

Interogări multi-relație. Operația de join. Operatori pe mulțimi. Subcereri necorelate.

I. [Obiective]

În acest laborator vom continua lucrul cu interogări **multi-relație** (acestea sunt cele care regăsesc date din mai multe tabele). Am introdus deja diferite tipuri de **join**. Vom relua aceste operații, vom analiza și o altă metodă de implementare a lor și de asemenea vom utiliza **operatori pe mulțimi** și **subcereri necorelate** (fără sincronizare).

Foarte utile în rezolvarea exercițiilor propuse vor fi **funcțiile SQL**, prezentate în laboratorul 2.

II. [Join]

Am implementat deja operația de **join** (compunere a tabelelor) în cadrul unor exemple relative la modelul luat în considerare (HR).

Join-ul este operația de regăsire a datelor din două sau mai multe tabele, pe baza valorilor comune ale unor coloane. De obicei, aceste coloane reprezintă cheia primară, respectiv cheia externă a tabelelor. Reamintim că pentru a realiza un **join** între **n** tabele, va fi nevoie de cel puțin **n – 1 condiții de join**.

Tipuri de join :

- **Inner join (equijoin, join simplu)** – corespunde situației în care valorile de pe coloanele ce apar în condiția de **join** trebuie să fie egale.
- **Nonequijoin** - condiția de **join** conține alți operatori decât operatorul egalitate.
- **Left | Right | Full Outer join** – un **outer join** este utilizat pentru a obține în rezultat și înregistrările care nu satisfac condiția de **join**. Operatorul pentru **outer join** este semnul plus inclus între paranteze (**+**), care se plasează în acea parte a condiției de **join** care este **deficitară în informație**. Efectul acestui operator este de a uni liniile tabelului care nu este deficitar în informație și cărora nu le corespunde nici o linie în celălalt tabel cu o linie cu valori **null**. Operatorul (**+**) poate fi plasat în orice parte a condiției de **join**, dar nu în ambele părți.

Full outer join – left outer join + right outer join.

Obs: O condiție care presupune un **outer join** nu poate utiliza operatorul **IN** și nu poate fi legată de altă condiție prin operatorul **OR**.

Obs: Un caz special al operației de join este **self join** – join-ul unui tabel cu el însuși. În ce situație concretă (relativ la modelul nostru) apare această operație?

Obs: **Alias**-urile pot fi utilizate oriunde, ca o notație mai scurtă, în locul denumirii tabelului. Ele pot avea lungimea de maxim 30 de caractere, dar este recomandat să fie scurte și sugestive. Dacă este atribuit un alias unui tabel din clauza **FROM**, atunci el trebuie să înlocuiască aparițiile numelui tabelului în instrucțiunea **SELECT**.

Un alias poate fi utilizat pentru a califica denumirea unei coloane. Calificarea unei coloane cu numele sau alias-ul tabelului se poate face :

- opțional, pentru claritate și pentru îmbunătățirea timpului de acces la baza de date;
- obligatoriu, ori de câte ori există o ambiguitate privind sursa coloanei. Ambiguitatea constă, de obicei, în existența unor coloane cu același nume în mai

mult de două tabele.

Join în standardul SQL3 (SQL:1999):

Pentru *join*, sistemul *Oracle* oferă și o sintaxă specifică, introdusă de către standardul SQL3 (SQL:1999). Această sintaxă nu aduce beneficii în privința performanței față de *join*-urile care se specifică în clauza *WHERE*. Tipurile de *join* conforme cu SQL3 sunt definite prin cuvintele cheie *CROSS JOIN* (pentru produs cartezian), *NATURAL JOIN*, *FULL OUTER JOIN*, clauzele *USING* și *ON*.

Sintaxa corespunzătoare standardului SQL3 este următoarea:

```
SELECT tabel_1.nume_coloană, tabel_2.nume_coloană
FROM tabel_1
[CROSS JOIN tabel_2]
| [NATURAL JOIN tabel_2]
| [JOIN tabel_2 USING (nume_coloană) ]
| [JOIN tabel_2 ON (tabel_1.nume_coloană = tabel_2.nume_coloană) ]
| [LEFT | RIGHT | FULL OUTER JOIN tabel_2
ON (tabel_1.nume_coloană = tabel_2.nume_coloană) ];
```

- *NATURAL JOIN* presupune existența unor coloane având același nume în ambele tabele. Clauza determină selectarea liniilor din cele două tabele, care au valori egale în aceste coloane. Dacă tipurile de date ale coloanelor cu nume identice sunt diferite, va fi returnată o eroare.

