

### Exemplul 7.1

```
pret_preferential.id_client_j%type,  
pret_preferential.id_categorie%type)
```

```
SELECT *  
FROM pret_preferential  
WHERE id_client_j = pc_client  
AND id_categorie = pc_categorie  
ORDER BY data_in DESC;
```

```
pret_preferential.id_client_j%type,  
pret_preferential.id_categorie%type)
```

```
pret_preferential.id_client_j%type,  
pret_preferential.id_categorie%type);
```

```
p_discount pret_preferential.discount%type,  
p_data_in DATE,  
p_data_sf DATE,  
p_categorie pret_preferential.id_categorie%type,  
p_client pret_preferential.id_client_j%type);
```

/

```
-- verifica daca un client are deja un pret  
-- preferential pentru acea categorie
```

```
pf_client pret_preferential.id_client_j%type,  
pf_categorie pret_preferential.id_categorie%type)
```

```

SELECT COUNT(*) INTO rezultat
FROM   pret_preferential
WHERE  id_client_j = pf_client
AND    id_categorie = pf_categorie
AND    SYSDATE BETWEEN data_in AND data_sf;

-- afiseaza preturile preferentiale avute pentru acea
-- categorie

pp_client      pret_preferential.id_client_j%type,
pp_categorie   pret_preferential.id_categorie%type)

                                LOOP

DBMS_OUTPUT.PUT_LINE('Pret preferential activ:');
DBMS_OUTPUT.PUT_LINE('Discount de '||
    i.discount*100 ||'% valabil in perioada '||
    i.data_in||' - '||i.data_sf);
DBMS_OUTPUT.PUT_LINE('Preturi pref. avute:');
ELSE
    DBMS_OUTPUT.PUT_LINE('Discount de '||
        i.discount*100 ||'% valabil in perioada '||
        i.data_in||' - '||i.data_sf);
    END IF;
END LOOP;

p_discount     pret_preferential.discount%type,
p_data_in      DATE,
p_data_sf      DATE,
p_categorie    pret_preferential.id_categorie%type,
p_client       pret_preferential.id_client_j%type)

DBMS_OUTPUT.PUT_LINE('Clientul are deja pret
    preferential pentru aceasta categorie');

```

```

                                ,p_discount,p_data_in,
                                p_data_sf,p_categorie,p_client);
DBMS_OUTPUT.PUT_LINE('Adaugare cu succes');

```

```

/

```

### Exemplul 7.2

```

--utilizarea pachetului

                                (0.1,SYSDATE,
                                ADD_MONTHS(SYSDATE,3),500,260);

BEGIN
    IF                                THEN

        DBMS_OUTPUT.PUT_LINE('Discount de '||
            i.discount*100 ||'% valabil in perioada '||
            i.data_in||' - '||i.data_sf);

    ELSE
        DBMS_OUTPUT.PUT_LINE('Clientul nu a avut pret
            preferential pentru aceasta categorie');
    END IF;
END;
/

```

### Exemplul 7.3

```
        produse.denumire%TYPE;
        produse.pret_unitar%TYPE;

SELECT
INTO
FROM   produse WHERE id_produs = p_id;
DBMS_OUTPUT.PUT_LINE('Produsul cautat este : '||
        v_den||' are pretul ' ||v_pret);

contor NUMBER := 0;

        (SELECT denumire, pret_unitar
FROM   produse
                                ) LOOP

        DBMS_OUTPUT.PUT_LINE(i.denumire||' are pretul ' ||
                                i.pret_unitar);
END LOOP;

        DBMS_OUTPUT.PUT_LINE('Nu exista produsul cautat');
ELSE
        DBMS_OUTPUT.PUT_LINE('Au fost gasite '||          ||
                                ' produse');
END IF;
END;
END;
/

EXECUTE                                (1000);

EXECUTE pack_ex3.afiseaza_produs('etichete');
```

#### Exemplul 7.4 - pachete fără corp

```
CREATE OR REPLACE PACKAGE constante IS

    cm_in_inch
    inch_in_cm

END;
/

DECLARE
    v_diagonala NUMBER := &p_diagonala;
BEGIN
    DBMS_OUTPUT.PUT_LINE('Diagonala de ' || v_diagonala ||
        'inch este echivalenta cu ' ||
                                || 'cm');

END;
/
```

