

Subcereri nesincronizate (necorelate). Gruparea datelor (1)

I. [Obiective]

Prezentarea conceptului de subcerere. Unde pot fi folosite subcererile?
Clasificarea subcererilor.

Introducere in gruparea datelor.

II. [Subcereri]

O **subcerere** este o comandă *SELECT* încapsulată într-o clauză a altei instrucțiuni *SQL*, numită instrucțiune „părinte”. Utilizând subcereri, se pot construi interogări complexe pe baza unor instrucțiuni simple. Subcererile mai sunt numite instrucțiuni *SELECT* imbricate sau interioare.

Subcererea returnează o valoare care este utilizată de către instrucțiunea „părinte”. Utilizarea unei subcereri este echivalentă cu efectuarea a două cereri secvențiale și utilizarea rezultatului cererii interne ca valoare de căutare în cererea externă (principală).

În general, subcererile pot apărea în clauzele *SELECT*, *FROM*, *WHERE*, *HAVING* ale comenzii *SELECT*.

Subcererile sunt de 2 tipuri :

➤ **Necorelate (nesincronizate)**, de forma :

```
SELECT   lista_select
FROM    nume_tabel
WHERE    expresie_operator (SELECT lista_select
                               FROM   nume_tabel);
```

- cererea internă este executată prima și determină o valoare (sau o mulțime de valori);
- cererea externă se execută o singură dată, utilizând valorile returnate de cererea internă.

➤ **Corelate (sincronizate)**, de forma :

```
SELECT nume_coloană_1[, nume_coloană_2 ...]
FROM   nume_tabel_1 extern
WHERE  expresie_operator
        (SELECT nume_coloană_1 [, nume_coloană_2 ...]
```

```
FROM   nume_tabel_2
WHERE  expresie_1 = extern.expresie_2);
```

- cererea externă determină o linie candidat;
- cererea internă este executată utilizând valoarea liniei candidat;
- valorile rezultate din cererea internă sunt utilizate pentru calificarea sau descalificarea liniei candidat;
- pașii precedenți se repetă până când nu mai există linii candidat.

Observație: operator poate fi:

- *single-row operator* (>, =, >=, <, <>, <=), care poate fi utilizat dacă subcererea returnează o singură linie;
- *multiple-row operator* (IN, ANY, ALL), care poate fi folosit dacă subcererea returnează mai mult de o linie.

Operatorul *NOT* poate fi utilizat în combinație cu *IN*, *ANY* și *ALL*.

III. [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. Aceste funcții 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*.

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).

IV. [Clauza HAVING]

- Clauza *HAVING* a comenzii *SELECT* 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*.

V. [Exercitii - subcereri necorelate]

1. Folosind subcereri, să se afișeze numele și data angajării pentru salariații care au fost angajați după Gates.

```
SELECT last_name, hire_date
FROM employees
WHERE hire_date > (SELECT hire_date
                   FROM employees
                   WHERE INITCAP(last_name)='Gates');
```

2. Folosind subcereri, scrieți o cerere pentru a afișa numele și salariul pentru toți colegii (din același departament) lui Gates. Se va exclude Gates.

```
SELECT last_name, salary
FROM employees
WHERE department_id IN (SELECT department_id
                       FROM employees
                       WHERE LOWER(last_name)='gates')
AND LOWER(last_name) <> 'gates';
```

În care caz nu se poate înlocui "=" cu "IN"??

3. Folosind subcereri, să se afișeze numele și salariul angajaților conduși direct de președintele companiei (acesta este considerat angajatul care nu are manager).
4. Scrieți o cerere pentru a afișa numele, codul departamentului și salariul angajaților al căror cod de departament și salariu coincid cu codul departamentului și salariul unui angajat care câștigă comision.

Soluție:

```
SELECT last_name, department_id, salary
```

```
FROM employees
WHERE (department_id, salary) IN
(
    SELECT department_id, salary
    FROM employees
    WHERE commission_pct IS NOT NULL
);
```

5. Rezolvați problema 6 din laboratorul precedent utilizând subcereri: să se afișeze codul, numele și salariul tuturor angajaților care câștigă mai mult decât salariul mediu pentru job-ul corespunzător și lucrează într-un departament cu cel puțin unul dintre angajații al căror nume conține litera “t”. Vom considera salariul mediu al unui job ca fiind egal cu media aritmetică a limitelor sale admise (specificate în coloanele *min_salary*, *max_salary* din tabelul JOBS).

Soluție:

```
SELECT e.employee_id,e.last_name,e.salary
FROM employees e
WHERE e.salary >
(
    SELECT (j.min_salary+j.max_salary)/2
    FROM jobs j
    WHERE j.job_id=e.job_id
)
AND e.job_id IN
(
    SELECT job_id
    FROM employees m
    WHERE e.department_id=m.department_id
    AND LOWER(m.last_name) LIKE '%t%'
);
```

6. Scrieti o cerere pentru a afișa angajații care câștigă mai mult decât oricare funcționar (job-ul conține șirul “CLERK”). Sortați rezultatele după salariu, în ordine descrescătoare.

Ce rezultat este returnat dacă se înlocuiește “ALL” cu “ANY”?

Soluție:

```
SELECT *
FROM employees e
WHERE salary > ALL
(
    SELECT salary
    FROM employees
    WHERE job_id LIKE '%CLERK'
);
```

7. Scrieți o cerere pentru a afișa numele, numele departamentului și salariul angajaților care nu câștigă comision, dar al căror șef direct câștigă comision.
8. Să se afișeze numele, departamentul, salariul și job-ul tuturor angajaților al căror salariu și comision coincid cu salariul și comisionul unui angajat din Oxford.

9. Să se afișeze numele angajaților, codul departamentului și codul job-ului salariaților al căror departament se află în Toronto.

VI. [Exercitii – gruparea datelor]

10. a) Funcțiile grup includ valorile *NULL* în calcule?
b) Care este deosebirea dintre clauzele *WHERE* și *HAVING*?
11. 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.
12. Să se afișeze minimul, maximul, suma și media salariilor pentru fiecare job.
13. Să se afișeze numărul de angajați pentru fiecare job.
14. Să se determine numărul de angajați care sunt șefi. Etichetați coloana “Nr. manageri”.
Observație: Este necesar cuvântul cheie *DISTINCT*. Ce obținem dacă îl omitem?
15. Să se afișeze diferența dintre cel mai mare și cel mai mic salariu mediu pe departamente. Etichetați coloana “Diferența”.
16. 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.
Observație: 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).
17. Să se afișeze codul și numele angajaților care câștigă mai mult decât salariul mediu din firmă. Se va sorta rezultatul în ordine descrescătoare a salariilor.
18. Pentru fiecare șef, să se afișeze codul său și salariul celui mai puțin plătit subordonat al său. 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.
19. Pentru departamentele în care salariul maxim depășește 3000\$, să se obțină codul, numele acestor departamente și salariul maxim pe departament.
20. 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.
21. Să se afișeze codul, numele departamentului și suma salariilor pe departamente.
22. Să se afișeze maximul salariilor medii pe departamente.

- 23.** Să se obțină codul, titlul și salariul mediu al job-ului pentru care salariul mediu este minim.
- 24.** Să se afișeze salariul mediu din firmă doar dacă acesta este mai mare decât 2500.
(clauza *HAVING* fără *GROUP BY*)