Coloanele având același nume în cele două tabele trebuie să nu fie precedate de numele sau *alias*-ul tabelului corespunzător.

- *JOIN tabel_2 USING nume_coloană* efectuează un *equijoin* pe baza coloanei cu numele specificat în sintaxă. Această clauză este utilă dacă există coloane având același nume, dar tipuri de date diferite. Coloanele referite în clauza *USING* **trebuie să nu conțină calificatori** (să nu fie precedate de nume de tabele sau *alias*-uri) în nici o apariție a lor în instrucțiunea SQL. Clauzele *NATURAL JOIN* și *USING* nu pot coexista în aceeași instrucțiune SQL.
- *JOIN tabel_2 ON tabel_1.nume_coloană = tabel_2.nume_coloană* efectuează un *equijoin* pe baza condiției exprimate în clauza *ON*. Această clauză permite specificarea separată a condițiilor de *join*, respectiv a celor de căutare sau filtrare (din clauza *WHERE*).
- *LEFT*, *RIGHT* și *FULL OUTER JOIN tabel_2 ON (tabel_1.nume_coloană = tabel_2.nume_coloană)* efectuează *outer join* la stânga, dreapta, respectiv în ambele părți pe baza condiției exprimate în clauza *ON*.

Un *join* care returnează rezultatele unui *inner join*, dar și cele ale *outer join*-urilor la stânga și la dreapta se numește *full outer join*.

III. [Operatori pe mulțimi]

Operatorii pe mulțimi combină rezultatele obținute din două sau mai multe interogări. Cererile care conțin operatori pe mulțimi se numesc **cereri compuse**. Există patru operatori pe mulțimi: *UNION*, *UNION ALL*, *INTERSECT* și *MINUS*.

Toți operatorii pe mulțimi au aceeași precedență. Dacă o instrucțiune SQL conține mai mulți operatori pe mulțimi, *server*-ul *Oracle* evaluează cererea de la stânga la dreapta (sau de sus în jos). Pentru a schimba această ordine de evaluare, se pot

utiliza paranteze.

- Operatorul **UNION** returnează toate liniile selectate de două cereri, eliminând duplicatele. Acest operator nu ignoră valorile *null* și are precedență mai mică decât operatorul *IN*.
- Operatorul **UNION ALL** returnează toate liniile selectate de două cereri, fără a elimina duplicatele. Precizările făcute asupra operatorului *UNION* sunt valabile și în cazul operatorului *UNION ALL*. În cererile asupra cărora se aplică *UNION ALL* nu poate fi utilizat cuvântul cheie *DISTINCT*.
- Operatorul **INTERSECT** returnează toate liniile comune cererilor asupra cărora se aplică. Acest operator nu ignoră valorile *null*.
- Operatorul **MINUS** determină liniile returnate de prima cerere care nu apar în rezultatul celei de-a doua cereri. Pentru ca operatorul *MINUS* să funcționeze, este necesar ca toate coloanele din clauza *WHERE* să se afle și în clauza *SELECT*.

Observații:

- În mod implicit, pentru toți operatorii cu excepția lui *UNION ALL*, rezultatul este ordonat crescător după valorile primei coloane din clauza *SELECT*.
- Pentru o cerere care utilizează operatori pe mulțimi, cu excepția lui *UNION ALL*, *server-ul Oracle* elimină liniile duplicate.
- În instrucțiunile *SELECT* asupra cărora se aplică operatori pe mulțimi, coloanele selectate trebuie să corespundă ca număr și tip de date. Nu este necesar ca numele coloanelor să fie identice. Numele coloanelor din rezultat sunt determinate de numele care apar în clauza *SELECT* a primei cereri.

IV. [Subcereri]

Prin intermediul subcererilor se pot construi interogări complexe pe baza unor instrucțiuni simple.

