UNIUNEA EUROPEANĂ

GUVERNUL ROMÂNIEI

Instrumente Structurale
2007-2013

Platformă de e-learning și curriculă e-content pentru învățământul superior tehnic

Proiect nr. 154/323 cod SMIS – 4428 cofinanțat de prin Fondul European de Dezvoltare Regională “Investiții pentru viitorul dumneavoastră”.

Programul Operaţional Sectorial Creşterea Competitivităţii Economice - POS CCE

UNIUNEA EUROPEANĂ

GUVERNUL ROMÂNIEI

Instrumente Structurale
2007-2013

Platformă de e-learning și curriculă e-content pentru învățământul superior tehnic

Baze de date

23. Subcereri care întorc mai multe valori

Introducere

- Subcererile care întorc mai multe valori pot fi clasificate astfel:
- **Subcereri ascunse**
  - **Subcereri care întorc o coloană**
  - **Subcereri care întorc o linie**
  - **Subcereri care întorc mai multe linii**
- **Subcereri corelate**
- **Subcereri pe un tabel temporar**
- **Subcereri pe clauza HAVING**
- **Subcereri pe clauza ORDER BY**

Subcereri care întorc o coloană

- Sunt subcereri care întorc mai multe valori şi folosesc operatorii ***IN*** şi ***NOT IN*** într-o construcție ca mai jos:
- **SELECT column 1, column 2, ...**

  FROM table 1

  WHERE column IN [NOT IN]

  (SELECT column

  FROM table 2

  WHERE condition);
- În cazul în care am folosi operatorul ‘=‘ în locul operatorului ***IN,*** s-ar genera o eroare pentru că subcererea întoarce mai mult de o valoare (ORA-01427). Trebuie specificat că operatorul ***IN*** se poate folosi în locul operatorului’**=** ‘, dar invers este greşit.

Exemple:

- Dacă vrem să aflăm salariile angajaților care au funcții similare funcțiilor aferente departamentului 20, folosim următoarea comandă :
- SQL>SELECT nume,functie,salariu

  FROM angajati

  WHERE functie IN

  (SELECT DISTINCT functie FROM angajati

  WHERE id_dep=20);
- Pentru a selecta toți angajații care nu s-au angajat în lunile decembrie, ianuarie şi februarie, folosim comanda:
- SQL>SELECT id_ang,nume,functie,data_ang FROM angajati

  WHERE id_ang NOT IN

  (SELECT DISTINCT id_ang FROM angajati WHERE

  to_char(data_ang,'MON') IN ('DEC','JAN','FEB'));

Subcereri care întorc o linie

- Sunt subcereri care întorc mai multe coloane, dar o singură linie şi folosesc operatorii ***IN*** şi ***NOT IN*****,** într-o construcție ca mai jos:
- **SELECT column 1, column 2, ...**

  FROM table 1

  WHERE (column 1,column 2...) IN [NOT IN]

  (SELECT column 1,column 2..

  FROM table 2

  WHERE condition);

- Numărul şi tipul coloanelor, din clauza *WHERE,* trebuie să coincidă cu cele din subcerere.

Exemplu:

- Dacă dorim să aflăm care angajați, din departamentul 30, au aceeaşi funcție ca cea a lui Tache Ionuț, folosim următoarea comandă:
- SQL>SELECT id_dep,nume,functie,data_ang FROM angajati

  WHERE (id_dep,functie) IN

  (SELECT id_dep,functie FROM angajati

  WHERE nume='TACHE IONUT');
- În acest caz, subcererea întoarce o singură linie, deoarece Tache are o singură funcție şi este angajat la un singur departament. În eventualitatea că pot exista mai mulți angajați cu acelaşi nume, se va folosi clauza *DISTINCT* în subcerere.

Subcereri care întorc mai multe linii

- Aceste subcereri au o construcție asemănătoare ca la punctul precedent, dar întorc mai multe linii cu mai multe coloane, drept pentru care se mai numesc şi subcereri care întorc un tabel. Trebuie specificat că în locul coloanelor se pot folosi şi expresii. Numărul de expresii specificate în clauza *WHERE* trebuie să coincidă cu numărul de expresii din subcecere şi să fie de acelaşi tip.
- **SELECT expr 1,... expr n**

  FROM table 1

  WHERE (expr 1,...expr k) IN [NOT IN]

  (SELECT expr 1, ...expr k

  FROM table 2

  WHERE condition);

Exemplu:

- Pentru a afla persoanele care au venit maxim pe fiecare departament, folosim următoarea comandă:
- **SQL> SELECT id_dep,nume,functie,salariu+nvl(comision,0) venit**

```
FROM angajati
WHERE (id_dep,salariu+nvl(comision,0)) IN
    (SELECT id_dep,max(salariu+nvl(comision,0))
        venit_max
     FROM angajati GROUP BY id_dep);
```

Subcereri corelate

- Subcererile corelate se execută o dată pentru fiecare linie candidat, prelucrată de cererea principală şi la execuție folosesc cel putin o valoare dintr-o coloană din cererea principală. O subcerere corelată se join-ează cu cererea exterioară, prin folosirea unei coloane a cererii exterioare în clauza predicatului cererii interioare. Construcția unei subcereri corelate este următoarea:

- **SELECT expr 1,... expr n**

  FROM table 1

  WHERE (expr 1,...expr k) IN [NOT IN]

