

### Exemplul 8.1

```
                                trig_ex1

BEGIN
  IF (TO_CHAR(SYSDATE,'D') = 1)
    OR (TO_CHAR(SYSDATE,'HH24') NOT BETWEEN 8 AND 20)
  THEN

                                , 'Operatiile asupra

                                tabelului sunt permise doar in
                                programul de lucru!');

  END IF;
END;
/
--stornare factura status = -2:

INSERT INTO facturi(id_factura, id_casa, id_client,
                    data, status, id_tip_plata)
VALUES (171, 402, 94, SYSDATE, -2, 10);

                                trig_ex1;
```

### Exemplul 8.2

```
CREATE OR REPLACE TRIGGER trig_ex2
  AFTER INSERT OR DELETE OR UPDATE on facturi
BEGIN
  IF (TO_CHAR(SYSDATE,'D') = 1)
    OR (TO_CHAR(SYSDATE,'HH24') NOT BETWEEN 8 AND 20)
  THEN

    RAISE_APPLICATION_ERROR(-20001, '
    este permisa doar in timpul programului de lucru!');

    RAISE_APPLICATION_ERROR(-20002, '
    este permisa doar in timpul programului de lucru!');

    RAISE_APPLICATION_ERROR(-20003, '
    sunt permise doar in timpul programului de lucru!');

  END IF;
END;
/
```

### Exemplul 8.3

#### --varianta1

```
CREATE OR REPLACE TRIGGER trig1_ex3
```

```
BEGIN
```

```
        (-20000, 'Nu puteti modifica  
        seria casei fiscale!');
```

```
END;
```

```
/
```

```
UPDATE case
```

```
SET      serie = serie||'_';
```

#### --varianta2

```
CREATE OR REPLACE PROCEDURE proc_trig_ex3
```

```
IS
```

```
BEGIN
```

```
    RAISE_APPLICATION_ERROR (-20000, 'Nu puteti modifica  
    seria casei fiscale!');
```

```
END;
```

```
/
```

```
CREATE OR REPLACE TRIGGER trig2_ex3
```

```
    BEFORE UPDATE OF serie ON case
```

```
    FOR EACH ROW
```

```
    WHEN (NEW.serie <> OLD.serie)
```

```
BEGIN
```

```
END;
```

```
/
```

#### --varianta3

```
CREATE OR REPLACE TRIGGER trig3_ex3
```

```
    BEFORE UPDATE OF serie ON case
```

```
    FOR EACH ROW
```

```
    WHEN (NEW.serie <> OLD.serie)
```

```
/
```

#### --varianta4

```
CREATE OR REPLACE TRIGGER trig4_ex3
```

```
    BEFORE UPDATE OF serie ON case
```

```
    FOR EACH ROW
```

```
BEGIN
```

```
    RAISE_APPLICATION_ERROR (-20000, 'Nu puteti  
    modifica seria casei fiscale!');
```

```
END;
```

```
/
```

#### Exemplul 8.4

```
CREATE OR REPLACE TRIGGER verifica_stoc

                                produse.stoc_curent%TYPE;

BEGIN
    SELECT
        INTO    v_limita
        FROM    produse
        WHERE   id_produc IN

        IF
        THEN
            RAISE_APPLICATION_ERROR(-20000,'Se depaseste '||
                'stocul impus. Cantitate permisa '||v_limita);
        END IF;
END;
/
```

#### Exemplul 8.5

```
CREATE OR REPLACE

(        produse.id_produc%TYPE,                NUMBER)
IS
BEGIN
    UPDATE produse
    SET    stoc_curent = stoc_curent +
    WHERE  id_produc =          ;
END;
/

CREATE OR REPLACE TRIGGER actualizeaza_stoc

BEGIN
    IF                THEN

    ELSIF                THEN

    ELSE

END IF;
END;
/
```

### Exemplul 8.6

```
-- coloana este populata cu null sau cu 0?
ALTER TABLE categorii
ADD nr_produce NUMBER DEFAULT 0;

-- coloana este populata cu null sau cu 0?
UPDATE categorii c
SET     nr_produce =
        (SELECT
          FROM   produse
          WHERE  id_categorie =
                                );

                                info_categorii_produce

SELECT p.*, c.denumire                                , nivel,
       id_parinte, nr_produce
FROM   produse p, categorii c
WHERE  p.id_categorie = c.id_categorie;

CREATE OR REPLACE TRIGGER actualizeaza_info

DECLARE

BEGIN

    IF                                THEN

        SELECT COUNT(*) INTO
        FROM   categorii
        WHERE  id_categorie =                                ;

        IF                                THEN

            VALUES (
                                ,:NEW.categ_denumire,
                                :NEW.nivel, :NEW.id_parinte, 1);

            VALUES (:NEW.id_produc, :NEW.denumire,
                    :NEW.descriere, :NEW.stoc_curent,
                    :NEW.stoc_impus, :NEW.pret_unitar,
                    :NEW.greutate, :NEW.volum, :NEW.tva,
                    :NEW.id_zona, :NEW.id_um,
                    :NEW.id_categorie, SYSDATE, SYSDATE,
                    :NEW.activ);

        ELSE

            VALUES (:NEW.id_produc, :NEW.denumire,
                    :NEW.descriere, :NEW.stoc_curent,
                    :NEW.stoc_impus, :NEW.pret_unitar,
```

```

        :NEW.greutate, :NEW.volum,
        :NEW.tva, :NEW.id_zona, :NEW.id_um,
        :NEW.id_categorie, SYSDATE, SYSDATE,
        :NEW.activ);

        SET      nr_produce =
        WHERE id_categorie =
    END IF;
ELSIF                                THEN

        WHERE id_produc =

        SET      nr_produce =
        WHERE id_categorie =

    ELSIF                                THEN

        SET      id_categorie =
        WHERE id_produc =

        SET      nr_produce =
        WHERE id_categorie =

        SET      nr_produce =
        WHERE id_categorie =

    ELSIF                                THEN

        SET      denumire =
        WHERE id_produc =
    ELSE
        RAISE_APPLICATION_ERROR(-20000,'Ai voie sa
        actualizezi doar categoria sau denumirea
        produsului!');
    END IF;
END;
/

