

- *SQL* dinamic permite construirea dinamică a comenzilor la momentul execuției.
- Pentru execuția dinamică a comenzilor *SQL* în *PL/SQL* există două tehnici:
 - utilizarea pachetului *DBMS_SQL*;
 - *SQL* dinamic nativ (este mai ușor de utilizat, mai rapid și are avantajul că suportă tipuri definite de utilizator).
- Comanda de bază utilizată pentru procesarea dinamică nativă a comenzilor *SQL* și a blocurilor *PL/SQL* este *EXECUTE IMMEDIATE*, care are următoarea sintaxă:

```
EXECUTE IMMEDIATE șir_dinamic  
[INTO {def_variabila [, def_variabila ...] | record} ]  
[USING [IN | OUT | IN OUT] argument_bind  
[, [IN | OUT | IN OUT] argument_bind ...] ]  
[ {RETURNING | RETURN}  
INTO argument_bind [, argument_bind ...] ];
```

șir_dinamic = șir de caractere care reprezintă o comandă *SQL* (fără caracter de terminare) sau un bloc *PL/SQL* (având caracter de continuare);

def_variabila = variabila în care se stochează valoarea coloanei selectate;

record = înregistrarea în care se depune o linie selectată;

argument_bind, dacă se referă la valori de intrare (*IN*) este o expresie (comandă *SQL* sau bloc *PL/SQL*), iar dacă se referă la valori de ieșire (*OUT*) este o variabilă ce va conține valoarea selectată de comanda *SQL* sau de blocul *PL/SQL*.

Clauza *INTO* este folosită pentru cereri care întorc o singură linie, iar clauza *USING* pentru a reține argumentele de legătură.

Pentru procesarea unei cereri care returnează mai multe linii sunt necesare instrucțiunile *OPEN...FOR*, *FETCH* și *CLOSE*.

Prin clauza *RETURNING* sunt precizate variabilele care conțin rezultatele.

1. Să se creeze un subprogram prin care se poate șterge orice tabel din baza de date (dat ca parametru subprogramului).

```
CREATE OR REPLACE PROCEDURE sterg_*** (tabel VARCHAR2) IS  
BEGIN  
    EXECUTE IMMEDIATE 'DROP TABLE ' || tabel;  
END;  
/  
  
CREATE TABLE tab_*** (col number(4));  
EXECUTE sterg_***('tab_***');
```

2. Să se creeze un subprogram prin care să se obțină numărul de salariați al căror salariu depășește o valoare dată ca parametru.

```
CREATE OR REPLACE FUNCTION numar_*** (val NUMBER)  
RETURN NUMBER AS  
    sir VARCHAR2(500);  
    rezultat NUMBER;  
BEGIN  
    sir := 'SELECT COUNT(*) FROM employees ' || 'WHERE salary >=  
:x';
```

```

EXECUTE IMMEDIATE sir
      INTO rezultat
      USING val;
RETURN rezultat;
END;
/

```

3. Utilizând SQL dinamic, să se creeze tabelul tab_*** (col VARCHAR2(15)), apoi să se insereze în acesta 10 linii de forma contor||i, să se tipărească conținutul tabelului utilizând un bloc anonim, iar în final să se șteargă tabelul tab_***.

```

SET SERVEROUTPUT ON
DECLARE
    sir      VARCHAR2(50);
    bloc    VARCHAR2(500);
BEGIN
    -- creare tabel
    EXECUTE IMMEDIATE
        'CREATE TABLE tab_*** (col VARCHAR2(15))';
    --inserare in tabel
    FOR i IN 1..10 LOOP
        sir := 'INSERT INTO tab_*** VALUES (''Contor ' || i || '')';
        EXECUTE IMMEDIATE sir;
    END LOOP;
    -- tiparire continut tabel
    bloc := 'BEGIN
                FOR i IN (SELECT * FROM tab_***) LOOP
                    DBMS_OUTPUT.PUT_LINE (i.col);
                END LOOP;
            END;';
    EXECUTE IMMEDIATE bloc;
    -- stergere tabel
    EXECUTE IMMEDIATE 'DROP TABLE tab_***';
END;
/

```

4. Să se creeze un pachet care să conțină:

- o funcție prin care se vor returna toți angajații care îndeplinesc o anumită condiție, dată ca parametru;
- o funcție prin care se vor returna toți angajații care au un anumit job_id, dat ca parametru;

```

CREATE OR REPLACE PACKAGE pachet_*** AS
    TYPE refcursor IS REF CURSOR;
    FUNCTION f1 (sir VARCHAR2) RETURN refcursor;
    FUNCTION f2 (sir VARCHAR2) RETURN refcursor;
END pachet_***;
/
CREATE OR REPLACE PACKAGE BODY pachet_*** AS
    FUNCTION f1 (sir VARCHAR2) RETURN refcursor IS
        rez refcursor;
        comanda VARCHAR2(500);
    BEGIN

