

Capitolul 11. OLAP în SQL

Cunoaștem cu toții zicala cu fratele, necuratul și puntea¹. O folosim destui, mai ales în momentele de compromis, firește, fiecare imaginându-și că celălalt e dracul. În materie de funcții analitice ce au fost introduse ca amendament la standardul SQL:1999 puntea este ANSI (American National Standard Institute), iar cei doi parteneri principali (care o fi necuratul ?) sunt IBM și Oracle.

Derulată de pe la începutul anului 1999, colaborarea dintre echipele conduse de Hamid Paresch la IBM (Almaden Research Laboratory) și Andy Witkowski la Oracle s-a concretizat rapid într-un set comun de specificații pe care l-au înaintat ANSI (de la ANSI, Jim Melton a jucat un rol important în acest proiect). În scurt timp, pe baza specificațiilor, ANSI a publicat Amendamentul 1 la standardul SQL:1999.

IBM a implementat o parte din opțiunile OLAP ale SQL încă din versiunea 6.1 a DB2-ului, în timp ce Oracle începând cu versiunea 8i2, ambele lansate pe piață în anul 1999. La scurt timp, și MS SQL Server a implementat câteva funcții analitice. De atunci, și versiunile standardului și cele ale serverelor de baze de date au evoluat continuu.

Definite uneori și funcții analitice, funcțiile/opțiunile OLAP (de la *On Line Analytical Processing*) extind nucleul SQL către probleme legate destul de strâns de ceea ce îndeobște este cunoscut sub titulaturile depozite de date (Data Warehouses), Data Mining etc. Din păcate, în acest capitol PostgreSQL-ul stă pe tușă, neavând implementată, până la versiunea curentă (8.3), nicio funcție OLAP.

11.1. Subtotaluri

Problema subtotalurilor în SQL a fost abordată cu mari eforturi încă din paragraful 7.2. Am convenit că, adunând mai multe fraze SELECT conectate prin UNION (plus grupările de rigoare), se poate obține o formă rezonabilă de raport. Peste câteva rânduri o să vedem cât de mult se simplifică lucrurile folosind operatorii ROLLUP, CUBE etc.

¹ Nu e prea elegant să te autocitezi sau să repeți glumele, mai ales atunci când sunt slabe. Însă, pur și simplu, nu m-am putut abține să nu inserez introducea din materialul publicat în PCReport-ul nr. 98 (noiembrie)/2000. Vezi [Fotache00-4] și [Fotache00-5]

11.1.1. Operatorul ROLLUP

Să se obțină situația vânzărilor pe clienți și zile în luna septembrie 2007, afișându-se câte un subtotal la nivel de client și un total general.

Dorim să obținem, mai întâi, raportul din figura 7.15 pentru care în paragraful 7.2 reuneam trei SELECT-uri, unul pentru rândul „curent”, altul pentru subtotalul la nivel de clienți și un altul pentru totalul general. Pentru asemenea gen de probleme avem operatorul ROLLUP aplicat după cum urmează:

```
SELECT DenCl AS DenumireClient, DataFact AS Data,
       ROUND(SUM(Cantitate * PretUnit * (1+ProcTVA)),0) AS Vinzari
FROM clienti c
      INNER JOIN facturi f ON c.CodCl=f.CodCl
      INNER JOIN liniifact lf ON lf.NrFact=f.NrFact
      INNER JOIN produse p ON lf.CodPr=p.CodPr
WHERE EXTRACT (YEAR FROM DataFact) = 2007
      AND EXTRACT (MONTH FROM DataFact) = 9
GROUP BY ROLLUP (DenCl, DataFact)
```

Ceea ce se obține în Oracle cu această interogare nu seamănă leită cu figura 7.2, însă e un bun punct de plecare – vezi figura 11.1. Clienții nu sunt dispuși în ordinea alfabetică a denumirii lor, însă liniile unui client sunt consecutive, iar rândul de total apare la sfârșit.

DENUMIRECLIENT	DATA	VINZARI
Client 2 SA	02-09-2007	127530
Client 2 SA	(null)	127530
Client 1 SRL	01-09-2007	4442960
Client 1 SRL	02-09-2007	110908
Client 1 SRL	10-09-2007	287855
Client 1 SRL	17-09-2007	179565
Client 1 SRL	(null)	5021287
Client 3 SRL	07-09-2007	5819668
Client 3 SRL	(null)	5819668
Client 5 SRL	01-09-2007	153442
Client 5 SRL	(null)	153442
Client 7 SRL	10-09-2007	136850
Client 7 SRL	(null)	136850
(null)	(null)	11258777

Figura 11.1. Operatorul ROLLUP (în Oracle) fără finisaje

MS SQL Server are o sintaxă ușor diferită față de standardul SQL:

```
SELECT DenCl AS DenumireClient, DataFact AS Data,
       ROUND(SUM(Cantitate * PretUnit * (1+ProcTVA)),0) AS Vinzari
FROM clienti c
      INNER JOIN facturi f ON c.CodCl=f.CodCl
```

```

INNER JOIN liniifact lf ON lf.NrFact=f.NrFact
INNER JOIN produse p ON lf.CodPr=p.CodPr
WHERE DATEPART (YEAR,DataFact) = 2007
      AND DATEPART(MONTH,DataFact) = 9
GROUP BY DenCl, DataFact WITH ROLLUP

```

În materie de formă a rezultatului, SQL Serverul prezintă avantajul ordonării implicite a liniilor după denumirea clientului și data facturării – vezi figura 11.2.

DenumireClient	Data	Vinzari
Client 1 SRL	2007-09-01 00:00:00	4442960.0000
Client 1 SRL	2007-09-02 00:00:00	110908.0000
Client 1 SRL	2007-09-10 00:00:00	287855.0000
Client 1 SRL	2007-09-17 00:00:00	179565.0000
Client 1 SRL	NULL	5021287.0000
Client 2 SA	2007-09-02 00:00:00	127530.0000
Client 2 SA	NULL	127530.0000
Client 3 SRL	2007-09-07 00:00:00	5819668.0000
Client 3 SRL	NULL	5819668.0000
Client 5 SRL	2007-09-01 00:00:00	153442.0000
Client 5 SRL	NULL	153442.0000
Client 7 SRL	2007-09-10 00:00:00	136850.0000
Client 7 SRL	NULL	136850.0000
NULL	NULL	11258777.0000

Figura 11.2. ROLLUP în MS SQL Server

DB2-ul are o sintaxă similară Oracle, cu excepția funcției EXTRACT:

```

SELECT ... FROM ...
WHERE YEAR(DataFact) = 2007 AND MONTH(DataFact) = 9
GROUP BY ROLLUP (DenCl, DataFact)

```

Ordonarea liniilor este una specială – vezi figura 11.3, valorile NULL fiind plasate înaintea celor nenule.

DENUMIRECLIENT	DATA	VINZARI
		11.258.777,0000
Client 1 SRL		5.021.287,0000
Client 2 SA		127.530,0000
Client 3 SRL		5.819.668,0000
Client 5 SRL		153.442,0000
Client 7 SRL		136.850,0000
Client 1 SRL	01.09.2007	4.442.960,0000
Client 1 SRL	02.09.2007	110.908,0000
Client 1 SRL	10.09.2007	287.855,0000
Client 1 SRL	17.09.2007	179.565,0000
Client 2 SA	02.09.2007	127.530,0000
Client 3 SRL	07.09.2007	5.819.668,0000
Client 5 SRL	01.09.2007	153.442,0000
Client 7 SRL	10.09.2007	136.850,0000

Figura 11.3. ROLLUP în DB2

În continuare, încercăm să scăpăm de NULL-uri și să inserăm „mesaje” lămuritoare. Cum textele de pe coloana dedicată datei facturilor ar genera o eroare de incompatibilitate a tipurilor de date, concatenăm (implicit în Oracle și explicit în SQL Server și DB2) DataFact cu un spațiu:

```
SELECT DenCl AS DenumireClient,
       COALESCE (CAST (DataFact AS CHAR(10)) ||
                '','subtotal (toate zilele)') AS Data,
       ROUND(SUM(Cantitate * PretUnit * (1+ProcTVA)),0) AS Vinzari
FROM clienti c
      INNER JOIN facturi f ON c.CodCl=f.CodCl
      INNER JOIN liniifact lf ON lf.NrFact=f.NrFact
      INNER JOIN produse p ON lf.CodPr=p.CodPr
WHERE EXTRACT (YEAR FROM DataFact) = 2007
      AND EXTRACT (MONTH FROM DataFact) = 9
GROUP BY ROLLUP (DenCl, DataFact)
```

DENUMIRECLIENT	DATA	VINZARI
Client 2 SA	02-09-2007	127530
Client 2 SA		127530
Client 1 SRL	01-09-2007	4442960
Client 1 SRL	02-09-2007	110908
Client 1 SRL	10-09-2007	287855
Client 1 SRL	17-09-2007	179565
Client 1 SRL		5021287
Client 3 SRL	07-09-2007	5819668
Client 3 SRL		5819668
Client 5 SRL	01-09-2007	153442
Client 5 SRL		153442
Client 7 SRL	10-09-2007	136850
Client 7 SRL		136850
(null)		11258777

Figura 11.4. Operatorul ROLLUP (în Oracle) cu finisaje defecte

Ceea ce obținem în Oracle ne derutează (un pic) – vezi figura 11.4. Descoperim, astfel, că, în Oracle, NULL concatenat cu spațiu nu e chiar NULL, ci spațiu (ne reamintim că prin adunarea unui număr cu un NULL obțineam NULL). Ciudat este că dacă portăm soluția în SQL Server:

```
SELECT DenCl AS DenumireClient,
       COALESCE (CAST (DataFact AS CHAR) + '','subtotal (toate zilele)') AS Data,
       ROUND(SUM(Cantitate * PretUnit * (1+ProcTVA)),0,0) AS Vinzari
FROM clienti c
```

```

INNER JOIN facturi f ON c.CodCl=f.CodCl
INNER JOIN liniifact lf ON lf.NrFact=f.NrFact
INNER JOIN produse p ON lf.CodPr=p.CodPr
WHERE DATEPART (YEAR,DataFact) = 2007
      AND DATEPART(MONTH,DataFact) = 9
GROUP BY DenCl, DataFact WITH ROLLUP

```

rezultatul este altul – vezi figura 11.5.

DenumireClient	Data	Vinzari
Client 1 SRL	Sep 1 2007 12:00AM	4442960.0000
Client 1 SRL	Sep 2 2007 12:00AM	110908.0000
Client 1 SRL	Sep 10 2007 12:00AM	287855.0000
Client 1 SRL	Sep 17 2007 12:00AM	179565.0000
Client 1 SRL	subtotal (toate zilele)	5021287.0000
Client 2 SA	Sep 2 2007 12:00AM	127530.0000
Client 2 SA	subtotal (toate zilele)	127530.0000
Client 3 SRL	Sep 7 2007 12:00AM	5819668.0000
Client 3 SRL	subtotal (toate zilele)	5819668.0000
Client 5 SRL	Sep 1 2007 12:00AM	153442.0000
Client 5 SRL	subtotal (toate zilele)	153442.0000
Client 7 SRL	Sep 10 2007 12:00AM	136850.0000
Client 7 SRL	subtotal (toate zilele)	136850.0000
NULL	subtotal (toate zilele)	11258777.0...

Figura 11.5. Un alt tratament concatenării spațiilor cu NULLi în MS SQL Server

DB2 se comportă similar SQL Server-ului, cu diferență că, în rezultat, linia de total general apare nu la sfârșit, ci la începutul listei. După aceasta urmează liniile de subtotal și apoi cele „curente”. În consecință, Oracle ne dă ghes să folosim o structură CASE:

```

SELECT COALESCE(DenCl, '{Total clienti}') AS DenumireClient,
      CASE WHEN DataFact IS NULL THEN 'subtotal (toate zilele)'
      ELSE CAST (DataFact AS CHAR(10)) || ' ' END AS Data,
      ROUND(SUM(Cantitate * PretUnit * (1+ProcTVA)),0) AS Vinzari
FROM clienti c
      INNER JOIN facturi f ON c.CodCl=f.CodCl
      INNER JOIN liniifact lf ON lf.NrFact=f.NrFact
      INNER JOIN produse p ON lf.CodPr=p.CodPr
WHERE EXTRACT (YEAR FROM DataFact) = 2007
      AND EXTRACT (MONTH FROM DataFact) = 9
GROUP BY ROLLUP (DenCl, DataFact)

```

Figura 11.6 ne arată că facem progrese remarcabile, însă la rândul de total general, pe care apare acum *{Total clienti}*, pe coloana Data apare șirul *subtotal (toate zilele)*, ceea ce, iarăși, ne irită un pic (nu știu de ce suntem așa de iritați astăzi !).

R 2	DENUMIRECLIENT	R 2	DATA	R 2	VINZARI
	Client 2 SA		02-09-2007		127530
	Client 2 SA		subtotal (toate zilele)		127530
	Client 1 SRL		01-09-2007		4442960
	Client 1 SRL		02-09-2007		110908
	Client 1 SRL		10-09-2007		287855
	Client 1 SRL		17-09-2007		179565
	Client 1 SRL		subtotal (toate zilele)		5021287
	Client 3 SRL		07-09-2007		5819668
	Client 3 SRL		subtotal (toate zilele)		5819668
	Client 5 SRL		01-09-2007		153442
	Client 5 SRL		subtotal (toate zilele)		153442
	Client 7 SRL		10-09-2007		136850
	Client 7 SRL		subtotal (toate zilele)		136850
	{Total clienti}		subtotal (toate zilele)		11258777

Figura 11.6. Un ghimpe Oracle la rândul de total general

În plus, nici acum clienții nu sunt ordonați alfabetic, așa că, imaginăm o nouă versiune în care să includem un CASE în CASE, plus o clauză de ordonare:

```

SELECT COALESCE(DenCl, '{TOTAL GENERAL}') AS DenumireClient,
       CASE
         WHEN DataFact IS NULL THEN
           CASE WHEN DenCl IS NULL THEN ''
                ELSE 'subtotal (toate zilele)'
           END
         ELSE CAST (DataFact AS CHAR(10)) || ''
       END AS Data,
       ROUND(SUM(Cantitate * PretUnit * (1+ProcTVA)),0) AS Vinzari
FROM clienti c
     INNER JOIN facturi f ON c.CodCl=f.CodCl
     INNER JOIN liniifact lf ON lf.NrFact=f.NrFact
     INNER JOIN produse p ON lf.CodPr=p.CodPr
WHERE EXTRACT (YEAR FROM DataFact) = 2007
      AND EXTRACT (MONTH FROM DataFact) = 9
GROUP BY ROLLUP (DenCl, DataFact)
ORDER BY 1,2

```

După „scrutarea” figurii 11.7 este evident că în Oracle avem o zi proastă, întrucât chiar dacă rândul total este în regulă, clienții sunt ordonați după denumire, liniile fiecărui client sunt ordonate aiurea, rândul de subtotal fiind plasat la începutul clientului, nu la final.

DENUMIRECLIENT	DATA	VINZARI
Client 1 SRL	subtotal (toate zilele)	5021287
Client 1 SRL	01-09-2007	4442960
Client 1 SRL	02-09-2007	110908
Client 1 SRL	10-09-2007	287855
Client 1 SRL	17-09-2007	179565
Client 2 SA	subtotal (toate zilele)	127530
Client 2 SA	02-09-2007	127530
Client 3 SRL	subtotal (toate zilele)	5819668
Client 3 SRL	07-09-2007	5819668
Client 5 SRL	subtotal (toate zilele)	153442
Client 5 SRL	01-09-2007	153442
Client 7 SRL	subtotal (toate zilele)	136850
Client 7 SRL	10-09-2007	136850
{TOTAL GENERAL}		11258777

Figura 11.7. Alt ghimpe Oracle (poziția liniei de subtotal la nivel de client)

Ciuda este cu atât mai mare, cu cât MS SQL Server ordonează liniile în rezultat cum trebuie, chiar și fără clauza ORDER BY:

```
SELECT COALESCE(DenCl, '{TOTAL GENERAL}') AS DenumireClient,
CASE
    WHEN DataFact IS NULL THEN
        CASE
            WHEN DenCl IS NULL THEN ''
            ELSE 'subtotal (toate zilele)'
        END
    ELSE CAST (DataFact AS CHAR) + ''
END AS Data,
ROUND(SUM(Cantitate * PretUnit * (1+ProcTVA)),0,0) AS Vinzari
FROM clienti c
INNER JOIN facturi f ON c.CodCl=f.CodCl
INNER JOIN liniifact lf ON lf.NrFact=f.NrFact
INNER JOIN produse p ON lf.CodPr=p.CodPr
WHERE DATEPART (YEAR,DataFact) = 2007
AND DATEPART(MONTH,DataFact) = 9
GROUP BY DenCl, DataFact WITH ROLLUP
```

Obținerea ordinii dorite în Oracle nu necesită un efort intelectual deosebit. Este suficient că în CASE-ul „principal”, pe ramura ELSE, să adăugăm un spațiu și înaintea DataFact (vezi textul italic din interogarea de mai jos:

```
SELECT COALESCE(DenCl, '{TOTAL GENERAL}') AS DenumireClient,
CASE
    WHEN DataFact IS NULL THEN
```

```

CASE WHEN DenCl IS NULL THEN ''
      ELSE 'subtotal (toate zilele)'
END
ELSE '' || DataFact || ''
END AS Data,
ROUND(SUM(Cantitate * PretUnit * (1+ProcTVA))) AS Vinzari
FROM
... continuare identică interogării Oracle precedente

```

Nu mai includem figura cu rezultatul, dar credeți-mă pe cuvânt că ordinea este așa cum ne-am propus-o. DB2 face nazuri similare, chiar și cu rândul de total general, așa că avem toate motivele să folosim varianta:

```

SELECT COALESCE('' || DenCl, '{TOTAL GENERAL}') AS DenumireClient,
CASE
  WHEN DataFact IS NULL THEN
    CASE WHEN DenCl IS NULL THEN ''
          ELSE 'subtotal (toate zilele)'
        END
      ELSE '' || CAST(DataFact AS CHAR(10)) || ''
    END AS Data,
ROUND(SUM(Cantitate * PretUnit * (1+ProcTVA)),0) AS Vinzari
FROM clienti c
  INNER JOIN facturi f ON c.CodCl=f.CodCl
  INNER JOIN liniifact lf ON lf.NrFact=f.NrFact
  INNER JOIN produse p ON lf.CodPr=p.CodPr
WHERE YEAR(DataFact) = 2007 AND MONTH(DataFact) = 9
GROUP BY ROLLUP (DenCl, DataFact)
ORDER BY 1,2

```

Care este totalul vânzărilor, pe clienți, localități, județe și totalul general ?

