

Funcții grup și clauzele GROUP BY, HAVING. Operatorii ROLLUP și CUBE.

I. [Funcții grup și clauza GROUP BY]

- Clauza *GROUP BY* este utilizată pentru a diviza liniile unui tabel în grupuri. Pentru a returna informația corespunzătoare fiecărui astfel de grup, pot fi utilizate funcțiile agregat. Ele pot apărea în clauzele:
 - *SELECT*
 - *ORDER BY*
 - *HAVING*.

Server-ul *Oracle* aplică aceste funcții fiecărui grup de linii și returnează un singur rezultat pentru fiecare mulțime.

- Dintre funcțiile grup definite în sistemul *Oracle*, se pot enumera: *AVG*, *SUM*, *MAX*, *MIN*, *COUNT*, *STDDEV*, *VARIANCE* etc. Tipurile de date ale argumentelor funcțiilor grup pot fi *CHAR*, *VARCHAR2*, *NUMBER* sau *DATE*.
 - Funcțiile *AVG*, *SUM*, *STDDEV* și *VARIANCE* operează numai asupra valorilor numerice.
 - Funcțiile *MAX* și *MIN* pot opera asupra valorilor numerice, caracter sau dată calendaristică.
- Absența clauzei *GROUP BY* conduce la aplicarea funcției grup pe mulțimea tuturor liniilor tabelului.
- Toate funcțiile grup, cu excepția lui *COUNT(*)*, ignoră valorile *null*. *COUNT(expresie)* returnează numărul de linii pentru care expresia dată nu are valoarea *null*. Funcția *COUNT* returnează un număr mai mare sau egal cu zero și nu întoarce niciodată valoarea *null*.
- Când este utilizată clauza *GROUP BY*, server-ul sortează implicit mulțimea rezultată în ordinea crescătoare a valorilor coloanelor după care se realizează gruparea.
- În clauza *GROUP BY* a unei cereri se pot utiliza operatorii *ROLLUP* și *CUBE*. Acești operatori sunt disponibili începând cu versiunea *Oracle8i*.
- Expresiile din clauza *SELECT* a unei cereri care conține opțiunea *GROUP BY* trebuie să reprezinte o proprietate unică de grup, adică fie un atribut de grupare, fie o funcție de agregare aplicată tuplurilor unui grup, fie o expresie formată pe baza primelor două. Toate expresiile din clauza *SELECT*, cu excepția funcțiilor de agregare, se trec în clauza *GROUP BY* (unde pot apărea cel mult 255 expresii).

II. [Clauza HAVING]

Opțiunea *HAVING* permite restricționarea grupurilor de linii returnate, la cele care îndeplinesc o anumită condiție.

Dacă această clauză este folosită în absența unei clauze *GROUP BY*, aceasta presupune că gruparea se aplică întregului tabel, deci este returnată o singură linie, care este reținută în rezultat doar dacă este îndeplinită condiția din clauza *HAVING*.

III. [Operatorul ROLLUP]

Acest operator furnizează valori agregat și superagregat corespunzătoare expresiilor din clauza *GROUP BY*. Operatorul *ROLLUP* poate fi folosit pentru extragerea de statistici și informații totalizatoare din mulțimile rezultate. Acest operator poate fi util la generarea de rapoarte, diagrame și grafice.

Operatorul *ROLLUP* creează grupări prin deplasarea într-o singură direcție, de la dreapta la stânga, de-a lungul listei de coloane specificate în clauza *GROUP BY*. Apoi, se aplică funcția agregat acestor grupări. Dacă sunt specificate n expresii în operatorul *ROLLUP*, numărul de grupări generate va fi $n + 1$. Liniile care se bazează pe valoarea primelor n expresii se numesc linii obișnuite, iar celelalte se numesc linii superagregat.