#### Exemplul 7.5 - pachete fără corp

```
CREATE OR REPLACE PACKAGE exceptii IS

END;
/

DECLARE
    v_id NUMBER(4) := &p_id;
    x VARCHAR2(100);
BEGIN
    SELECT denumire INTO x
    FROM   produse
    WHERE  id_produs = v_id;
EXCEPTION
    WHEN                                     THEN
        DBMS_OUTPUT.PUT_LINE('Nu exista produsul specificat');
        DBMS_OUTPUT.PUT_LINE(
                                );
END;
/
```

### Exemplul 7.6 - persistența variabilelor

```
CREATE OR REPLACE PACKAGE variabila_globala
IS
    v VARCHAR2(100)
END;
/

-- 2 sesiuni - 2 utilizatori
--sesiune 1 - utilizator1
BEGIN

    ||' in sesiunea 1';
    DBMS_OUTPUT.PUT_LINE(variabila_globala.v);
END;
/

BEGIN
    DBMS_OUTPUT.PUT_LINE(variabila_globala.v);
END;
/

--sesiune 2 - utilizator2
BEGIN

    ||' in sesiunea 2';
    DBMS_OUTPUT.PUT_LINE(
        curs_plsql.variabila_globala.v);
END;
/

BEGIN
    DBMS_OUTPUT.PUT_LINE(
        curs_plsql.variabila_globala.v);
END;
/

-- 2 sesiuni - acelasi utilizator
--sesiune 1 - utilizator1
--acelasi cod ca mai sus

--sesiune 2 - utilizator1

    ||' in sesiunea 2';
    DBMS_OUTPUT.PUT_LINE(variabila_globala.v);
/
```

### Exemplul 7.9

```
BEGIN
  DBMS_OUTPUT.PUT('Astazi ');
  DBMS_OUTPUT.PUT('este ');
  DBMS_OUTPUT.PUT(SYSDATE);
  DBMS_OUTPUT.NEW_LINE;
END;
/

BEGIN
  DBMS_OUTPUT.PUT_LINE('Astazi este ' || SYSDATE);

END;
/
```

### Exemplul 7.10

```
DECLARE
-- paramentrii de tip OUT pt procedura GET_LINE

    linie2 VARCHAR2(255);
    stare2 INTEGER;
    linie3 VARCHAR2(255);
    stare3 INTEGER;

v_id          NUMBER(4);
v_denumire    VARCHAR2(100);
v_pret        produse.pret_unitar%TYPE;

BEGIN
    SELECT id_produc,denumire, pret_unitar
    INTO    v_id, v_denumire, v_pret
    FROM    produse
    WHERE   id_produc = 1000;

-- se introduce o linie in buffer fara caracter
-- de terminare linie

        (' 1 - '||v_id|| ' ');

-- se incearca extragerea liniei introdusa
-- in buffer si starea acesteia

        (linie1,stare1);

-- se depunde informatie pe aceeasi linie in buffer

        (' 2 - '||v_denumire|| ' ');

-- se inchide linia depusa in buffer si se extrage
-- linia din buffer

        (linie2,stare2);

-- se introduc informatii pe aceeasi linie
-- si se afiseaza informatia

        (' 3 - ID: ' ||v_id|| ' Pret: '
         || v_pret);
        (linie3,stare3);

-- se afiseaza ceea ce s-a extras
    DBMS_OUTPUT.PUT_LINE('linie1 = '|| linie1||
                          '; stare1 = '||stare1);
    DBMS_OUTPUT.PUT_LINE('linie2 = '|| linie2||
                          '; stare2 = '||stare2);
    DBMS_OUTPUT.PUT_LINE('linie3 = '|| linie3||
                          '; stare3 = '||stare3);

END;
/
```



