

FUNCTII SQL PENTRU RAPORTARE

1. Obțineți pentru fiecare factură a clientului 146 valoarea cumulată după fiecare produs.

Observații:

- partiția va fi creată pe baza facturilor; fiecare submulțime din partiție va avea dimensiunea unei facturi, numărul de linii care o alcătuiesc depinzând de numărul de coduri de produse prezente pe factura respectivă;
- ordonarea în cadrul submulțimilor partiției se realizează după codul produsului;
- fereastra pentru care se efectuează calculul pentru fiecare linie are o margine fixă (începutul submulțimii) și una mobilă (care este chiar linia curentă).

```
SUM(pret_unitar_vanzare*cantitate)
      OVER (PARTITION BY factura
            ORDER BY      produs_id
            ROWS UNBOUNDED PRECEDING)
```

Explicații:

- Fereastră definită se constituie în cadrul submulțimilor formate de valorile comune ale atributului *factura*.
- În cadrul submulțimilor, liniile sunt ordonate crescător după valorile atributului *produs_id*, iar intervalul liniilor asupra căruia se aplică funcția de agregare *SUM(pret_unitar_vanzare*cantitate)* este alcătuit din prima linie a submulțimii și linia curentă (*ROWS UNBOUNDED PRECEDING*).

2. Obțineți ziua din 2007 în care s-a încasat cea mai mare sumă din vânzarea fiecărui produs din categoria „Markere permanente“.

```
MAX(SUM(pret_unitar_vanzare*cantitate))
      OVER (PARTITION BY v.produs_id)
```

3. Afișați pentru fiecare zi a lunii martie din 2007, următoarele valori:

- vânzările din ziua respectivă;
- vânzările cumulate (inclusiv ale zilei curente);
 - o fereastră cu limită fizică care:
 - începe de la începutul submulțimii (*ROWS UNBOUNDED PRECEDING*)
 - se termină cu linia curentă
- vânzările din ziua curentă cumulate cu cele din ziua precedentă.
 - o fereastră cu limită logică care:
 - începe cu ziua calendaristică anterioară
 - se termină cu ziua curentă.

Observații:

- Dacă pentru intervalul definit fizic se ia în calcul linia anterioară, indiferent cărei zi îi corespunde, în cazul intervalului calendaristic, valoarea zilei curente se cumulează numai dacă linia precedentă se referă la ziua calendaristică precedentă.
- De exemplu, pentru data de *19 martie 2007* valorile coloanelor *valoare_cummulata* și *azi_si_ieri* coincid, deoarece pe *18 martie 2007* martie nu există nicio factură emisă.

```
SUM(valoare)
  OVER (ORDER BY timp_id
        ROWS UNBOUNDED PRECEDING)
  AS valoare_cumulata
SUM(valoare)
  OVER (ORDER BY timp_id
        RANGE INTERVAL '1' DAY PRECEDING)
  AS azi_si_ieri
```

4. Să se obțină valorile tranzacționate cu fiecare client:

- în fiecare trimestru al anului 2007;
- valoarea cumulată la nivel de trimestru.

```
SUM(SUM(pret_unitar_vanzare*cantitate))
  OVER (PARTITION BY client_id
        ORDER BY      client_id, trimestru
        ROWS UNBOUNDED PRECEDING) AS client_vanzari
```

5. Obțineți evoluția mediei vânzărilor pentru clientul având codul 146:

- a. pe ultimele trei luni (luna curentă și cele două luni anterioare)

```
AVG(SUM(pret_unitar_vanzare*cantitate))  
    OVER (ORDER BY client_id, luna  
          ROWS 2 PRECEDING)  
AS media_3_luni
```

- b. centrată pe trei luni (luna precedentă, luna curentă și luna următoare)

```
AVG(SUM(pret_unitar_vanzare*cantitate))  
    OVER (ORDER BY client_id, luna  
          ROWS BETWEEN 1 PRECEDING AND 1 FOLLOWING)  
AS media_centrata_3_luni
```

6. Obțineți evoluția mediei vânzărilor centrată pe trei zile (ziua anterioară, ziua curentă și ziua următoare), pentru clientul având codul 233.

```
AVG(SUM(pret_unitar_vanzare*cantitate))  
    OVER (PARTITION BY client_id ORDER BY timp_id  
          RANGE BETWEEN INTERVAL '1' DAY PRECEDING  
          AND INTERVAL '1' DAY FOLLOWING)  
AS medie_centrata_3_zile
```

7. Afișați pentru fiecare produs din categoria „Plicuri securizate“, numele celui mai scump produs din această categorie.

