UNIUNEA EUROPEANĂ

GUVERNUL ROMÂNIEI

Instrumente Structurale
2007-2013

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

Baze de date 1

7. Subcereri în SQL (1)

SUBCERERI

- O subcerere este o cerere SELECT inclusă într-o altă cerere SQL.
- Astfel de construcţii se folosesc în cazul în care rezultatul dorit nu se poate obţine cu o singură parcurgere a datelor.
- Exemplu: pentru a afla cine este studentul cu cel mai mare punctaj sunt necesare două parcurgeri ale tabelei STUD:
  - prima calculează punctajul maxim iar ulterior
  - a doua afişează datele dorite despre studentul sau studenţii cu acel punctaj.

STUD

MATR	NUME	AN	GRUPA	DATAN	LOC	TUTOR	PUNCTAJ	CODS
1456	GEORGE	4	1141A	12-MAR-82	BUCURESTI		2890	11
1325	VASILE	2	1122A	05-OCT-84	PITESTI	1456	390	11
1645	MARIA	3	1131B	17-JUN-83	PLOIESTI		1400	11
3145	ION	1	2112B	24-JAN-85	PLOIESTI	3251	1670	21
2146	STANCA	4	2141A	15-MAY-82	BUCURESTI		620	21
3251	ALEX	5	2153B	07-NOV-81	BRASOV		1570	21
2215	ELENA	2	2122A	29-AUG-84	BUCURESTI	2146	890	21
4311	ADRIAN	3	2431A	31-JUL-83	BUCURESTI		450	24
3514	FLOREA	5	2452B	03-FEB-81	BRASOV		3230	24
1925	OANA	2	2421A	20-DEC-84	BUCURESTI	4311	760	24
2101	MARIUS	1	2412B	02-SEP-85	PITESTI	3514	310	24
4705	VOICU	2	2421B	19-APR-84	BRASOV	4311	1290	24

SPEC si BURSA

CODS	NUME	DOMENIU
11	MATEMATICA	STIINTE EXACTE
21	GEOGRAFIE	UMANIST
24	ISTORIE	UMANIST

TIP	PMIN	PMAX	SUMA
FARA BURSA	0	399	
BURSA SOCIALA	400	899	100
BURSA DE STUDIU	900	1799	150
BURSA DE MERIT	1800	2499	200
BURSA DE EXCEPTIE	2500	9999	300

PARCURGERI

- Cum o cerere SELECT specifică o singură parcurgere a datelor rezultă că pentru rezolvarea problemei sunt necesare două astfel de cereri, prima (subcererea) furnizând datele necesare pentru a doua (cererea principală).

SUBCERERILE APAR:

- în expresiile logice din clauzele WHERE şi HAVING
- în clauza ORDER BY a unei cereri SELECT; valoarea returnată de subcerere pentru fiecare linie a rezultatului va determina ordinea de afişare a acestora.
- în clauza SELECT; valoarea returnată de subcerere va fi prezentă în rezultatul final.
- în clauza FROM; în acest caz ele sunt asimilate unor tabele temporare din care se calculează rezultatul cererii care le include.

SUBCERERI IN EXPRESII LOGICE

- Rezultatul unui SELECT, dacă este nevid, este întotdeauna o tabelă. Din punct de vedere al modului de folosire al unei subcereri există însă diferenţe în funcţie de forma rezultatului acesteia:

1. Subcereri care întorc o singură valoare
2. Subcereri care întorc o coloană
3. Subcereri care întorc o tabelă.

- În continuare este prezentat modul de utilizare pentru fiecare tip în parte

O VALOARE

- În acest caz valoarea întoarsă de subcerere poate fi folosită ca oricare alta în comparaţii care includ operatorii obişnuiţi: <, <=, >, >=, = şi <>.
- Exemplu: studentul cu cel mai mare punctaj: subcererea întoarce valoarea maximă a punctajului din tabela STUD iar cererea care o include numele studentului sau studenţilor care au acel punctaj şi valoarea acestuia:

```
SELECT NUME, PUNCTAJ
FROM STUD
WHERE PUNCTAJ =
          (SELECT MAX(PUNCTAJ) FROM STUD)
```

REZULTAT

- Rezultatul obţinut este:

```
NUME          PUNCTAJ
----------    -------
FLOREA           3230
```

REGULI PENTRU SUBCERERI (1)

- Subcererea trebuie să fie întotdeauna în partea dreaptă a comparaţiei, ca în exemplul de mai sus.
- Subcererea se pune obligatoriu între paranteze.
- Deoarece rezultatul este folosit pentru calculele cererii principale, ordinea valorilor returnate nu este importantă. De aceea subcererile nu pot conţine clauza ORDER BY.

REGULI PENTRU SUBCERERI (2)

- Dacă subcererea întoarce mai multe linii se semnalează eroarea *ORA-01427: single-row subquery returns more than one row.*
- Dacă subcererea nu întoarce nici o linie, comparaţia în care e implicată se evaluează la FALS.
- O cerere poate conţine una sau mai multe subcereri, acestea putând fi pe acelaşi nivel sau incluse una în alta.

