Exemplul 6.2

```
CREATE OR REPLACE PROCEDURE proc_ex2
 (v_id IN produse.id_produs%TYPE, v_procent NUMBER)
AS
  exceptie EXCEPTION;
BEGIN
  UPDATE  produse
  SET     pret_unitar =
          ROUND(pret_unitar + pret_unitar*v_procent,2)
  WHERE   id_produs = v_id;

  IF SQL%NOTFOUND THEN
     RAISE exceptie;
  END IF;

EXCEPTION
  WHEN exceptie THEN
  RAISE_APPLICATION_ERROR (-20000,'Nu exista produsul');
END;
/
```

Exemplul 6.5

```
DECLARE
   v_den  categorii.denumire%TYPE := '&p_den';
   v_nivel categorii.nivel%TYPE := &p_niv;
   v_parinte categorii.id_parinte%TYPE;

   PROCEDURE proc_ex5
     (id categorii.id_categorie%TYPE DEFAULT 2000,
      den  categorii.denumire%TYPE,
      nivel categorii.nivel%TYPE,
      parinte categorii.id_parinte%TYPE)
   IS
   BEGIN
      INSERT INTO categorii
      VALUES(id,den,nivel,parinte);
      DBMS_OUTPUT.PUT_LINE('Au fost inserate '||
                SQL%ROWCOUNT ||' linii');
   END;
BEGIN
   v_parinte := &p_parinte;
   proc_ex5(den=>v_den,nivel=>v_nivel,
            parinte=>v_parinte);
END;
/
```

Exemplul 6.6

```
CREATE OR REPLACE PROCEDURE proc_ex6
(p_id  IN categorii.id_categorie%TYPE,
 p_den OUT categorii.denumire%TYPE,
 p_parinte OUT categorii.id_parinte%TYPE) IS
BEGIN
  SELECT denumire, id_parinte INTO p_den,p_parinte
  FROM   categorii
  WHERE  id_categorie = p_id;
END;
/

DECLARE
  v_den     categorii.denumire%TYPE;
  v_parinte categorii.id_parinte%TYPE;
BEGIN
  proc_ex6(369, v_den, v_parinte);
  DBMS_OUTPUT.PUT_LINE(v_den || ' ' ||v_parinte);
END;
/
```

Exemplul 6.7

```
CREATE OR REPLACE PROCEDURE proc_ex6
(p_id  IN OUT categorii.id_categorie%TYPE,
 p_den OUT categorii.denumire%TYPE) IS
BEGIN
  SELECT denumire, id_parinte INTO p_den, p_id
  FROM   categorii
  WHERE  id_categorie = p_id;
END;
/

DECLARE
  v_den categorii.denumire%TYPE;
  v_id categorii.id_parinte%TYPE := 369;
BEGIN
  proc_ex6(v_id, v_den);
  DBMS_OUTPUT.PUT_LINE(v_den || ' ' ||v_id);
END;
/
```

Exemplul 6.8

```
CREATE FUNCTION func_ex8
(p_id IN clienti.id_client%TYPE,
 p_an NUMBER DEFAULT 2000)
RETURN NUMBER
IS
 rezultat NUMBER;
BEGIN
  SELECT COUNT(DISTINCT id_produs) INTO rezultat
  FROM   facturi f, facturi_produse fp
  WHERE  EXTRACT(YEAR FROM data) = p_an
  AND    id_client = p_id;

  RETURN rezultat;
END;
/
```

Exemplul 6.14

```
CREATE OR REPLACE FUNCTION func_ex14(v_id NUMBER)
RETURN VARCHAR2
IS
  rez VARCHAR2(150):='';
BEGIN
  FOR i IN (SELECT denumire
            FROM   categorii
            START WITH id_categorie = v_id
            CONNECT BY   PRIOR id_parinte  =
                         id_categorie) LOOP
      rez := i.denumire||'/'||rez ;
  END LOOP;

  RETURN rez;
END;
/

SELECT id_produs, denumire,
       func_ex14(id_categorie) AS arbore
FROM   produse
WHERE  ROWNUM <=10;
```

Exemplul 6.15

```
DECLARE
  v_nr_trimestru NUMBER(10);
  v_nr_luna      NUMBER(10);

 FUNCTION nr_produse (p_id produse.id_produs%TYPE,
                      p_luna NUMBER)
    RETURN NUMBER IS
    rezultat NUMBER(10);
 BEGIN
    SELECT SUM(cantitate)
    INTO   rezultat
    FROM   facturi f, facturi_produse fp
    WHERE  f.id_factura = fp.id_factura
    AND    fp.id_produs = p_id
    AND    EXTRACT(MONTH FROM data)=p_luna;
    RETURN rezultat;
 END;

 FUNCTION nr_produse (p_id produse.id_produs%TYPE,
                      p_trimestru CHAR)
    RETURN NUMBER IS
    rezultat NUMBER(10);
 BEGIN
    SELECT SUM(cantitate)
    INTO   rezultat
    FROM   facturi f, facturi_produse fp
    WHERE  f.id_factura = fp.id_factura
    AND    fp.id_produs = p_id
    AND    TO_CHAR(data,'q') = p_trimestru;
    RETURN rezultat;
 END;

BEGIN
  v_nr_luna := nr_produse(1275,4);
  DBMS_OUTPUT.PUT_LINE('Nr produse vandute in luna
       aprilie: '||v_nr_luna);
  v_nr_trimestru := nr_produse(1275,'2');
  DBMS_OUTPUT.PUT_LINE('Nr produse vandute in
       trimestrul 2: '||v_nr_trimestru);
END;
/
```

Exemplul 6.16

```
CREATE OR REPLACE FUNCTION fibonacci(n NUMBER)
   RETURN NUMBER RESULT_CACHE IS
BEGIN
  IF (n =0) OR (n =1) THEN
    RETURN 1;
  ELSE
    RETURN fibonacci(n - 1) + fibonacci(n - 2);
  END IF;
END;
/
```