SUBCERERI

O subcerere este o declaratie SELECT care este inclusa in alta declaratie SELECT, utilizata pentru a obtine rezultate intermediare.

Exemplu:

```
SELECT column1, column2, ...
FROM table
WHERE column =  (SELECT column
                 FROM table
                 WHERE condition)
```

Subcererea este adesea referita ca un subSELECT sau ca un SELECT interior; subcererea se executa prima si rezultatul sau este folosit pentru a completa conditia cererii principale (externe). Folosirea subcererilor permite construirea de comenzi puternice pornind de la unele simple.

Subcereri care intorc o linie

Pentru a gasi angajatul care cistiga salariul minim din companie (salariul minim este o cantitate necunoscuta), trebuie parcursi doi pasi:

Gasirea salariului minim:

```
SELECT MIN(SALARY)
FROM EMPLOYEES;
```

Gasirea angajatului care cistiga salariul minim:

```
SELECT FIRST_NAME, JOB_ID, SALARY
FROM EMPLOYEES
WHERE SALARY = (cel mai mic salariu care este cunoscut)
```

Putem combina cele doua cereri:

```
SELECT FIRST_NAME, JOB_ID, SALARY
FROM EMPLOYEES
WHERE SALARY = (SELECT MIN(SALARY)
                FROM EMPLOYEES);
```

Prelucrarea subcererilor

O declarartie SELECT poate fi considerata ca un bloc. Exemplul de mai sus consta din doua blocuri - cererea exterioara si cererea interioara.

Declaratia SELECT interioara este executata prima, producind un rezultat al cererii: 800. Blocul extern este apoi prelucrat si foloseste valoarea intoarsa de cererea interioara pentru a completa conditia de cautare. Cererea in final va arata in felul urmator:

```
SELECT FIRST_NAME, SALARY, DEPARTMENT_ID
FROM EMPLOYEES
WHERE SALRY = 800;
```

In exemplul de mai sus, cererea interna intorce o singura valoare. Cind o subcerere intoarce doar o valoare, se poate utiliza un operator logic sau un operator de comparatie. De exemplu: =, <, >, <=, etc.

Pentru a gasi toti angajatii ce au aceeasi functie ca BLAKE, vom introduce:

```
SELECT FIRST_NAME, JOB_ID
FROM EMPLOYEES
WHERE JOB_ID = (SELECT JOB_ID
                FROM EMPLOYEES
                WHERE FIRST_NAME = 'BLAKE');
```

Cererea interioara intoarce functia lui BLAKE, care este folosita in WHERE din cererea principala.

Subcereri care intorc mai mult de o valoare

Urmatoarea cerere gaseste angajatii care au salariul egal cu salariul minim din fiecare departament:

```
SELECT FIRST_NAME, SALARY, DEPARTMENT_ID
FROM EMPLOYEES
WHERE SALARY IN
                (SELECT MIN(SALARY)
                 FROM EMPLOYEES
                 GROUP BY DEPARTMENT_ID);
```

cererea interioara are clauza GROUP BY. Aceasta inseamna ca va intoarce mai mult decit o valoare. Deci este nevoie sa folosim un operator SQL (in acest caz, operatorul IN, deoarece rezulta o lista de valori).

Deoarece se compara doar valorile salariilor, cererea exterioara poate intoarce mai multe valori pentru un acelasi departament. Prin urmare, cererea poate fi rescrisa pentru a gasi salariile minime din fiecare departament.

Compararea a mai multor valori

Urmatoarea cerere gaseste angajatii care cistiga salariul cel mai mic din departamentul lor:

```
SELECT FIRST_NAME, SALARY, DEPARTMENT_ID
FROM EMPLOYEES
WHERE (SALARY, DEPARTMENT_ID) IN
                          (SELECT MIN(SALARY), DEPARTMENT_ID
                           FROM EMPLOYEES
                           GROUP BY DEPARTMENT_ID);
```

Cererea de mai sus compara o pereche de coloane.

Observatie : coloanele din partea stinga a conditiei de cautare sint intre paranteze si fiecare coloana este separata printr-o virgula.

Coloanele listate in clauza SELECT a subcererii trebuie sa se potriveasca in numar si tip de date cu coloanele cu care ele sint comparate in cererea externa.

Erori intilnite

Cind o subcerere intoarce mai mult decit o valoare si este folosit un operator de comparatie pentru o singura valoare, SQL*Plus refuza cererea:

```
SELECT FIRST_NAME, SALARY, DEPARTMENT_ID
FROM EMPLOYEES
WHERE SALARY = (SELECT MIN(SALARY)
                FROM EMPLOYEES
                GROUP BY DEPARTMENT_ID);
```

SAU

```
SELECT FIRST_NAME, JOB_ID
FROM EMPLOYEES
WHERE JOB_ID = (SELECT JOB_ID
                FROM EMPLOYEES
                WHERE ENAME = 'SMITHE');
```