ALTI OPERATORI

- În afară de operatorii uzuali de comparaţie, în cazul acestui tip de subcereri se pot folosi şi operatorii BETWEEN, LIKE şi IS NULL.
- Exemplele următoare reprezintă cereri valide conţinând şi BETWEEN sau LIKE. Folosirea lui IS NULL pentru o subcerere este relevantă doar în cazul subcererilor corelate prezentate într-un alt subcapitol.

SUBCERERI IN BETWEEN

- Afişarea numelui şi punctajului pentru studenţii având un punctaj egal cu cel mediu +/- 30%. Se foloseşte operatorul BETWEEN având ca parametri două subcereri:

```
SELECT NUME, PUNCTAJ
FROM STUD
WHERE PUNCTAJ BETWEEN
   (SELECT AVG(PUNCTAJ) FROM STUD)*0.7 AND
   (SELECT AVG(PUNCTAJ) FROM STUD)*1.3;
```

SUBCERERI IN LIKE

- Afişarea aceloraşi date ca mai sus pentru studenţii având un nume care nu începe cu aceeaşi literă cu a studentului cu punctaj maxim:

```
SELECT NUME, PUNCTAJ
FROM STUD
WHERE NUME NOT LIKE SUBSTR(
  (SELECT NUME FROM STUD
    WHERE PUNCTAJ = (SELECT MAX(PUNCTAJ)
   FROM STUD)
 ) , 1, 1) || '%';
```

SUBCERERI IN LIKE (2)

- Există două niveluri de imbricare:
- Subcererea de nivel 2 întoarce valoarea maximă a punctajului.
- Subcererea de nivel 1 întoarce numele studentului cu acel punctaj.
- În cererea principală este decupată prima literă a acestui nume, folosind funcţia SUBSTR(rezultat, 1, 1) şi este concatenată cu caracterul % pentru a forma un şablon folosit apoi de condiţia NOT LIKE.

ERORI (1)

- Subcererea întoarce mai multe valori. În acest caz nu vom obţine un rezultat ci mesajul de eroare menţionat anterior:

```
SELECT NUME, PUNCTAJ
FROM STUD
WHERE PUNCTAJ = (SELECT PUNCTAJ FROM
  STUD);
```

ERORI (2)

- Subcererea nu întoarce nici o valoare. În acest caz vom obţine un rezultat vid (fără nici o linie), condiţia evaluându-se la FALS:

```
SELECT NUME, PUNCTAJ
FROM STUD
WHERE PUNCTAJ =
    (SELECT MAX(PUNCTAJ) FROM STUD WHERE
  CODS = 100)
```

O COLOANA

- În acest caz valorile coloanei întoarse de subcerere sunt asimilate unei mulţimi.
- Condiţia trebuie să folosească operatorul IN sau negatul acestuia NOT IN şi nu operatori de comparaţie.

EXEMPLU

- Cererea din exemplul următor afişează lista tuturor studenţilor de la specializarea cu codul 21 care sunt tutori ai altor studenţi. Pentru a fi tutor, matricola studentului trebuie să aparţină mulţimii valorilor aflate pe coloana TUTOR din tabela STUD calculată cu ajutorul subcererii:

```
SELECT NUME, CODS
FROM STUD
WHERE MATR IN (SELECT TUTOR FROM STUD)
  AND CODS = 21;
```

REZULTAT

- Rezultatul va conţine datele pentru doi studenţi:

```
NUME          CODS
---------- -------
STANCA          21
ALEX            21
```

- Observaţie: NOT IN returnează întotdeauna valoarea fals în cazul în care mulţimea conţine valori nule.

NOT IN

- Din această cauză pentru a obţine lista studenţilor de la această specializare care nu sunt tutori nu se poate folosi cererea:

```
SELECT NUME, CODS
FROM STUD
WHERE MATR NOT IN (SELECT TUTOR FROM
  STUD) AND CODS = 21;
```

- deoarece subcererea întoarce o coloană care conţine şi valori nule.

NOT IN – cont.

- În astfel de cazuri este necesară eliminarea acestor valori din rezultat prin adăugarea unei condiţii suplimentare de tip IS NOT NULL:

```
SELECT NUME, CODS
FROM STUD
WHERE MATR NOT IN
(SELECT TUTOR FROM STUD
 WHERE TUTOR IS NOT NULL)
AND CODS = 21;
```

SUBCERERI CU GROUP BY

- În exemplul următor subcererea foloseşte o clauză GROUP BY pentru a genera punctajele maxime pentru fiecare specializare. Cererea principală afişează studenţii care au un punctaj egal cu vreuna dintre valorile returnate:

```
SELECT NUME, PUNCTAJ, CODS
FROM STUD
WHERE PUNCTAJ IN
   (SELECT MAX(PUNCTAJ)
    FROM STUD
    GROUP BY CODS);
```

SUBCERERI CU GROUP BY (2)

```
SELECT NUME, PUNCTAJ, CODS
FROM STUD
WHERE PUNCTAJ IN
   (SELECT MAX(PUNCTAJ)
    FROM STUD
    GROUP BY CODS);
```

- Întâmplător, rezultatul conţine chiar studenţii cu cel mai mare punctaj pentru fiecare specializare.
- În cazul general însă rezultatul poate conţine şi studenţi care nu au punctajul maxim la specializarea lor dar egal cu maximul unei alte specializări.

