

## **Laborator 3 PL/SQL**

### **Cursoare**

Serverul Oracle alocă o zonă de memorie, care se numește **cursor**, ori de câte ori este înaintată o cerere SQL.

Cursoarele pot fi de două feluri:

- **implicite** (create și gestionate de PL/SQL în mod automat)  
Cursoarele implicite sunt declarate de PL/SQL în mod implicit pentru toate comenzile LMD și comanda SELECT, inclusiv comenzile care întorc o singură linie.
- **explicite** (create și gestionate de utilizator).  
Cursoarele explicite sunt create pentru cereri care întorc mai mult de o linie.

Mulțimea de linii procesate întoarse de o cerere multiple-row se numește **set activ**.

Un cursor este o modalitate de a parcurge setul activ linie cu linie.

Etapele utilizării unui cursor:

a) **Declarare.**

Definește numele și structura cursorului, împreună cu clauza SELECT care va popula cursorul cu date. Cererea este validată, dar nu executată.

În secțiunea declarativă prin intermediul cuvântului cheie CURSOR:

```
CURSOR c_num_curs [ (parametru tip_de_date, ..) ] IS  
comanda SELECT;
```

b) **Deschidere** (comanda OPEN).

Este executată cererea și se populează cursorul cu date.

```
OPEN c_num_curs [ (parametru, ...) ];
```

c) **Încărcare** (comanda FETCH).

Încarcă o linie din cursor (indicată de pointerul cursorului) în variabile și mută pointerul la linia următoare.

Numărul de variabile din clauza INTO trebuie să se potrivească cu lista SELECT din definiția cursorului.

```
FETCH c_num_curs INTO variabil1, ...;
```

d) **Verificare**

Se verifică dacă s-a ajuns la finalul setului activ folosind atributul %NOTFOUND sau %FOUND.

```
c_num_curs%NOTFOUND = TRUE, dacă nicio linie nu a fost procesată
```

```
c_num_curs%FOUND = TRUE, dacă cel puțin o linie a fost procesată
```

Dacă nu s-a ajuns la final mergi la (c).

e) **Închidere** cursor (dacă rămâne deschis cursorul consumă din resursele serverului)

```
CLOSE c_num_curs;
```

Șterge datele din cursor și îl închide. Acesta poate fi redeschis pentru o actualizare a datelor.

Alte atribute utile:

- **%ROWCOUNT** reprezintă numărul de linii procesate;
- **%ISOPEN** este TRUE în cazul în care cursorul este deschis.

```

DECLARE
    declarare cursor
BEGIN
    deschidere cursor (OPEN)
    WHILE rămân linii de recuperat LOOP
        recuperare linie rezultat (FETCH)
        ...
    END LOOP
    închidere cursor (CLOSE)
    ...
END;
```

1. Obțineți pentru fiecare departament numele acestuia și numărul de angajați, într-una din următoarele forme:

“În departamentul <nume departament> nu lucrează angajati”.

“În departamentul <nume departament> lucrează un angajat”.

“În departamentul <nume departament> lucrează <numar> angajati”.

Rezolvați problema folosind un **cursor explicit**.

**TEMĂ:** Rezolvați problema în SQL.

```

DECLARE
    v_nr      number(4);
    v_nume    departments.department_name%TYPE;
    CURSOR c IS
        SELECT department_name nume, COUNT(employee_id) nr
        FROM    departments d, employees e
        WHERE   d.department_id=e.department_id(+)
        GROUP BY department_name;
BEGIN
    OPEN c;
    LOOP
        FETCH c INTO v_nume,v_nr;
        EXIT WHEN c%NOTFOUND;
        IF v_nr=0 THEN
            DBMS_OUTPUT.PUT_LINE('In departamentul '|| v_nume||
                                ' nu lucreaza angajati');
        ELSIF v_nr=1 THEN
            DBMS_OUTPUT.PUT_LINE('In departamentul '|| v_nume||
                                ' lucreaza un angajat');
        ELSE
            DBMS_OUTPUT.PUT_LINE('In departamentul '|| v_nume||
                                ' lucreaza '|| v_nr||' angajati');
        END IF;
    END LOOP;
    CLOSE c;
END;
```

2. Rezolvați exercițiul 1 menținând informațiile din cursor în colecții. Comentați. Procesăți toate liniile din cursor, încărcând la fiecare pas câte 5 linii.

