- o Gdy potrzebujemy zapisać wyniki zapytania na dłużej, np. do dalszej analizy.

Dodatkowe

Zapamiętaj dodatkowe dwie komendy: CAST oraz CASE. Zadania te będą wykorzystywane na teście praktycznym.

CASE jest to wyrażenie w języku SQL, które umożliwia warunkowe przypisywanie wartości do kolumny wynikowej. Używa się go do tworzenia niestandardowych kolumn, które zależą od warunków logicznych lub wartości innych kolumn. Składnia wyrażenia CASE obejmuje określenie warunków za pomocą klauzuli WHEN, a następnie przypisanie wartości do kolumny wynikowej za pomocą klauzuli THEN. Można także wykorzystać klauzulę ELSE do zdefiniowania wartości domyślnej dla przypadków, w których nie spełniony zostanie żaden warunek. Oto przykładowe użycie wyrażenia CASE:

```
SELECT

product_name,
price,

CASE

WHEN price > 100 THEN 'Drogi'
WHEN price > 50 THEN 'Średni'
ELSE 'Tani'
END AS 'Cena'

FROM
products
```

W tym przykładzie wykorzystujemy wyrażenie CASE, aby przyporządkować każdemu produktowi wartość "Drogi", "Średni" lub "Tani", w zależności od jego ceny.

CAST jest to funkcja w języku SQL, która umożliwia konwersję jednego typu danych na inny. Używa się jej, gdy chcemy dokonać operacji matematycznych lub porównań między różnymi typami danych, lub gdy chcemy zwrócić wartość w określonym formacie. Składnia funkcji CAST obejmuje określenie wartości, która ma być przekonwertowana, a następnie określenie docelowego typu danych. Oto przykładowe użycie funkcji CAST:

```
SELECT

customer_name,

CAST(order_date AS DATE) AS 'Data zamówienia'

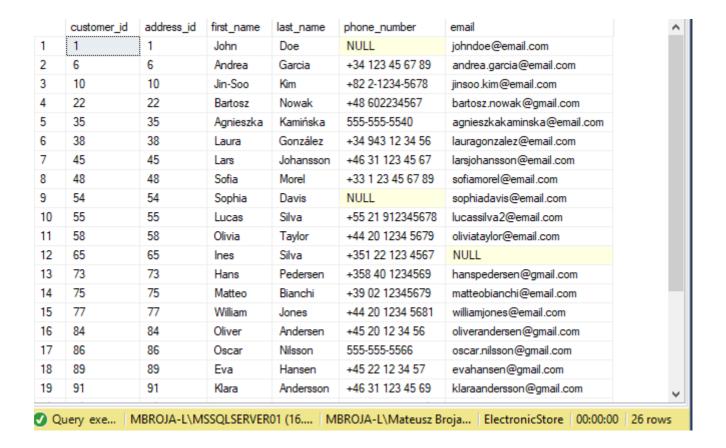
FROM

orders
```

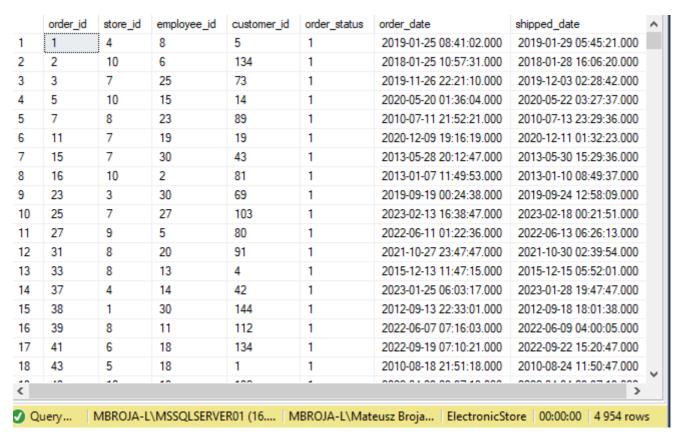
W tym przykładzie wykorzystujemy funkcję CAST, aby przekonwertować datę zamówienia (która jest przechowywana jako wartość daty i godziny) na typ danych DATE, aby wyświetlić tylko datę bez czasu.