Operatorii SOME/ANY, ALL

Operatorii ANY sau ALL pot fi folositi pentru subcererile care intorc mai mult de o valoare. Ei sint folositi in clauzele WHERE sau HAVING in legatura cu operatorii logici (=, !=, <, >, >=, <=).
ANY (sau sinonimul sau SOME) compara o valoare cu fiecare valoare intoarsa de o subcerere.
Pentru a afisa angajatii care cistiga mai mult decit cel mai mic salariu din departamentul 30, introducem :

```
SELECT FIRST_NAME, SALARY, JOB_ID, DEPARTMENT_ID
FROM EMPLOYEES
WHERE SALARY > SOME (SELECT DISTINCT SALARY
                     FROM EMPLOYEES
                     WHERE DEPARTMENT_ID = 30)
ORDER BY SALARY DESC;
```

Cererea principala intoarce angajatii care cistiga un salariu mai mare ca salariul minim din departamentul 30. Asa ca '> ANY' inseamna mai mare ca minim. '=ANY' este echivalent cu IN.

Cind se foloseste SOME/ANY, DISTINCT este frecvent folosit pentru a impiedica sa se selecteze liniile de mai multe ori.

ALL compara o valoare cu fiecare valoare intoarsa de o subcerere. Urmatoarea cerere gaseste angajatii care cistiga mai mult ca fiecare angajat din departamentul 30:

```
SELECT FIRST_NAME, SALARY, JO_ID, DEPARTMENT_ID
FROM EMPLOYEES
WHERE SALARY > ALL (SELECT DISTINCT SALARY
                    FROM EMPLOYEES
                    WHERE  DEPARTMENT_ID = 30)
ORDER BY SALARY DESC;
```

Cererea intoarce acei angajati ai caror salariu este mai mare ca salariul maxim din departamentul 30, prin urmare mai mare ca fiecare salariu din departament.

Operatorul NOT poate fi folosit cu IN, ANY sau ALL.

Ordonarea datelor cu subcereri

Nu poate exista o clauza ORDER BY intr-o subcerere.

Regula este ca poate exista doar o singura clauza ORDER BY pentru o declaratie SELECT si, daca este specificata, trebuie sa fie ultima clauza din comanda SELECT.

Imbricarea subcererilor

Subcererile pot fi imbricate (folosite in interiorul unei subcereri):

Afisati numele, functia si data angajarii pentru angajatii al caror salariu este mai mare ca cel mai mare salariu din orice departament de vinzari.

```
SELECT FISRT_NAME, JOB_ID, HIREDATE, SALARY
FROM EMPLOYEES
```

```
WHERE SALARY > (SELECT MAX(SALARY)
                FROM EMPLOYEES
                WHERE DEPARTMENT_ID = (SELECT DEPARTMENT_ID
                                       FROM DEPARTMENTS
                                       WHERE DEPARTMENT_NAME = 'SALES'));
```

Limitele de imbricare

Limita nivelelor de imbricare pentru o subcerere este 255.

Reguli de scriere a cererilor

- cererea interioara trebuie sa fie inclusa intre paranteze si trebuie sa fie in partea dreapta a conditiei.
- subcererile nu pot avea clauza ORDER BY.
- clauza ORDER BY apare la sfirsitul declaratiei SELECT principale.
- coloanele multiple din lista din SELECT a cererii interioare trebuie sa corespunda in tip si numar coloanelor din cererea exterioara.
- pot fi folositi operatorii logici si SQL si ANY si ALL.
- subcererile pot
    - intoarce una sau mai multe linii;
    - intoarce una sau mai multe coloane;
    - folosi GROUP BY sau functii de grup;
    - fi folosite inlantuite cu predicate multiple AND sau OR in aceesi cerere externa.
    - uni tabele.
    - recupera dintr-o tabela diferita de cea a cererii exterioare.
    - apare in declaratii SELECT, UPDATE, DELETE, INSERT, CREATE TABLE.
    - fi corelate cu o cerere exterioara.
    - folosi operatori de multimi.

Subcereri corelate

O subcerere corelata este o subcerere care este executata o data pentru fiecare linie candidat considerata de cererea principala si care la executie foloseste o valoare dintr-o coloana din cererea exterioara. Aceasta determina ca subcererea corelata sa fie prelucrata intr-un mod diferit de subcererea obisnuita.

O subcerere corelata este identificata prin folosirea unei coloane a cererii exterioare in clauza predicatului cererii interioare.

Cererea interioara este dirijata de cererea externa.

Pasii de executie ai unei subcereri corelate :

1. Se obtine linia candidat. (de cererea exterioara)
2. Se executa cererea interioara folosind valoarea liniei candidat.
3. Se folosesc valorile rezultate din cererea interioara pentru a pastra sau pentru a inlatura linia candidat.
4. Se repeta pina nu mai ramine nici o linie candidat.

Putem folosi o subcerere corelata pentru a gasi angajatii care cistiga un salariu mai mare ca salariul mediu al departamentului lor:

```
SELECT FIRST_NAME, LAST_NAME, SALARY, E.DEPARTMENT_ID
FROM EMPLOYEES E
WHERE SALARY > (SELECT AVG(SALARY)
             FROM EMPLOYEES
             WHERE DEPARTMENT_ID = E.DEPARTMENT_ID)
ORDER BY DEPARTMENT_ID;
```

Putem observa ca este o cerere corelata pentru ca am folosit o coloana din SELECT-ul extern in clauza WHERE din SELECT-ul interior.

