1. Definiti un tabel de LOG-uri in care sa puteti adăuga operațiile DML efectuate asupra unui tabel asociativ

din schema companie comercială.

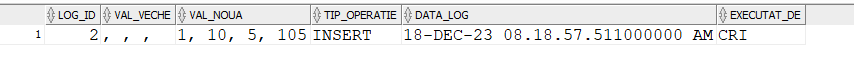
În acest tabel vor fi stocate ID-ul, vechea valoare, noua valoare, tipul operației,

momentul în care s-a executat operația și de către cine (userul curent autentificat -

SELECT USER FROM dual).

Implementați un trigger care populează acest tabel cu informațiile menționate mai sus.





CREATE TABLE log\_facturi\_contin\_produse\_cco (

log\_id NUMBER PRIMARY KEY,

val\_veche VARCHAR2(200),

val\_noua VARCHAR2(200),

tip\_operatie VARCHAR2(10),

data\_log TIMESTAMP default SYSTIMESTAMP,

executat\_de VARCHAR2(100)

);

drop table log\_facturi\_contin\_produse\_cco;

/

SET SERVEROUTPUT ON

/

CREATE SEQUENCE log\_sequence

START WITH 1

MAXVALUE 99999999999999999999999999

INCREMENT BY 1

NOCACHE

NOCYCLE;

/

CREATE OR REPLACE TRIGGER tigger\_log

BEFORE INSERT OR UPDATE OR DELETE ON facturi\_contin\_produse

FOR EACH ROW

DECLARE

v\_tip\_operatie VARCHAR2(10);

v\_user VARCHAR2(100);

BEGIN

SELECT USER INTO v\_user FROM dual;

-- Determinăm tipul operației

IF INSERTING THEN

v\_tip\_operatie := 'INSERT';

ELSIF UPDATING THEN

v\_tip\_operatie := 'UPDATE';

ELSIF DELETING THEN

v\_tip\_operatie := 'DELETE';

END IF;

INSERT INTO log\_facturi\_contin\_produse\_cco (

log\_id,

val\_veche,

val\_noua,

tip\_operatie,

executat\_de

) VALUES (

log\_sequence.nextval,

TO\_CHAR(:OLD.ID\_FACTURA) || ', ' || TO\_CHAR(:OLD.ID\_PRODUS) || ', ' || TO\_CHAR(:OLD.CANTITATE) || ', ' || TO\_CHAR(:OLD.PRET\_FACTURARE),

TO\_CHAR(:NEW.ID\_FACTURA) || ', ' || TO\_CHAR(:NEW.ID\_PRODUS) || ', ' || TO\_CHAR(:NEW.CANTITATE) || ', ' || TO\_CHAR(:NEW.PRET\_FACTURARE),

v\_tip\_operatie,

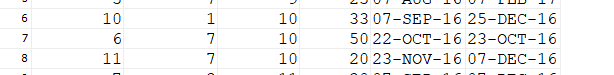
v\_user

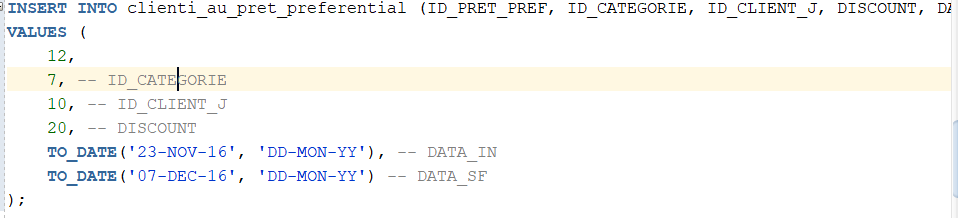
);

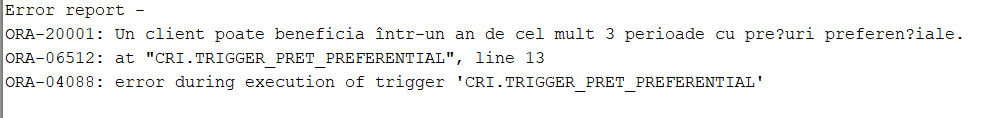
END;

/

2. Implementați cu ajutorul unui trigger următoarea restricție:

un client poate beneficia într-un an de cel mult 3 perioade cu prețuri preferențiale. 





CREATE OR REPLACE TRIGGER trigger\_pret\_preferential

BEFORE INSERT ON clienti\_au\_pret\_preferential

FOR EACH ROW

DECLARE

v\_nr\_perioade NUMBER;

v\_an1 NUMBER;

v\_an2 NUMBER;

BEGIN

v\_an2 := EXTRACT(YEAR FROM :NEW.DATA\_SF);

v\_an1:= EXTRACT(YEAR FROM :NEW.DATA\_IN);

FOR i in v\_an1..v\_an2 LOOP

SELECT COUNT(\*)

INTO v\_nr\_perioade

FROM clienti\_au\_pret\_preferential

WHERE ID\_CLIENT\_J = :NEW.ID\_CLIENT\_J

AND EXTRACT(YEAR FROM DATA\_IN) = i;

IF v\_nr\_perioade >= 3 THEN

RAISE\_APPLICATION\_ERROR(-20001, 'Un client poate beneficia într-un an de cel mult 3 perioade cu prețuri preferențiale.');

END IF;

END LOOP;

END;

3. Pe un tabel dependent din schema companie comercială implementați cu ajutorul unui trigger

--o constrângere de integritate la alegere.

--Observație: trebuie să apară explicit pe ce tabel și care este constrangerea implementată.

Tabel: facturi\_contin\_produse

Constrangerea: stocul curent al produsului sa fie mai mare sau egal cu cantittea din factura (verificam ca se cumpara cat e disponibil)

CREATE OR REPLACE TRIGGER trigger\_verificare\_stoc

BEFORE INSERT OR UPDATE ON facturi\_contin\_produse

FOR EACH ROW

DECLARE

v\_stoc\_curent NUMBER;

BEGIN

SELECT STOC\_CURENT

INTO v\_stoc\_curent

FROM produse

WHERE ID\_PRODUS = :NEW.ID\_PRODUS;

IF :NEW.CANTITATE > v\_stoc\_curent THEN

RAISE\_APPLICATION\_ERROR(-20001, 'Cantitatea din factură depășește stocul curent al produsului.');

END IF;

END;

/