**Temă:** Rezolvați problema folosind cursorul și o singură colecție.

Rezolvați problema folosind doar colecții.

```

DECLARE
  TYPE    tab_nume IS TABLE OF departments.department_name%TYPE;
  TYPE    tab_nr IS TABLE OF NUMBER(4);
  t_nr    tab_nr;
  t_nume  tab_nume;
  CURSOR c IS
    SELECT department_name nume, COUNT(employee_id) nr
    FROM    departments d, employees e
    WHERE   d.department_id=e.department_id(+)
    GROUP BY department_name;
BEGIN
  OPEN c;
  FETCH c BULK COLLECT INTO t_nume, t_nr;
  CLOSE c;
  FOR i IN t_nume.FIRST..t_nume.LAST LOOP
    IF t_nr(i)=0 THEN
      DBMS_OUTPUT.PUT_LINE('In departamentul '|| t_nume(i)||
                           ' nu lucreaza angajati');
    ELSIF t_nr(i)=1 THEN
      DBMS_OUTPUT.PUT_LINE('In departamentul '||t_nume(i)||
                           ' lucreaza un angajat');
    ELSE
      DBMS_OUTPUT.PUT_LINE('In departamentul '|| t_nume(i)||
                           ' lucreaza '|| t_nr(i)|| ' angajati');
    END IF;
  END LOOP;
END;
/

```

### 3. Rezolvați exercițiul 1 folosind un **ciclu cursor**.

```

DECLARE
  CURSOR c IS
    SELECT department_name nume, COUNT(employee_id) nr
    FROM    departments d, employees e
    WHERE   d.department_id=e.department_id(+)
    GROUP BY department_name;
BEGIN
  FOR i in c LOOP
    IF i.nr=0 THEN
      DBMS_OUTPUT.PUT_LINE('In departamentul '|| i.nume||
                           ' nu lucreaza angajati');
    ELSIF i.nr=1 THEN
      DBMS_OUTPUT.PUT_LINE('In departamentul '|| i.nume ||
                           ' lucreaza un angajat');
    ELSE
      DBMS_OUTPUT.PUT_LINE('In departamentul '|| i.nume||
                           ' lucreaza '|| i.nr|| ' angajati');
    END IF;
  END LOOP;
END;
/

```

#### 4. Rezolvați exercițiul 1 folosind un **ciclu cursor cu subcereri**.

```

BEGIN
  FOR i IN (SELECT department_name nume, COUNT(employee_id) nr
            FROM departments d, employees e
            WHERE d.department_id=e.department_id(+)
            GROUP BY department_name) LOOP
    IF i.nr=0 THEN
      DBMS_OUTPUT.PUT_LINE('In departamentul '|| i.nume||
                           ' nu lucreaza angajati');
    ELSIF i.nr=1 THEN
      DBMS_OUTPUT.PUT_LINE('In departamentul '|| i.nume ||
                           ' lucreaza un angajat');
    ELSE
      DBMS_OUTPUT.PUT_LINE('In departamentul '|| i.nume||
                           ' lucreaza '|| i.nr||' angajati');
    END IF;
  END LOOP;
END;
/

```

#### 5. Obțineți primii 3 manageri care au cei mai mulți subordonați. Afișați numele managerului, respectiv numărul de angajați.

a. Rezolvați problema folosind un cursor **explicit**.

b. Modificați rezolvarea anterioară astfel încât să obțineți primii 4 manageri care îndeplinesc condiția. Observați rezultatul obținut și specificați dacă la punctul a s-a obținut top 3 manageri?

**TEMĂ:** Rezolvați problema în SQL.

```

DECLARE
  v_cod      employees.employee_id%TYPE;
  v_nume     employees.last_name%TYPE;
  v_nr       NUMBER(4);
  CURSOR c IS
    SELECT   sef.employee_id cod, MAX(sef.last_name) nume,
            count(*) nr
    FROM     employees sef, employees ang
    WHERE    ang.manager_id = sef.employee_id
    GROUP BY sef.employee_id
    ORDER BY nr DESC;
BEGIN
  OPEN c;
  LOOP
    FETCH c INTO v_cod,v_nume,v_nr;
    EXIT WHEN c%ROWCOUNT>3 OR c%NOTFOUND;
    DBMS_OUTPUT.PUT_LINE('Managerul '|| v_cod ||
                        ' avand numele '|| v_nume ||
                        ' conduce '|| v_nr||' angajati');
  END LOOP;
  CLOSE c;
END;
/