```

### Exemplul 8.7

```
CREATE TABLE audit_user
(nume_bd          VARCHAR2(50),
 user_logat       VARCHAR2(30),
 eveniment        VARCHAR2(20),
 tip_object_referit VARCHAR2(30),
 nume_object_referit VARCHAR2(30),
 data             TIMESTAMP(3));

CREATE OR REPLACE TRIGGER audit_schema

BEGIN
    INSERT INTO audit_user
    VALUES (

END;
/

CREATE TABLE tabel (coloana_1 number(2));
ALTER TABLE tabel ADD (coloana_2 number(2));
INSERT INTO tabel VALUES (1,2);
CREATE INDEX ind_tabel ON tabel(coloana_1);

SELECT * FROM audit_user;
```

### Exemplul 8.8

```
CREATE TABLE log_user(nume_user  VARCHAR2(30),
                      data         TIMESTAMP,
                      moment        VARCHAR2(20));

CREATE OR REPLACE PROCEDURE insert_log IS
BEGIN
    INSERT INTO log_user
    VALUES (SYS.LOGIN_USER, SYSTIMESTAMP, 'after logon');
END;
/

CREATE OR REPLACE TRIGGER logon_logoff_after

/

CREATE OR REPLACE TRIGGER logon_logoff_before

BEGIN

    VALUES (SYS.LOGIN_USER, SYSTIMESTAMP, 'before logoff');
END;
/

SELECT * FROM log_user;
```

### Exemplul 8.9

```
CREATE TABLE erori
(nume_bd          VARCHAR2(50),
 user_logat       VARCHAR2(30),
 data             TIMESTAMP(3),
 eroare           VARCHAR2(2000));

CREATE OR REPLACE TRIGGER log_erori

BEGIN
    INSERT INTO erori
    VALUES (SYS.DATABASE_NAME, SYS.LOGIN_USER,
            SYSTIMESTAMP,
            );
END;
/

CREATE TABLE a (id NUMBER(2));
INSERT INTO a VALUES (123);
ALTER TABLE a DROP (b);
SELECT * FROM abc;

SELECT * FROM erori;
```

### Exemplul 8.10

```
CREATE TABLE erori
(nume_bd          VARCHAR2(50),
 user_logat       VARCHAR2(30),
 data             TIMESTAMP(3),
 eroare           VARCHAR2(2000));

CREATE OR REPLACE TRIGGER log_eroare

BEGIN
    IF                                THEN
        INSERT INTO erori
        VALUES (SYS.DATABASE_NAME, SYS.LOGIN_USER,
                SYSTIMESTAMP,
                DBMS_UTILITY.FORMAT_ERROR_STACK);
    END IF;
END;
/

ALTER TABLE ab DROP (b);
SELECT * FROM abc;

SELECT * FROM erori;
```

### Exemplul 8.11

```
CREATE OR REPLACE PROCEDURE create_trigger
    (v_nume VARCHAR2)
IS

BEGIN
    SELECT
    INTO
    FROM
    WHERE

    DBMS_OUTPUT.PUT (

    DBMS_OUTPUT.PUT_LINE (      );
end;
/

EXECUTE create_trigger('verifica_stoc')
```

### Exemplul 8.12

```
-- Trigger-ul realizeaza actualizari in cascada:
-- actualizarea cheii primare din tabelul parinte
-- determina
-- actualizarea cheii externe din tabelul copil

CREATE OR REPLACE TRIGGER modifica_copil

BEGIN
    UPDATE

    SET

    WHERE
END;
/
-- constrangerea de cheie externa este definita
-- (cu optiuni la stegere sau nu)
-- exista produse in categoria 428
-- actualizarea urmatoare este permisa
UPDATE    categorii
SET       id_categorie = 7000
WHERE     id_categorie = 428;
```