O subcerere (subinterogare) este o comandă *SELECT* integrată într-o clauză a altei instrucțiuni *SQL*, numită instrucțiune „părinte” sau instrucțiune exterioară. Subcererile mai sunt numite instrucțiuni *SELECT* imbricate sau interioare.

Rezultatele subcererii sunt utilizate în cadrul cererii exterioare, pentru a determina rezultatul final. În funcție de modul de evaluare a subcererii în raport cu cererea exterioară, subcererile pot fi:

- nesincronizate (necorelate) sau
- sincronizate (corelate).

Prima clasă de subcereri este evaluată dinspre interior către exterior, adică interogarea externă acționează pe baza rezultatului cererii interne. Al doilea tip de subcerere este evaluat invers, adică interogarea externă furnizează valori cererii interne, iar rezultatele subcererii sunt transferate cererii externe.

- Subcererile nesincronizate care apar în clauza *WHERE* a unei interogări sunt de forma următoare:

[illegible]

- 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.

➤ Subcererile sincronizate care apar în clauza *WHERE* a unei interogări au următoarea formă generală:

```
SELECT expresie_ext_1[, expresie_ext_2 ...]
FROM   nume_tabel_1 extern
WHERE  expresie_condiție operator
        (SELECT expresie
         FROM   nume_tabel_2
         WHERE  expresie = extern.expresie_ext);
```

- 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.

Obs: 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*.

Cuvintele cheie *ANY* și *ALL* pot fi utilizate cu subcererile care produc o singură coloană de valori. Dacă subcererea este precedată de către cuvântul cheie *ALL*, atunci condiția va fi adevărată numai dacă este satisfăcută de către toate valorile produse de subcerere. Astfel, <*ALL* are semnificația „mai mic decât minimul“, iar >*ALL* este echivalent cu „mai mare decât maximul“. Dacă subcererea este precedată de către cuvântul cheie *ANY*, condiția va fi adevărată dacă este satisfăcută de către oricare (una sau mai multe) dintre valorile produse de subcerere. În comparații, <*ANY* are semnificația „mai mic decât maximul“; >*ANY* înseamnă „mai mare decât minimul“; =*ANY* este echivalent cu operatorul *IN*.

Dacă subcererea returnează mulțimea vidă, atunci condiția *ALL* va returna valoarea *true*, iar condiția *ANY* va returna valoarea *false*. Standardul *ISO* permite utilizarea cuvântului cheie *SOME*, în locul lui *ANY*.

V. [Exerciții - join]

1. Scrieți o cerere pentru a se afișa numele, luna (în litere) și anul angajării pentru toți salariații din același departament cu Gates, al căror nume conține litera „a“. Se va exclude Gates. Se vor da 2 soluții pentru determinarea apariției literei „A“ în nume. De asemenea, pentru una din metode se va da și varianta join-ului conform standardului *SQL99*.

2. Să se afișeze codul și numele angajaților care lucrează în același departament cu cel puțin un angajat al cărui nume conține litera „t“. Se vor afișa, de asemenea, codul și

numele departamentului respectiv. Rezultatul va fi ordonat alfabetic după nume. Se vor da 2 soluții pentru join (condiție în clauza WHERE și sintaxa introdusă de standardul SQL3).

Salvați cererea într-un fișier *p2/3.sql*.

3. Sa se afișeze numele, salariul, titlul job-ului, orașul și țara în care lucrează angajații conduși direct de King. Se vor da 2 soluții pentru join.

4. Executați comenzile SQL*Plus următoare:

SET LINESIZE 120

SET PAGESIZE 20

după care rulați comanda de la exercițiul precedent (e suficient */* pentru rularea buffer-ului). Ce observați? Ce efect au comenzile SET LINESIZE n, SET PAGESIZE n?

5. Sa se afișeze codul departamentului, numele departamentului, numele și job-ul tuturor angajaților din departamentele al căror nume conține șirul 'ti'. De asemenea, se va lista salariul angajaților, în formatul "\$99,999.00". Rezultatul se va ordona alfabetic după numele departamentului, și în cadrul acestuia, după numele angajaților.