```

## 6. Rezolvați exercițiul 5 folosind un **ciclu cursor**.

```

DECLARE
  CURSOR c IS
    SELECT      sef.employee_id cod, MAX(sef.last_name) nume,
               count(*) nr
    FROM        employees sef, employees ang
    WHERE       ang.manager_id = sef.employee_id
    GROUP BY    sef.employee_id
    ORDER BY    nr DESC;
BEGIN
  FOR i IN c LOOP
    EXIT WHEN c%ROWCOUNT>3 OR c%NOTFOUND;
    DBMS_OUTPUT.PUT_LINE('Managerul ' || i.cod ||
                          ' avand numele ' || i.nume ||
                          ' conduce ' || i.nr || ' angajati');
  END LOOP;
END;
/

```

## 7. Rezolvați exercițiul 5 folosind un **ciclu cursor cu subcereri**.

```

DECLARE
  top number(1) := 0;
BEGIN
  FOR i IN (SELECT      sef.employee_id cod, MAX(sef.last_name) nume,
                       count(*) nr
            FROM        employees sef, employees ang
            WHERE       ang.manager_id = sef.employee_id
            GROUP BY    sef.employee_id
            ORDER BY    nr DESC)
  LOOP
    DBMS_OUTPUT.PUT_LINE('Managerul ' || i.cod ||
                          ' avand numele ' || i.nume ||
                          ' conduce ' || i.nr || ' angajati');

    Top := top+1;
    EXIT WHEN top=3;
  END LOOP;
END;
/

```

## 8. Modificați exercițiul 1 astfel încât să obțineți doar departamentele în care lucrează cel puțin $x$ angajați, unde $x$ reprezintă un număr introdus de la tastatură. Rezolvați problema folosind toate cele trei tipuri de cursoare studiate.

```

DECLARE
  v_x      number(4) := &p_x;
  v_nr     number(4);
  v_nume   departments.department_name%TYPE;

```

```

CURSOR c (paramentru NUMBER) IS
  SELECT department_name nume, COUNT(employee_id) nr
  FROM   departments d, employees e
  WHERE  d.department_id=e.department_id
  GROUP BY department_name
  HAVING COUNT(employee_id)> paramentru;
BEGIN
  OPEN c(v_x) ;
  LOOP
    FETCH c INTO v_nume,v_nr;
    EXIT WHEN c%NOTFOUND;
    DBMS_OUTPUT.PUT_LINE('In departamentul '|| v_nume||
                          ' lucreaza '|| v_nr||' angajati');
  END LOOP;
  CLOSE c;
END;
/

```

```

DECLARE
  v_x      number(4) := &p_x;
  CURSOR c (paramentru NUMBER) IS
    SELECT department_name nume, COUNT(employee_id) nr
    FROM   departments d, employees e
    WHERE  d.department_id=e.department_id
    GROUP BY department_name
    HAVING COUNT(employee_id)> paramentru;
BEGIN
  FOR i in c(v_x) LOOP
    DBMS_OUTPUT.PUT_LINE('In departamentul '|| i.nume||
                          ' lucreaza '|| i.nr||' angajati');
  END LOOP;
END;
/

```

```

DECLARE
  v_x      number(4) := &p_x;
  BEGIN
    FOR i in (SELECT department_name nume, COUNT(employee_id) nr
              FROM   departments d, employees e
              WHERE  d.department_id=e.department_id
              GROUP BY department_name
              HAVING COUNT(employee_id)> v_x)
    LOOP
      DBMS_OUTPUT.PUT_LINE('In departamentul '|| i.nume||
                            ' lucreaza '|| i.nr||' angajati');
    END LOOP;
  END;
/

```

9. Măriți cu 1000 salariile celor care au fost angajați în 2000 (din tabelul emp\_\*\*\*) blocând liniile înainte de actualizare (**cursor SELECT FOR UPDATE**).

**Observație:** Uneori este necesară blocarea liniilor înainte ca acestea să fie șterse sau reactualizate. Blocarea se poate realiza (atunci când cursorul este deschis) cu ajutorul comenzii *SELECT* care conține clauza *FOR UPDATE*.