Pe baza soluției de la problema precedentă, bănuiam că nu mai avem ce comenta:

```

SELECT COALESCE(Judet, '{TOTAL GENERAL}') AS Judet,
CASE  WHEN Judet IS NULL THEN ''
      ELSE COALESCE(Loc, '{Subtotal-JUDET}' || Judet)
END AS Loc,
CASE  WHEN Loc IS NULL THEN ''
      ELSE COALESCE(DenCl, '{Subtotal-LOCALITATE}' || Loc)
END AS DenCl,
ROUND(SUM(Cantitate * PretUnit * (1+ProcTVA)),0) AS Vinzari
FROM judete j

```



```

INNER JOIN coduri_postale cp ON j.Jud=cp.Jud
INNER JOIN clienti c ON cp.CodPost=c.CodPost
INNER JOIN facturi f ON c.CodCl=f.CodCl
INNER JOIN liniifact lf ON lf.NrFact=f.NrFact
INNER JOIN produse p ON lf.CodPr=p.CodPr
GROUP BY ROLLUP (Judet, Loc, DenCl)
ORDER BY 1,2,3

```

Și în acest caz (Oracle) ordinea dispunerii liniilor în rezultat este, în ciuda sforțărilor noastre (clauza ORDER BY, folosirea acoladei care are codul ASCII superior oricărei litere), aiurită – vezi figura 11.8.

R	JUDET	R	LOC	R	DENCL	R	VINZARI
	Iasi		Iasi		Client 1 SRL		15061538
	Iasi		Iasi		Client 2 SA		361335
	Iasi		Iasi		{Subtotal-LOCALITATE}Iasi		15422873
	Iasi		Pascani		Client 4		9479110
	Iasi		Pascani		{Subtotal-LOCALITATE}Pascani		9479110
	Iasi		{Subtotal-JUDET}Iasi				24901983
	Neamt		Roman		Client 6 SA		6021707
	Neamt		Roman		{Subtotal-LOCALITATE}Roman		6021707
	Neamt		{Subtotal-JUDET}Neamt				6021707
	Timis		{Subtotal-JUDET}Timis				832812
	Timis		Timisoara		Client 5 SRL		432400
	Timis		Timisoara		Client 7 SRL		400412
	Timis		Timisoara		{Subtotal-LOCALITATE}Timisoara		832812
	{TOTAL GENERAL}						49170335
	Vaslui		{Subtotal-JUDET}Vaslui				17413835
	Vaslui		Vaslui		Client 3 SRL		17413835
	Vaslui		Vaslui		{Subtotal-LOCALITATE}Vaslui		17413835

Figura 11.8. (Alte) depresii în Oracle

Pentru a obține rezultatul corect, cu liniile dispuse în ordine firească nu avem decât să renunțăm, din interogarea de mai sus, la clauza ORDER BY !!!!- vezi figura 11.9. Nu aș recomanda, însă, această soluție (este foarte posibil ca pe calculatoarele dvs. ordinea să fie alta), ci inserarea unui spațiu (cum am procedat cu DataFact în Oracle la problema precedentă) în structura CASE pentru afișarea valorilor coloanei Judet:

```

SELECT
    CASE WHEN Judet IS NULL THEN '{TOTAL GENERAL}'
        ELSE ' ' || Judet
    END AS Judet,
    CASE WHEN Judet IS NULL THEN ''
        ELSE COALESCE(Loc, '{Subtotal-JUDET}' || Judet)
    END AS Loc,

```

```

CASE WHEN Loc IS NULL THEN ''
      ELSE COALESCE(DenCl, '{Subtotal-LOCALITATE}' || Loc)
END AS DenCl,
ROUND(SUM(Cantitate * PretUnit * (1+ProcTVA)),0) AS Vinzari
FROM
... identic interogării Oracle precedente

```

JUDET	LOC	DENCL	VINZARI
Iasi	Iasi	Client 1 SRL	15061538
Iasi	Iasi	Client 2 SA	361335
Iasi	Iasi	{Subtotal-LOCALITATE}Iasi	15422873
Iasi	Pascani	Client 4	9479110
Iasi	Pascani	{Subtotal-LOCALITATE}Pascani	9479110
Iasi	{Subtotal-JUDET}Iasi		24901983
Neamt	Roman	Client 6 SA	6021707
Neamt	Roman	{Subtotal-LOCALITATE}Roman	6021707
Neamt	{Subtotal-JUDET}Neamt		6021707
Timis	Timisoara	Client 5 SRL	432400
Timis	Timisoara	Client 7 SRL	400412
Timis	Timisoara	{Subtotal-LOCALITATE}Timisoara	832812
Timis	{Subtotal-JUDET}Timis		832812
Vaslui	Vaslui	Client 3 SRL	17413835
Vaslui	Vaslui	{Subtotal-LOCALITATE}Vaslui	17413835
Vaslui	{Subtotal-JUDET}Vaslui		17413835
{TOTAL GENERAL}			49170335

Figura 11.9. Ordonare corectă în Oracle la folosirea funcției ROLLUP

Soluția a fost aplicată deja în DB2, așa că nu mai formulăm și sintaxa acestei interogări pe dialectul SQL de la IBM.

11.1.2. ROLLUP și GROUPING

Structura alternativă de tip CASE poate utiliza și o nouă funcție introdusă în SQL:1999 – GROUPING. Argumentul acesteia este coloana de grupare, rezultatul furnizat fiind 1, atunci când coloana respectivă este inclusă într-un grup de agregare superior, sau 0 pentru liniile “normale” (din afara subtotalurilor). Pentru edificare, să examinăm rezultatul din figura 11.10 produs de interogarea (sintaxă DB2/Oracle):

```

SELECT Judet, Loc, DenCl,
      SUM(Cantitate * PretUnit * (1+ProcTVA)) AS Vinzari,
      GROUPING (Judet) AS Grup_Judet,
      GROUPING (Loc) AS Grup_Loc,

```

```

GROUPING (DenCl) AS Grup_DenCl
FROM judete j
INNER JOIN coduri_postale cp ON j.Jud=cp.Jud
INNER JOIN clienti c ON cp.CodPost=c.CodPost
INNER JOIN facturi f ON c.CodCl=f.CodCl
INNER JOIN liniifact lf ON lf.NrFact=f.NrFact
INNER JOIN produse p ON lf.CodPr=p.CodPr
GROUP BY ROLLUP (Judet, Loc, DenCl)

```

JUDET	LOC	DENCL	VINZARI	GRUP_JUDET	GRUP_LOC	GRUP_DENCL
Iasi	Iasi	Client 2 SA	361335	0	0	0
Iasi	Iasi	Client 1 SRL	15061538	0	0	0
Iasi	Iasi	(null)	15422873	0	0	1
Iasi	Pascani	Client 4	9479109,5	0	0	0
Iasi	Pascani	(null)	9479109,5	0	0	1
Iasi	(null)	(null)	24901982,5	0	1	1
Neamt	Roman	Client 6 SA	6021706,5	0	0	0
Neamt	Roman	(null)	6021706,5	0	0	1
Neamt	(null)	(null)	6021706,5	0	1	1
Timis	Timisoara	Client 5 SRL	432400	0	0	0
Timis	Timisoara	Client 7 SRL	400411,5	0	0	0
Timis	Timisoara	(null)	832811,5	0	0	1
Timis	(null)	(null)	832811,5	0	1	1
Vaslui	Vaslui	Client 3 SRL	17413834,5	0	0	0
Vaslui	Vaslui	(null)	17413834,5	0	0	1
Vaslui	(null)	(null)	17413834,5	0	1	1
(null)	(null)	(null)	49170335	1	1	1

Figura 11.10. Valori furnizate de funcția GROUPING

Valoarea 1 returnată de funcția GROUPING pentru un atribut indică un subtotal pentru acel atribut, relativ la atributele din stânga (în ordinea declarată în clauza ROLLUP). Pe baza funcției GROUPING, rezultatul din figura 11.9 poate fi obținut și astfel:

```

SELECT CASE
    WHEN GROUPING (Judet) = 1 THEN '{TOTAL GENERAL}'
    ELSE '' || Judet
END AS Judet,
CASE
    WHEN GROUPING (Judet) = 1 THEN ''
    WHEN GROUPING(Loc) = 1 THEN '{Subtotal-JUDET}' || Judet
    ELSE '' || Loc
END AS Loc,
CASE

```

```

        WHEN GROUPING (Judet) = 1 OR GROUPING(Loc) = 1 THEN ''
        WHEN GROUPING (DenCl) = 1 THEN '{Subtotal-LOCALITATE}' || Loc
        ELSE '' || DenCl
    END AS DenCl,
    ROUND(SUM(Cantitate * PretUnit * (1+ProcTVA)),0) AS Vinzari
FROM judete j
    INNER JOIN coduri_postale cp ON j.Jud=cp.Jud
    INNER JOIN clienti c ON cp.CodPost=c.CodPost
    INNER JOIN facturi f ON c.CodCl=f.CodCl
    INNER JOIN liniifact lf ON lf.NrFact=f.NrFact
    INNER JOIN produse p ON lf.CodPr=p.CodPr
GROUP BY ROLLUP (Judet, Loc, DenCl)
ORDER BY 1,2,3

```

Interogarea funcționează fără modificări în DB2. Schimbând operatorul de concatenare (din || în +) și clauza GROUP BY din *GROUP BY ROLLUP (Judet, Loc, DenCl)* în *GROUP BY Judet, Loc, DenCl WITH ROLLUP*, interogarea funcționează și în MS SQL Server.

11.1.3. ROLLUP-uri parțiale

Este de dorit, uneori, ca subtotalizarea atributelor de grupare să fie parțială: fie nu interesează totalul general, fie subtotalizarea este necesară numai pentru anumite atribute sau grupuri de atribute. Schimbăm cerințele de raportare, în sensul că acum *vânzările din luna septembrie 2007 ne interesează pe facturi, clienți, localități și județe. Subtotalurile, însă, trebuie calculate numai la nivel de client și localitate, iar totalul general nu ne privește*. Interogarea Oracle ce furnizează răspunsul (figura 11.11) este:

```

SELECT CASE
    WHEN GROUPING (Judet) = 1 THEN '{TOTAL GENERAL}'
    ELSE Judet
END AS Judet,
CASE
    WHEN GROUPING (Judet) = 1 THEN ''
    WHEN GROUPING(Loc) = 1 THEN '{Subtotal-JUDET}' || Judet
    ELSE Loc
END AS Loc,
CASE
    WHEN GROUPING (Judet) = 1 OR GROUPING(Loc) = 1 THEN ''
    WHEN GROUPING (DenCl) = 1 THEN '{Subtotal-LOCALITATE}' || Loc
    ELSE '' || DenCl
END AS DenCl,

```

```

CASE
    WHEN GROUPING (Judet) = 1 OR GROUPING(Loc) = 1
        OR GROUPING (DenCl) = 1 THEN ''
    WHEN GROUPING (f.NrFact) = 1 THEN '{Subtotal-CLIENT}' || DenCl
    ELSE '' || CAST(f.NrFact AS CHAR(8)) || ''
END AS Factura,
ROUND(SUM(Cantitate * PretUnit * (1+ProcTVA)),0) AS Vinzari
FROM judete j
    INNER JOIN coduri_postale cp ON j.Jud=cp.Jud
    INNER JOIN clienti c ON cp.CodPost=c.CodPost
    INNER JOIN facturi f ON c.CodCl=f.CodCl
    INNER JOIN liniifact lf ON lf.NrFact=f.NrFact
    INNER JOIN produse p ON lf.CodPr=p.CodPr
WHERE EXTRACT (YEAR FROM DataFact)=2007 AND
    EXTRACT (MONTH FROM DataFact)=9
GROUP BY Judet, Loc, ROLLUP (DenCl, f.NrFact)
ORDER BY 1,2,3,4

```

R2	JUDET	R2	LOC	R2	DENCL	R2	FACTURA	R2	VINZARI
	Iasi		Iasi		Client 1 SRL		3111		4442960
	Iasi		Iasi		Client 1 SRL		3115		110908
	Iasi		Iasi		Client 1 SRL		3117		287855
	Iasi		Iasi		Client 1 SRL		3118		179565
	Iasi		Iasi		Client 1 SRL		{Subtotal-CLIENT}Client 1 SRL		5021287
	Iasi		Iasi		Client 2 SA		3113		127530
	Iasi		Iasi		Client 2 SA		{Subtotal-CLIENT}Client 2 SA		127530
	Iasi		Iasi		{Subtotal-LOCALITATE}Iasi				5148817
	Timis		Timisoara		Client 5 SRL		3112		153442
	Timis		Timisoara		Client 5 SRL		{Subtotal-CLIENT}Client 5 SRL		153442
	Timis		Timisoara		Client 7 SRL		3116		136850
	Timis		Timisoara		Client 7 SRL		{Subtotal-CLIENT}Client 7 SRL		136850
	Timis		Timisoara		{Subtotal-LOCALITATE}Timisoara				290292
	Vaslui		Vaslui		Client 3 SRL		3119		5819668
	Vaslui		Vaslui		Client 3 SRL		{Subtotal-CLIENT}Client 3 SRL		5819668
	Vaslui		Vaslui		{Subtotal-LOCALITATE}Vaslui				5819668

Figura 11.11. ROLLUP parțial

Încludând funcțiile EXTRACT cu YEAR și MONTH se obține varianta funcționabilă în DB2. Prin clauza *GROUP BY Judet, Loc, ROLLUP (DenCl, F.NrFact)* se vor calcula două tipuri de subtotaluri:

- unul pentru valori distincte ale combinației (Judet, Loc și DenCl) și
- altul pentru valori distincte ale combinației (Judet, Loc).

Reținem, deci: o coloană declarată în ROLLUP determină calcularea subtotalurilor pentru valori distincte ale celorlalte coloane. Astfel, interogarea Oracle următoare produce raportul din figura 11.12.

```

SELECT
    CASE
        WHEN GROUPING (Judet) = 1 THEN '{TOTAL GENERAL}'
        ELSE Judet
    END AS Judet,
    CASE
        WHEN GROUPING (Judet) = 1 THEN ''
        WHEN GROUPING(Loc) = 1 THEN '{Subtotal-JUDET}' || Judet
        ELSE Loc
    END AS Loc,
    CASE
        WHEN GROUPING (Judet) = 1 OR GROUPING(Loc) = 1 THEN ''
        WHEN GROUPING (DenCl) = 1 THEN '{Subtotal-LOCALITATE}' || Loc
        ELSE DenCl
    END AS DenCl,
    CASE
        WHEN GROUPING (Judet) = 1 OR GROUPING(Loc) = 1
            OR GROUPING (DenCl) = 1 THEN ''
        WHEN GROUPING (f.NrFact) = 1 THEN '{Subtotal-CLIENT}' || DenCl
        ELSE CAST (f.NrFact AS CHAR(8)) || ''
    END AS Factura,
    ROUND(SUM(Cantitate * PretUnit * (1+ProcTVA)),0) AS Vinzari
FROM judete j
    INNER JOIN coduri_postale cp ON j.Jud=cp.Jud
    INNER JOIN clienti c ON cp.CodPost=c.CodPost
    INNER JOIN facturi f ON c.CodCl=f.CodCl
    INNER JOIN liniifact lf ON lf.NrFact=f.NrFact
    INNER JOIN produse p ON lf.CodPr=p.CodPr
WHERE EXTRACT (YEAR FROM DataFact)=2007
    AND EXTRACT (MONTH FROM DataFact)=9
GROUP BY Judet, ROLLUP(Loc, DenCl), f.NrFact
ORDER BY 1,2,3,4

```

Rezultatul este destul de anapoda, dar nici interogarea nu-i mai prejos ! Practic, după fiecare valoare distinctă a combinației atributelor din GROUP BY se calculează un subtotal pe județ-localitate și altul pe județ (deoarece attributele din ROLLUP sunt DenCl și Loc).

În SQL Server ROLLUP-urile parțiale nu sunt posibile direct, ci doar prin grupări clasice, și reuniuni, așa că nu mai cheltuim timp și spațiu cu redactarea interogării care să afișeze rezultatul din figura 11.12.

JUDET	LOC	DENCL	FACTURA	VINZARI
Iasi	Iasi	Client 1 SRL	3111	4442960
Iasi	Iasi	{Subtotal-LOCALITATE}Iasi		4442960
Iasi	{Subtotal-JUDET}Iasi			4442960
Iasi	Iasi	Client 2 SA	3113	127530
Iasi	Iasi	{Subtotal-LOCALITATE}Iasi		127530
Iasi	{Subtotal-JUDET}Iasi			127530
Iasi	Iasi	Client 1 SRL	3115	110908
Iasi	Iasi	{Subtotal-LOCALITATE}Iasi		110908
Iasi	{Subtotal-JUDET}Iasi			110908
Iasi	Iasi	Client 1 SRL	3117	287855
Iasi	Iasi	{Subtotal-LOCALITATE}Iasi		287855
Iasi	{Subtotal-JUDET}Iasi			287855
Iasi	Iasi	Client 1 SRL	3118	179565
Iasi	Iasi	{Subtotal-LOCALITATE}Iasi		179565
Iasi	{Subtotal-JUDET}Iasi			179565
Timis	Timisoara	Client 5 SRL	3112	153442
Timis	Timisoara	{Subtotal-LOCALITATE}Timisoara		153442
Timis	{Subtotal-JUDET}Timis			153442
Timis	Timisoara	Client 7 SRL	3116	136850
Timis	Timisoara	{Subtotal-LOCALITATE}Timisoara		136850
Timis	{Subtotal-JUDET}Timis			136850
Vaslui	Vaslui	Client 3 SRL	3119	5819668
Vaslui	Vaslui	{Subtotal-LOCALITATE}Vaslui		5819668
Vaslui	{Subtotal-JUDET}Vaslui			5819668

Figura 11.12. Un alt ROLLUP parțial, dar ceva mai curios

11.2. Analize multidimensionale. Operatorii CUBE și GROUPING SETS

De multe ori, pentru a produce o impresie mai puternică partenerilor de discuții, analiza datelor trebuie să fie una multi-dimensională. Apanaj, prin excelență, al... Excel-ului, sau pretențioaselor instrumente OLAP, acest paragraf este menit să demonstreze că prin SQL:1999 sunt create condiții ideale pentru acest gen de operațiuni.