Dacă în clauza *GROUP BY* sunt specificate n coloane, pentru a produce subtotaluri fără operatorul *ROLLUP* ar fi necesare $n + 1$ instrucțiuni *SELECT* conectate prin *UNION ALL*. Aceasta ar face execuția cererii ineficientă pentru că fiecare instrucțiune *SELECT* determină accesarea tabelului. Operatorul *ROLLUP* determină rezultatele efectuând un singur acces la tabel și este util atunci când sunt implicate multe coloane în producerea subtotalurilor.

Ilustrăm aplicarea acestui operator prin urmatorul exemplu.

Exemplu:

Pentru departamentele având codul mai mic decât 50, să se afișeze:

- pentru fiecare departament și pentru fiecare an al angajării (corespunzător departamentului respectiv), valoarea totală a salariilor angajaților în acel an;
- valoarea totală a salariilor pe departamente (indiferent de anul angajării);
- valoarea totală a salariilor (indiferent de anul angajării și de departament).

```
SELECT department_id, TO_CHAR(hire_date, 'yyyy'), SUM(salary)
FROM employees
WHERE department_id < 50
GROUP BY ROLLUP(department_id, TO_CHAR(hire_date, 'yyyy'));
```

Instrucțiunea precedentă va avea un rezultat de forma:

DEPARTMENT_ID	TO_CHAR(hire_date,'yyyy')	SUM(SALARY)
10	1987	4400
10		4400
20	1996	13000
20	1997	6000
20		19000
30	1994	11000
30	1995	3100
30	1997	5700
30	1998	2600
30	1999	2500
30		24900
40	1994	6500
40		6500
		54800

În rezultatul prezentat anterior se pot distinge 3 tipuri de linii.

- 1) Prima linie reprezintă suma salariilor angajaților în 1987 din departamentul care are codul 10. În mod similar se interpretează liniile din rezultat care au toate coloanele completate.

- 2) Linia a doua conține valoarea totală a salariilor din departamentul al cărui cod este 10. La fel se interpretează toate liniile care se disting prin faptul că valoarea coloanei *TO_CHAR(hire_date, 'dd')* este *null*.
- 3) Ultima linie conține suma salariilor tuturor angajaților din departamentele al căror cod este mai mic decât 50. Întrucât această linie corespunde totalului general, ea conține valoarea *null* pe toate coloanele, cu excepția câmpului *SUM(salary)*.

IV. [Operatorul CUBE]

Operatorul *CUBE* grupează liniile selectate pe baza valorilor tuturor combinațiilor posibile ale expresiilor specificate și returnează câte o linie totalizatoare pentru fiecare grup. Acest operator este folosit pentru a produce mulțimi de rezultate care sunt utilizate în rapoarte. În vreme ce *ROLLUP* produce subtotalurile doar pentru o parte dintre combinațiile posibile, operatorul *CUBE* produce subtotaluri pentru toate combinațiile posibile de grupări specificate în clauza *GROUP BY*, precum și un total general.

Dacă există *n* coloane sau expresii în clauza *GROUP BY*, vor exista 2^n combinații posibile superagregat. Din punct de vedere matematic, aceste combinații formează un cub *n*-dimensional, de aici provenind numele operatorului. Pentru producerea de subtotaluri fără ajutorul operatorului *CUBE* ar fi necesare 2^n instrucțiuni *SELECT* conectate prin *UNION ALL*.

Exemplu:

Pentru departamentele având codul mai mic decât 50 să se afișeze:

- valoarea totală a salariilor corespunzătoare fiecărui an de angajare, din cadrul fiecărui departament;
- valoarea totală a salariilor din fiecare departament (indiferent de anul angajării);
- valoarea totală a salariilor corespunzătoare fiecărui an de angajare (indiferent de departament);
- valoarea totală a salariilor (indiferent de departament și de anul angajării).

```
SELECT department_id, TO_CHAR(hire_date, 'yyyy'), SUM(salary)
FROM employees
WHERE department_id < 50
GROUP BY CUBE(department_id, TO_CHAR(hire_date, 'yyyy'));
```

În plus față de rezultatul corespunzător operației *ROLLUP*, operatorul *CUBE* va produce linii care reprezintă suma salariilor pentru fiecare an de angajare corespunzător unui departament având codul mai mic decât 50. Aceste linii se disting prin faptul că valoarea coloanei *department_id* este *null*.

