Funcţii grup şi clauzele GROUP BY, HAVING.

I. [Funcţii grup şi clauza GROUP BY]

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

Server-ul *Oracle* aplică aceste funcţii fiecărui grup de linii şi returnează un singur rezultat pentru fiecare mulţime.

- Dintre **funcţiile grup** definite în sistemul *Oracle*, se pot enumera: ***AVG***, ***SUM***, ***MAX***, ***MIN***, ***COUNT***, ***STDDEV***, ***VARIANCE*** etc. Tipurile de date ale argumentelor funcţiilor grup pot fi *CHAR*, *VARCHAR2*, *NUMBER* sau *DATE*.
    - Funcţiile *AVG*, *SUM*, *STDDEV* şi *VARIANCE* operează numai asupra valorilor numerice.
    - Funcţiile *MAX* şi *MIN* pot opera asupra valorilor numerice, caracter sau dată calendaristică.

- Absenţa clauzei **GROUP BY** conduce la aplicarea funcţiei grup pe mulţimea tuturor liniilor tabelului.
- Toate funcţiile grup, cu excepţia lui *COUNT*(*), ignoră valorile *null*. ***COUNT*(*expresie*) returnează numărul de linii** pentru care expresia dată nu are valoarea *null*. Funcţia *COUNT* returnează un număr mai mare sau egal cu zero şi nu întoarce niciodată valoarea *null*.
- Când este utilizată clauza *GROUP BY*, *server*-ul sortează implicit mulţimea rezultată în ordinea crescătoare a valorilor coloanelor după care se realizează gruparea.
- În clauza *GROUP BY* a unei cereri se pot utiliza operatorii *ROLLUP* şi *CUBE*. Aceşti operatori sunt disponibili începând cu versiunea *Oracle8i*.
- Expresiile din clauza *SELECT* a unei cereri care conţine opţiunea *GROUP BY* trebuie să reprezinte o proprietate unică de grup, adică fie un atribut de grupare, fie o funcţie de agregare aplicată tuplurilor unui grup, fie o expresie formată pe baza primelor două. Toate expresiile din clauza *SELECT*, cu excepţia funcţiilor de agregare, se trec în clauza *GROUP BY* (unde pot apărea cel mult 255 expresii).

II. [Clauza HAVING]

Opţiunea ***HAVING*** permite restricţionarea grupurilor de linii returnate, la cele care îndeplinesc o anumită condiţie.

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

III. [Exerciţii – funcţii grup şi clauzele GROUP BY, HAVING]

1. a) Functiile grup includ valorile *NULL* in calcule?
   b) Care este deosebirea dintre clauzele *WHERE* şi *HAVING*?

2. Să se afişeze **cel mai mare salariu**, **cel mai mic salariu**, **suma** şi **media salariilor** tuturor angajaţilor. Etichetaţi coloanele Maxim, Minim, Suma, respectiv Media. Sa se rotunjeasca media salariilor.

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

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

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

4. Să se afişeze **numărul de angajaţi** pentru **FIECARE** departament.

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

5. Să se determine **numărul de angajaţi** care sunt şefi. Etichetati coloana “Nr. manageri”.

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

6. Să se afişeze **diferenţa** dintre **cel mai mare si cel mai mic salariu**. Etichetati coloana “Diferenta”.

7. Scrieţi o cerere pentru a se afişa **numele departamentului**, **locaţia**, **numărul de angajaţi** şi **salariul mediu** pentru angajaţii 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 funcţiilor grup (a se vedea ultima observaţie de la punctul I).

8. Să se afişeze **codul** şi **numele angajaţilor** 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 şef**, să se afişeze **codul** său şi **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$. Sortaţi rezultatul în ordine descrescătoare a salariilor.

10. Pentru departamentele in care salariul maxim depăşeşte 3000$, să se obţină **codul**, **numele** acestor departamente şi **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.

```
SELECT MIN(AVG(salary))
FROM     employees
GROUP BY job_id;
```

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

13. Sa se obtina **codul**, **titlul** şi **salariul mediu** al job-ului pentru care salariul mediu este minim.

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

15. Să se afişeze **suma salariilor** pe departamente şi, **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 **aceeaşi zi a lunii** în care cei mai multi dintre salariati au fost angajati.

18. Să se obţină **numărul departamentelor** care au cel puţin 15 angajaţi.

19. Să se obţină **codul departamentelor** şi **suma salariilor** angajaţilor care lucrează în acestea, în ordine crescătoare. Se consideră departamentele care **au mai mult** de 10 angajaţi şi al căror **cod este diferit** de 30.

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

21. Să se calculeze **comisionul mediu** din firmă, luând în considerare **toate** liniile din tabel.

Obs: Funcţiile grup ignoră valorile *null*. Prin urmare, instrucţiunea

```
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 funcţiile *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. [Exerciţii – DECODE]

22. Scrieţi o cerere pentru a afişa **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 afişeze **numărul total de angajaţi** şi, din acest total, numărul celor care au fost angajaţi în 1997, 1998, 1999 si 2000. Denumiti capetele de tabel in mod corespunzator.

V. [Exerciţii – 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 numeşte ***view in-line***.

24. Să se afişeze **codul**, **numele departamentului** şi **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;
```

25. Să se afişeze **numele**, **salariul**, **codul departamentului** si **salariul mediu** din departamentul respectiv.

26. Modificaţi cererea anterioară, pentru a determina şi **listarea numărului de angajaţi** din departamente.

```
SELECT last_name, salary, department_id, SalMediu, NrAng
FROM employees join (SELECT ROUND(AVG(salary)) SalMediu, department_id,
                        count(employee_id) NrAng
                     FROM employees
                     GROUP BY department_id
                     )
   USING (department_id);
```