11.2.1. Cuburi & cubulețe

Pentru început, interesează vizualizarea vânzărilor din luna septembrie 2007 pe două axe, *produse* și *clienți*, ca în figura 11.13.

CLIENT	PRODUS	VINZARI
Client 1 SRL	Produs 1	324989
Client 1 SRL	Produs 2	411584
Client 1 SRL	Produs 5	4284714
Client 2 SA	Produs 2	127530
Client 3 SRL	Produs 2	41584
Client 3 SRL	Produs 3	33320
Client 3 SRL	Produs 4	84530
Client 3 SRL	Produs 5	5660235
Client 5 SRL	Produs 2	95430
Client 5 SRL	Produs 3	58013
Client 7 SRL	Produs 2	136850
Client 1 SRL	{}	5021287
Client 2 SA	{}	127530
Client 3 SRL	{}	5819668
Client 5 SRL	{}	153442
Client 7 SRL	{}	136850
{}	Produs 1	324989
{}	Produs 2	812977
{}	Produs 3	91333
{}	Produs 4	84530
{}	Produs 5	9944949
{}	{}	11258777

Figura 11.13. Analiză pe două dimensiuni

Întrucât *produse* și *clienți* sunt două „variabile” independente, vor exista patru variante de grupare a datelor:

- grupare după client și produs,
- grupare numai după client,
- grupare numai după produs și
- un grup pentru total general.

Soluțiile redactate urmând sintaxa SQL-92 sunt impresionante ca întindere, iar dacă analiza se realizează după trei, patru... variabile, impresionantul se transformă în halucinant. Iată prima interogare Oracle ce obține un raport de genul celui din figura 11.13.

```
SELECT ' ' || DenCl AS Client, ' ' || DenPr AS Produs,
       ROUND(SUM(Cantitate * PretUnit * (1+ProcTVA)),0) AS Vinzari
FROM clienti c
```



```

        INNER JOIN facturi f ON c.CodCl=f.CodCl
        INNER JOIN liniifact lf ON lf.NrFact=f.NrFact
        INNER JOIN produse p ON lf.CodPr=p.CodPr
WHERE EXTRACT (YEAR FROM DataFact)=2007 AND
        EXTRACT (MONTH FROM DataFact)=9
GROUP BY DenCl, DenPr
UNION
SELECT DenCl AS Client, '{}' AS Produs,
        ROUND(SUM(Cantitate * PretUnit * (1+ProcTVA)),0) AS Vinzari
FROM clienti c
        INNER JOIN facturi f ON c.CodCl=f.CodCl
        INNER JOIN liniifact lf ON lf.NrFact=f.NrFact
        INNER JOIN produse p ON lf.CodPr=p.CodPr
WHERE EXTRACT (YEAR FROM DataFact)=2007 AND
        EXTRACT (MONTH FROM DataFact)=9
GROUP BY DenCl
UNION
SELECT '{}' AS Client, DenPr AS Produs,
        ROUND(SUM(Cantitate * PretUnit * (1+ProcTVA)),0) AS Vinzari
FROM clienti c
        INNER JOIN facturi f ON c.CodCl=f.CodCl
        INNER JOIN liniifact lf ON lf.NrFact=f.NrFact
        INNER JOIN produse p ON lf.CodPr=p.CodPr
WHERE EXTRACT (YEAR FROM DataFact)=2007 AND
        EXTRACT (MONTH FROM DataFact)=9
GROUP BY DenPr
UNION
SELECT '{}' AS Client, '{}' AS Produs,
        ROUND(SUM(Cantitate * PretUnit * (1+ProcTVA)),0) AS Vinzari
FROM clienti c
        INNER JOIN facturi f ON c.CodCl=f.CodCl
        INNER JOIN liniifact lf ON lf.NrFact=f.NrFact
        INNER JOIN produse p ON lf.CodPr=p.CodPr
WHERE EXTRACT (YEAR FROM DataFact)=2007 AND
        EXTRACT (MONTH FROM DataFact)=9

```

Pentru astfel de situații există un operator cuceritor prin simplitate: CUBE. Cu ajutorul acestuia, se poate redacta următoarea variantă care produce raportul din figura 11.14:

```

SELECT DenCl AS Client, DenPr AS Produs,
        ROUND(SUM(Cantitate * PretUnit * (1+ProcTVA)),0) AS Vinzari

```

```

FROM clienti c
    INNER JOIN facturi f ON c.CodCl=f.CodCl
    INNER JOIN liniifact lf ON lf.NrFact=f.NrFact
    INNER JOIN produse p ON lf.CodPr=p.CodPr
WHERE EXTRACT (YEAR FROM DataFact)=2007 AND
    EXTRACT (MONTH FROM DataFact)=9
GROUP BY CUBE (DenCl, DenPr )
ORDER BY DenCl, DenPr

```

CLIENT	PRODUS	VINZARI
Client 1 SRL	Produs 1	324989
Client 1 SRL	Produs 2	411584
Client 1 SRL	Produs 5	4284714
Client 1 SRL	(null)	5021287
Client 2 SA	Produs 2	127530
Client 2 SA	(null)	127530
Client 3 SRL	Produs 2	41584
Client 3 SRL	Produs 3	33320
Client 3 SRL	Produs 4	84530
Client 3 SRL	Produs 5	5660235
Client 3 SRL	(null)	5819668
Client 5 SRL	Produs 2	95430
Client 5 SRL	Produs 3	58013
Client 5 SRL	(null)	153442
Client 7 SRL	Produs 2	136850
Client 7 SRL	(null)	136850
(null)	Produs 1	324989
(null)	Produs 2	812977
(null)	Produs 3	91333
(null)	Produs 4	84530
(null)	Produs 5	9944949
(null)	(null)	11258777

Figura 11.14. Un cub neprelucrat

În DB2 trebuie înlocuite cele două funcții EXTRACT, iar sintaxa MS SQL Server este similară:

```

SELECT DenCl AS Client, DenPr AS Produs,
    ROUND(SUM(Cantitate * PretUnit * (1+ProcTVA)),0) AS Vinzari
FROM clienti c
    INNER JOIN facturi f ON c.CodCl=f.CodCl
    INNER JOIN liniifact lf ON lf.NrFact=f.NrFact
    INNER JOIN produse p ON lf.CodPr=p.CodPr
WHERE DATEPART(YEAR,DataFact)=2007 AND DATEPART(MONTH,DataFact)=9

```

GROUP BY DenCl, DenPr WITH CUBE

CLIENT	PRODUS	VINZARI
Client 1 SRL	Produs 1	324989
Client 1 SRL	Produs 2	411584
Client 1 SRL	Produs 5	4284714
Client 1 SRL	Subtotal client	5021287
Client 2 SA	Produs 2	127530
Client 2 SA	Subtotal client	127530
Client 3 SRL	Produs 2	41584
Client 3 SRL	Produs 3	33320
Client 3 SRL	Produs 4	84530
Client 3 SRL	Produs 5	5660235
Client 3 SRL	Subtotal client	5819668
Client 5 SRL	Produs 2	95430
Client 5 SRL	Produs 3	58013
Client 5 SRL	Subtotal client	153442
Client 7 SRL	Produs 2	136850
Client 7 SRL	Subtotal client	136850
Subtotal produs	Produs 1	324989
Subtotal produs	Produs 2	812977
Subtotal produs	Produs 3	91333
Subtotal produs	Produs 4	84530
Subtotal produs	Produs 5	9944949
Total general		11258777

Figura 11.15. Un cub ceva mai lustruit

Pentru un plus de atractivitate, se poate recurge la clauza GROUPING astfel:

```
SELECT CASE
```

```
    WHEN GROUPING(DenCl)=1 AND GROUPING(DenPr)=1 THEN 'Total general'
```

```
    WHEN GROUPING(DenCl)=1 AND GROUPING(DenPr)=0
```

```
        THEN 'Subtotal produs '
```

```
    ELSE DenCl
```

```
    END AS Client,
```

```
    CASE
```

```
        WHEN GROUPING(DenCl)=1 AND GROUPING(DenPr)=1 THEN ''
```

```
        WHEN GROUPING(DenCl)=0 AND GROUPING(DenPr)=1
```

```
            THEN 'Subtotal client '
```

```
        ELSE DenPr
```

```
    END AS Produs,
```

```

ROUND(SUM(Cantitate * PretUnit * (1+ProcTVA)),0) AS Vinzari
FROM clienti c
    INNER JOIN facturi f ON c.CodCl=f.CodCl
    INNER JOIN liniifact lf ON lf.NrFact=f.NrFact
    INNER JOIN produse p ON lf.CodPr=p.CodPr
WHERE EXTRACT (YEAR FROM DataFact)=2007 AND
    EXTRACT (MONTH FROM DataFact)=9
GROUP BY CUBE (DenCl, DenPr )
ORDER BY DenCl, DenPr

```

Drept răsplată, forma raportului este cea din figura 11.15. Se mai cuvine de adăugat că atributele care definesc dimensiunile de analiză trebuie să fie de același tip. Interogarea funcționează în DB2 (aveți grijă la EXTRACT) și în SQL Server, schimbând funcțiile EXTRACT cu DATEPART și clauza GROUP BY.

Un mare avantaj al CUBE-ului iese la iveală când suntem în criză de idei, sau rapoartele sunt prea „subțiri”. Trecerea de la analiza pe două axe la cea pe trei axe: clienți, produse și zile, este cât se poate de spumoasă:

```

SELECT
    CASE
        WHEN GROUPING(DenCl)=1 AND GROUPING(DenPr)=1
            AND GROUPING(DataFact)=1 THEN '{{{Total general}}}'
        WHEN GROUPING(DenCl)=1 AND GROUPING(DenPr)=1 AND
            GROUPING(DataFact)=0 THEN '{{Subtotal pentru ziua}}'
        WHEN GROUPING(DenCl)=1 AND GROUPING(DenPr)=0 AND
            GROUPING(DataFact)=1 THEN '{{Subtotal pentru produs}}'
        WHEN GROUPING(DenCl)=1 AND GROUPING(DenPr)=0 AND
            GROUPING(DataFact)=0 THEN '{Subtotal produs/zi}'
        ELSE DenCl
    END AS Client,
    CASE
        WHEN GROUPING(DenCl)=1 AND GROUPING(DenPr)=1 AND
            GROUPING(DataFact)=1 THEN '{{{}}}'
        WHEN GROUPING(DenCl)=1 AND GROUPING(DenPr)=1 AND
            GROUPING(DataFact)=0 THEN '{{{}}}'
        WHEN GROUPING(DenCl)=0 AND GROUPING(DenPr)=1 AND
            GROUPING(DataFact)=0 THEN '{Subtotal client/zi}'
        WHEN GROUPING(DenCl)=0 AND GROUPING(DenPr)=1 AND
            GROUPING(DataFact)=1 THEN '{{Subtotal client}}'
        ELSE ' ' || DenPr
    END AS Produs,

```

```

CASE
    WHEN GROUPING(DenCl)=1 AND GROUPING(DenPr)=1 AND
         GROUPING(DataFact)=1 THEN '{{{}}}'
    WHEN GROUPING(DenCl)=1 AND GROUPING(DenPr)=0 AND
         GROUPING(DataFact)=1 THEN '{{}}'
    WHEN GROUPING(DenCl)=0 AND GROUPING(DenPr)=0 AND
         GROUPING(DataFact)=1 THEN '{Subtotal client/produs}'
    ELSE '' || DataFact || ''
END AS Zi,
ROUND(SUM(Cantitate * PretUnit * (1+ProcTVA)),0) AS Vinzari
FROM clienti c
    INNER JOIN facturi f ON c.CodCl=f.CodCl
    INNER JOIN liniifact lf ON lf.NrFact=f.NrFact
    INNER JOIN produse p ON lf.CodPr=p.CodPr
WHERE EXTRACT (YEAR FROM DataFact)=2007 AND
      EXTRACT (MONTH FROM DataFact)=9
GROUP BY CUBE (DenCl, DenPr, DataFact )
ORDER BY 1,2,3

```

Rezultatul, fiind mai lung decât o pagină, este afișat sub formă de tabel (Tabelul 11.1). Atenție, este posibil să aveți probleme la ordonarea rezultatului în Oracle.

Tabelul 11.1 Analiză pe trei dimensiuni

CLIENT	PRODUS	ZI	VINZARI
Client 1 SRL	Produs 1	01-09-2007	67830
Client 1 SRL	Produs 1	10-09-2007	124355
Client 1 SRL	Produs 1	17-09-2007	132804
Client 1 SRL	Produs 1	{Subtotal client/produs}	324989
Client 1 SRL	Produs 2	01-09-2007	90416
Client 1 SRL	Produs 2	02-09-2007	110908
Client 1 SRL	Produs 2	10-09-2007	163500
Client 1 SRL	Produs 2	17-09-2007	46761
Client 1 SRL	Produs 2	{Subtotal client/produs}	411584
Client 1 SRL	Produs 5	01-09-2007	4284714
Client 1 SRL	Produs 5	{Subtotal client/produs}	4284714
Client 1 SRL	{{Subtotal client}}		5021287
Client 1 SRL	{Subtotal client/zi}	01-09-2007	4442960
Client 1 SRL	{Subtotal client/zi}	02-09-2007	110908
Client 1 SRL	{Subtotal client/zi}	10-09-2007	287855
Client 1 SRL	{Subtotal client/zi}	17-09-2007	179565
Client 2 SA	Produs 2	02-09-2007	127530

Client 2 SA	Produs 2	{Subtotal client/produs}	127530
Client 2 SA	{{Subtotal client}}		127530
Client 2 SA	{Subtotal client/zi}	02-09-2007	127530
Client 3 SRL	Produs 2	07-09-2007	41584
Client 3 SRL	Produs 2	{Subtotal client/produs}	41584
Client 3 SRL	Produs 3	07-09-2007	33320
Client 3 SRL	Produs 3	{Subtotal client/produs}	33320
Client 3 SRL	Produs 4	07-09-2007	84530
Client 3 SRL	Produs 4	{Subtotal client/produs}	84530
Client 3 SRL	Produs 5	07-09-2007	5660235
Client 3 SRL	Produs 5	{Subtotal client/produs}	5660235
Client 3 SRL	{{Subtotal client}}		5819668
Client 3 SRL	{Subtotal client/zi}	07-09-2007	5819668
Client 5 SRL	Produs 2	01-09-2007	95430
Client 5 SRL	Produs 2	{Subtotal client/produs}	95430
Client 5 SRL	Produs 3	01-09-2007	58013
Client 5 SRL	Produs 3	{Subtotal client/produs}	58013
Client 5 SRL	{{Subtotal client}}		153442
Client 5 SRL	{Subtotal client/zi}	01-09-2007	153442
Client 7 SRL	Produs 2	10-09-2007	136850
Client 7 SRL	Produs 2	{Subtotal client/produs}	136850
Client 7 SRL	{{Subtotal client}}		136850
Client 7 SRL	{Subtotal client/zi}	10-09-2007	136850
{{Subtotal pentru produs}}	Produs 1	{{}}	324989
{{Subtotal pentru produs}}	Produs 2	{{}}	812977
{{Subtotal pentru produs}}	Produs 3	{{}}	91333
{{Subtotal pentru produs}}	Produs 4	{{}}	84530
{{Subtotal pentru produs}}	Produs 5	{{}}	9944949
{{Subtotal pentru ziua}}	{{}}	01-09-2007	4596402
{{Subtotal pentru ziua}}	{{}}	02-09-2007	238438
{{Subtotal pentru ziua}}	{{}}	07-09-2007	5819668
{{Subtotal pentru ziua}}	{{}}	10-09-2007	424705
{{Subtotal pentru ziua}}	{{}}	17-09-2007	179565
{Subtotal produs/zi}	Produs 1	01-09-2007	67830
{Subtotal produs/zi}	Produs 1	10-09-2007	124355
{Subtotal produs/zi}	Produs 1	17-09-2007	132804
{Subtotal produs/zi}	Produs 2	01-09-2007	185845
{Subtotal produs/zi}	Produs 2	02-09-2007	238438
{Subtotal produs/zi}	Produs 2	07-09-2007	41584

{Subtotal produs/zi}	Produs 2	10-09-2007	300350
{Subtotal produs/zi}	Produs 2	17-09-2007	46761
{Subtotal produs/zi}	Produs 3	01-09-2007	58013
{Subtotal produs/zi}	Produs 3	07-09-2007	33320
{Subtotal produs/zi}	Produs 4	07-09-2007	84530
{Subtotal produs/zi}	Produs 5	01-09-2007	4284714
{Subtotal produs/zi}	Produs 5	07-09-2007	5660235
{{{Total general}}}	{{{}}}	{{{}}}	11258777

DB2 reclamă funcțiile YEAR și MONTH în locul EXTRACT și o funcție CAST necesară conversiei datei facturii în șir de caractere. Pentru a porta soluția în SQL Server, în afara funcției DATEPART și sintaxei GROUP BY... WITH CUBE, este necesară, ca în DB2, funcția CAST:

SELECT

CASE

**WHEN GROUPING(DenCl)=1 AND GROUPING(DenPr)=1
AND GROUPING(DataFact)=1 THEN '{{{Total general}}}'**
**WHEN GROUPING(DenCl)=1 AND GROUPING(DenPr)=1 AND
GROUPING(DataFact)=0 THEN '{{Subtotal pentru ziua}}'**
**WHEN GROUPING(DenCl)=1 AND GROUPING(DenPr)=0 AND
GROUPING(DataFact)=1 THEN '{{Subtotal pentru produs}}'**
**WHEN GROUPING(DenCl)=1 AND GROUPING(DenPr)=0 AND
GROUPING(DataFact)=0 THEN '{Subtotal produs/zi}'**
ELSE DenCl

END AS Client,

CASE

**WHEN GROUPING(DenCl)=1 AND GROUPING(DenPr)=1 AND
GROUPING(DataFact)=1 THEN '{{{}}}'**
**WHEN GROUPING(DenCl)=1 AND GROUPING(DenPr)=1 AND
GROUPING(DataFact)=0 THEN '{{{}}}'**
**WHEN GROUPING(DenCl)=0 AND GROUPING(DenPr)=1 AND
GROUPING(DataFact)=0 THEN '{Subtotal client/zi}'**
**WHEN GROUPING(DenCl)=0 AND GROUPING(DenPr)=1 AND
GROUPING(DataFact)=1 THEN '{{Subtotal client}}'**
ELSE '' + DenPr

END AS Produs,

CASE

**WHEN GROUPING(DenCl)=1 AND GROUPING(DenPr)=1 AND
GROUPING(DataFact)=1 THEN '{{{}}}'**
**WHEN GROUPING(DenCl)=1 AND GROUPING(DenPr)=0 AND
GROUPING(DataFact)=1 THEN '{{{}}}'**

```

        WHEN GROUPING(DenCl)=0 AND GROUPING(DenPr)=0 AND
            GROUPING(DataFact)=1 THEN '{Subtotal client/produs}'
        ELSE '' + CAST (DataFact AS VARCHAR(12)) + ''
    END AS Zi,
    ROUND(SUM(Cantitate * PretUnit * (1+ProcTVA)),0) AS Vinzari
FROM clienti c
    INNER JOIN facturi f ON c.CodCl=f.CodCl
    INNER JOIN liniifact lf ON lf.NrFact=f.NrFact
    INNER JOIN produse p ON lf.CodPr=p.CodPr
WHERE DATEPART(YEAR,DataFact)=2007 AND DATEPART(MONTH,DataFact)=9
GROUP BY DenCl, DenPr, DataFact WITH CUBE

```

11.2.2. Operatorul GROUPING SETS

Uneori, prea multă detaliere, precum cea din tabelul de mai sus, devine obositoare. Prin GROUPING SETS se poate regla granularitatea agregărilor. Dacă, spre exemplu, în interogarea anterioară se dorește urmărirea zilnică a vânzărilor pentru fiecare client (și toate produsele) și pentru fiecare produs (și toți clienții), se poate recurge la varianta următoare:

CLIENT	PRODUS	ZI	VINZARI
Client 1 SRL	Produs 1	{Subtotal client/produs}	324989
Client 1 SRL	Produs 2	{Subtotal client/produs}	411584
Client 1 SRL	Produs 5	{Subtotal client/produs}	4284714
Client 2 SA	Produs 2	{Subtotal client/produs}	127530
Client 3 SRL	Produs 2	{Subtotal client/produs}	41584
Client 3 SRL	Produs 3	{Subtotal client/produs}	33320
Client 3 SRL	Produs 4	{Subtotal client/produs}	84530
Client 3 SRL	Produs 5	{Subtotal client/produs}	5660235
Client 5 SRL	Produs 2	{Subtotal client/produs}	95430
Client 5 SRL	Produs 3	{Subtotal client/produs}	58013
Client 7 SRL	Produs 2	{Subtotal client/produs}	136850
{{Subtotal pentru ziua}}	{{}}	01-09-2007	4596402
{{Subtotal pentru ziua}}	{{}}	02-09-2007	238438
{{Subtotal pentru ziua}}	{{}}	07-09-2007	5819668
{{Subtotal pentru ziua}}	{{}}	10-09-2007	424705
{{Subtotal pentru ziua}}	{{}}	17-09-2007	179565

Figura 11.16. Clauza GROUPING SETS

```

SELECT ...
FROM ... identic interogării anterioare
WHERE ...

```


GROUP BY GROUPING SETS ((DenCl, DenPr), DataFact)**ORDER BY 1,2,3**

În clauza GROUP BY a fost introdus operatorul GROUPING SETS prin care se precizează două grupuri de agregare, unul alcătuit din combinația (DenCl, DenPr), celălalt din data facturii, astfel încât se obține un rezultat cum este cel din figura 11.16.

Cu modificările amintite în exemplele precedente, interogarea funcționează și în DB2. Dacă se dorește o situație sintetică cu subtotaluri ale celor trei atribute, plus un total general (obținut prin perechea de paranteze fără argumente) se schimbă clauza GROUP BY astfel:

```
SELECT      ...
FROM        ...identic interogării anterioare
WHERE       ...
GROUP BY GROUPING SETS (DenCl, DenPr, DataFact, ( ) )
ORDER BY 1,2,3
```

CLIENT	PRODUS	ZI	VINZARI
Client 1 SRL	{{Subtotal client}}		5021287
Client 2 SA	{{Subtotal client}}		127530
Client 3 SRL	{{Subtotal client}}		5819668
Client 5 SRL	{{Subtotal client}}		153442
Client 7 SRL	{{Subtotal client}}		136850
{{Subtotal pentru produs}}	Produs 1	{{}}	324989
{{Subtotal pentru produs}}	Produs 2	{{}}	812977
{{Subtotal pentru produs}}	Produs 3	{{}}	91333
{{Subtotal pentru produs}}	Produs 4	{{}}	84530
{{Subtotal pentru produs}}	Produs 5	{{}}	9944949
{{Subtotal pentru ziua}}	{{}}	01-09-2007	4596402
{{Subtotal pentru ziua}}	{{}}	02-09-2007	238438
{{Subtotal pentru ziua}}	{{}}	07-09-2007	5819668
{{Subtotal pentru ziua}}	{{}}	10-09-2007	424705
{{Subtotal pentru ziua}}	{{}}	17-09-2007	179565
{{Total general}}	{{}}	{{}}	11258777

Figura 11.17. GROUPING SETS – exemplu 2

Cele 16 linii din figura 11.17 sunt obținute astfel: 5 clienți pentru care sunt vânzări în sept. 2007, plus 5 produse vândute, plus 5 zile în care s-au întocmit facturi, plus linia totalului general.

Interesant este și faptul că cei trei operatori pot fi combinați. Spre exemplu, dacă GROUP BY are forma:

```
...
GROUP BY
GROUPING SETS ( ROLLUP (DenCl, DenPr), DataFact)
```

...

prin execuția consultării se obțin liniile din figura 11.18.

SQL Server nu are, după bruma noastră de cunoștințe, opțiunea GROUPING SETS.

CLIENT	PRODUS	ZI	VINZARI
Client 1 SRL	Produs 1	{Subtotal client/produs}	324989
Client 1 SRL	Produs 2	{Subtotal client/produs}	411584
Client 1 SRL	Produs 5	{Subtotal client/produs}	4284714
Client 1 SRL	{{Subtotal client}}		5021287
Client 2 SA	Produs 2	{Subtotal client/produs}	127530
Client 2 SA	{{Subtotal client}}		127530
Client 3 SRL	Produs 2	{Subtotal client/produs}	41584
Client 3 SRL	Produs 3	{Subtotal client/produs}	33320
Client 3 SRL	Produs 4	{Subtotal client/produs}	84530
Client 3 SRL	Produs 5	{Subtotal client/produs}	5660235
Client 3 SRL	{{Subtotal client}}		5819668
Client 5 SRL	Produs 2	{Subtotal client/produs}	95430
Client 5 SRL	Produs 3	{Subtotal client/produs}	58013
Client 5 SRL	{{Subtotal client}}		153442
Client 7 SRL	Produs 2	{Subtotal client/produs}	136850
Client 7 SRL	{{Subtotal client}}		136850
{{Subtotal pentru ziua}}	{{}}	01-09-2007	4596402
{{Subtotal pentru ziua}}	{{}}	02-09-2007	238438
{{Subtotal pentru ziua}}	{{}}	07-09-2007	5819668
{{Subtotal pentru ziua}}	{{}}	10-09-2007	424705
{{Subtotal pentru ziua}}	{{}}	17-09-2007	179565
{{{Total general}}}	{{{}}}	{{{}}}	11258777

Figura 11.18. GROUPING SETS și ROLLUP

11.2.3. CUBE-uri parțiale

Discuția se poartă în termeni similari ROLLUP-ului parțial. Subtotalurile și combinațiile posibile sunt limitate la atributele-dimensiuni incluse între paranteze. Spre exemplu, dacă GROUP BY are forma:

GROUP BY atribut1, CUBE (atribut2, atribut3)

vor fi calculate patru subtotaluri, pentru combinațiile:

- (atribut1, atribut2, atribut3) – GROUP BY-ul obișnuit,
- (atribut1, atribut2),
- (atribut1, atribut3),
- (atribut1).

Astfel, interogarea:

```

SELECT CASE
    WHEN GROUPING(DenCl)=1 AND GROUPING(DenPr)=1
        AND GROUPING(DataFact)=1 THEN '{{{Total general}}}'
    WHEN GROUPING(DenCl)=1 AND GROUPING(DenPr)=1
        AND GROUPING(DataFact)=0 THEN '{{Subtotal pentru ziua}}'
    WHEN GROUPING(DenCl)=1 AND GROUPING(DenPr)=0 AND
        GROUPING(DataFact)=1 THEN '{{Subtotal pentru produs}}'
    WHEN GROUPING(DenCl)=1 AND GROUPING(DenPr)=0 AND
        GROUPING(DataFact)=0 THEN '{{Subtotal produs/zi}}'
    ELSE DenCl
END AS Client,
CASE
    WHEN GROUPING(DenCl)=1 AND GROUPING(DenPr)=1 AND
        GROUPING(DataFact)=1 THEN '{{{}}}'
    WHEN GROUPING(DenCl)=1 AND GROUPING(DenPr)=1 AND
        GROUPING(DataFact)=0 THEN '{{{}}}'
    WHEN GROUPING(DenCl)=0 AND GROUPING(DenPr)=1 AND
        GROUPING(DataFact)=0 THEN '{{Subtotal client/zi}}'
    WHEN GROUPING(DenCl)=0 AND GROUPING(DenPr)=1 AND
        GROUPING(DataFact)=1 THEN '{{Subtotal client}}'
    ELSE '' || DenPr
END AS Produs,
CASE
    WHEN GROUPING(DenCl)=1 AND GROUPING(DenPr)=1 AND
        GROUPING(DataFact)=1 THEN '{{{}}}'
    WHEN GROUPING(DenCl)=1 AND GROUPING(DenPr)=0 AND
        GROUPING(DataFact)=1 THEN '{{{}}}'
    WHEN GROUPING(DenCl)=0 AND GROUPING(DenPr)=0 AND
        GROUPING(DataFact)=1 THEN '{{Subtotal client/produs}}'
    ELSE '' || CAST (DataFact AS CHAR(10)) || ''
END AS Zi,
ROUND(SUM(Cantitate * PretUnit * (1+ProcTVA)),0) AS Vinzari
FROM clienti c
    INNER JOIN facturi f ON c.CodCl=f.CodCl
    INNER JOIN liniifact lf ON lf.NrFact=f.NrFact
    INNER JOIN produse p ON lf.CodPr=p.CodPr
WHERE EXTRACT (YEAR FROM DataFact)=2007 AND
        EXTRACT (MONTH FROM DataFact)=9
GROUP BY DenCl, CUBE (DenPr, DataFact )
ORDER BY 1,2,3

```

va genera liniile tabelului 11.2.

Tabel 11.2. CUBE parțial

CLIENT	PRODUS	ZI	VINZARI
Client 1 SRL	Produs 1	01-09-2007	67830
Client 1 SRL	Produs 1	10-09-2007	124355
Client 1 SRL	Produs 1	17-09-2007	132804
Client 1 SRL	Produs 1	{Subtotal client/produs}	324989
Client 1 SRL	Produs 2	01-09-2007	90416
Client 1 SRL	Produs 2	02-09-2007	110908
Client 1 SRL	Produs 2	10-09-2007	163500
Client 1 SRL	Produs 2	17-09-2007	46761
Client 1 SRL	Produs 2	{Subtotal client/produs}	411584
Client 1 SRL	Produs 5	01-09-2007	4284714
Client 1 SRL	Produs 5	{Subtotal client/produs}	4284714
Client 1 SRL	{{Subtotal client}}		5021287
Client 1 SRL	{Subtotal client/zi}	01-09-2007	4442960
Client 1 SRL	{Subtotal client/zi}	02-09-2007	110908
Client 1 SRL	{Subtotal client/zi}	10-09-2007	287855
Client 1 SRL	{Subtotal client/zi}	17-09-2007	179565
Client 2 SA	Produs 2	02-09-2007	127530
Client 2 SA	Produs 2	{Subtotal client/produs}	127530
Client 2 SA	{{Subtotal client}}		127530
Client 2 SA	{Subtotal client/zi}	02-09-2007	127530
Client 3 SRL	Produs 2	07-09-2007	41584
Client 3 SRL	Produs 2	{Subtotal client/produs}	41584
Client 3 SRL	Produs 3	07-09-2007	33320
Client 3 SRL	Produs 3	{Subtotal client/produs}	33320
Client 3 SRL	Produs 4	07-09-2007	84530
Client 3 SRL	Produs 4	{Subtotal client/produs}	84530
Client 3 SRL	Produs 5	07-09-2007	5660235
Client 3 SRL	Produs 5	{Subtotal client/produs}	5660235
Client 3 SRL	{{Subtotal client}}		5819668
Client 3 SRL	{Subtotal client/zi}	07-09-2007	5819668
Client 5 SRL	Produs 2	01-09-2007	95430
Client 5 SRL	Produs 2	{Subtotal client/produs}	95430
Client 5 SRL	Produs 3	01-09-2007	58013
Client 5 SRL	Produs 3	{Subtotal client/produs}	58013
Client 5 SRL	{{Subtotal client}}		153442
Client 5 SRL	{Subtotal client/zi}	01-09-2007	153442
Client 7 SRL	Produs 2	10-09-2007	136850
Client 7 SRL	Produs 2	{Subtotal client/produs}	136850

Client 7 SRL	{{Subtotal client}}		136850
Client 7 SRL	{Subtotal client/zi}	10-09-2007	136850

În schimb, dacă GROUP BY are forma:

...

GROUP BY CUBE((DenCl, DenPr), DataFact)

vor fi calculate patru subtotaluri (tabel 11.3), pentru combinațiile:

- (DenCl, DenPr, DataFact) – GROUP BY-ul obișnuit
- (DenCl, DataFact)
- (DenPr, DataFact)
- (DataFact).

Tabel 11.3. Alt CUBE parțial

CLIENT	PRODUS	ZI	VINZARI
Client 1 SRL	Produs 1	01-09-2007	67830
Client 1 SRL	Produs 1	10-09-2007	124355
Client 1 SRL	Produs 1	17-09-2007	132804
Client 1 SRL	Produs 1	{Subtotal client/produs}	324989
Client 1 SRL	Produs 2	01-09-2007	90416
Client 1 SRL	Produs 2	02-09-2007	110908
Client 1 SRL	Produs 2	10-09-2007	163500
Client 1 SRL	Produs 2	17-09-2007	46761
Client 1 SRL	Produs 2	{Subtotal client/produs}	411584
Client 1 SRL	Produs 5	01-09-2007	4284714
Client 1 SRL	Produs 5	{Subtotal client/produs}	4284714
Client 2 SA	Produs 2	02-09-2007	127530
Client 2 SA	Produs 2	{Subtotal client/produs}	127530
Client 3 SRL	Produs 2	07-09-2007	41584
Client 3 SRL	Produs 2	{Subtotal client/produs}	41584
Client 3 SRL	Produs 3	07-09-2007	33320
Client 3 SRL	Produs 3	{Subtotal client/produs}	33320
Client 3 SRL	Produs 4	07-09-2007	84530
Client 3 SRL	Produs 4	{Subtotal client/produs}	84530
Client 3 SRL	Produs 5	07-09-2007	5660235
Client 3 SRL	Produs 5	{Subtotal client/produs}	5660235
Client 5 SRL	Produs 2	01-09-2007	95430
Client 5 SRL	Produs 2	{Subtotal client/produs}	95430
Client 5 SRL	Produs 3	01-09-2007	58013
Client 5 SRL	Produs 3	{Subtotal client/produs}	58013
Client 7 SRL	Produs 2	10-09-2007	136850
Client 7 SRL	Produs 2	{Subtotal client/produs}	136850
{{Subtotal pentru ziua}}	{{}}	01-09-2007	4596402

{{Subtotal pentru ziua}}	{{}}	02-09-2007	238438
{{Subtotal pentru ziua}}	{{}}	07-09-2007	5819668
{{Subtotal pentru ziua}}	{{}}	10-09-2007	424705
{{Subtotal pentru ziua}}	{{}}	17-09-2007	179565
{{{Total general}}}	{{{}}}	{{{}}}	11258777

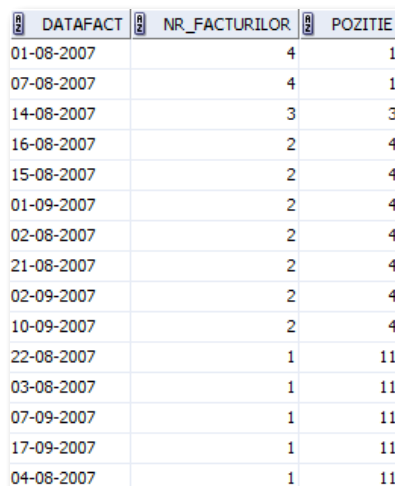
Nici în materie de cuburi parțiale MS SQL Server nu are vreo clauză/opțiune.