Observati ca alias-ul este necesar doar pentru a indeparta ambiguitatea pentru numele coloanelor.

Sa analizam exemplul de mai sus folosind tabela EMP :

Cererea principala

1. Se selecteaza prima linie candidat – PRESUPUNEM CA Smith din departamentul 20 cistiga 800.
2. EMPLOYEES in clauza FROM are alias-ul E care obtine coloana DEPARTMENT_ID referita in clauza WHERE a cererii interioare.
3. Clauza WHERE compara 800 cu valoarea intoarsa de cererea interioara.

   Cererea interioara
4. Calculeaza AVG(SALARY) pentru departamentul angajatului.
5. Valoarea departamentului din clauza WHERE este departamentul candidatului (E.DEPARTMENT_ID), valoare transmisa cererii interioare din coloana DEPARTMENT_ID a cererii exterioare.
6. AVG(SALARY) pentru departamentul lui Smith - 20 - este 2175.
7. Linia candidat nu indeplineste conditia, asa ca este indepartata.
8. Se repeta de la pasul 1 pentru urmatoarea linie candidat; ALLEN din de- partamentul 30 cistiga 1600

Selectia liniilor candidat continua cu verificarea conditiei ce apare in re- zultatul cererii.

Operatorul EXISTS

Operatorul EXISTS este frecvent folosit cu subcererile corelate. El testeaza daca o valoare exista (NOT EXISTS specifica daca ceva nu exista). Daca valoarea exista se intoarce TRUE; daca valoarea nu exista se intoarce FALSE.

Pentru a gasi angajatii ce au cel putin un subordonat, introducem:

```
SELECT EMPLOYEE_ID, FIRST_NAME, JOB_ID, DEPARTMENT_ID
FROM EMPLOYEES E
WHERE EXISTS (SELECT EMPLOYEE_ID
              FROM EMPLOYEES P
              WHERE P.MANAGER_ID = E.EMPLOYEE_ID)
ORDER BY EMPLOYEE_ID;
```

Sa gasim toti angajatii al caror departament nu este in tabela DEPT:

```
SELECT EMPLOYEE_ID, FIRST_NAME, DEPARTMENT_ID
FROM EMPLOYEES E
WHERE NOT EXISTS (SELECT DEPARTMENT_ID
                  FROM DEPARTMENTS D
                  WHERE D.DEPARTMENT_ID = E.DEPARTMENT_ID);
```

De ce sa folosim o subcerere corelata ?

Subcererea corelata este un mod de a 'citi' fiecare linie din tabela si de a compara valorile din fiecare linie cu datele inrudite. Adica, o subcerere corelata este folosita pentru a raspunde la intrebari cu mai multe subpuncte al caror raspuns depinde de valoarea din fiecare linie din cererea externa.

NOT EXISTS contra NOT IN

Desi intr-o subcerere o operatie NOT IN poate fi la fel de eficienta ca si NOT EXISTS, NOT EXISTS este mult mai sigur daca subcererea intoarce niste valori NULL, fata de NOT IN pentru care conditia se evalueaza la FALSE cind in lista de comparatii sint incluse valori NULL.

Considerind urmatoarea cerere care gaseste angajatii ce nu au nici un subordonat:

```
SELECT FIRST_NAME, JOB_ID
FROM EMPLOYEES
WHERE EMPLOYEE_ID NOT IN (SELECT MANAGER_ID
                    FROM EMPLOYEES);
```

Nici o linie nu va fi intoarsa de cererea de mai sus, deoarece coloana MANAGER_ID contine o valoare NULL.

Cererea corecta este:

```
SELECT FIRST_NAME, JOB_ID
 FROM EMPLOYEES E
 WHERE NOT EXISTS (SELECT MANAGER_ID
                   FROM EMPLOYEES
                   WHERE MANAGER_ID = E.EMPLOYEE_ID);
```

Tema

1. Gasiti angajatii care cistiga cel mai mare salariu pentru fiecare tip de functie. Sortati in ordinea descrescatoare a salariului.
2. Gasiti angajatii care cistiga salariul minim pentru functia lor. Afisati rezultatul in ordine crescatoare a salariului.
3. Gasiti cei mai recenti angajati din fiecare departament. Ordonati dupa data angajarii.
4. Afisati urmatoarele detalii pentru orice angajat care cistiga un salariu mai mare ca media pentru departamentul lor. Sortati dupa numarul departamentului.
5. Care sint primii trei angajati in functie de salariul cistigat? Afisati numele lor si salariul.
6. In ce an s-au angajat cei mai multi in companie? Afisati anul si numarul angajatilor.
7. Modificati intrebarea 4 pentru a afisa si salariul mediu pentru departament.
8. Scrieti o subcerere care afiseaza o '*' linga linia celui mai recent angajat. Afisati ENAME, HIREDATE si coloana (maxdate) completata cu '*'.