6. Sa se afișeze numele angajaților, numărul departamentului, numele departamentului, orașul și job-ul tuturor salariaților al caror departament este localizat în Oxford.

7. Sa se modifice fișierul *p2/3.sql* pentru a afișa codul, numele și salariul tuturor angajaților care castiga mai mult decât salariul mediu pentru job-ul corespunzător și lucrează într-un departament cu cel puțin unul din angajații al caror nume conține litera "t". Salvați ca *p7/3.sql*. Executați cererea.

Obs: Salariul mediu pentru un job se va considera drept media aritmetică a valorilor minime și maxime admise pentru acesta (media valorilor coloanelor *min_salary* și *max_salary*).

8. Să se afișeze numele salariaților și numele departamentelor în care lucrează. Se vor afișa și salariații care nu au asociat un departament. (right outer join, 2 variante).

9. Să se afișeze numele departamentelor și numele salariaților care lucrează în ele. Se vor afișa și departamentele care nu au salariați. (left outer join, 2 variante)

10. Cum se poate implementa *full outer join*?

Obs: *Full outer join* se poate realiza fie prin reuniunea rezultatelor lui *right outer join* și *left outer join*, fie utilizând sintaxa introdusă de standardul SQL99.

VI. [Exerciții - operatori pe mulțimi]

11. Se cer codurile departamentelor al căror nume conține șirul "re" sau în care lucrează angajați având codul job-ului "SA_REP".

Cum este ordonat rezultatul?

12. Ce se întâmplă dacă înlocuim *UNION* cu *UNION ALL* în comanda precedentă?

13. Sa se obtina codurile departamentelor in care nu lucreaza nimeni (nu este introdus nici un salariat in tabelul *employees*). Se cer două soluții.

Obs: Operatorii pe mulțimi pot fi utilizați în subcereri. Coloanele care apar în clauza *WHERE* a interogării trebuie să corespundă, ca număr și tip de date, celor din clauza *SELECT* a subcererii.

SELECT department_id "Cod departament"
FROM departments

MINUS

```
SELECT department_id  
FROM employees;
```

```
SELECT department_id  
FROM departments  
WHERE department_id NOT IN (SELECT DISTINCT NVL(department_id,0)  
                             FROM employees);
```

? În a doua variantă, de ce este nevoie de utilizarea funcției NVL?

14. Se cer codurile departamentelor al căror nume conține șirul "re" și în care lucrează angajați având codul job-ului "HR_REP".

15. Să se determine codul angajaților, codul job-urilor și numele celor al căror salariu este mai mare decât 3000 sau este egal cu media dintre salariul minim și cel maxim pentru job-ul respectiv.

```
SELECT employee_id, job_id, last_name  
FROM employees  
WHERE (job_id, salary)  
      IN (SELECT job_id, salary  
          FROM employees  
          WHERE salary > 3000  
          UNION  
          SELECT job_id, (min_salary+max_salary)/2  
          FROM jobs);
```

VII. [Exercitii - subcereri necorelate (nesincronizate)]

16. 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');
```

17. 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';
```

? Se putea pune "=" în loc de "IN"? În care caz nu se poate face această înlocuire?

18. Folosind subcereri, să se afișeze numele și salariul angajaților conduși direct de președintele companiei (acesta este considerat angajatul care nu are manager).

19. Scrieți o cerere pentru a afișa numele, codul departamentului și salariul angajaților al căror număr de departament și salariu coincid cu numărul departamentului și salariul unui angajat care castiga comision.

20. Rezolvați problema 7 utilizând subcereri.

21. Scrieti o cerere pentru a afisa angajatii care castiga mai mult decat oricare functionar (job-ul conține șirul "CLERK"). Sortati rezultatele dupa salariu, in ordine descrescatoare. (ALL)

? Ce rezultat este returnat dacă se înlocuiește "ALL" cu "ANY"?

22. Scrieți o cerere pentru a afișa numele, numele departamentului și salariul angajaților care nu câștigă comision, dar al căror șef direct coincide cu șeful unui angajat care câștigă comision.

23. Sa se afiseze numele, departamentul, salariul și job-ul tuturor angajatilor al caror salariu si comision coincid cu salariul si comisionul unui angajat din Oxford.

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