11.3. Clasamente – soluții clasice și OLAP

Problema TOP-urilor a fost atacată (cu oarecare succes) încă din paragraful 9.2, soluțiile fiind subconsultările pe mai multe niveluri, sau cele proprietate – LIMIT, TOP și FETCH. De asemenea, în paragraful 9.4.2 au fost prezentate câteva soluții proprietate Oracle bazate pe ROWNUM și subconsultări în clauza FROM.

Să se afișeze clasamentul zilelor în ordinea descrescătoare a numărului de facturi emise.

Problema care ne privește acum este indicarea poziției fiecărei zile – vezi figura 11.19. După cum am văzut în paragraful 9.4.2, în Oracle este posibilă ordonarea liniilor unei subconsultări din clauza FROM, așa că soluția următoare, bazată pe pseudo-coloana ROWNUM, funcționează:



DATAFACT	NR_FACTURILOR	POZITIE
01-08-2007	4	1
07-08-2007	4	1
14-08-2007	3	3
16-08-2007	2	4
15-08-2007	2	4
01-09-2007	2	4
02-08-2007	2	4
21-08-2007	2	4
02-09-2007	2	4
10-09-2007	2	4
22-08-2007	1	11
03-08-2007	1	11
07-09-2007	1	11
17-09-2007	1	11
04-08-2007	1	11

Figura 11.19. Un clasament veritabil

```
SELECT DataFact, Nr_Facturilor, ROWNUM AS Pozitie
FROM
  (SELECT DataFact, COUNT(*) AS Nr_Facturilor
   FROM facturi
   GROUP BY DataFact
```

ORDER BY Nr_Facturilor DESC)

Chiar dacă ne putem declara satisfăcuți de rezultat (figura 11.20), se poate argumenta că 7 august 2007 este pe nedrept “retrogradată” pe locul 2 al clasamentului, atât timp cât are același număr de zile ca și 1 august și nu există nici un criteriu suplimentar de departajare. Interogarea de mai sus este corectă sută la sută doar atunci când, fie toate valorile ordonate sunt diferite, fie când există un număr suficient de criterii de balotaj care să confere unicitate fiecărei poziții a clasamentului.

DATAFACT	NR_FACTURILOR	POZITIE
01-08-2007	4	1
07-08-2007	4	2
14-08-2007	3	3
16-08-2007	2	4
15-08-2007	2	5
01-09-2007	2	6
02-08-2007	2	7
21-08-2007	2	8
02-09-2007	2	9
10-09-2007	2	10
22-08-2007	1	11
03-08-2007	1	12
07-09-2007	1	13
17-09-2007	1	14
04-08-2007	1	15

Figura 11.20. Pseudo-clasamentul obținut printr-o subconsultare

Există remedii „proprietary” bazate pe expresii-tabelă și corelări în clauza SELECT. Interogarea Oracle următoare obține un rezultat precum cel din figura 11.19:

```

WITH top1 AS
  (SELECT DataFact, Nr_Facturilor, ROWNUM AS Pozitie
   FROM
     (SELECT DataFact, COUNT(*) AS Nr_Facturilor
      FROM facturi
      GROUP BY DataFact
      ORDER BY Nr_Facturilor DESC)
  )
SELECT DataFact, Nr_Facturilor,
  CASE WHEN Nr_Facturilor = (SELECT Nr_facturilor
                             FROM top1
                             WHERE pozitie = t2.pozitie-1
                          )
  THEN (SELECT MIN(Pozitie) FROM top1
        WHERE Nr_facturilor=t2.Nr_Facturilor)

```

```

ELSE Pozitie
END AS Pozitie
FROM top1 t2

```

Pentru astfel de situații, *Amendamentul OLAP din SQL:1999* a introdus două funcții ideale, RANK și DENSE_RANK. În prealabil, însă, câteva chestiuni ce țin de logica execuției unui asemenea gen de consultări. Procesarea interogărilor ce utilizează funcții analitice se derulează în trei etape:

1. Se operează toate joncțiunile, se constituie grupurile și se efectuează selecția asupra grupurilor, altfel spus, se execută clauzele JOIN (sau FROM/WHERE), WHERE, GROUP BY și HAVING.
2. Se „aranjează” rezultatul în vederea aplicării funcțiilor analitice și se efectuează calculele (sunt create partițiile), apoi
3. Funcțiile analitice sunt aplicate linie cu linie în cadrul fiecărei partiții.

Pasul 3 este operațional numai dacă interogarea prezintă la sfârșit o clauză ORDER BY, ceea ce atrage ordonarea finală a rezultatului în conformitate cu criteriile specificate. Partițiile sunt seturi de linii create după delimitarea grupurilor prin GROUP BY, astfel încât pot constitui subiectul (sau obiectul) oricărei funcții de agregare (SUM, AVG....). Constituirea unei partiții se poate face în funcție de valorile unuia sau mai multor atribute sau expresii de atribute.

Pentru a obține o situație identică celei din figura 11.19, soluția comună DB2/Oracle/SQL Server este:

```

SELECT DataFact, COUNT(*) AS Nr_Facturilor,
       RANK() OVER (ORDER BY COUNT(*) DESC) AS Pozitie
FROM facturi
GROUP BY DataFact

```

Funcția RANK ierarhizează după valorile obținute de COUNT(*) liniile rezultatului și, ceea ce este cel mai important, la valori egale, atribuie fiecărei linii aceeași poziție în clasament.

Luăm un alt exemplu care să pună în valoare și partițiile: *Să se afișeze, pentru fiecare zi de vânzări din luna august 2007, topul celor mai "valoroase" facturi.*

Să analizăm problema:

- partiția are dimensiunea unei zile; numărul liniilor depinde de numărul facturilor întocmite în ziua respectivă; prin urmare, atributul de partiționare este DataFact;
- în cadrul unei partiții, clasamentul se face în funcție de valorile calculate de funcția SUM;
- în rezultatul final, pentru fiecare factură, liniile sunt dispuse în funcție de valorile DataFact și NrFact, deci, la prezentare, se păstrează ordinea cronologică

Iată și soluția DB2/Oracle/MS SQL Server (în DB2 și SQL Server funcția EXTRACT trebuie înlocuită cu YEAR/MONTH sau DATEPART), rezultatul fiind cel din figura 11.21:


```

SELECT DataFact, F.NrFact,
       ROUND(SUM(Cantitate * PretUnit * (1+ProcTVA)),0) AS ValFact,
RANK() OVER
  (PARTITION BY DataFact
    ORDER BY SUM(Cantitate * PretUnit * (1+ProcTVA)) DESC)
AS Pozitie
FROM produse p
  INNER JOIN liniifact lf ON p.CodPr=lf.CodPr
  INNER JOIN facturi f ON lf.NrFact=f.NrFact
WHERE EXTRACT (YEAR FROM DataFact)=2007 AND
      EXTRACT (MONTH FROM DataFact)=8
GROUP BY DataFact, F.NrFact
ORDER BY 1, 4

```

DATAFACT	NRFACT	VALFACT	POZITIE
01-08-2007	1114	6021707	1
01-08-2007	1111	4346038	2
01-08-2007	1112	125516	3
01-08-2007	1113	106275	4
02-08-2007	1115	151238	1
02-08-2007	1116	126713	2
03-08-2007	1117	222050	1
04-08-2007	1118	201975	1
07-08-2007	1119	5774499	1
07-08-2007	1121	4737838	2
07-08-2007	1120	97664	3
14-08-2007	2111	4442960	1
14-08-2007	2112	153442	2
14-08-2007	2113	127530	3
15-08-2007	2116	136850	1
15-08-2007	2115	110908	2
16-08-2007	2117	287855	1
16-08-2007	2118	179565	2
21-08-2007	2119	5819668	1
21-08-2007	2121	4741272	2

Figura 11.21. Topurile zilnice ale facturilor

Factura 1111 este a doua ca valoare pe 1 august, 2007, cea mai mare valoare în această zi având factura 1114.

Într-o singură frază SELECT se pot întocmi, pentru aceleași date, clasamente diferite prin specificarea mai multor partiții și criterii. Astfel, interogarea (Oracle/DB2/SQL Server)²:

```
SELECT DataFact, DenPr,
       ROUND(SUM(Cantitate * PretUnit * (1+ProcTVA)),0) AS ValFact,
       RANK() OVER (PARTITION BY DataFact
                    ORDER BY SUM(Cantitate * PretUnit * (1+ProcTVA)) DESC)
       AS Pozitie_Zi,
       RANK() OVER (PARTITION BY DenPr
                    ORDER BY SUM(Cantitate * PretUnit * (1+ProcTVA)) DESC)
       AS Pozitie_Prod
FROM produse p
     INNER JOIN liniifact lf ON p.CodPr=lf.CodPr
     INNER JOIN facturi f ON lf.NrFact=f.NrFact
WHERE EXTRACT (YEAR FROM DataFact)=2007 AND
      EXTRACT (MONTH FROM DataFact)=9
GROUP BY DataFact, DenPr
ORDER BY DataFact, DenPr
```

calculează valoarea vânzărilor pe zilele din luna septembrie 2007 (DataFact) pentru fiecare produs. Cele două funcții RANK determină poziția vânzărilor pentru un produs și o zi în totalul vânzărilor pe ziua respectivă (prima) și poziția în clasamentul tuturor zilelor în care s-a vândut produsul respectiv- vezi figura 11.22.

Pe prima linie a rezultatului apar ziua de 1 septembrie 2007 și produsul 1, pentru care valoarea vânzărilor în această zi este de 67830 RON. 67830 este, ca mărime, a treia valoare pentru ziua respectivă - *Pozitie_Zi* - (pe primul loc în 1 sept. este produsul 5, iar al doilea este produs 2). Între toate zilele în care s-a vândut produsul 1 (1, 10 și 17 septembrie) - *Pozitie_Prod*-, valoarea 67830 este tot a treia (pe primul loc sunt vânzările din 17 sept., iar pe locul 2 cele din 10 sept.). Altfel spus, pentru 1 septembrie 2007, produsul 1 este al treilea în topul pe produsul al zile, (cel mai bine vândut în această zi fiind produsul 5), iar, dintre toate zilele în care a fost vândut Produs 1, data de 1 septembrie este a treia în topul vânzărilor pe zile (cel mai bine s-a vândut pe 17 septembrie).

² Sub aceeași rezervă a înlocuirii EXTRACT cu YEAR/MONTH sau DATEPART.

DATAFACT	DENPR	VALFACT	POZITIE_ZI	POZITIE_PROD
01-09-2007	Produs 1	67830	3	3
01-09-2007	Produs 2	185845	2	3
01-09-2007	Produs 3	58013	4	1
01-09-2007	Produs 5	4284714	1	2
02-09-2007	Produs 2	238438	1	2
07-09-2007	Produs 2	41584	3	5
07-09-2007	Produs 3	33320	4	2
07-09-2007	Produs 4	84530	2	1
07-09-2007	Produs 5	5660235	1	1
10-09-2007	Produs 1	124355	2	2
10-09-2007	Produs 2	300350	1	1
17-09-2007	Produs 1	132804	1	1
17-09-2007	Produs 2	46761	2	4

Figura 11.22. Două partiții de ordonare

Se poate alcătui un clasament nu numai pe baza unei simple funcții agregat, dar și prin combinarea funcției RANK cu funcțiile ROLLUP și CUBE. Rezultate nu sunt însă întotdeauna limpezi. Spre exemplu, consultarea DB2/Oracle următoare produce raportul din figura 11.23³.

```

SELECT Judet, DenPr AS "Produs",
       ROUND(SUM(Cantitate * PretUnit * (1+ProcTVA)),0) AS Vinzari,
       RANK() OVER (PARTITION BY Judet
                    ORDER BY SUM(Cantitate * PretUnit * (1+ProcTVA)) DESC)
       AS "Poz_in_Judet",
       RANK() OVER (PARTITION BY DenPr
                    ORDER BY SUM(Cantitate * PretUnit * (1+ProcTVA)) DESC)
       AS "Poz_in_Produs"
FROM judete j
     INNER JOIN coduri_postale cp ON j.Jud=cp.Jud
     INNER JOIN clienti c ON cp.CodPost=c.CodPost
     INNER JOIN facturi f ON c.CodCl=f.CodCl
     INNER JOIN liniifact lf ON lf.NrFact=f.NrFact
     INNER JOIN produse p ON lf.CodPr=p.CodPr

```

³ În MS SQL Server interogarea funcționează cu o ușoară modificare a clauzei GROUP BY (GROUP BY Judet, DenPr WITH ROLLUP), iar liniile de total general și subtotal vor fi plasate la începutul raportului/grupurilor.

GROUP BY ROLLUP (Judet, DenPr)

ORDER BY Judet, DenPr

JUDET	Produs	VINZARI	Poz_in_Judet	Poz_in_Produs
Iasi	Produs 1	988533	4	1
Iasi	Produs 2	1896546	3	1
Iasi	Produs 5	22016904	2	1
Iasi	(null)	24901983	1	2
Neamt	Produs 2	81641	3	4
Neamt	Produs 4	55754	4	2
Neamt	Produs 5	5884312	2	3
Neamt	(null)	6021707	1	4
Timis	Produs 2	681087	2	2
Timis	Produs 3	151725	3	1
Timis	(null)	832812	1	5
Vaslui	Produs 2	124751	4	3
Vaslui	Produs 3	99960	5	2
Vaslui	Produs 4	245904	3	1
Vaslui	Produs 5	16943220	2	2
Vaslui	(null)	17413835	1	3
(null)	(null)	49170335	1	1

Figura 11.23. RANK și ROLLUP

Este clar că *Poz_in_Judet* va fi întotdeauna 1 în liniile dedicate subtotalului pentru fiecare dintre județe. Spre exemplu, Produs 1 nu este al patrulea ca vânzări în județul Iași, ci al treilea. Valorile *Poz_in_Produs* pentru județe sunt însă corecte (nu necesită ajustări). Județul în care Produs 4 s-a vândut cel mai bine este Vaslui (245904 RON), iar al doilea clasat este județul Neamț (55754 RON).

Dacă examinăm liniile subtotalurilor pe județe și linia totalului general, aflăm, spre exemplu, din penultima linie, că județul Vaslui este al treilea ca volum total al vânzărilor. Poziția exactă a Vasluiului în topul județelor este însă a doua, deoarece prima îi revine automat totalului general (ultima linie).

Lucrurile stau aidoma și în cazul operatorului CUBE. Deși raportul (figura 11.24) legat de vânzările pe clienți și produse din luna septembrie 2007, furnizat de următoarea frază SELECT valabilă deopotrivă în Oracle și DB2 (cu modificările amintite), este bogat în informații, câteva dintre acestea trebuie nuanțate.

SELECT

```

CASE WHEN GROUPING (DenCl) = 1
      THEN '{SUBTOTAL produs}' ELSE DenCl
END AS Client,
CASE WHEN GROUPING (DenPr) = 1
      THEN '{SUBTOTAL client}' ELSE DenPr
END AS Produs,
SUM(Cantitate * PretUnit * (1+ProcTVA)) AS Vinzari,
RANK() OVER (PARTITION BY DenCl

```

```

ORDER BY SUM(Cantitate * PretUnit * (1+ProcTVA)) DESC)
AS Poz_Client,
RANK() OVER (PARTITION BY DenPr
ORDER BY SUM(Cantitate * PretUnit * (1+ProcTVA)) DESC)
AS Poz_Produs
FROM judete j
INNER JOIN coduri_postale cp ON j.Jud=cp.Jud
INNER JOIN clienti c ON cp.CodPost=c.CodPost
INNER JOIN facturi f ON c.CodCl=f.CodCl
INNER JOIN liniifact lf ON lf.NrFact=f.NrFact
INNER JOIN produse p ON lf.CodPr=p.CodPr
WHERE EXTRACT (YEAR FROM DataFact)=2007 AND
EXTRACT (MONTH FROM DataFact)=9
GROUP BY CUBE (DenCl, DenPr )
ORDER BY DenCl, DenPr

```

RANK	CLIENT	RANK	PRODUS	RANK	VINZARI	RANK	POZ_CLIENT	RANK	POZ_PRODUS
	Client 1 SRL		Produs 1		324989		4		1
	Client 1 SRL		Produs 2		411584		3		2
	Client 1 SRL		Produs 5		4284714		2		3
	Client 1 SRL		{SUBTOTAL client}		5021287		1		3
	Client 2 SA		Produs 2		127530		1		4
	Client 2 SA		{SUBTOTAL client}		127530		1		6
	Client 3 SRL		Produs 2		41583,5		4		6
	Client 3 SRL		Produs 3		33320		5		3
	Client 3 SRL		Produs 4		84529,5		3		1
	Client 3 SRL		Produs 5		5660235		2		2
	Client 3 SRL		{SUBTOTAL client}		5819668		1		2
	Client 5 SRL		Produs 2		95429,5		2		5
	Client 5 SRL		Produs 3		58012,5		3		2
	Client 5 SRL		{SUBTOTAL client}		153442		1		4
	Client 7 SRL		Produs 2		136849,5		1		3
	Client 7 SRL		{SUBTOTAL client}		136849,5		1		5
	{SUBTOTAL produs}		Produs 1		324989		4		1
	{SUBTOTAL produs}		Produs 2		812976,5		3		1
	{SUBTOTAL produs}		Produs 3		91332,5		5		1
	{SUBTOTAL produs}		Produs 4		84529,5		6		1
	{SUBTOTAL produs}		Produs 5		9944949		2		1
	{SUBTOTAL produs}		{SUBTOTAL client}		11258776,5		1		1

Figura 11.24. RANK și CUBE

Astfel, ca vânzări, pentru Client 1 SRL, Produs 5 este pe locul 1 (4284714 RONi), și nu pe locul 2, cum indică a doua linie din figură. La modul general, poziția produselor în vânzările corespunzătoare unui client (*Poz_Client*) este "umflată" cu 1, datorită subtotalurilor de la nivel de client. Excepție fac clienții cărora li s-a

vândut un singur produs (Client 2 SA și Client 7 SRL) la care ambele valori ale *Poz_Client* sunt 1.