### Exemplul 7.11

```
DECLARE
-- parametru de tip OUT pentru NEW_LINES
-- colectie de siruri de caractere

-- parametru de tip IN OUT pentru NEW_LINES

v_id          NUMBER(4);
v_denumire    VARCHAR2(100);
v_pret        produse.pret_unitar%TYPE;

BEGIN
    SELECT id_produc,denumire, pret_unitar
    INTO    v_id, v_denumire, v_pret
    FROM    produse
    WHERE   id_produc = 1000;

-- se maresc dimensiunea bufferului
                                (1000000);

    DBMS_OUTPUT.PUT(' 1 - ID: '||v_id|| ' ');
    DBMS_OUTPUT.PUT(' 2 - DEN: '||v_denumire|| ' ');
    DBMS_OUTPUT.NEW_LINE;
    DBMS_OUTPUT.PUT_LINE(' 3 - ID: ' ||v_id||
                        ' PRET: ' || v_pret);
    DBMS_OUTPUT.PUT_LINE(' 4 - ID: ' ||v_id||
                        ' DEN: '|| v_denumire||' PRET: ' ||v_pret);
-- se afiseaza ceea ce s-a extras

    DBMS_OUTPUT.PUT_LINE('In buffer sunt '||
                        nr_linii ||' linii');

    DBMS_OUTPUT.put_line('Linia '||i ||': '||

END;
/
```

### Exemplul 7.12

```
CREATE OR REPLACE PROCEDURE valoare_vanzari_per_zi
IS
    valoare NUMBER(10);
BEGIN
    SELECT                                INTO valoare
    FROM    facturi_produce fp, facturi f
    WHERE   f.id_factura = fp.id_factura
    AND     data=SYSDATE;
    DBMS_OUTPUT.PUT_LINE('Pana la aceasta ora '||
        's-au efectuat vanzari in valoare de '||
        valoare ||'lei');
END;
/
--varianta 1 - definire job

BEGIN

    -- întoarce numărul jobului,
    -- printr-o variabilă de legătură

    -- codul PL/SQL care trebuie executat

    -- data de start a execuției (dupa 3 secunde)

    -- intervalul de timp la care se repetă
    -- execuția = 3secunde

END;
/

-- numarul jobului

-- informatii despre joburi
SELECT
FROM

-- lansarea jobului la momentul dorit
BEGIN
    -- presupunand ca jobul are codul 90 atunci:

END;
/
```

```

-- stergerea unui job
BEGIN

END;
/

SELECT JOB, NEXT_DATE, WHAT
FROM    USER_JOBS;

--varianta 2 - definire job

CREATE OR REPLACE PACKAGE pachet_job
IS

END;
/

CREATE OR REPLACE PACKAGE body pachet_job
IS
    FUNCTION obtine_job RETURN NUMBER IS
    BEGIN

        END;
END;
/

BEGIN

    -- întoarce numărul jobului,
    -- printr-o variabilă de legătură

    -- codul PL/SQL care trebuie executat
    WHAT => 'valoare_vanzari_per_zi;',

-- data de start a execuției (dupa 3 secunde)
    NEXT_DATE => SYSDATE+3/86400,

    -- intervalul de timp la care se repetă
    -- execuția = 3secunde
    INTERVAL => 'SYSDATE+3/86400');
END;
/

-- informatii despre joburi
SELECT JOB, NEXT_DATE, WHAT
FROM    USER_JOBS
WHERE

```

```

-- lansarea jobului la momentul dorit
BEGIN

END;
/
-- stergerea unui job
BEGIN

END;
/

SELECT JOB, NEXT_DATE, WHAT
FROM    USER_JOBS
WHERE   JOB = pachet_job.obtine_job;

```

#### Exemplul 7.14

```

CREATE OR REPLACE PROCEDURE scriu_fisier
(
)
IS

CURSOR c IS
SELECT
        denumire,
        SUM(cantitate) cantitate_totala
FROM    facturi_produce fp, produse p
WHERE   fp.id_produc = p.id_produc
GROUP BY denumire
HAVING  SUM(cantitate) > 500
ORDER BY SUM(cantitate) DESC;
v_lista c%ROWTYPE;
BEGIN

        'CANTITATI VANDUTE PER PRODUS

        SYSDATE);

UTL_FILE.NEW_LINE(v_file);
UTL_FILE.NEW_LINE(v_file);
UTL_FILE.PUTF(v_file,
        'Denumire produs          Cantitate');
UTL_FILE.NEW_LINE(v_file);
UTL_FILE.PUTF(v_file,
        '-----');

OPEN c;
LOOP
        FETCH c INTO v_lista;
        EXIT WHEN c%NOTFOUND;