V. [Exerciții – funcții grup și clauzele GROUP BY, HAVING]

1. a) Funcțiile grup includ valorile *NULL* în calcule?
b) Care este deosebirea dintre clauzele *WHERE* și *HAVING*?
2. Să se afișeze cel mai mare salariu, cel mai mic salariu, suma și media salariilor tuturor angajaților. Etichetați coloanele Maxim, Minim, Suma, respectiv Media. Să se rotunjească rezultatele. Să se salveze instrucțiunea SQL într-un fișier *p214.sql*.

```
SELECT MAX(salary) Maxim, _____, _____, _____
FROM employees;
```

3. Să se modifice fișierul *p114.sql* pentru a se afișa minimul, maximul, suma și media salariilor pentru fiecare job. Salvati acest fișier ca *p214.sql*. Executați cererea.

```
SELECT job_id, MAX(salary) Maxim, _____, _____, _____
FROM employees
```

GROUP BY job_id;

4. Să se afișeze numărul de angajați pentru fiecare job.

```
SELECT job_id, COUNT(*)  
FROM employees  
GROUP BY _____;
```

5. Să se determine numărul de angajați care sunt șefi. Etichetati coloana "Nr. manageri".

```
SELECT COUNT(DISTINCT manager_id) "Nr. Manageri"  
FROM employees;
```

? De ce am folosit cuvântul cheie DISTINCT? Ce am fi obținut dacă îl omiteam?

6. Să se afișeze diferența dintre cel mai mare și cel mai mic salariu. Etichetati coloana "Diferența".

7. Scrieți o cerere pentru a se afișa numele departamentului, locația, numărul de angajați și salariul mediu pentru angajații din acel departament. Coloanele vor fi etichetate corespunzător.

```
SELECT d.department_name "Departament", l.city "Locatie",  
COUNT(*) "Nr angajati", AVG(salary) "Salariu mediu"  
FROM locations l, departments d, employees e  
WHERE l.location_id = d.location_id  
AND d.department_id = e.department_id  
GROUP BY department_name, l.city;
```

!!!Obs: În clauza *GROUP BY* se trec obligatoriu toate coloanele prezente în clauza *SELECT*, care nu sunt argument al funcțiilor grup (a se vedea ultima observație de la punctul I).

8. Să se afișeze codul și numele angajaților care câștiga mai mult decât salariul mediu din firmă. Se va sorta rezultatul în ordine descrescătoare a salariilor.
9. Pentru fiecare șef, să se afișeze codul său și salariul celui mai prost platit subordonat. Se vor exclude cei pentru care codul managerului nu este cunoscut. De asemenea, se vor exclude grupurile în care salariul minim este mai mic de 1000\$. Sortați rezultatul în ordine descrescătoare a salariilor.
10. Pentru departamentele în care salariul maxim depășește 3000\$, să se obțină codul, numele acestor departamente și salariul maxim pe departament.
11. Care este salariul mediu minim al job-urilor existente? Salariul mediu al unui job va fi considerat drept media aritmetică a salariilor celor care îl practică.
12. Să se afișeze codul, numele departamentului și suma salariilor pe departamente.
13. Să se afișeze maximul salariilor medii pe departamente.
14. Sa se obtina codul, titlul și salariul mediu al job-ului pentru care salariul mediu este minim.
15. Să se afișeze salariul mediu din firmă doar dacă acesta este mai mare decât 2500. (clauza *HAVING* fără *GROUP BY*)
16. Să se afișeze suma salariilor pe departamente și, în cadrul acestora, pe job-uri.
! Modificați această cerere astfel încât să se afișeze numele departamentelor și titlul job-urilor.