Analog, prima poziție a listei pentru *Poz_Produs* este cea a subtotalului pe produs, iar, spre exemplu, cel mai bun cumpărător al Produs 2 este Client 1 SRL. Faptul că, la Produs 1, în dreptul Clientului 1 SRL este indicată poziția corectă, acest lucru se datorează vânzării produsului 1 unui singur client.

Ținând seama de interferența subtotalurilor, am fi tentați să decrementăm rezultatele funcțiilor RANK cu o unitate:

```
SELECT CASE WHEN GROUPING (DenCl) = 1 THEN '{SUBTOTAL produs}'
          ELSE DenCl
        END AS Client,
CASE WHEN GROUPING (DenPr) = 1 THEN '{SUBTOTAL client}'
      ELSE DenPr
    END AS Produs,
ROUND(SUM(Cantitate * PretUnit * (1+ProcTVA)),0) AS Vinzari,
RANK() OVER (PARTITION BY DenCl
              ORDER BY SUM(Cantitate * PretUnit * (1+ProcTVA)) DESC) - 1
          AS Poz_Client,
RANK() OVER (PARTITION BY DenPr
              ORDER BY SUM(Cantitate * PretUnit * (1+ProcTVA)) DESC) - 1
          AS Poz_Produs
FROM judete j
      INNER JOIN coduri_postale cp ON j.Jud=cp.Jud
      INNER JOIN clienti c ON cp.CodPost=c.CodPost
      INNER JOIN facturi f ON c.CodCl=f.CodCl
      INNER JOIN liniifact lf ON lf.NrFact=f.NrFact
      INNER JOIN produse p ON lf.CodPr=p.CodPr
WHERE EXTRACT (YEAR FROM DataFact)=2007 AND
      EXTRACT (MONTH FROM DataFact)=9
GROUP BY CUBE (DenCl, DenPr )
ORDER BY DenCl, DenPr
```

Rezultatele ar fi perfecte, dacă n-ar exista produse vândute unui singur client (Produs 1) și clienți care au cumpărat un singur produs (Client 2 și Client 7) – vezi figura 11.25. Pentru aceste două spețe, *Poz_Client*, respectiv *Poz_Produs* sunt 0, chiar dacă respectivul produs/client ar fi singur și, poziția ar trebui să fie, inevitabil, 1. Am modificat, astfel, gluma aceea răsuflată: să alergi de unul singur și ajungi pe locul zero !

CLIENT	PRODUS	VINZARI	POZ_CLIENT	POZ_PRODUS
Client 1 SRL	Produs 1	324989	3	0
Client 1 SRL	Produs 2	411584	2	1
Client 1 SRL	Produs 5	4284714	1	2
Client 1 SRL	{SUBTOTAL client}	5021287	0	2
Client 2 SA	Produs 2	127530	0	3
Client 2 SA	{SUBTOTAL client}	127530	0	5
Client 3 SRL	Produs 2	41584	3	5
Client 3 SRL	Produs 3	33320	4	2
Client 3 SRL	Produs 4	84530	2	0
Client 3 SRL	Produs 5	5660235	1	1
Client 3 SRL	{SUBTOTAL client}	5819668	0	1
Client 5 SRL	Produs 2	95430	1	4
Client 5 SRL	Produs 3	58013	2	1
Client 5 SRL	{SUBTOTAL client}	153442	0	3
Client 7 SRL	Produs 2	136850	0	2
Client 7 SRL	{SUBTOTAL client}	136850	0	4
{SUBTOTAL produs}	Produs 1	324989	3	0
{SUBTOTAL produs}	Produs 2	812977	2	0
{SUBTOTAL produs}	Produs 3	91333	4	0
{SUBTOTAL produs}	Produs 4	84530	5	0
{SUBTOTAL produs}	Produs 5	9944949	1	0
{SUBTOTAL produs}	{SUBTOTAL client}	11258777	0	0

Figura 11.25. Încercare de corectarea a pozițiilor din clasament

O soluție mai elegantă se bazează pe modificarea modului de declarare a partițiilor. Pentru a pune în evidență cele două modalități, se afișează câte două poziții pentru clienți/produse, folosindu-se câte două funcții RANK: una identică cu exemplul anterior și o alta care beneficiază de serviciile GROUPING, după cum urmează:

SELECT

```

CASE WHEN GROUPING (DenCl) = 1
      THEN '{SUBTOTAL produs}' ELSE DenCl END AS Client,
CASE WHEN GROUPING (DenPr) = 1 THEN '{SUBTOTAL client}'
      ELSE DenPr END AS Produs,
ROUND(SUM(Cantitate * PretUnit * (1+ProcTVA)),0) AS Vinzari,
RANK() OVER (PARTITION BY DenCl
              ORDER BY SUM(Cantitate * PretUnit * (1+ProcTVA)) DESC)
AS Poz_Client_1,
RANK() OVER (PARTITION BY DenCl, GROUPING(DenPr)
              ORDER BY SUM(Cantitate * PretUnit * (1+ProcTVA)) DESC)
AS Poz_Client_2,
RANK() OVER (PARTITION BY DenPr
              ORDER BY SUM(Cantitate * PretUnit * (1+ProcTVA)) DESC)

```

```

AS Poz_Produs_1,
RANK() OVER (PARTITION BY DenPr, GROUPING (DenCl)
ORDER BY SUM(Cantitate * PretUnit * (1+ProcTVA)) DESC)
AS Poz_Produs_2
FROM judete j
INNER JOIN coduri_postale cp ON j.Jud=cp.Jud
INNER JOIN clienti c ON cp.CodPost=c.CodPost
INNER JOIN facturi f ON c.CodCl=f.CodCl
INNER JOIN liniifact lf ON lf.NrFact=f.NrFact
INNER JOIN produse p ON lf.CodPr=p.CodPr
WHERE EXTRACT (YEAR FROM DataFact)=2007 AND
EXTRACT (MONTH FROM DataFact)=9
GROUP BY CUBE (DenCl, DenPr )
ORDER BY GROUPING(DenCl), DenCl, GROUPING(DenPr), DenPr

```

Coloanele *Poz_Client_2* și *Poz_Produs_2* conțin, grație includerii clauzelor *GROUPING* în funcțiile *RANK*, valorile care ne interesează – vezi figura 11.26.

R	CLIENT	R	PRODUS	R	VINZARI	R	POZ_CLIENT_1	R	POZ_CLIENT_2	R	POZ_PRODUS_1	R	POZ_PRODUS_2
	Client 1 SRL		Produs 1		324989		4		3		1		1
	Client 1 SRL		Produs 2		411584		3		2		2		1
	Client 1 SRL		Produs 5		4284714		2		1		3		2
	Client 1 SRL		{SUBTOTAL client}		5021287		1		1		3		2
	Client 2 SA		Produs 2		127530		1		1		4		3
	Client 2 SA		{SUBTOTAL client}		127530		1		1		6		5
	Client 3 SRL		Produs 2		41584		4		3		6		5
	Client 3 SRL		Produs 3		33320		5		4		3		2
	Client 3 SRL		Produs 4		84530		3		2		1		1
	Client 3 SRL		Produs 5		5660235		2		1		2		1
	Client 3 SRL		{SUBTOTAL client}		5819668		1		1		2		1
	Client 5 SRL		Produs 2		95430		2		1		5		4
	Client 5 SRL		Produs 3		58013		3		2		2		1
	Client 5 SRL		{SUBTOTAL client}		153442		1		1		4		3
	Client 7 SRL		Produs 2		136850		1		1		3		2
	Client 7 SRL		{SUBTOTAL client}		136850		1		1		5		4
	{SUBTOTAL produs}		Produs 1		324989		4		3		1		1
	{SUBTOTAL produs}		Produs 2		812977		3		2		1		1
	{SUBTOTAL produs}		Produs 3		91333		5		4		1		1
	{SUBTOTAL produs}		Produs 4		84530		6		5		1		1
	{SUBTOTAL produs}		Produs 5		9944949		2		1		1		1
	{SUBTOTAL produs}		{SUBTOTAL client}		11258777		1		1		1		1

Figura 11.26. CUBE și RANK, cu și fără GROUPING

Care sunt cele mai mari cinci prețuri unitare din LINIIFACT ?

În paragraful 9.2 am prezentat o soluție ingenioasă bazată pe subconsultări în cascadă, iar în paragraful 9.4.2 variantă Oracle bazată pe ROWNUM. Din păcate, în alte servere de baze de date decât Oracle, pseudocoloana ROWNUM nu este operațională. În schimb, DB2, Oracle și SQL Server au implementat o nouă funcție OLAP din SQL:1999, funcție cu o logică similară - ROW_NUMBER -, astfel încât se poate redacta o soluție comună al cărei rezultat este cel din stânga figurii 11.27.

```
SELECT PretUnit
```

```
FROM (SELECT PretUnit, ROW_NUMBER () OVER (ORDER BY PretUnit DESC)
```

```
      AS Poz
```

```
FROM liniifact ) p1
```

```
WHERE Poz <=5
```

PRETUNIT
7064
7064
7064
7060
7060

PRETUNIT
7064
7060
6300
1705
1410

Figura 11.27. Două aplicări ale funcției ROW_NUMBER()

Funcția ROW_NUMBER întoarce numărul de ordine al fiecărei linii în partiția declarată, funcționând depotrivă în DB2, Oracle și SQL Server. Dacă ne-ar interesa cele mai mari cinci prețuri unitare *distincte*, atunci am schimba interogarea de mai sus:

```
SELECT PretUnit
```

```
FROM
```

```
(SELECT PretUnit, ROW_NUMBER () OVER (ORDER BY PretUnit DESC) AS
```

```
Poz
```

```
FROM
```

```
(SELECT DISTINCT PretUnit
```

```
FROM liniifact
```

```
) t1
```

```
) t2
```

```
WHERE Poz <=5
```

rezultatul fiind acum cel din dreapta figurii 11.27.

Care sunt cele mai mari cinci prețuri unitare de vânzare, produsele și facturile în care apar cele cinci prețuri maxime ?

Și această soluție este comună - DB2/Oracle/SQL Server:

```
SELECT DISTINCT NrFact, DenPr, lf.PretUnit
```

```
FROM liniifact lf
```

```
INNER JOIN produse p ON lf.CodPr=p.CodPr
```

```
INNER JOIN
```

```

        (SELECT Poz, PretUnit
        FROM
            (SELECT PretUnit, ROW_NUMBER () OVER
                (ORDER BY PretUnit DESC) AS Poz
            FROM liniifact
            ) p1
        WHERE Poz <=5 ) PMAX
        ON lf.PretUnit >= PMAX.PretUnit
    ORDER BY lf.PretUnit DESC

```

Să se afișeze ziua sau zilele în care s-au emis cele mai multe facturi.

O interogare comună, bazată pe funcția analitică ROW_NUMBER, poate fi:

```

        SELECT ROW_NUMBER() OVER (ORDER BY Nr_Facturilor DESC) AS Pozitie,
            DataFact AS Zi,
            Nr_Facturilor
    FROM
        (SELECT DataFact, COUNT(*) AS Nr_Facturilor
        FROM facturi
        GROUP BY DataFact
        HAVING COUNT(*) >= ANY
            (SELECT Nr_Facturilor
            FROM
                (SELECT Nr_Facturilor, ROW_NUMBER() OVER
                    (ORDER BY Nr_Facturilor DESC) AS Poz
                FROM
                    (SELECT DISTINCT COUNT(*) AS Nr_Facturilor
                    FROM facturi
                    GROUP BY DataFact
                    ) X1
                ) X2
            WHERE X2.Poz <=1
            )
        ) X3
    ORDER BY Nr_Facturilor DESC

```

Folosind funcția RANK, ambele zile au același număr în top, 1. Și noua variantă este valabilă atât în DB2 și Oracle, cât și în SQL Server.

```

SELECT *
FROM
    (SELECT RANK() OVER (ORDER BY COUNT(*) DESC) AS Pozitie,
        DataFact AS Zi,

```

```

COUNT(*) AS Nr_Facturilor
FROM facturi
GROUP BY DataFact) X
WHERE Pozitie <= 1

```

Să se afișeze cele mai mari două valori ale numărului zilnic de facturi emise, precum și datele în care s-au înregistrat respectivele valori.

Modificăm, în penultima interogare, ce utilizează funcția ROW_NUMBER, doar 1 cu 2, adică, în loc de X2.Poz <=1 vom scrie X2.Poz <=2. Însă, cu funcția RANK, soluția este (o idee) mai scurtă:

```

SELECT RANK() OVER (ORDER BY COUNT(*) DESC) AS Pozitie,
       DataFact AS Zi, COUNT(*) AS Nr_Facturilor
FROM facturi
GROUP BY DataFact
HAVING COUNT(*) >= ANY
( SELECT Nr_Facturilor
  FROM
    (SELECT RANK() OVER (ORDER BY Nr_Facturilor DESC) AS Poz,
     Nr_Facturilor
    FROM
      (SELECT DISTINCT COUNT(*) AS Nr_Facturilor
       FROM facturi
       GROUP BY DataFact) X1
    ) X2
  WHERE Poz <= 2
)

```

RZ	POZITIE	RZ	ZI	RZ	NR_FACTURILOR
	1		01-08-2007		4
	1		07-08-2007		4
	3		14-08-2007		3

Figura 11.28. Indicarea corectă a poziției din clasament

Care sunt cel mai bine vândute două produse ale fiecărei zile din luna august 2007 ?
Practic, interesează o listă precum cea din figura 11.29.

DATAFACT	DENPR	VINZARI	POZITIE_ZI
01-08-2007	Produs 5	10085012	1
01-08-2007	Produs 2	363570	2
02-08-2007	Produs 2	277950	1
03-08-2007	Produs 1	113050	1
03-08-2007	Produs 2	109000	2
04-08-2007	Produs 1	166005	1
04-08-2007	Produs 2	35970	2
07-08-2007	Produs 5	10246138	1
07-08-2007	Produs 2	253698	2
14-08-2007	Produs 5	4284714	1
14-08-2007	Produs 2	313375	2
15-08-2007	Produs 2	247757	1
16-08-2007	Produs 1	257159	1
16-08-2007	Produs 2	210261	2
21-08-2007	Produs 5	10283623	1
21-08-2007	Produs 2	159467	2

Figura 11.29. Cele mai bine vândute două produse din fiecare zi

Soluția următoare este una comună (cu excepția PostgreSQL și înlocuind, în DB2 și MS SQL Server, EXTRACT cu YEAR/MONTH):

SELECT *

FROM

(SELECT DataFact, DenPr,

ROUND(SUM(Cantitate * PretUnit * (1+ProcTVA)),0) AS Vinzari,

RANK() OVER (PARTITION BY DataFact

ORDER BY SUM(Cantitate * PretUnit * (1+ProcTVA)) DESC)

AS Pozitie_Zi

FROM produse p INNER JOIN liniifact lf ON p.CodPr=lf.CodPr

INNER JOIN facturi f ON lf.NrFact=f.NrFact

WHERE EXTRACT(YEAR FROM DataFact)=2007 AND

EXTRACT (MONTH FROM DataFact)=8

GROUP BY DataFact, DenPr) x1

WHERE Pozitie_Zi <= 2

ORDER BY DataFact, Pozitie_Zi

Asemănătoare funcției RANK este DENSE_RANK ce, spre deosebire de prima, nu lasă goluri în clasament atunci când pe poziția precedentă sunt două sau mai multe linii. Diferența este pusă în valoare de interogarea următoare și rezultatul acestuia din figura 11.30.