Comanda *SELECT* are următoarea extensie PL/SQL pentru blocarea explicită a înregistrărilor ce urmează a fi prelucrate (modificate sau șterse):

```
SELECT ... FROM ... WHERE ...GROUP BY ...ORDER BY ...
FOR UPDATE [OF lista_coloane] [NOWAIT | WAIT n];
```

În cazul în care liniile selectate de cerere nu pot fi blocate din cauza altor blocări atunci:

- dacă se folosește *NOWAIT* apare imediat eroarea *ORA-00054*;
- dacă nu se folosește *NOWAIT* atunci se așteaptă până când liniile sunt deblocate;
- dacă se folosește *WAIT n*, atunci se așteaptă un număr determinat de secunde pentru ca liniile ce trebuie selectate pentru modificare să fie deblocate.

Pentru a modifica o anumită linie întoarsă de un astfel de cursor se folosește clauza:

```
WHERE CURRENT OF nume_cursor
```

Această clauză apare la finalul unei comenzi *UPDATE* și face referință la un cursor care este deschis și pentru care s-a realizat cel puțin o încărcare (*FETCH*).

```
SELECT last_name, hire_date, salary
FROM emp_***
WHERE TO_CHAR(hire_date, 'yyyy') = 2000;
```

```
DECLARE
  CURSOR c IS
    SELECT *
    FROM emp_***
    WHERE TO_CHAR(hire_date, 'YYYY') = 2000
    FOR UPDATE OF salary NOWAIT;
BEGIN
  FOR i IN c LOOP
    UPDATE emp_***
    SET salary= salary+1000
    WHERE CURRENT OF c;
  END LOOP;
END;
/
```

```
SELECT last_name, hire_date, salary
FROM emp_***
WHERE TO_CHAR(hire_date, 'yyyy') = 2000;

ROLLBACK;
```

- 10.** Pentru fiecare dintre departamentele 10, 20, 30 și 40, obțineți numele precum și lista numelor angajaților care își desfășoară activitatea în cadrul acestora. Rezolvați problema folosind:
- a. cele trei tipuri de cursoare studiate;
  - b. expresii cursor.

**Observație:** În *Oracle9i* a fost introdus conceptul de expresie cursor care întoarce un cursor imbricat (*nested cursor*).

Varianta 1.1 – cursor clasic **Temă**Varianta 1.2 – ciclu cursor **Temă**Varianta 1.3 – ciclu cursor cu subcereri

```

BEGIN
  FOR v_dept IN (SELECT department_id, department_name
                 FROM   departments
                 WHERE  department_id IN (10,20,30,40))
  LOOP
    DBMS_OUTPUT.PUT_LINE('-----');
    DBMS_OUTPUT.PUT_LINE ('DEPARTAMENT '||v_dept.department_name);
    DBMS_OUTPUT.PUT_LINE('-----');
    FOR v_emp IN (SELECT last_name
                  FROM   employees
                  WHERE  department_id = v_dept.department_id)
    LOOP
      DBMS_OUTPUT.PUT_LINE (v_emp.last_name);
    END LOOP;
  END LOOP;
END;
/

```

Varianta 2 – expresii cursor

```

DECLARE
  TYPE refcursor IS REF CURSOR;
  CURSOR c_dept IS
    SELECT department_name,
           CURSOR (SELECT last_name
                    FROM   employees e
                    WHERE  e.department_id = d.department_id)
    FROM   departments d
    WHERE  department_id IN (10,20,30,40);
  v_num_dept  departments.department_name%TYPE;
  v_cursor    refcursor;
  v_num_emp   employees.last_name%TYPE;
BEGIN
  OPEN c_dept;
  LOOP
    FETCH c_dept INTO v_num_dept, v_cursor;
    EXIT WHEN c_dept%NOTFOUND;
    DBMS_OUTPUT.PUT_LINE('-----');
    DBMS_OUTPUT.PUT_LINE ('DEPARTAMENT '||v_num_dept);
    DBMS_OUTPUT.PUT_LINE('-----');
    LOOP
      FETCH v_cursor INTO v_num_emp;
      EXIT WHEN v_cursor%NOTFOUND;
      DBMS_OUTPUT.PUT_LINE (v_num_emp);
    END LOOP;
  END LOOP;
  CLOSE c_dept;
END;
/

```



11. Declarați un **cursor dinamic** care întoarce linii de tipul celor din tabelul emp\_\*\*\*. În funcție de o opțiune introdusă de la tastatură (una dintre valorile 1, 2 sau 3) deschideți cursorul astfel încât să regăsească:

- toate informațiile din tabelul emp\_\*\*\* (pentru opțiunea 1);
- doar angajații având salariul cuprins între 10000 și 20000 (pentru opțiunea 2);
- doar salariații angajați în anul 2000 (pentru opțiunea 3).

