

Funkcje okna

Czas: 2 × 45 min

praca indywidualna lub w parach

Cel: Nauczenie się liczenia rankingów, sum narastających, porównań z poprzednim okresem i udziałów w grupie bez utraty pojedynczych wierszy — za pomocą funkcji okna `OVER()`.

Po zajęciach potrafisz:

- odróżnia `GROUP BY` (skleja wiersze) od `OVER()` (zachowuje wiersze)
- numeruje i szereguje wiersze funkcjami `ROW_NUMBER`, `RANK` i `DENSE_RANK`, także w grupach (`PARTITION BY`)
- liczy sumy narastające, średnie kroczące i udziały procentowe w grupie
- porównuje wiersz z poprzednim i następnym (`LAG`, `LEAD`)
- wybiera top-N w każdej grupie przez CTE z funkcją okna

Sprzęt i materiały

- komputer w pracowni z XAMPP (MariaDB 10.4) lub MySQL 8.4
- baza sklep zaimportowana z pliku `sklep.sql` (folder `Pliki_do_zadan`)

Oprogramowanie

- **XAMPP (Apache + MariaDB + phpMyAdmin)** — panel sterowania XAMPP, konsola: przycisk Shell albo
`C:\xampp\mysql\bin\mysql.exe -u root`
apachefriends.org/download.html
- **phpMyAdmin** — zakładka SQL do uruchamiania zapytań; wchodzi w skład XAMPP
localhost/phpmyadmin
- **VS Code lub Notepad++** — do pisania pliku `.sql` z zapytaniami
code.visualstudio.com

Przygotowanie przed zajęciami

- Powtórz z zadania 5 zapis CTE (`WITH ... AS`).

Bezpieczeństwo

- Zadanie wymaga tylko odczytu danych i utworzenia jednego widoku. Nie zmieniaj danych w tabelach bazy sklep.

Przebieg zadania

1 Okno bez sklejania wierszy

Widok z wartością każdego zamówienia nieanulowanego będzie podstawą dalszych zapytań.

```
CREATE VIEW v_sprzedaz AS
SELECT z.id_zamowienia, z.id_klienta, z.data_zamowienia,
       SUM(pz.ilosc * pz.cena_jedn) AS wartosc
FROM zamowienia z
JOIN pozycje_zamowienia pz ON pz.id_zamowienia = z.id_zamowienia
WHERE z.status <> 'anulowane'
GROUP BY z.id_zamowienia, z.id_klienta, z.data_zamowienia;
```

1. Zaimportuj `sklep.sql`, utwórz plik `r2_06_nazwisko.sql` i widok `v_sprzedaz` (kod powyżej). Sprawdź: 11 zamówień o łącznej wartości 31639.00.
2. Z1. Dla każdego produktu z kategorią wypisz cenę, średnią cenę jego kategorii (`AVG ... OVER (PARTITION BY ...)`) i różnicę od tej średniej. Posortuj po kategorii i cenie malejąco.
3. Z2. Porównaj liczbę wierszy: `AVG(cena)` z `GROUP BY id_kategorii` oraz `AVG(cena) OVER (PARTITION BY id_kategorii)` bez `GROUP BY`. Wyjaśnij różnicę.
4. Z3. Spróbuj wybrać 3 najdroższe produkty warunkiem `WHERE ROW_NUMBER() OVER (ORDER BY cena DESC) <= 3`. Zapisz błąd.

2 Rankingi

1. Z4. Dla każdego klienta, który złożył zamówienie (wszystkie statusy), policz zamówienia i obok wypisz `ROW_NUMBER`, `RANK` i `DENSE_RANK` według tej liczby malejąco. W `ROW_NUMBER` dodaj drugie kryterium sortowania (nazwisko), żeby numeracja była powtarzalna. Opisz, czym różnią się trzy kolumny przy remisach.
2. Z5. Ponumeruj produkty w każdej kategorii od najdroższego (`ROW_NUMBER` z `PARTITION BY`).
3. Z6. Korzystając z Z5 i CTE, wypisz po 2 najdroższe produkty z każdej kategorii (top-N w grupie).
4. Z7. Podziel produkty na 3 grupy cenowe funkcją `NTILE(3)` (od najdroższych). Ile produktów trafiło do każdej grupy?

3 Sumy narastające i udziały

1. Z8. Z widoku `v_sprzedaz` wypisz zamówienia w kolejności dat z wartością i sumą narastającą sprzedaży (`SUM ... OVER (ORDER BY data_zamowienia)`).
2. Z9. Dla klientów 1, 2 i 4 wypisz ich zamówienia z sumą narastającą osobno dla każdego klienta oraz liczbą wszystkich zamówień klienta w każdym wierszu (`COUNT(*) OVER (PARTITION BY ...)`).
3. Z10. Dla każdego zamówienia z `v_sprzedaz` podaj jego udział procentowy w sprzedaży jego miesiąca (1 miejsce po przecinku).
4. Z11. Dla pracowników wypisz pensję, najwyższą pensję w zespole tego samego przełożonego i udział pensji w całym funduszu płac (`SUM(pensja) OVER ()`).

4 Porównania z sąsiednimi wierszami

1. Z12. W CTE policz sprzedaż w każdym miesiącu (z `v_sprzedaz`). Wypisz miesiąc, wartość, wartość z poprzedniego miesiąca (`LAG`), zmianę w złotych i w procentach (zabezpiecz dzielenie funkcją `NULLIF`).
2. Z13. Dla zamówień klientów 1, 2 i 4 (z tabeli `zamowienia`) wypisz datę następnego zamówienia tego samego klienta (`LEAD`) i liczbę dni do niego (`DATEDIFF`).
3. Z14. Dla zamówień z `v_sprzedaz` oblicz średnią kroczącą z 3 ostatnich zamówień: `AVG(wartosc) OVER (ORDER BY data_zamowienia ROWS BETWEEN 2 PRECEDING AND CURRENT ROW)`.

W sprawozdaniu umieść

- ☐ plik `.sql` z zapytaniami Z1–Z14
- ☐ zrzuty ekranu wyników Z4, Z6, Z8 i Z12
- ☐ wyjaśnienie różnic ROW_NUMBER / RANK / DENSE_RANK na przykładzie Z4
- ☐ odpowiedzi na pytania kontrolne

Pytania kontrolne

1. Dlaczego funkcji okna nie można użyć w WHERE i jak to obejść?
2. Co zmienia PARTITION BY w porównaniu z samym OVER (ORDER BY ...)?
3. Klienci mają liczby zamówień 3, 2, 2, 1. Jakie numery dostaną od ROW_NUMBER, RANK i DENSE_RANK?
4. Dlaczego pierwsza wartość LAG w Z12 to NULL i jak to wpływa na zmianę procentową?

Kryteria oceny

Kryterium	Punkty
OVER z PARTITION BY i porównanie z GROUP BY (Z1–Z3)	3
Rankingi i top-N w grupie (Z4–Z7)	5
Sumy narastające i udziały (Z8–Z11)	5
LAG, LEAD i średnia krocząca (Z12–Z14)	4
Odpowiedzi na pytania kontrolne	3
Razem	20

Sprawozdanie przygotuj na wzorze dołączonym do zadania (plik .docx o tym samym numerze).