Observație:

- Funcțiile *FIRST_VALUE* și *LAST_VALUE* sunt nedeterminate.
- Dacă ar exista două produse cu același preț maxim atunci va fi ales ca produs cu preț maxim, acel produs care este returnat primul de subcerere.

```
FIRST_VALUE(denumire)  
    OVER (ORDER BY pret_unitar_curent DESC  
          ROWS BETWEEN UNBOUNDED PRECEDING  
          AND UNBOUNDED FOLLOWING)  
AS den_prod_cu_pret_maxim
```

8. Obțineți pentru fiecare produs procentul zilnic al vânzărilor, calculat prin raportare la prima zi de vânzare a acestuia.

```
ROUND(valoare/FIRST_VALUE(valoare)
      OVER (PARTITION BY produs_id
            ORDER BY timp_id)
      * 100,2) AS procent_din_zil
```

9. Pentru fiecare subgrupă de produse, să se obțină orașul care a înregistrat vânzări maxime în anul 2007.

```
MAX(SUM(pret_unitar_vanzare*cantitate ))
    OVER (PARTITION BY categorie_3)
AS max_val
```

10. Obțineți cele mai bine vândute 5 produse din fiecare grupă, care contribuie cu mai mult de 20% la vânzările raionului din care face parte.

```
SUM(SUM(pret_unitar_vanzare*cantitate ))
    OVER (PARTITION BY categorie_1) AS RAION,
SUM(SUM(pret_unitar_vanzare*cantitate))
    OVER (PARTITION BY categorie_2) AS GRUPA,
DENSE_RANK()
    OVER (PARTITION BY categorie_2
          ORDER BY SUM(pret_unitar_vanzare*cantitate))
AS RANG
```

11. Pentru fiecare tip de plată obțineți valoarea procentuală a vânzărilor realizate raportată la vânzările totale.

Observație:

- Funcția *RATIO_TO_REPORT* calculează proporția unei valori raportată la suma tuturor valorilor.

```
SUM(SUM(pret_unitar_vanzare*cantitate))
    OVER () AS TOTAL_VANZ
RATIO_TO_REPORT(SUM(pret_unitar_vanzare*cantitate))
    OVER () AS RATIO_REP
```

- 12.** Comparați vânzările zilei curente cu cele ale zilei anterioare, respectiv ale zilei următoare (deplasarea este 1).

Observații :

- Funcția *LAG* asigură acces la o altă linie prin deplasare cu una sau mai multe linii anterior poziției curente.
- Funcția *LEAD* asigură acces la o altă linie prin deplasare cu una sau mai multe linii ulterior poziției curente.

```
LAG (SUM (pret_unitar_vanzare*cantitate)
      OVER (ORDER BY timp_id) AS LAG1,
LEAD (SUM (pret_unitar_vanzare*cantitate))
      OVER (ORDER BY timp_id) AS LEAD1
```

- 13.** Obțineți diferențele zilnice dintre vânzările fiecărui produs (diferența dintre valoarea vânzărilor de la o zi la alta pentru fiecare produs).

```
valoare_vanzari-LAG (valoare_vanzari, 1)
      OVER (PARTITION BY produs_id ORDER BY timp_id)
AS diferenta
```

- 14.** Obțineți evoluția zilnică a vânzărilor realizate.

```
valoare_vanzari-LAG (valoare_vanzari, 1)
      OVER ( ORDER BY timp_id)
AS diferenta
```