Verificați ce se întâmplă în cazul în care introduceți o valoare diferită de 1, 2 sau 3. Modificați corespunzător.

```

DECLARE
  TYPE      emp_tip IS REF CURSOR RETURN employees%ROWTYPE;
  -- sau
  -- TYPE    emp_tip IS REF CURSOR;

  v_emp      emp_tip;
  v_optiune  NUMBER := &p_optiune;
  v_ang      employees%ROWTYPE;
BEGIN
  IF v_optiune = 1 THEN
    OPEN v_emp FOR SELECT *
                      FROM employees;
  ELSIF v_optiune = 2 THEN
    OPEN v_emp FOR SELECT *
                      FROM employees
                      WHERE salary BETWEEN 10000 AND 20000;
  ELSIF v_optiune = 3 THEN
    OPEN v_emp FOR SELECT *
                      FROM employees
                      WHERE TO_CHAR(hire_date, 'YYYY') = 2000;
  ELSE
    DBMS_OUTPUT.PUT_LINE('Optiune incorecta');
  END IF;

  LOOP
    FETCH v_emp INTO v_ang;
    EXIT WHEN v_emp%NOTFOUND;
    DBMS_OUTPUT.PUT_LINE(v_ang.last_name);
  END LOOP;

  DBMS_OUTPUT.PUT_LINE('Au fost procesate ' || v_emp%ROWCOUNT
                        || ' linii');

  CLOSE v_emp;
END;
/

```

12. Citiți de la tastatură o valoare *n*. Prin intermediul unui **cursor deschis cu ajutorul unui șir dinamic** obțineți angajații având salariul mai mare decât *n*. Pentru fiecare linie regăsită de cursor afișați următoarele informații:

- numele și salariul dacă angajatul nu are comision;
- numele, salariul și comisionul dacă angajatul are comision.

```
DECLARE
  TYPE empref IS REF CURSOR;
  v_emp empref;
  v_nr INTEGER := &n;
BEGIN
  OPEN v_emp FOR
    'SELECT employee_id, salary, commission_pct ' ||
    'FROM employees WHERE salary > :bind_var'
    USING v_nr;
  -- introduceți liniile corespunzătoare rezolvarii problemei
END;
/
```

## EXERCITII

- E1.** Pentru fiecare job (titlu – care va fi afișat o singură dată) obțineți lista angajaților (nume și salariu) care lucrează în prezent pe jobul respectiv. Tratați cazul în care nu există angajați care să lucreze în prezent pe un anumit job. Rezolvați problema folosind:
- cursoare clasice
  - ciclu cursoare
  - ciclu cursoare cu subcereri
  - expresii cursor
- E2.** Modificați exercițiul anterior astfel încât să obțineți și următoarele informații:
- un număr de ordine pentru fiecare angajat care va fi resetat pentru fiecare job
  - pentru fiecare job
    - o numărul de angajați
    - o valoarea lunară a veniturilor angajaților
    - o valoarea medie a veniturilor angajaților
  - indiferent job
    - o numărul total de angajați
    - o valoarea totală lunară a veniturilor angajaților
    - o valoarea medie a veniturilor angajaților
- E3.** Modificați exercițiul anterior astfel încât să obțineți suma totală alocată lunar pentru plata salariilor și a comisioanelor tuturor angajaților, iar pentru fiecare angajat cât la sută din această sumă câștigă lunar.
- E4.** Modificați exercițiul anterior astfel încât să obțineți pentru fiecare job primii 5 angajați care câștigă cel mai mare salariu lunar. Specificați dacă pentru un job sunt mai puțin de 5 angajați.
- E5.** Modificați exercițiul anterior astfel încât să obțineți pentru fiecare job top 5 angajați. Dacă există mai mulți angajați care respectă criteriul de selecție care au același salariu, atunci aceștia vor ocupa aceeași poziție în top 5.
- E6.** Adaptați cerința exercițiului 10 pentru diagrama proiectului prezentată la materia Baze de Date din anul I. Rezolvați subpunctul (a) al acestui exercițiu în PL/SQL, folosind baza de date proprie.