Baze de date-Anul 2
Laborator 3 SQL

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 şi o altă metodă de implementare a lor şi de asemenea vom utiliza ***operatori pe mulțimi*** şi ***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 şi 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 deficient în informație şi 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* şi 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 însuşi. 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 şi 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 şi 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 acelaşi nume în mai

mult de două tabele.

Join* în standardul SQL3 (SQL:1999)*:

Pentru *join*, sistemul *Oracle* oferă şi 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* şi *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 acelaşi 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 acelaşi 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 acelaşi 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* şi *USING* nu pot coexista în aceeaşi 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* şi *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 şi cele ale *outer join*-urilor la stânga şi la dreapta se numeşte *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* şi *MINUS*.

Toți operatorii pe mulțimi au aceeaşi 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* şi 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 şi 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 şi 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 duplicat.
- În instrucțiunile *SELECT* asupra cărora se aplică operatori pe mulțimi, coloanele selectate trebuie să corespundă ca număr şi 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:

```
SELECT   expresie1, expresie2, ...
FROM     nume_tabel1
WHERE    expresie_condiție operator (SELECT   expresie
                                     FROM     nume_tabel2);
```

- cererea internă este executată prima şi 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;
- paşii 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* şi *ALL*.

Cuvintele cheie *ANY* şi *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. Scrieti o cerere pentru a se afisa numele, luna (în litere) şi anul angajării pentru toti salariatii din acelasi 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 şi varianta join-ului conform standardului *SQL99*.

2. Sa se afiseze codul şi numele angajaților care lucrează în acelasi departament cu cel puțin un angajat al cărui nume conține litera “t”. Se vor afişa, de asemenea, codul şi

numele departamentului respectiv. Rezultatul va fi ordonat alfabetic după nume. Salvați cererea într-un fişier *p2l3.sql*.

! Dați şi soluția care utilizează sintaxa specifică *Oracle* pentru join.

3. Sa se afiseze numele, salariul, titlul job-ului, oraşul şi țara în care lucrează angajatii condusi direct de King.

! Dați mai multe metode de rezolvare a acestui exercițiu.

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 afiseze codul departamentului, numele departamentului, numele si job-ul tuturor angajatilor din departamentele al căror nume conține şirul 'ti'. De asemenea, se va lista salariul angajaților, în formatul "$99,999.00". Rezultatul se va ordona alfabetic după numele departamentului, şi în cadrul acestuia, după numele angajaților.

6. Sa se afiseze numele angajatilor, numarul departamentului, numele departamentului, oraşul si job-ul tuturor salariatilor al caror departament este localizat in Oxford.

7. Sa se modifice fisierul *p2l3.sql* pentru a afisa codul, numele si salariul tuturor angajatilor care castiga mai mult decat salariul mediu pentru job-ul corespunzător si lucreaza intr-un departament cu cel putin unul din angajatii al caror nume contine litera "t". Salvati ca *p7l3.sql*. Executati cererea.

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

9. Să se afişeze numele departamentelor şi numele salariaților care lucrează în ele. Se vor afişa şi 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* şi *left outer join*, fie utilizând sintaxa specifică standardului SQL99.

VI. [Exerciții - operatori pe mulțimi]
11. Se cer codurile departamentelor al căror nume conține şirul "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 şi 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 şirul “re” şi în care lucrează angajați având codul job-ului “HR_REP”.

15. Să se determine codul angajaților, codul job-urilor şi numele celor al căror salariu este mai mare decât 3000 sau este egal cu media dintre salariul minim şi 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]

16. Folosind subcereri, să se afişeze numele şi 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 afişa numele şi salariul pentru toți colegii (din acelaşi 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”? In care caz nu se poate face această înlocuire?

18. Folosind subcereri, să se afişeze numele şi salariul angajaților conduşi direct de preşedintele companiei (acesta este considerat angajatul care nu are manager).

19. Scrieti o cerere pentru a afişa numele, codul departamentului si salariul angajatilor al caror număr de departament si salariu coincid cu numarul departamentului si 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 şirul “CLERK”). Sortati rezultatele dupa salariu, in ordine descrescatoare. (*ALL*)

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

22. Scrieți o cerere pentru a afişa numele, numele departamentului şi salariul angajaților care nu câştigă comision, dar al căror şef direct coincide cu şeful unui angajat care câştigă comision.

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

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