Zadania

1. Znajdź klientów, którzy złożyli zamówienia w ciągu ostatnich 30 dni. Użyj CTE.



2. Znajdź zamówienia, które mają status 1 (order_status = 1). Użyj CTE.

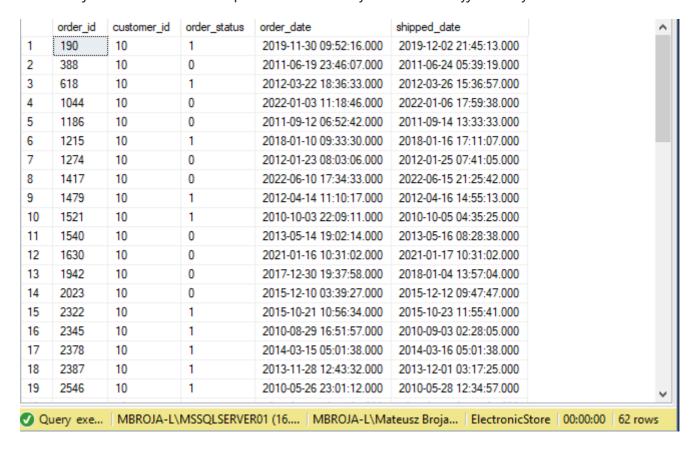


3. Znajdź produkty, których cena jest mniejsza niż 50 zł. Użyj tabele tymczasowe.

	product_id	product_name	brand	price
1	29	G203 LIGHTSYNC Wired Gaming Mouse	Logitech	39.99
2	144	K120	Logitech	19.99
3	145	DeathAdder	Razer	39.99

Query exec... | MBROJA-L\MSSQLSERVER01 (16.... | MBROJA-L\Mateusz Broja... | ElectronicStore | 00:00:00 | 3 rows

4. Znajdź zamówienia złożone przez klienta o identyfikatorze 10. Użyj tabele tymczasowe.



5. Znajdź nazwy kategorii, które mają więcej niż 5 produktów. Zwróć również liczbę produktów dla każdej kategorii.

	category_name	ProductCount
1	Laptop	15
2	Keyboard	16
3	Mouse	17
4	Monitor	17
5	Headphones	17
6	Printer	17
7	Tablet	17
8	Smartphone	19
9	Camera	12
10	Speakers	9
11	PC	18

Query exe...

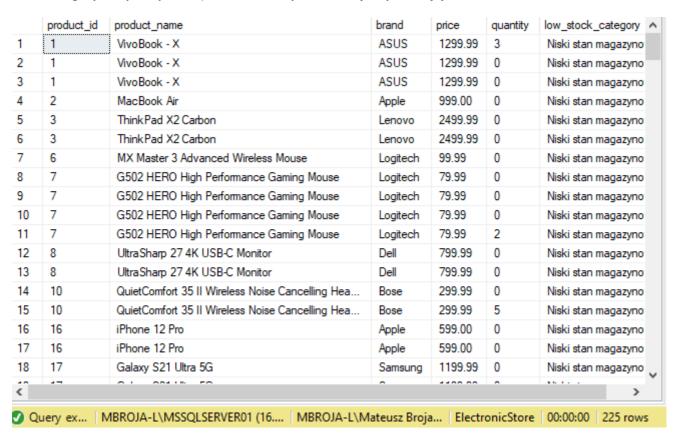
MBROJA-L\MSSQLSERVER01 (16.... | MBROJA-L\Mateusz Broja... | ElectronicStore | 00:00:00 | 11 rows

6. Znajdź klientów, którzy nie złożyli zamówienia w ciągu ostatnich 6 miesięcy, i przypisz im kategorię "Nieaktywni". Wynik zapisz do tabeli tymczasowej i wyświetl jej zawartość.