### Exemplul 8.13

```
-- Trigger-ul realizeaza actualizari in cascada:  
-- actualizarea cheii externe din tabelul copil  
-- determina  
-- actualizarea cheii primare din tabelul parinte  
  
CREATE OR REPLACE TRIGGER modifica_parinte  
  
BEGIN  
    UPDATE categorii  
    SET     id_categorie = :NEW.id_categorie  
    WHERE  id_categorie = :OLD.id_categorie;  
END;  
/  
  
--actualizarea urmatoare este permisa  
UPDATE     produse  
SET        id_categorie = 7000  
WHERE      id_categorie = 428;
```

### Exemplul 8.14

```
-- daca ambii trigger-i definiti anterior ar fi activi  
-- simultan, atunci urmatoarele comenzi nu ar fi  
-- permise  
-- eroarea aparuta  
-- "table is mutating,  
-- trigger/function may not see it"  
  
UPDATE     categorii  
SET        id_categorie = 7000  
WHERE      id_categorie = 428;  
  
UPDATE     produse  
SET        id_categorie = 7000  
WHERE      id_categorie = 428;
```

### Exemplul 8.15

```
CREATE OR REPLACE TRIGGER trig_ex15

DECLARE
    v_denumire VARCHAR2(50);
BEGIN
    SELECT denumire INTO v_denumire
    FROM    categorii
    WHERE   id_categorie = :OLD.id_categorie;
END;
/
--trigger-ul consulta tabelul de care este asociat
--comanda urmatoare nu este permisa
--eroarea aparuta "table is mutating
--trigger/function may not see it"

DELETE FROM categorii WHERE id_categorie = 428;

-- comanda urmatoare este permisa
-- (categoria 7000 nu exista in tabel)
DELETE FROM categorii WHERE id_categorie = 7000;
```

### Exemplul 8.16

```
-- Trigger-ul realizeaza stergeri in cascada:
-- stergerea unei inregistrari din tabelul parinte
-- determina
-- stergerea inregistrarilor copil asociate
CREATE OR REPLACE TRIGGER sterge_copil

BEGIN

END;
/
-- Cazul 1 - constrangerea de cheie externa nu are
-- optiuni de stergere specificate
-- urmatoarea comanda este permisa
DELETE FROM categorii WHERE id_categorie = 428;

-- Cazul 2 - constrangerea de cheie externa are
-- optiuni de stergere (CASCADE/SET NULL)
-- comanda urmatoare nu este permisa
-- eroarea aparuta
-- "table is mutating,
-- trigger/function may not see it"
DELETE FROM categorii WHERE id_categorie = 428;
```

### Exemplul 8.17

```
--Varianta 1
CREATE OR REPLACE TRIGGER trig_17

DECLARE
    nr NUMBER(1);
BEGIN
    SELECT
    FROM
    WHERE
    AND

    IF          THEN
        RAISE_APPLICATION_ERROR(-20000,'Clientul are
        deja numarul maxim de promotii permis anual');
    END IF;
END;
/
-- clientul 10 are deja 3 promotii asociate
-- apare mesajul din trigger
INSERT INTO pret_preferential
VALUES (101,0.1, sysdate,sysdate+30, 500, 10);

-- clientul 20 are doar 2 promotii asociate
-- linia este inserata
INSERT INTO pret_preferential
VALUES (101,0.1, sysdate,sysdate+30, 500, 20);
rollback;

--comenzile urmatoare determina eroare mutating
INSERT INTO pret_preferential
SELECT 101,0.1, sysdate,sysdate+30, 500, 20
FROM    DUAL;

UPDATE pret_preferential
SET     id_client_j = 120
WHERE   id_client_j = 70;
```

```

--Varianta 2
CREATE OR REPLACE PACKAGE pachet
AS
    TYPE tip_rec IS RECORD
        (id pret_preferential.id_client_j%TYPE,
         nr NUMBER(1));
    TYPE tip_ind IS TABLE OF tip_rec
        INDEX BY PLS_INTEGER;
    t tip_ind;
    contor NUMBER(2) := 0;
END;
/

CREATE OR REPLACE TRIGGER trig_17_comanda

BEGIN

    SELECT          COUNT(*)

    FROM    pret_preferential
    WHERE   EXTRACT(YEAR FROM data_in) =
            EXTRACT(YEAR FROM SYSDATE)

END;
/

CREATE OR REPLACE TRIGGER trig_17_linie

BEGIN
    FOR i in 1..pachet.t.last LOOP
        IF

            THEN
                RAISE_APPLICATION_ERROR(-20000,'Clientul '||
                    :NEW.id_client_j||' depaseste numarul '||
                    'maxim de promotii permis anual');
            END IF;
        END LOOP;

END;
/

```

```
-- linia este inserata
INSERT INTO pret_preferential
VALUES (102,0.1, sysdate,sysdate+30, 500, 120);
-- linia este inserata
INSERT INTO pret_preferential
SELECT 103,0.1, sysdate,sysdate+30, 501, 120
FROM    DUAL;

-- se depaseste limita impusa
-- apare mesajul din trigger
INSERT INTO pret_preferential
SELECT * FROM pret_pref;

UPDATE pret_preferential
SET    id_client_j = 120
WHERE  id_client_j = 40;

UPDATE pret_preferential
SET    id_client_j = 210
WHERE  id_client_j in (40, 130,140);
```