

Funcții grup și clauzele **GROUP BY, **HAVING**.**

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. [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 media salariilor.

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

3. Să se modifice problema 2 pentru a se afișa **minimul, maximul, suma și media salariilor** pentru **FIECARE** job.

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

4. Să se afișeze **numărul de angajați** pentru **FIECARE** departament.

```
SELECT COUNT(...), department_id  
FROM _____  
GROUP BY _____;
```

5. Să se determine **numărul de angajați** care sunt șefi. Etichetați coloana "Nr. manageri".
6. Să se afișeze **diferența** dintre **cel mai mare și cel mai mic salariu**. Etichetați coloana "Diferența".

```
SELECT max(salary)-min(salary) Diferenta  
FROM employees;
```

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.

!!!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 au salariul mai mare decât salariul mediu din firmă. Se va sorta rezultatul în ordine descrescătoare a salariilor.

```
SELECT employee_id, first_name, last_name
FROM employees
WHERE salary > (SELECT AVG(salary)
                FROM employees)
ORDER BY salary DESC;
```

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**.

```
SELECT department_id, department_name, MAX(salary)
FROM departments JOIN employees USING(department_id)
GROUP BY department_id, department_name
HAVING MAX(salary) >= 3000;
```

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.

```
SELECT _____
FROM employees
GROUP BY _____;
```

12. Să se afișeze **maximul salariilor medii** pe departamente.

13. Să se obțină **codul**, **titlul** și **salariul mediu** al job-ului pentru care salariul mediu este minim.

14. Să se afișeze **salariul mediu** din firmă doar dacă acesta este mai mare decât 2500. (clauza *HAVING* fără *GROUP BY*)

15. Să se afișeze **suma salariilor** pe departamente și, **în cadrul acestora**, pe job-uri.

```
SELECT department_id, job_id, SUM(salary)
FROM employees
GROUP BY department_id, job_id;
```

16. 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.

a) *SELECT e.department_id, d. department_name, COUNT(*)
FROM departments d JOIN employees e
ON (d.department_id = e.department_id)
WHERE e.department_id IN (SELECT department_id
FROM employees
GROUP BY department_id
HAVING COUNT(*) < 4)
GROUP BY e.department_id, d.department_name;*

Sau:

SELECT e.department_id, d.department_name, COUNT()
FROM employees e JOIN departments d
ON (d.department_id = e.department_id)
GROUP BY e.department_id, d.department_name
HAVING COUNT(*)<4;*

17. Sa se afiseze **salariatii** care au fost angajati în **aceeași zi a lunii** în care cei mai multi dintre salariatii au fost angajati.

18. Să se obțină **numărul departamentelor** care au cel puțin 15 angajați.

19. 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.

20. Care sunt angajatii care **au mai avut cel puțin doua joburi**?

21. 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;
```

IV. [Exerciții – DECODE]

22. 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

.....				
.....				

```
SELECT job_id, SUM(DECODE(department_id, 30, salary)) Dep30,
              SUM(DECODE(department_id, 50, salary)) Dep50,
              SUM(DECODE(department_id, 80, salary)) Dep80,
              SUM(salary) Total
FROM employees
GROUP BY job_id;
```

Metoda 2: (cu subcereri corelate în clauza SELECT)

```
SELECT job_id, (SELECT SUM(salary)
                FROM employees
                WHERE department_id = 30
                AND job_id = e.job_id) Dep30,
              (SELECT SUM(salary)
                FROM employees
                WHERE department_id = 50
                AND job_id = e.job_id) Dep50,
              (SELECT SUM(salary)
                FROM employees
                WHERE department_id = 80
                AND job_id = e.job_id) Dep80,
              SUM(salary) Total
FROM employees e
GROUP BY job_id;
```

23. 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.

V. [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**.

24. Să se afișeze **codul**, **numele departamentului** și **suma salariilor** pe departamente.
25. Să se afișeze **numele**, **salariul**, **codul departamentului** și **salariul mediu** din departamentul respectiv.
26. Modificați cererea anterioară, pentru a determina și **listarea numărului de angajați** din departamente.