	customer_id	first_name	last_name	email	phone_number	inactive_category
1	4	Emilie	Dubois	emilie.dubois@email.com	+33 6 12 34 56 78	Nieaktywni
2	5	Lukas	Müller	lukas.muller@email.com	+49 1234 567890	Nieaktywni
3	11	Femanda	Souza	femanda.souza@email.com	+55 11 1234-5678	Nieaktywni
4	20	Aarav	Patel	aarav.patel@gmail.com	+91 9988776655	Nieaktywni
5	26	Maria	Garcia	mariagarcia@email.com	+34 615123456	Nieaktywni
6	34	Piotr	Wiśniewski	piotrwisniewski@email.com	555-555-5539	Nieaktywni
7	81	Noah	Sorensen	noahsorensen@gmail.com	+45 42 12 34 56	Nieaktywni
8	98	Laura	Müller	lauramuller@gmail.com	+49 30 12345681	Nieaktywni
9	129	Tomasz	Sikora	tomaszsikora@gmail.com	+48 515 678 901	Nieaktywni
10	146	Patrycja	Czerwińska	patrycjaczerwinska@gmail.com	+48 546 789 012	Nieaktywni

Query exe... | MBROJA-L\MSSQLSERVER01 (16.... | MBROJA-L\Mateusz Broja... | ElectronicStore | 00:00:00 | 10 rows

7. Znajdź produkty, których ilość w magazynie jest mniejsza niż 10, i przypisz im kategorię "Niski stan magazynowy". Wynik zapisz do tabeli tymczasowej i wyświetl jej zawartość.



8. Znajdź adresy klientów, którzy zamówili produkty za więcej niż 1000 \$ w ciągu ostatnich 3 miesięcy. Zwróć również liczbę zamówień i łączną wartość zamówień dla każdego adresu.

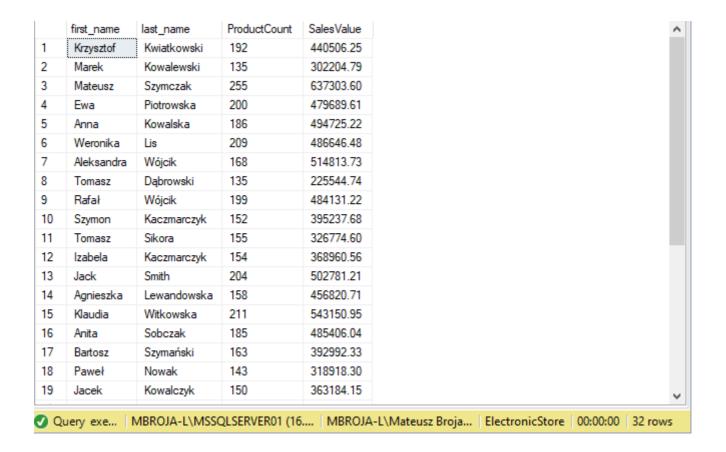


9. Znajdź sklepy, które nie sprzedały produktów w ciągu ostatnich 15 dni. Zwróć również datę ostatniego zamówienia dla każdego sklepu.

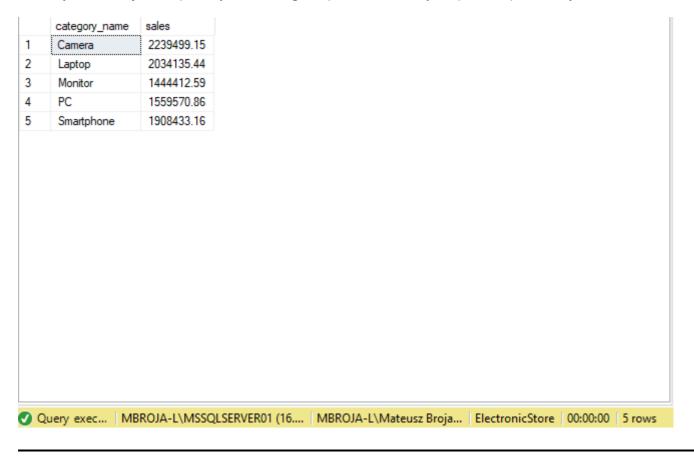
	store_name	country	city	street	postal_code	LastOrderDate
1	Electric World	USA	New York	789 Broadway	10011	2023-02-13 19:06:04.000
2	RTV World	Canada	Toronto	654 King St	M5V 2B6	2023-02-08 09:05:39.000
3	MegaMarket	Poland	Warsaw	ul. Marszalkowska 116/122	00-001	2023-02-12 06:47:26.000
4	MaxiMart	Poland	Krakow	ul. Pawia 5	31-154	2023-02-14 23:45:16.000
5	Sprzęt RTV	Poland	Gdansk	ul. Dluga 5	80-828	2023-02-14 07:33:08.000
6	MiniMart	Poland	Wroclaw	ul. Swidnicka 15	50-067	2023-02-15 00:31:55.000
7	Komputery Świat	Poland	Poznan	ul. Garbary 59	61-758	2023-02-13 16:38:47.000
8	GigaStore	France	Paris	1 Rue de la Paix	75001	2023-02-14 22:21:25.000
9	RTV i AGD	Spain	Madrid	Calle de Fuencarral 200	28010	2023-02-13 14:20:04.000
10	MediaExpert	Germany	Berlin	Friedrichstraße 85	10117	2023-02-13 03:04:17.000