SELECT DataFact, COUNT(*) AS Nr_Facturilor,

RANK() OVER (ORDER BY COUNT(*) DESC) AS Poz_RANK,

DENSE_RANK() OVER (ORDER BY COUNT(*) DESC)

AS Poz_DENSE_RANK
 FROM facturi
 GROUP BY DataFact

DATAFACT	NR_FACTURILOR	POZ_RANK	POZ_DENSE_RANK
01-08-2007	4	1	1
07-08-2007	4	1	1
14-08-2007	3	3	2
16-08-2007	2	4	3
15-08-2007	2	4	3
01-09-2007	2	4	3
02-08-2007	2	4	3
21-08-2007	2	4	3
02-09-2007	2	4	3
10-09-2007	2	4	3
22-08-2007	1	11	4
03-08-2007	1	11	4
07-09-2007	1	11	4
17-09-2007	1	11	4
04-08-2007	1	11	4

Figura 11.30. Diferența dintre RANK și DENSE_RANK

Cu DENSE_RANK se simplifică și interogarea ce afișează cele mai mari două valori ale numărului zilnic de facturi emise în luna august 2007, precum și datele (calendaristice) respective:

```
SELECT DENSE_RANK() OVER (ORDER BY COUNT(*) DESC) AS Pozitie,
       DataFact AS Zi, COUNT(*) AS Nr_Facturilor
FROM facturi
WHERE EXTRACT(YEAR FROM DataFact)=2007 AND
      EXTRACT (MONTH FROM DataFact)=8
GROUP BY DataFact
HAVING COUNT(*) >= ANY
      (SELECT Nr_Facturilor
       FROM
          (SELECT DENSE_RANK() OVER (ORDER BY COUNT(*) DESC) AS Poz,
                COUNT(*) AS Nr_Facturilor
           FROM facturi
           WHERE EXTRACT(YEAR FROM DataFact)=2007 AND
                 EXTRACT (MONTH FROM DataFact)=8
           GROUP BY DataFact
          ) X2
       WHERE Poz <= 2
      )
```

POZITIE	ZI	NR_FACTURILOR
1	01-08-2007	4
1	07-08-2007	4
2	14-08-2007	3

Figura 11.31. DENSE_RANK pentru obținerea primelor două valori ale numărului zilnic de facturi

În afara folositelor ROW_NUMBER, RANK și DENSE_RANK, în Oracle funcțiile OLAP au fost gândite să-și apropie și mai mult o categorie profesională sedusă, de mult timp, de programe precum SPSS sau chiar Excel, și anume statisticienii. Apar astfel funcțiile: CUME_DIST, PERCENT_RANK⁴, NTILE⁵. Pentru a le compara, folosim interogarea următoare al cărei rezultat este vizualizat în figura 11.32.

```
SELECT
    DENSE_RANK() OVER (ORDER BY COUNT(*) DESC) AS Pozitie,
    DataFact AS Zi,
    COUNT(*) AS Nr_Facturilor,
    ROUND(PERCENT_RANK() OVER (ORDER BY COUNT(*) DESC ),2)
        AS Poz_Proc_Zi,
    ROUND(CUME_DIST() OVER (ORDER BY COUNT(*) DESC ),2)
        AS Distrib_Cume_Zi,
    NTILE(3) OVER (ORDER BY COUNT(*) DESC ) AS Tertile_Zi,
    NTILE(4) OVER (ORDER BY COUNT(*) DESC ) AS Cuartile_Zi
FROM facturi
GROUP BY DataFact
```

⁴ Funcțiile CUME_DIST și PERCENT_RANK nu sunt însă implementate nici în DB2, nici în MS SQL Server

⁵ Funcția NTILE nu este implementată în DB2

POZITIE	ZI	NR_FACTURILOR	POZ_PROC_ZI	DISTRIB_CUME_ZI	TERTILE_ZI	CUARTILE_ZI
1	2007-08-01	4	0	0,13	1	1
1	2007-08-07	4	0	0,13	1	1
2	2007-08-14	3	0,14	0,2	1	1
3	2007-08-02	2	0,21	0,67	2	1
3	2007-09-01	2	0,21	0,67	2	2
3	2007-08-15	2	0,21	0,67	1	2
3	2007-08-16	2	0,21	0,67	1	2
3	2007-08-21	2	0,21	0,67	2	2
3	2007-09-10	2	0,21	0,67	2	3
3	2007-09-02	2	0,21	0,67	2	3
4	2007-08-22	1	0,71	1	3	3
4	2007-08-03	1	0,71	1	3	3
4	2007-09-17	1	0,71	1	3	4
4	2007-09-07	1	0,71	1	3	4
4	2007-08-04	1	0,71	1	3	4

Figura 11.32. Funcțiile Oracle PERCENT_RANK, CUME_DIST, și NTILE

- PERCENT_RANK întoarce poziția procentuală a unei linii în cadrul unei partiții, calculată prin formula: (poziția liniei în partiție - 1) / (nr. liniilor din partiție - 1)
- CUME_DIST calculează poziția unei valori specificate x relativă la o mulțime M de N valori: $CUME_DIST(x) = \frac{\text{numărul de valori (diferite de, sau egale cu x) din M ce preced pe x (în ordinea specificată)}}{N}$
- NTILE(3) calculează tertilele, iar NTILE(4) calculează cuartilele.

Din păcate, cunoștințele mele de statistică nu s-au ameliorat de la prima ediție a cărții, așa că ne oprim (tot) aici.

11.4. Ferestre pentru funcțiile analitice

Una dintre cele mai importante noțiuni introduse de Amendamentul OLAP al SQL:1999 este *ferestra*. Fereastra se definește în cadrul unei partiții și se referă la intervalul liniilor luat în calcul pentru linia curentă. Mărimea ferestrei se poate specifica fie fizic, printr-un număr de linii, fie logic, printr-un interval de tip dată calendaristică/timp sau interval de valori. Calculele se efectuează pentru fiecare linie din cadrul ferestrei, fereastră mișcătoare între o poziție de start și una de final. Linia curentă servește ca punct de referință pentru determinarea începutului și sfârșitului ferestrei.

Specificațiile unei ferestre privesc trei componente: partiționarea, ordonarea și grupurile de agregare. Orice funcție de agregare poate fi utilizată în cadrul unei ferestre: SUM, AVG, MIN, MAX, STDDEV, VARIANCE, COUNT. Pe lângă acestea, Oracle (începând cu versiunea 8i) mai oferă suport pentru alte câteva funcții statistice: VAR_SAMP, VAR_POP, STDDEV_SAMP s.a.m.d., pe care, din lipsă de cunoștințe de specialitate, nu le discutăm aici. Valorile agregate pot fi cumulative, mișcătoare sau centrate.

Să se afișeze, pentru fiecare zi, valoarea cumulată a vânzărilor după fiecare factură emisă în luna septembrie 2007

- partiția are dimensiunea unei zile; numărul de linii care o alcătuiesc depinde de numărul de facturi emise în ziua respectivă;
- ordonarea în cadrul partiției se face după valoarea NrFact; prin urmare, poziția de început a fiecărei partiții este dată de prima factură din ziua respectivă, iar cea de sfârșit de factura ce are cel mare număr pentru DataFact ce definește partiția curentă;
- fereastra pentru care se efectuează calculul, pentru fiecare linie, are o margine fixă – începutul partiției și una mobilă care este chiar linia curentă.

SQL:1999 a introdus clauza WINDOW în care se declară cele trei elemente enumerate anterior, partiționarea, ordonarea și grupurile de agregare. Soluția problemei formulate mai sus este:

```
SELECT DataFact AS Zi, NrFact, ValoareFact,
       SUM(ValoareFact) OVER W1 AS Val_Cumulata
FROM
  (SELECT DataFact, F.NrFact, SUM(Cantitate * PretUnit * (1+ProcTVA))
   AS ValoareFact
  FROM produse p INNER JOIN liniifact lf ON p.CodPr=lf.CodPr
   INNER JOIN facturi f ON lf.NrFact=f.NrFact
 WHERE EXTRACT(YEAR FROM DataFact)=2007 AND
       EXTRACT (MONTH FROM DataFact)=9
  GROUP BY DataFact, F.NrFact
   ) VALFACT
WINDOW W1 AS (PARTITION BY DataFact ORDER BY NrFact
              ROWS UNBOUNDED PRECEDING)
```

Fereastra definită are numele W1 și se constituie în cadrul partițiilor formate de valorile comune ale atributului DataFact. În cadrul partițiilor, liniile sunt ordonate crescător după valorile atributului NrFact, iar intervalul liniilor asupra căruia se aplică funcția de agregare SUM(...) este alcătuit din prima linie a partiției și linia curentă (ROWS UNBOUNDED PRECEDING).

Altă variantă SQL:1999 permite declararea “în linie” a ferestrei, variantă utilizată de DB2 și Oracle (înlocuind YEAR și MONTH cu EXTRACT-uri), parametrii ferestrei fiind definiți direct în clauza SELECT, după OVER (rezultatul este cel din figura 11.33):

```
SELECT DataFact AS Zi, NrFact, ValoareFact,
       SUM(ValoareFact) OVER (PARTITION BY DataFact
                              ORDER BY NrFact ROWS UNBOUNDED PRECEDING)
       AS Val_Cumulata
```


FROM

```
(SELECT DataFact, F.NrFact,
      ROUND(SUM(Cantitate * PretUnit * (1+ProcTVA)),0) AS ValoareFact
FROM produse p INNER JOIN liniifact lf ON p.CodPr=lf.CodPr
      INNER JOIN facturi f ON lf.NrFact=f.NrFact
WHERE YEAR(DataFact)=2007 AND MONTH(DataFact)=9
GROUP BY DataFact, F.NrFact
) VALFACT
```

ZI	NRFACT	VALOAREFACT	VAL_CUMULATA
01.09.2007	3.111	4.442.960,0000	4.442.960,0000
01.09.2007	3.112	153.442,0000	4.596.402,0000
02.09.2007	3.113	127.530,0000	127.530,0000
02.09.2007	3.115	110.908,0000	238.438,0000
07.09.2007	3.119	5.819.668,0000	5.819.668,0000
10.09.2007	3.116	136.850,0000	136.850,0000
10.09.2007	3.117	287.855,0000	424.705,0000
17.09.2007	3.118	179.565,0000	179.565,0000

Figura 11.33. Valoarea cumulată, după fiecare factură, pentru fiecare zi

Din păcate, MS SQL Server este prea puțin generos în materie de ferestre. Există o clauză de partiționare, dar funcția-agregat (de obicei, SUM) nu se poate aplica decât la nivelul întregii partiții. Deci, fereastra este chiar partiția. Ceea ce putem calcula este doar ceva de genul:

Care este contribuția fiecărei facturi emise în luna septembrie 2007 în totalul vânzărilor din ziua respectivă ?

Fereastra este o zi (DataFact), rezultatul din figura 11.34 fiind furnizat de soluția comună SQL Server/DB2:

```
SELECT DataFact AS Zi, NrFact, ValoareFact,
      SUM(ValoareFact) OVER (PARTITION BY DataFact) AS Val_Cumulata
FROM
  (SELECT f.NrFact, DataFact,
        ROUND(SUM(Cantitate * PretUnit * (1+ProcTVA)),0) AS ValoareFact
  FROM produse p INNER JOIN liniifact lf ON p.CodPr=lf.CodPr
        INNER JOIN facturi f ON lf.NrFact=f.NrFact
  WHERE YEAR(DataFact)=2007 AND MONTH (DataFact)=9
  GROUP BY f.NrFact, DataFact
  ) VALFACT
ORDER BY 1,2
```

Partea spumoasă – ferestrele „culisante” nu sunt la îndemână în MS SQL Server prin clauzele OVER/PARTITION.

Zi	NrFact	ValoareFact	Val_Cumulata
2007-09-01 00:00:00	3111	4442960.0000	4596402.0000
2007-09-01 00:00:00	3112	153442.0000	4596402.0000
2007-09-02 00:00:00	3113	127530.0000	238438.0000
2007-09-02 00:00:00	3115	110908.0000	238438.0000
2007-09-07 00:00:00	3119	5819668.0000	5819668.0000
2007-09-10 00:00:00	3116	136850.0000	424705.0000
2007-09-10 00:00:00	3117	287855.0000	424705.0000
2007-09-17 00:00:00	3118	179565.0000	179565.0000

Figura 11.34. O fereastră cât o zi

Să se calculeze (și afișeze) soldul fiecărui client după fiecare vânzare și încasare din luna august 2007.

Cele două operațiuni care influențează soldul (restul de plată al) unui client sunt vânzările (facturile emise) și încasările. Orice vânzare crește soldul, sold care indică cuantumul obligațiilor de plată ale clientului către firma noastră, iar printr-o încasare un client își mai diminuează din obligații. Mai întâi, trebuie să obținem o situație cronologică a facturărilor și încasărilor, ceea ce presupune o reuniune (sintaxă Oracle):

```
SELECT F.NrFact, DataFact AS Data,
       ROUND(SUM(Cantitate * PretUnit * (1+ProcTVA)),0) AS Facturat, 0 AS Incasat
FROM produse p
     INNER JOIN liniifact lf ON p.CodPr=lf.CodPr
     INNER JOIN facturi f ON lf.NrFact=f.NrFact
WHERE EXTRACT (YEAR FROM DataFact)=2007 AND
      EXTRACT (MONTH FROM DataFact)=8
GROUP BY F.NrFact, DataFact
UNION
SELECT NrFact, DataInc AS Data, 0 AS Facturat, Transa AS Incasat
FROM incasari i INNER JOIN incasfact if ON i.CodInc=if.CodInc
WHERE EXTRACT (YEAR FROM DataInc)=2007 AND
      EXTRACT (MONTH FROM DataInc)=8
```

Ultimele două coloane din rezultat (vezi figura 11.35) reprezintă cele două mărimi care influențează soldul. Atributul Data reprezintă, după caz, fie data facturii, fie data încasării. În continuare, jonționăm rezultatul acestei consultări cu tabela CLIENTI, apoi construim o partiție pentru fiecare client, iar soldul clientului va fi suma glisantă („mișcătoare”) a expresiei *Facturat – Incasat*.

Nr	NRFAC	Nr	DATA	Nr	FACTURAT	Nr	INCASAT
	1111		01-08-2007		4346038		0
	1111		15-08-2007		0		53996
	1112		01-08-2007		125516		0
	1112		15-08-2007		0		125516
	1113		01-08-2007		106275		0
	1113		17-08-2007		0		106275
	1114		01-08-2007		6021707		0
	1115		02-08-2007		151238		0
	1116		02-08-2007		126713		0
	1117		03-08-2007		222050		0
	1117		16-08-2007		0		9754
	1117		17-08-2007		0		9754
	1117		18-08-2007		0		3696
	1118		04-08-2007		201975		0
	1118		15-08-2007		0		101975
	1118		16-08-2007		0		100000
	1119		07-08-2007		5774499		0
	1120		07-08-2007		97664		0

Figura 11.35. Rezultat intermediar în calcularea soldului (pe client) după fiecare operațiune

```

SELECT DenCl, Data, NrFact, Facturat, Incasat,
       SUM(Facturat-Incasat) OVER (PARTITION BY DenCl ORDER BY NrFact
                                   ROWS UNBOUNDED PRECEDING) AS Sold_Client
FROM (
  SELECT DenCl,t.*
  FROM
    (SELECT F.NrFact, DataFact AS Data,
             ROUND(SUM(Cantitate * PretUnit * (1+ProcTVA)),0)
             AS Facturat, 0 AS Incasat
     FROM produse p
      INNER JOIN liniifact lf ON p.CodPr=lf.CodPr
      INNER JOIN facturi f ON lf.NrFact=f.NrFact
     WHERE EXTRACT (YEAR FROM DataFact)=2007 AND
           EXTRACT (MONTH FROM DataFact)=8
     GROUP BY F.NrFact, DataFact
      UNION
     SELECT NrFact, DataInc AS Data, 0 AS Facturat, Transa AS Incasat
     FROM incasari i INNER JOIN incasfact if ON i.CodInc=if.CodInc
     WHERE EXTRACT (YEAR FROM DataInc)=2007 AND
           EXTRACT (MONTH FROM DataInc)=8
  )

```

```

) t INNER JOIN facturi f ON t.NrFact=f.NrFact
    INNER JOIN clienti c ON f.CodCl=c.CodCl
)

```

O bună parte din rezultat este prezentată în figura 11.36. Ultima coloană este suma culisantă a diferenței dintre facturat și încasat.

RZ	DENCL	RZ	NRFACT	RZ	DATA	RZ	FACTURAT	RZ	INCASAT	RZ	SOLD_CLIENT
	Client 1 SRL		1111		01-08-2007		4346038		0		4346038
	Client 1 SRL		1111		15-08-2007		0		53996		4292042
	Client 1 SRL		1115		02-08-2007		151238		0		4443280
	Client 1 SRL		1117		18-08-2007		0		3696		4439584
	Client 1 SRL		1117		17-08-2007		0		9754		4429830
	Client 1 SRL		1117		16-08-2007		0		9754		4420076
	Client 1 SRL		1117		03-08-2007		222050		0		4642126
	Client 1 SRL		1118		15-08-2007		0		101975		4540151
	Client 1 SRL		1118		16-08-2007		0		100000		4440151
	Client 1 SRL		1118		04-08-2007		201975		0		4642126
	Client 1 SRL		1120		07-08-2007		97664		0		4739790
	Client 1 SRL		1120		16-08-2007		0		7315		4732475
	Client 1 SRL		2111		14-08-2007		4442960		0		9175435
	Client 1 SRL		2115		15-08-2007		110908		0		9286343
	Client 1 SRL		2117		16-08-2007		287855		0		9574198
	Client 1 SRL		2118		16-08-2007		179565		0		9753763
	Client 2 SA		1113		01-08-2007		106275		0		106275
	Client 2 SA		1113		17-08-2007		0		106275		0
	Client 2 SA		2113		14-08-2007		127530		0		127530
	Client 3 SRL		1119		07-08-2007		5774499		0		5774499
	Client 3 SRL		2119		21-08-2007		5819668		0		11594167
	Client 4		1121		07-08-2007		4737838		0		4737838

Figura 11.36. Soldul clientului după fiecare operațiune (facturare sau încasare)

Interogarea următoare, funcționabilă în Oracle și DB2 (dacă înlocuim funcția `EXTRACT`) al cărui rezultat este cel din figura 11.37, ilustrează patru moduri de construire a unei ferestre:

- Coloana `Fact_Precedte_Crta/Zi` calculează suma valorilor facturilor din ziua respectivă, începând cu prima și terminând cu factura curentă (`ROWS UNBOUNDED PRECEDING`);
- Coloana `Fact_Crta_Precedta/Zi` însumează valoarea facturii curente și cea a precedentei (`ROWS 1 PRECEDING`) din ziua respectivă;
- La calculul `3Fact_Ant_Crt_Urmat/Zi` se însumează valorile facturilor: curentă, precedentă și următoare (`ROWS BETWEEN 1 PRECEDING AND 1 FOLLOWING`);

- În fine, *Crta_Urmatoarele/Zi* însumează valoarea facturii curente și tuturor celor ce-i succed în ziua respectivă (*ROWS BETWEEN CURRENT ROW AND UNBOUNDED FOLLOWING*).

```

SELECT DataFact AS Zi, NrFact, ValoareFact AS "Factura_Crt",
      SUM(ValoareFact) OVER (PARTITION BY DataFact ORDER BY NrFact
        ROWS UNBOUNDED PRECEDING) AS "Fact_Precedte_Crta/Zi",
      SUM(ValoareFact) OVER (PARTITION BY DataFact ORDER BY NrFact
        ROWS 1 PRECEDING) AS "Fact_Crta_Precedta/Zi",
      SUM(ValoareFact) OVER (PARTITION BY DataFact ORDER BY NrFact
        ROWS BETWEEN 1 PRECEDING AND 1 FOLLOWING)
        AS "3Fact:_Ant_Crt_Urmat/Zi",
      SUM(ValoareFact) OVER (PARTITION BY DataFact ORDER BY NrFact
        ROWS BETWEEN CURRENT ROW AND UNBOUNDED FOLLOWING)
        AS "Crta_Urmatoarele/Zi"
FROM
  (SELECT DataFact, F.NrFact,
    ROUND(SUM(Cantitate * PretUnit * (1+ProcTVA)),0) AS ValoareFact
  FROM produse p INNER JOIN liniifact lf ON p.CodPr=lf.CodPr
    INNER JOIN facturi f ON lf.NrFact=f.NrFact
  WHERE EXTRACT(YEAR FROM DataFact)=2007 AND
    EXTRACT (MONTH FROM DataFact)=8
  GROUP BY DataFact, F.NrFact
  ) VALFACT