  (SELECT expr 1, ...expr k

  FROM table 2

  WHERE table 2.expr x = table 1.expr y [AND,OR] ..);

Exemple:

- Dacă vrem să aflăm persoanele care au salariul peste valoarea medie a salariului pe departamentul din care fac parte, folosim comanda următoare:
- SQL> SELECT a.id_dep,a.nume,a.functie,a.salariu

```
FROM angajati a WHERE a.salariu >
        (SELECT avg(salariu) salariu_mediu FROM angajati b
        WHERE b.id_dep=a.id_dep )
ORDER By id_dep;
```

- Dacă vrem să indexăm salariile cu 10% şi comisioanele angajaților să le aducem la nivelul comisionului mediu pe fiecare departament, numai pentru persoanele angajate înainte de 1-iun-1981, folosim următoarea comandă:
- SQL> UPDATE angajati a SET (salariu, comision) =

```
        (SELECT AVG(salariu) * 1.1, AVG(comision) FROM angajati b
        WHERE b.id_dep = a.id_dep)
WHERE data_ang<= '1-JUN-1981';
```

Observații

- În cazul unei subcereri ascunse, comanda *SELECT* din subcerere rulează prima şi se execută o singură dată, întorcând valori ce vor fi folosite de cererea principală.
- Pentru o cerere corelată, subcererea se execută de mai multe ori, câte o dată pentru fiecare linie prelucrată de cererea principală, deci cererea interioară este condusă de cererea exterioară.
- Paşii de execuție ai unei subcereri corelate sunt:
- se obține linia candidat prin procesarea cererii exterioare;
- se execută cererea interioară corelată cu valoarea liniei candidat;
- se folosesc valorile rezultate din cererea interioară, pentru a prelucra linia candidat;
- se repetă până nu mai rămâne nicio linie candidat.
- Trebuie specificat că, deşi subcererea corelată se execută repetat, aceasta nu înseamnă ca subcererile corelate sunt mai puțin eficiente decât subcererile ascunse.

Subcereri pe un tabel temporar

- Aceste subcereri se întâlnesc în cazul în care se foloseşte o subcerere la nivelul clauzei *FROM* din cererea principală.
- **SELECT t1.column1,t1.column2,.. t2.expr 1,... t2.expr k**

  FROM table t1,

  (SELECT expr 1, ...expr n

  FROM table

  WHERE conditions) t2

  WHERE conditions

  ORDER BY expr;

Exemplu:

- Pentru a afla salariul maxim pe fiecare departament, se poate face o construcție ca mai jos, unde subcererea întoarce un tabel temporar cu salariul maxim pe fiecare departament:
- SQL>SELECT b.id_dep,a.den_dep,b.sal_max_dep

```
FROM departamente a,  (SELECT  id_dep, max(salariu)
        sal_max_dep FROM angajati  GROUP BY id_dep) b
WHERE a.id_dep=b.id_dep
ORDER BY b.id_dep;
```

Subcereri pe clauza HAVING

- Aceste subcereri sunt prinse în clauza *HAVING* într-o construcție ca cea de mai jos :
- **SELECT expr 1,... expr n**

  FROM table 1

  WHERE conditions

  HAVING expr, (operator)

  (SELECT expr 1, ...expr k

  FROM table 2

  WHERE conditions)

  GROUP BY expresion;
- Trebuie reținut că o clauză *HAVING* se foloseşte totdeauna împreună cu clauza *GROUP BY* şi într-o clauză *HAVING* se pot folosi funcții de grup.

Exemple:

- Pentru a afla ce departamente au cei mai mulți angajați pe aceeaşi funcție, folosim următoarea comandă:
-

```
SQL>SELECT d.den_dep, a.functie,count(a.id_ang) nr_ang
     FROM  angajati a, departamente d
    WHERE  a.id_dep = d.id_dep
    HAVING count(a.id_ang) = (SELECT max(count(id_ang))
                              FROM angajati GROUP BY id_dep, functie)
    GROUP BY d.den_dep, a.functie;
```

- Dacă vrem să aflăm ce angajat are salariul maxim în firmă, putem face o construcție ca mai jos:
-

```
SQL> SELECT a.nume,a.functie,a.data_ang,a.salariu
     FROM angajati a,
       (SELECT id_dep,max(salariu) sal_max_dep FROM angajati  GROUP BY id_dep) b
     HAVING a.salariu=max(b.sal_max_dep)
     GROUP BY a.nume,a.functie,a.data_ang,a.salariu
     ORDER BY a.nume;
```

Subcereri pe clauza ORDER BY

- Se poate folosi o subcerere şi pe clauza ORDER BY, dar numai pentru cereri corelate şi trebuie să returneze totdeauna o singură valoare, având următoarea construcție:
- **SELECT expr 1,... expr n,**

  FROM table 1

  WHERE conditions

  ORDER BY

  (SELECT expr FROM table 2

  WHERE table 2.expr x = table 1.expr y [AND,OR] ..)

  ASC/DESC

Exemplu:

- Dacă vrem să facem o listă cu angajații din departamentele 10 şi 20 şi să-i ordonăm descrescător după număr, pe fiecare departament, folosim comanda următoare:
-

```
SQL> SELECT id_dep,nume,functie FROM angajati a
        WHERE id_dep in (10,20)
        ORDER BY
           (SELECT count(*) FROM angajati b
            WHERE  a.id_dep=b.id_dep) DESC
```