17. Să se afișeze numele departamentului si cel mai mic salariu din departamentul avand cel mai mare salariu mediu.
18. Sa se afiseze codul, numele departamentului si numarul de angajati care lucreaza in acel departament pentru:
 - a) departamentele in care lucreaza mai putin de 4 angajati;
 - b) departamentul care are numarul maxim de angajati.
19. Sa se afiseze salariatii care au fost angajati în aceeași zi a lunii în care cei mai multi dintre salariați au fost angajați.
20. Să se obțină numărul departamentelor care au cel puțin 15 angajați.
21. Să se obțină codul departamentelor și suma salariilor angajaților care lucrează în acestea, în ordine crescătoare. Se consideră departamentele care au mai mult de 10 angajați și al căror cod este diferit de 30.
22. Sa se afiseze codul, numele departamentului, numarul de angajati si salariul mediu din departamentul respectiv, impreuna cu numele, salariul si jobul angajatilor din acel departament. Se vor afișa și departamentele fără angajați (outer join).
23. Scrieti o cerere pentru a afisa, pentru departamentele avand codul > 80, salariul total pentru fiecare job din cadrul departamentului. Se vor afisa orasul, numele departamentului, jobul si suma salariilor. Se vor eticheta coloanele corespunzator.
24. Care sunt angajatii care au mai avut cel putin doua joburi?
25. Să se calculeze comisionul mediu din firmă, luând în considerare toate liniile din tabel.

Obs: Funcțiile grup ignoră valorile *null*. Prin urmare, instrucțiunea

```
SELECT AVG(commission_pct)
FROM employees;
```

va returna media valorilor pe baza liniilor din tabel pentru care există o valoare diferită de *null*. Astfel, reiese că suma valorilor se împarte la numărul de valori diferite de *null*. Calculul mediei pe baza tuturor liniilor din tabel se poate realiza utilizând funcțiile *NVL*, *NVL2* sau *COALESCE*:

```
SELECT AVG(NVL(commission_pct, 0))
FROM employees;
```

O altă variantă este dată de o cerere de forma:

```
SELECT SUM(commission_pct)/COUNT(*)
FROM employees;
```

VI. [Exerciții – ROLLUP și CUBE]

26. Analizați cele 2 exemple prezentate mai sus, referitor la operatorii *ROLLUP* și *CUBE*.

VII. [Exerciții – DECODE]

27. Scrieți o cerere pentru a afișa job-ul, salariul total pentru job-ul respectiv pe departamente si salariul total pentru job-ul respectiv pe departamentele 30, 50, 80. Se vor eticheta coloanele corespunzător. Rezultatul va apărea sub forma de mai jos:

Job	Dep30	Dep50	Dep80	Total

.....				
.....				

28. Să se creeze o cerere prin care să se afișeze numărul total de angajați și, din acest total, numărul celor care au fost angajați în 1997, 1998, 1999 și 2000. Denumiți capetele de tabel în mod corespunzător.

[Exerciții – subcereri în clauza FROM]

Obs: Subcererile pot apărea în clauza SELECT, WHERE sau FROM a unei cereri. O subcerere care apare în clauza FROM se mai numește *view in-line*.

29. Să se afișeze codul, numele departamentului și suma salariilor pe departamente.

```
SELECT d.department_id, department_name, a.suma
FROM departments d, (SELECT department_id, SUM(salary) suma
                     FROM employees
                     GROUP BY department_id) a
WHERE d.department_id = a.department_id;
```

30. Să se afișeze numele, salariul, codul departamentului și salariul mediu din departamentul respectiv.

31. Modificați cererea anterioară, pentru a determina și listarea numărului de angajați din departamente.

32. Pentru fiecare departament, să se afișeze numele acestuia, numele și salariul celor mai prost plătiți angajați din cadrul său.