```

ZI	NRFACT	Factura_Crt	Fact_Precedte_Crta/Zi	Fact_Crta_Precedta/Zi	3Fact:_Ant_Crt_Urmat/Zi	Crta_Urmatoarele/Zi
01-08-2007	1111	4346038	4346038	4346038	4471554	10599536
01-08-2007	1112	125516	4471554	4471554	4577829	6253498
01-08-2007	1113	106275	4577829	231791	6253498	6127982
01-08-2007	1114	6021707	10599536	6127982	6127982	6021707
02-08-2007	1115	151238	151238	151238	277951	277951
02-08-2007	1116	126713	277951	277951	277951	126713
03-08-2007	1117	222050	222050	222050	222050	222050
04-08-2007	1118	201975	201975	201975	201975	201975
07-08-2007	1119	5774499	5774499	5774499	5872163	10610001
07-08-2007	1120	97664	5872163	5872163	10610001	4835502
07-08-2007	1121	4737838	10610001	4835502	4835502	4737838
14-08-2007	2111	4442960	4442960	4442960	4596402	4723932
14-08-2007	2112	153442	4596402	4596402	4723932	280972
14-08-2007	2113	127530	4723932	280972	280972	127530
15-08-2007	2115	110908	110908	110908	247758	247758
15-08-2007	2116	136850	247758	247758	247758	136850
16-08-2007	2117	287855	287855	287855	467420	467420
16-08-2007	2118	179565	467420	467420	467420	179565
21-08-2007	2119	5819668	5819668	5819668	10560940	10560940
21-08-2007	2121	4741272	10560940	10560940	10560940	4741272

Figura 11.37. Patru moduri de definire a ferestrelor

Dacă în SELECT-ul precedent mărimea fereastrelor era definită fizic, prin numărul de linii ce preced sau succed liniei curente, în următoarea interogare se pun față în față ferestre definite fizic, prin număr de linii, și logic, prin intervale (de zile).

```
SELECT Client, DataFact AS Zi, Vinzari,
       SUM(Vinzari) OVER (PARTITION BY Client ORDER BY DataFact
                          ROWS 1 PRECEDING) AS Crt_Prec1_Linii,
       SUM(Vinzari) OVER (PARTITION BY Client ORDER BY DataFact
                          RANGE INTERVAL '1' DAY PRECEDING) AS Crt_Prec1_Zile,
       SUM(Vinzari) OVER (PARTITION BY Client ORDER BY DataFact
                          ROWS BETWEEN 1 PRECEDING AND 1 FOLLOWING)
                          AS P1_Act_U1_Linii,
       SUM(Vinzari) OVER (PARTITION BY Client ORDER BY DataFact
                          RANGE BETWEEN INTERVAL '1' DAY PRECEDING AND
                          INTERVAL '1' DAY FOLLOWING) AS P1_Act_U1_Zile
FROM
  (SELECT DenCl AS Client, DataFact,
          ROUND(SUM(Cantitate * PretUnit * (1+ProcTVA)),0) AS Vinzari
   FROM produse p INNER JOIN liniifact lf ON p.CodPr=lf.CodPr
          INNER JOIN facturi f ON lf.NrFact=f.NrFact
          INNER JOIN clienti c ON f.CodCl=c.CodCl
   WHERE EXTRACT(YEAR FROM DataFact)=2007 AND
          EXTRACT (MONTH FROM DataFact)=9
   GROUP BY DenCl, DataFact
  ) VALFACT
```

- Coloana *Crt_Prec1_Linii* însumează, după cum am văzut în precedenta interogare, valoarea facturii curente și cea a precedentei (*ROWS 1 PRECEDING*);
- La calculul valorii *Crt_Prec1_Zile* se ia în calcul linia precedentă, numai dacă data căruia îi este asociată este imediat înaintea datei din linia curentă (*RANGE INTERVAL '1' DAY PRECEDING*) – vezi, pentru Client 1, situația zilei de 7 august;

CLIENT	ZI	VINZARI	CRT_PREC1_LINII	CRT_PREC1_ZILE	P1_ACT_U1_LINII	P1_ACT_U1_ZILE
Client 1 SRL	01-09-2007	4442960	4442960	4442960	4553868	4553868
Client 1 SRL	02-09-2007	110908	4553868	4553868	4841723	4553868
Client 1 SRL	10-09-2007	287855	398763	287855	578328	287855
Client 1 SRL	17-09-2007	179565	467420	179565	467420	179565
Client 2 SA	02-09-2007	127530	127530	127530	127530	127530
Client 3 SRL	07-09-2007	5819668	5819668	5819668	5819668	5819668
Client 5 SRL	01-09-2007	153442	153442	153442	153442	153442
Client 7 SRL	10-09-2007	136850	136850	136850	136850	136850

Figura 11.38. Ferestelor definite logic și fizic

Curios, deși interogarea respectă specificațiile SQL-99 și funcționează în Oracle, în DB2 nu am reușit să o fac operațională.

11.5. Comparații și ponderi

Atunci când, într-o partiție, se dorește raportarea valorilor liniei curente la cele din prima sau ultima linie dintr-un interval sau fereastră, este rezonabil să se folosească funcțiile `FIRST_VALUE` și `LAST_VALUE` implementate în DB2 și Oracle, nu însă și în SQL Server.

Care este raportul zilnic al vânzărilor pe luna septembrie 2007, relativ la prima zi de vânzări ?

```
SELECT
    DataFact AS Zi,Vinzari,
    FIRST_VALUE(Vinzari) OVER (ORDER BY DataFact) AS Vnz_Zi1,
    ROUND(Vinzari / FIRST_VALUE(Vinzari) OVER (ORDER BY DataFact),2)
        AS Raport_Crt_Prima_Zi
FROM
    (SELECT DataFact,
        ROUND(SUM(Cantitate * PretUnit * (1+ProcTVA)),0) AS Vinzari
    FROM produse p INNER JOIN liniifact lf ON p.CodPr=lf.CodPr
        INNER JOIN facturi f ON lf.NrFact=f.NrFact
    WHERE EXTRACT(YEAR FROM DataFact)=2007 AND
        EXTRACT (MONTH FROM DataFact)=9
    GROUP BY DataFact
    )
```

ZI	VINZARI	VNZ_ZI1	RAPORT_CRT_PRIMA_ZI
01-09-2007	4596402	4596402	1
02-09-2007	238438	4596402	0,05
07-09-2007	5819668	4596402	1,27
10-09-2007	424705	4596402	0,09
17-09-2007	179565	4596402	0,04

Figura 11.39. Exemplu de utilizare a funcției `FIRST_VALUE`

- Să se determine zilele din luna august 2007 în care s-a vândut cel mai bine fiecare produs.

Una dintre soluții se bazează pe o subconsultare ce întrebuințează funcția `FIRST_VALUE`:

```
SELECT DenPr, Datafact, Vinzari
FROM
    (SELECT DenPr, DataFact,
```

```

ROUND(SUM(Cantitate * PretUnit * (1+ProcTVA)),0) AS Vinzari,
FIRST_VALUE(ROUND(SUM(Cantitate * PretUnit * (1+ProcTVA)),0))
OVER (PARTITION BY DenPr ORDER BY
ROUND(SUM(Cantitate * PretUnit * (1+ProcTVA)),0) DESC)
AS Vnz_Max
FROM produse p INNER JOIN liniifact lf ON p.CodPr=lf.CodPr
INNER JOIN facturi f ON lf.NrFact=f.NrFact
WHERE EXTRACT(YEAR FROM DataFact)=2007 AND
EXTRACT (MONTH FROM DataFact)=8
GROUP BY DenPr, DataFact
) t
WHERE Vinzari = Vnz_Max

```

DENPR	DATAFACT	VINZARI	VNZ_MAX
Produs 1	16-08-2007	257159	257159
Produs 1	04-08-2007	166005	257159
Produs 1	03-08-2007	113050	257159
Produs 1	14-08-2007	67830	257159
Produs 1	01-08-2007	59500	257159
Produs 2	01-08-2007	363570	363570
Produs 2	14-08-2007	313375	363570
Produs 2	02-08-2007	277950	363570
Produs 2	07-08-2007	253698	363570
Produs 2	15-08-2007	247757	363570
Produs 2	16-08-2007	210261	363570
Produs 2	21-08-2007	159467	363570
Produs 2	03-08-2007	109000	363570
Produs 2	04-08-2007	35970	363570
Produs 3	14-08-2007	58013	58013
Produs 3	01-08-2007	35700	58013
Produs 3	21-08-2007	33320	58013
Produs 3	07-08-2007	33320	58013
Produs 4	21-08-2007	84530	84530
Produs 4	07-08-2007	76845	84530
Produs 4	01-08-2007	55754	84530
Produs 5	21-08-2007	10283623	10283623
Produs 5	07-08-2007	10246138	10283623
Produs 5	01-08-2007	10085012	10283623
Produs 5	14-08-2007	4284714	10283623

Figura 11.40. Determinarea, pentru fiecare produs, a vânzărilor și a maximului zilnic din luna august 2007

Subconsultarea:

```
SELECT DenPr, DataFact,
       ROUND(SUM(Cantitate * PretUnit * (1+ProcTVA)),0) AS Vinzari,
       FIRST_VALUE(ROUND(SUM(Cantitate * PretUnit * (1+ProcTVA)),0))
       OVER (PARTITION BY DenPr ORDER BY
            ROUND(SUM(Cantitate * PretUnit * (1+ProcTVA)),0) DESC)
       AS Vnz_Max
FROM produse p INNER JOIN liniifact lf ON p.CodPr=lf.CodPr
     INNER JOIN facturi f ON lf.NrFact=f.NrFact
WHERE EXTRACT(YEAR FROM DataFact)=2007 AND
      EXTRACT (MONTH FROM DataFact)=8
GROUP BY DenPr, DataFact
```

obține rezultatul din figura 11.40. În final, răspunsul problemei este cel din figura 11.41.

DENPR	DATAFACT	VINZARI
Produs 1	16-08-2007	257159
Produs 2	01-08-2007	363570
Produs 3	14-08-2007	58013
Produs 4	21-08-2007	84530
Produs 5	21-08-2007	10283623

Figura 11.41. Zilele în care s-a vândut cel mai bine fiecare produs

Pentru respectarea adevărului istoric, se cuvine de remarcat că, în locul cosmopolitei funcții `FIRST_VALUE`, am fi putut folosi în subconsultare și tocita `MAX`, bineînțeles, într-o haină nouă:

```
SELECT DenPr, DataFact, ROUND(Vinzari,0) AS Vinzari
FROM
    (SELECT DenPr, DataFact,
            SUM(Cantitate * PretUnit * (1+ProcTVA)) AS Vinzari,
            MAX(SUM(Cantitate * PretUnit * (1+ProcTVA))) OVER
            (PARTITION BY DenPr
             ORDER BY SUM(Cantitate * PretUnit * (1+ProcTVA)) DESC)
            AS Vnz_Max
    FROM produse p INNER JOIN liniifact lf ON p.CodPr=lf.CodPr
         INNER JOIN facturi f ON lf.NrFact=f.NrFact
    WHERE EXTRACT(YEAR FROM DataFact)=2007 AND
          EXTRACT (MONTH FROM DataFact)=8
    GROUP BY DenPr, DataFact
    )
WHERE Vinzari=Vnz_Max
```

Atunci când se dorește determinarea: contribuției unui produs la totalul vânzărilor, părții unui client în totalul datornicilor, procentului unui produs din valoarea unei facturi, ponderii salariului directorului general (sau al șefilor de compartimente) în totalul fondului de salarii al unei întreprinderi s.a.m.d. se poate apela la serviciile funcției `RATIO_TO_REPORT` care calculează raportul dintre o valoare (atribut/expresie) și suma totală a valorii respective pentru toate liniile din fereastră. Funcția este implementată în Oracle, nu și în DB2 și MS SQL Server.

Să se afișeze structura vânzărilor pe județe.

```
SELECT Judet, Vinzari,
       ROUND(RATIO_TO_REPORT(Vinzari) OVER (),2) AS Pondere_Judet
FROM
  (SELECT Judet,
           ROUND(SUM(Cantitate * PretUnit * (1+ProcTVA)),0) AS Vinzari
   FROM judete j
        INNER JOIN coduri_postale cp ON j.Jud=cp.Jud
        INNER JOIN clienti c ON cp.CodPost=c.CodPost
        INNER JOIN facturi f ON c.CodCl=f.CodCl
        INNER JOIN liniifact lf ON lf.NrFact=f.NrFact
        INNER JOIN produse p ON lf.CodPr=p.CodPr
   GROUP BY Judet
  )
```

După cum se observă, avantajul funcției este că nu necesită precizarea explicită a numitorului. Rezultatul interogării este prezentat în figura 11.42.

JUDET	VINZARI	PONDERE_JUDET
Iasi	24901983	0,51
Neamt	6021707	0,12
Timis	832812	0,02
Vaslui	17413835	0,35

Figura 11.42. Contribuția fiecărui județ în totalul vânzărilor

Să se calculeze contribuția produselor la valoarea fiecărei facturi emise în luna septembrie 2007.

```
SELECT Factura, Produs, Vinzari,
       ' || ROUND(RATIO_TO_REPORT(Vinzari) OVER (PARTITION BY Factura),2)
       * 100 || '%' AS Pondere_Prod_Fact
FROM
  (SELECT NrFact AS Factura, DenPr AS Produs,
           ROUND(SUM(Cantitate * PretUnit * (1+ProcTVA)),0) AS Vinzari
   FROM produse p INNER JOIN liniifact lf ON p.CodPr=lf.Codpr
        INNER JOIN facturi f ON lf.NrFact=f.NrFact
  )
```

```

WHERE EXTRACT(YEAR FROM DataFact)=2007 AND
      EXTRACT (MONTH FROM DataFact)=9
GROUP BY NrFact, DenPr
)

```

FACTURA	PRODUS	VINZARI	PONDERE_PROD_FACT
3111	Produs 1	67830	2%
3111	Produs 2	90416	2%
3111	Produs 5	4284714	96%
3112	Produs 2	95430	62%
3112	Produs 3	58013	38%
3113	Produs 2	127530	100%
3115	Produs 2	110908	100%
3116	Produs 2	136850	100%
3117	Produs 1	124355	43%
3117	Produs 2	163500	57%
3118	Produs 1	132804	74%
3118	Produs 2	46761	26%
3119	Produs 2	41584	1%
3119	Produs 3	33320	1%
3119	Produs 4	84530	1%
3119	Produs 5	5660235	97%

Figura 11.43. Ponderea fiecărui produs în fiecare factură emisă în luna septembrie 2007

La analiza dinamicii unei mărimi, se raportează valoarea din linia curentă la cea dintr-o altă linie plasată la o anumită distanță. Spre exemplu, pentru compararea vânzărilor fiecărei luni calendaristice la aceeași lună din anul precedent, va trebui să împărțim valoarea de pe linia curentă la o valoare aflată cu 12 linii/luni înainte. Apare astfel noțiunea de deplasament înapoi (LAG) și înainte (LEAD). Firește, soluția funcționează numai dacă nu există luni de întrerupere, adică intervalul este continuu. În interogarea următoare, al cărei rezultat este prezentat în figura 11.44, deplasamentul este, pentru ambele funcții, 1.

```

SELECT Zi, Vinzari,
      LAG(Vinzari,1) OVER (ORDER BY Zi) AS Vnz_LAG,
      LEAD(Vinzari,1) OVER (ORDER BY Zi) AS Vnz_LEAD
FROM
  (SELECT DataFact AS Zi,
        ROUND(SUM(Cantitate * PretUnit * (1+ProcTVA)),0) AS Vinzari
  FROM produse p
    INNER JOIN liniifact lf ON P.CodPr=lf.Codpr
    INNER JOIN facturi f ON lf.NrFact=f.NrFact
  WHERE EXTRACT(YEAR FROM DataFact)=2007 AND
        EXTRACT (MONTH FROM DataFact)=9
  GROUP BY DataFact)

```

Funcțiile LAG și LEAD extrag vânzările din ziua precedentă, respectiv următoare. Locurile libere din cadrul listei reprezintă valori NULL. Ambele funcții sunt implementate în DB2 și Oracle, nu însă și în MS SQL Server.

ZI	VINZARI	VNZ_LAG	VNZ_LEAD
01-09-2007	4596402	(null)	238438
02-09-2007	238438	4596402	5819668
07-09-2007	5819668	238438	424705
10-09-2007	424705	5819668	179565
17-09-2007	179565	424705	(null)

Figura 11.44. Vânzările din ziua curentă, precedentă și următoare

Care este evoluția vânzărilor de la o zi la alta, pentru luna septembrie 2007 ?

```

SELECT Zi, Vinzari,
       COALESCE(LAG(Vinzari,1) OVER (ORDER BY Zi),0) AS Vnz_Preced,
       Vinzari - COALESCE((LAG(Vinzari,1) OVER (ORDER BY Zi)),0) AS Diferenta
FROM
  (SELECT DataFact AS Zi,
         ROUND(SUM(Cantitate * PretUnit * (1+ProcTVA)),0) AS Vinzari
   FROM produse p
    INNER JOIN liniifact lf ON p.CodPr=lf.Codpr
    INNER JOIN facturi f ON lf.NrFact=f.NrFact
   WHERE EXTRACT(YEAR FROM DataFact)=2007 AND
         EXTRACT (MONTH FROM DataFact)=9
   GROUP BY DataFact
  )

```

ZI	VINZARI	VNZ_PRECED	DIFERENTA
01-09-2007	4596402	0	4596402
02-09-2007	238438	4596402	-4357964
07-09-2007	5819668	238438	5581230
10-09-2007	424705	5819668	-5394963
17-09-2007	179565	424705	-245140

Figura 11.45. Evoluția zilnică a vânzărilor pentru septembrie 2007