✓ Query exe... | MBROJA-L\MSSQLSERVER01 (16.... | MBROJA-L\Mateusz Broja... | ElectronicStore | 00:00:00 | 10 rows

10. Znajdź pracowników, którzy pracują w sklepie, w którym sprzedano najwięcej produktów w ciągu ostatnich 3 miesięcy. Zwróć również liczbę sprzedanych produktów i łączną wartość sprzedaży dla każdego pracownika. Nie uwzględniaj discount dla uproszczenia.



11. Znajdź sumę sprzedaży dla każdej kategorii produktów w ostatnim roku i zapisz wynik do tabeli tymczasowej. Następnie wyświetl kategorie produktów, których sprzedaż przekroczyła 1000000 \$.



12. Znajdź zamówienia, które zostały złożone przez klientów mieszkających w Polsce, i przypisz im kategorię "Krajowy". Wynik zapisz do tabeli tymczasowej i wyświetl jej zawartość.

	order_id	customer_id	order_status	order_date	shipped_date	domestic_category	^
1	2	134	1	2018-01-25 10:57:31.000	2018-01-28 16:06:20.000	Krajowy	
2	4	146	0	2018-05-11 01:28:01.000	2018-05-12 12:02:15.000	Krajowy	
3	10	145	0	2012-09-13 05:53:47.000	2012-09-16 14:05:54.000	Krajowy	
4	14	140	0	2016-01-15 05:36:35.000	2016-01-19 21:38:03.000	Krajowy	
5	19	124	0	2011-03-31 13:19:18.000	2011-04-02 16:36:54.000	Krajowy	
6	21	126	0	2018-06-08 12:02:14.000	2018-06-10 16:50:10.000	Krajowy	
7	22	31	0	2014-09-06 23:08:18.000	2014-09-11 01:16:52.000	Krajowy	
8	26	23	0	2014-03-27 07:24:53.000	2014-03-29 18:07:08.000	Krajowy	
9	29	146	0	2010-08-06 22:32:48.000	2010-08-10 11:33:19.000	Krajowy	
10	35	137	0	2015-12-24 15:09:24.000	2015-12-28 00:43:16.000	Krajowy	
11	36	35	0	2022-03-10 15:13:40.000	2022-03-11 20:45:16.000	Krajowy	
12	38	144	1	2012-09-13 22:33:01.000	2012-09-18 18:01:38.000	Krajowy	
13	41	134	1	2022-09-19 07:10:21.000	2022-09-22 15:20:47.000	Krajowy	
14	46	125	0	2013-09-06 20:45:01.000	2013-09-08 19:05:35.000	Krajowy	
15	47	37	0	2011-01-31 14:16:09.000	2011-02-03 18:54:23.000	Krajowy	
16	49	132	1	2022-04-03 03:37:19.000	2022-04-04 03:37:19.000	Krajowy	
17	52	100	1	2015-07-13 22:51:08.000	2015-07-17 16:07:41.000	Krajowy	
18	59	29	1	2011-02-22 00:09:25.000	2011-02-23 00:09:25.000	Krajowy	
19	63	124	1	2019-12-27 13:54:56.000	2019-12-31 03:31:01.000	Krajowy	_
y Q	uery I	MBROJA-L\M	SSQLSERVER01	(16 MBROJA-L\Mateu	usz Broja ElectronicSto	re 00:00:00 3 492	