Triggeri
- eliminare mutating table

Sa se implementeze cu ajutorul unui declansator restrictia ca intr-un departament pot lucra maximum 45 de angajati.

```
CREATE OR REPLACE TRIGGER TrLimitaDep_***
AFTER INSERT ON emp_***
--la BEFORE nu apare aceasta eroare
FOR EACH ROW
DECLARE
v_Max_emp CONSTANT NUMBER := 45;
v_emp_curent NUMBER;
BEGIN
SELECT COUNT(*) INTO v_emp_curent
FROM emp_***
WHERE department_id = :NEW.department_id;
IF v_emp_curent + 1 > v_Max_emp THEN
RAISE_APPLICATION_ERROR(-20000,'Prea multi angajati in
departamentul avand codul ' || :NEW.department_id);
END IF;
END TrLimitaDep_***;
/

--Testam trigger-ul.

INSERT INTO emp_spr VALUES
(employees_seq.NEXTVAL,'Prenume','Nume','a','t',SYSDATE,'IT_PROG',
10000,0.1,100,50);

/*se va executa de mai multe ori astfel incat sa existe mai mult de 45 de angajati in
departamentul respectiv.
Obs: Declansatorul TrLimitaDep_*** consulta chiar tabelul (emp_***) la care este asociat
declansatorul (mutating).
Tabelul emp_*** este mutating doar pentru un declansator la nivel de linie.
O solutie pentru acest problema este crearea a doi declansatori, unul la nivel de linie si
altul la nivel de instructiune:
 - în declansatorul la nivel de linie se inregistreaza valoarea lui :NEW.department_id, dar
nu va fi interogat tabelul emp_***.
 - interogarea va fi facuta in declansatorul la nivel de instructiune si va folosi valoarea
inregistrata in declansatorul la nivel de linie.
O modalitate pentru a inregistra valoarea lui :NEW.department_id este utilizarea unui
tablou indexat in interiorul unui pachet.
*/

CREATE OR REPLACE PACKAGE PdepDate_***
AS
TYPE t_cod IS TABLE OF dept_***.department_id%TYPE
INDEX BY BINARY_INTEGER;
v_cod_dep t_cod;
```

```
v_NrIntrari BINARY_INTEGER := 0;
END PdepDate_***;
/

CREATE OR REPLACE TRIGGER TrLLimitaDep_***
BEFORE INSERT ON emp_***
FOR EACH ROW
BEGIN
PDepDate_***.v_NrIntrari := PDepDate_***.v_NrIntrari + 1;
PdepDate_***.v_cod_dep (PdepDate_***.v_NrIntrari) :=
:NEW.department_id;
END TrLLimitaDep_***;
/

CREATE OR REPLACE TRIGGER TrILimitaDep_***
BEFORE INSERT ON emp_***
DECLARE
v_Max_emp CONSTANT NUMBER := 45;
v_emp_curent NUMBER;
v_cod_dep dep_***.department_id%TYPE;
BEGIN

/* Parcurge fiecare departament inserat sau actualizat si
verifica daca se incadreaza in limita stabilita */

FOR v_LoopIndex IN 1..PdepDate_***.v_NrIntrari LOOP
 v_cod_dep := PdepDate_***.v_cod_dep(v_LoopIndex);
 SELECT COUNT(*)
 INTO v_emp_curent
 FROM emp_***
 WHERE department_id = v_cod_dep;
 IF v_emp_curent >= v_Max_emp THEN
  RAISE_APPLICATION_ERROR(-20000, 'Prea multi angajati in
departamentul avand codul: ' || v_cod_dep);
 END IF;
END LOOP;

/* Reseteaza contorul deoarece urmatoarea executie
va folosi date noi */

PdepDate_***.v_NrIntrari := 0;
END TrILimitaDep_***;
/

/*Obs: Aceast solutie functioneaza pentru departamentele nou introduse.
Pentru testare:
- introducem un nou departament
- introducem mai mult de 45 de angajati in departamentul inserat anterior
(eventual, cu o comanda de tip INSERT INTO ... (SELECT ...)).*/
```

Trigger care va determina stergeri si modificari in cascada
Sa se creeze un declansator care:
a) daca este eliminat un departament, va sterge toti angajatii care lucreaza in departamentul respectiv;
b) daca se schimba codul unui departament, va modifica aceasta valoare pentru fiecare angajat care lucreaza in departamentul respectiv.

```
CREATE OR REPLACE TRIGGER dep_cascada_***
BEFORE DELETE OR UPDATE OF department_id ON dept_***
FOR EACH ROW
BEGIN
IF DELETING THEN
 DELETE FROM emp_***
 WHERE department_id = :OLD.department_id;
END IF;
IF UPDATING AND :OLD.department_id != :NEW.department_id THEN
 UPDATE emp_***
 SET department_id = :NEW.department_id
 WHERE department_id = :OLD.department_id;
END IF;
END dep_cascada_***;

/* Declansatorul anterior realizeaza constrangerea de integritate
UPDATE sau ON DELETE CASCADE, adica stergerea sau modificarea
cheii primare a unui tabel „parinte“ se va reflecta si asupra
inregistrărilor corespunzătoare din tabelul „copil“.

Executarea acestuia, pe tabelul dept_*** (tabelul „parinte“), va
duce la efectuarea a doua tipuri de operatii pe tabelul emp_***
(tabelul „copil“).
La eliminarea unui departament din tabelul dept_***, se vor sterge
toti angajatii din acel departament.*/

DELETE FROM  dept_***
WHERE  department_id = 30;

/* La modificarea codului unei departament din tabelul dept_***,
se va actualiza codul departamentului atat in tabelul dept_***,
cât si in tabelul emp_***.*/

UPDATE  dept_***
SET     department_id = 7
WHERE   department_id = 30;

/* Se presupune ca asupra tabelului emp_*** exista o constrangere
de integritate:
FOREIGN KEY (department_id) REFERENCES dept_***(department_id) ON
DELETE CASCADE/SET NULL
```

In acest caz pentru operatia de stergere sistemul Oracle va afisa un mesaj de eroare prin care se precizeaza ca **tabelul dept_*** este mutating**, iar constrangerea definita mai sus nu poate fi verificata.

ORA-04091: table MASTER.dept_*** is mutating, trigger/function may not see it*/