SOME/ANY SI ALL

- Este posibilă folosirea operatorilor de comparaţie uzuali (<, >, =, etc.) în conjuncţie cu o cerere care întoarce o coloană dacă aceasta este prefixată cu unul din operatorii SOME, ANY şi ALL.
- Semnificaţia lor este următoarea:
- SOME şi ANY: condiţia este adevărată dacă măcar o valoare dintre cele returnate de subcerere verifică comparaţia respectivă.
- ALL: condiţia este adevărată dacă toate valorile returnate de subcerere verifică comparaţia respectivă.

EXEMPLE (1)

- Lista studenţilor care au un punctaj mai mare decât al vreunui student de la specializarea cu cod 11:

```
SELECT NUME, PUNCTAJ
FROM STUD
WHERE PUNCTAJ >
   SOME(SELECT PUNCTAJ FROM STUD
        WHERE CODS = 11);
```

- Înlocuirea lui SOME cu ANY duce la obţinerea aceluiaşi rezultat, cei doi operatori efectuând aceeaşi operaţie

EXEMPLE (2)

- Lista studenţilor care au un punctaj mai mare decât al tuturor studenţilor de la specializarea 11:

```
SELECT NUME, PUNCTAJ
FROM STUD
WHERE PUNCTAJ >
   ALL(SELECT PUNCTAJ FROM STUD
        WHERE CODS = 11);
```

O TABELA

- În cazul în care subcererea întoarce un rezultat care are mai multe coloane acesta este asimilat cu o mulţime de linii şi se poate folosi operatorul IN în următorul mod:

```
WHERE (lista_de_expresii) IN (subcerere)
```

REGULI

1. Lista de expresii trebuie încadrată de paranteze rotunde.
2. Numărul de coloane din rezultatul subcererii trebuie să fie egal cu numărul de expresii din listă.
3. Corespondenţa între valorile expresiilor din listă şi coloanele rezultatului este poziţională.

REGULI - cont

4. Tipurile elementelor corespondente trebuie să fie aceleaşi sau convertibile automat unul la celălalt (sistemul Oracle face conversia automată între tipurile şir de caractere şi numere/date calendaristice).
5. Condiţia este adevărată dacă rezultatul subcererii conţine măcar o linie formată din valorile expresiilor din listă.
6. Dacă rezultatul subcererii este vid întreaga condiţie este evaluată la *fals*.

EXEMPLU

- Pentru a afla care sunt studenţii cu cel mai mare punctaj de la fiecare specializare, cererea este următoarea:

```
SELECT NUME, PUNCTAJ, CODS
FROM STUD
WHERE (CODS, PUNCTAJ) IN
            (SELECT CODS, MAX(PUNCTAJ)
             FROM STUD
             GROUP BY CODS);
```

- Codiţia va fi adevărată dacă punctajul studentului este egal cu un punctaj maxim întors de subcerere şi în acelaşi timp codul specializării este al celei pentru care s-a calculat maximul respectiv.

OBSERVATIE

- Observaţie: Din cauza faptului că două valori nule nu sunt egale între ele un cuplu de tipul (NULL, valoare) nu va fi considerat egal cu el însuşi.
- Din acest motiv, cererea următoare nu va returna date despre studenţii care nu au un tutor asociat (au o valoare nulă pe această coloană) deşi în aparenţă condiţia ar trebui să fie adevărată pentru orice student al specializării având codul 11:

EXEMPLU

```
SELECT MATR, NUME, CODS, TUTOR
FROM STUD
WHERE (MATR, NUME, CODS, TUTOR) IN
     (SELECT MATR, NUME, CODS, TUTOR
      FROM STUD
      WHERE CODS = 11);
```

REZULTATE

```
 MATR NUME           CODS TUTOR
----- ---------- ----- -----
 1325 VASILE          11  1456
```

- deşi rezultatul subcererii (executată separat) este următorul:

```
MATR NUME           CODS TUTOR
----- ---------- ----- -----
 1456 GEORGE          11
 1325 VASILE          11  1456
 1645 MARIA           11
```

Bibliografie

1. **Hector Garcia-Molina, Jeffrey D. Ullman, Jennifer D. Widom**: *Database Systems: The Complete Book*, Prentice-Hall, Englewood Cliffs, NJ, 2002.
2. **F. Rădulescu** : *Oracle SQL, PL/SQL*, Editura Printech, ISBN 973-718-203-02005