```

```

        comanda := 'SELECT * FROM employees ' || sir;
        OPEN rez FOR comanda;
        RETURN rez;
    END;
FUNCTION f2 (sir VARCHAR2) RETURN refcursor IS
    rez refcursor;
    comanda VARCHAR2(500);
    BEGIN
        comanda := 'SELECT * FROM employees WHERE job_id
= :j';
        OPEN rez FOR comanda USING sir;
        RETURN rez;
    END;
END pachet_***;
/
DECLARE
    v_emp employees%ROWTYPE;
    v_cursor pachet_***.refcursor;
BEGIN
    -- deschide cursor
    v_cursor := pachet_***.f1 ('WHERE salary >10000');
    -- parcurge cursor si tipareste rezultate
    LOOP
        FETCH v_cursor INTO v_emp;
        EXIT WHEN v_cursor%NOTFOUND;
        DBMS_OUTPUT.PUT_LINE(v_emp.last_name|| ' ' ||v_emp.salary);
    END LOOP;
    CLOSE v_cursor;

    DBMS_OUTPUT.PUT_LINE ('*****');
    -- deschide cursor
    v_cursor := pachet_***.f2 ('SA_MAN');
    -- parcurge cursor si tipareste rezultate
    LOOP
        FETCH v_cursor INTO v_emp;
        EXIT WHEN v_cursor%NOTFOUND;
        DBMS_OUTPUT.PUT_LINE (v_emp.last_name|| ' ' ||v_emp.job_id);
    END LOOP;
    CLOSE v_cursor;
END;
/

```

5. Utilizând SQL dinamic și tablouri imbricate, să se afișeze pentru fiecare departament codul și numele său.

```

DECLARE
    TYPE refc IS REF CURSOR;
    TYPE t_cod IS TABLE OF NUMBER;
    TYPE t_nume IS TABLE OF VARCHAR2(50);
    cursor_dept refc;
    cod t_cod;
    nume t_nume;
BEGIN

```

```

        DBMS_OUTPUT.PUT_LINE ('***** Varianta 1 *****');
        OPEN cursor_dept FOR 'SELECT department_id, department_name
FROM departments';
        FETCH cursor_dept BULK COLLECT INTO cod, nume;
        CLOSE cursor_dept;
        FOR i IN cod.FIRST..cod.LAST LOOP
            DBMS_OUTPUT.PUT_LINE (cod(i)|| ' ' || nume(i));
        END LOOP;

        DBMS_OUTPUT.PUT_LINE ('***** Varianta 2 *****');

        EXECUTE IMMEDIATE 'SELECT department_id, department_name
FROM departments '
        BULK COLLECT INTO cod, nume;
        FOR i IN cod.FIRST..cod.LAST LOOP
            DBMS_OUTPUT.PUT_LINE (cod(i)|| ' ' || nume(i));
        END LOOP;
END;
/

```

6. Utilizarea variabilelor de legătură ca argumente de tip *OUT* (numai comenzile INSERT, UPDATE și DELETE permit acest lucru).

Exemplul 1

```

DECLARE
    TYPE tablou IS TABLE OF VARCHAR2(60);
    v_tab      tablou;
    valoare    NUMBER := 1000;
    comanda    VARCHAR2(200);
BEGIN
    comanda := 'UPDATE emp_*** SET salary = salary + :a WHERE
job_id='''SA_MAN'''
                RETURNING last_name INTO :b';
    EXECUTE IMMEDIATE comanda
    USING valoare RETURNING BULK COLLECT INTO v_tab;
    FOR i IN v_tab.FIRST.. v_tab.LAST LOOP
        DBMS_OUTPUT.PUT_LINE (v_tab (i));
    END LOOP;
END;
/

```

Exemplul 2

```

DECLARE
    TYPE t_nr   IS TABLE OF NUMBER;
    TYPE t_nume IS TABLE OF VARCHAR2(30);
    nr  t_nr;
    nume t_nume;
BEGIN
    nr := t_nr(110, 120, 130, 140, 150);
    FORALL i IN 1..5
        EXECUTE IMMEDIATE
            'UPDATE emp_*** SET salary = salary*1.1
            WHERE employee_id = :1

```

```
        RETURNING last_name INTO :2'  
    USING nr(i) RETURNING BULK COLLECT INTO nume;  
    FOR i IN nume.FIRST..nume.LAST LOOP  
        DBMS_OUTPUT.PUT_LINE (nume (i));  
    END LOOP;  
END;  
/
```