```

```

        UTL_FILE.NEW_LINE(v_file);

        UTL_FILE.PUT(v_file, ' ');
        UTL_FILE.PUT(v_file, v_lista.cantitate_totala);
    END LOOP;
    CLOSE c;

END;
/
--creare director la nivelul sistemului de operare
--D:/OracleW7/Directory/curs_plsql

--setare valoare parametru de initializare Oracle
--utl_file_dir = D:/OracleW7/Directory/curs_plsql

--atribuire privilegiu CREATE ANY DIRECTORY
--utilizatorului
--conectare user sys
GRANT CREATE ANY DIRECTORY TO curs_plsql;

--conectare user curs_plsql
--definire director
CREATE DIRECTORY curs_plsql
AS 'D:/OracleW7/Directory/curs_plsql';

--apel procedura
(
    'D:\OracleW7\Directory\curs_plsql',

```

### Exemplul 7.15

```

DECLARE
    comanda      VARCHAR2(200);
    nume_cursor  NUMBER;
BEGIN
    comanda :=
    nume_cursor :=

        (nume_cursor, comanda,

        );

        (nume_cursor);

END;
/

```

### Exemplul 7.16

```
CREATE OR REPLACE PROCEDURE sterge_linii
(
                                ) AS
  nume_cursor INTEGER;
BEGIN
  nume_cursor := DBMS_SQL.OPEN_CURSOR;
                    (nume_cursor,
                                ,
                                );
  nr_linii :=                                (nume_cursor);
                    (nume_cursor);
END;
/
-- DBMS_SQL.V7 reprezintă modul (versiunea 7)
-- în care Oracle -- tratează comenzile SQL

VARIABLE linii_sterse NUMBER

PRINT linii_sterse
```

### Exemplul 7.17

```
DECLARE
  sir VARCHAR2(50);
  bloc VARCHAR2(500);
BEGIN
  -- creare tabel

  --inserare in tabel
  FOR i IN 1..10 LOOP
    sir := 'INSERT INTO tabel
            VALUES (''Contor ' || i || ''')';

  END LOOP;
  -- tiparire continut tabel
  bloc :=
    FOR i IN (SELECT * FROM tabel) LOOP
      DBMS_OUTPUT.PUT_LINE (i.col);
    END LOOP;

  -- stergere tabel

END;
/
```

### Exemplul 7.18

```
CREATE OR REPLACE PROCEDURE
    sterg_tabel(nume VARCHAR2) IS
BEGIN
    EXECUTE IMMEDIATE

    -- EXECUTE IMMEDIATE
    --
    --
END;
/
EXECUTE sterg_tabel(                )

CREATE OR REPLACE PROCEDURE sterg_tabele
IS
BEGIN
    FOR I IN (SELECT
                FROM
                ) LOOP

        ;

    END LOOP;
END;
/
```

### Exemplul 7.19

```
DECLARE
    TYPE tip_imb IS TABLE OF
        produse.id_produș%TYPE;
    t            tip_imb;
    v_procent    NUMBER(3,2) := &p_procent;
    v_categorie  VARCHAR2(50) := '&p_categorie';
    comanda      VARCHAR2(500);
BEGIN

    SET pret_unitar =

    WHERE id_categorie =
        (SELECT id_categorie
         FROM   categorii
         )

EXECUTE IMMEDIATE

FOR i IN 1..t.LAST LOOP
    DBMS_OUTPUT.PUT_LINE (t(i));
END LOOP;
END;
/
```

### Exemplul 7.20

```
DECLARE
    TYPE tip_imb IS TABLE OF produse%ROWTYPE;
    t              tip_imb;
    v_categorie VARCHAR2(50) := '&p_categorie';
    comanda       VARCHAR2(500);
BEGIN

    FROM   produse
    WHERE  id_categorie =
           (SELECT id_categorie
            FROM   categorii

EXECUTE IMMEDIATE

FOR i IN 1..t.LAST LOOP
    DBMS_OUTPUT.PUT_LINE (t(i).denumire);
END LOOP;
END;
/
```

### Exemplul 7.21

```
CREATE OR REPLACE PROCEDURE
    colecteaza_statistici
    (p_obiect_tip  IN VARCHAR2,
     p_obiect_nume IN VARCHAR2)
IS
BEGIN

END;
/

EXECUTE colecteaza_statistici (

SELECT
    EMPTY_BLOCKS, AVG_ROW_LEN
FROM   USER_TABLES
WHERE  TABLE_NAME = 'PRODUSE';
```