Cartouche du document

Année : ING 2
Activité : Travail dirigé

Objectifs

L'objectif de cette série d'exercices est :

- de se familiariser avec la gestion d'erreur dans un bloc PL/SQL
- de gérer des transactions indépendantes
- d'approfondir les notions de triggers et de curseurs

Sommaire des exercices

Corps des exercices

1 - Gestion de classes

Énoncé :

On gère des classes dans un collège. Une classe est composée de collégiens. Un collégien est décrit par la relation Collegien (idEtudiant,Nom,Prenom)

Question 1)

Énoncé de la question

Ecrire une procédure qui reçoit un triplet idEtudiant,Nom,Prenom et qui

- insère un nouveau collégien si la clé idEtudiant n'existe pas dans la table ;
- met à jour le nom et lé prénom d'un nouveau collégien si la clé idEtudiant existe dans la table.

Solution de la question

```
---- la TABLE collegien
CREATE TABLE Collegien
-- Création de table
(
idEtudiant number,
nom varchar2(20),
prenom varchar2(20),
CONSTRAINT pk_collegien PRIMARY KEY(idEtudiant)
);
-- la précédure pour créer ou mettre à jour un collégien --
CREATE PROCEDURE pcdMajCollegien(pidEtudiant number, pnom varchar2, pprenom
varchar2) IS
BEGIN
    INSERT INTO collegien VALUES(pidEtudiant, pnom, pprenom);
EXCEPTION
    WHEN DUP_VAL_ON_INDEX THEN
        UPDATE collegien SET nom = pnom, prenom = pprenom WHERE idEtudiant =
pidEtudiant;
```

```
END;
/
```

2 - Gestion de commandes

Énoncé :

Soit le MLD suivant :

produit (idproduit,designationprod,prixprod,stockprod)

commande (idcommande,datecom,montanttotalcom)

ligne_commande (idligcom, #idcommande,quantiteprod,#idproduit)

ligne_erreur (idcommande,idligcom,idproduit,quantiteprod,stockprod)

On désire, à chaque insertion d'une ligne de commande mettre à jour automatiquement le montant total de la commande correspondante ainsi que le stock du produit concerné par la ligne de commande.

Cette mise à jour ne doit pas se faire si la quantité demandée dans la ligne de commande est strictement supérieure à la quantité actuellement en stock. Dans ce cas, une erreur doit être inscrite dans la table appropriée.

Question 1)

Énoncé de la question

Expliquer pourquoi on doit écrire un trigger avant insertion sur la table ligne_commande.

Solution de la question

Se conférer à la réponse globale de l'exercice

Question 2)

Énoncé de la question

Ecrire les procédures.

Une ligne de commande non satisfaite devra faire l'objet d'une insertion dans la table LIGNEERREUR par l'intermédiaire d'une gestion d'exception.

1) Pour exécuter dans une fonction une transaction indépendante de la transaction courante, on utilise la clause PRAGMA AUTONOMOUS_TRANSACTION;. Cette clause apparaît dans la zone de déclaration des variables du bloc PL/SQL

2) Pour provoquer un rollback de la transaction courante en raison d'une erreur, on utilise la fonction raise_application_error(noErreur,msgErreur) où noErreur doit être un nombre négatif strictement inférieur à -20000 (les autres codes d'erreur sont réservés par Oracle).

Ci-dessous, vous trouverez le script de création des tables en vue tester le trigger.

```
---- On détruit les tables
DROP TABLE produit cascade constraints;
DROP TABLE commande cascade constraints;
DROP TABLE lignecommande cascade constraints;
DROP TABLE ligneerreur cascade constraints;
-- on crée les tables
CREATE TABLE produit
-- Création de table
(idproduit varchar2(5),
```

```
desprod varchar(50),
prixprod number(8,2),
stockprod number(4),
constraint pk_produit primary key(idproduit)
);
CREATE TABLE commande
-- Création de table
(idcommande varchar2(5) primary key,
datecde date,
montanttotalcom number(10,2),
constraint pk_commande primary key(idcommande)
);
CREATE TABLE ligne_commande
-- Création de table
(idlig varchar2(5),
idcommande varchar2(5),
quantiteprod number(3),
idproduit varchar2(5) REFERENCES produit(idproduit),
constraint pk_ligne_commande primary key (idlig,idcommande)
);
CREATE TABLE ligne_erreur
-- Création de table
(
idcommande varchar2(5),
idlig varchar2(5),
idproduit varchar2(5),
quantiteprod number(3),
stockprod number(4)
);
ALTER TABLE ligne_commande add constraint fk_lig_com1 foreign key(idcommande)
REFERENCES commande(idcommande);
ALTER TABLE ligne_commande add constraint fk_lig_com2 foreign key(idproduit)
REFERENCES produit(idproduit);
INSERT INTO produit VALUES ('P1','desc P1',50,50);
INSERT INTO produit VALUES ('P2','des P2',500,0);
INSERT INTO commande VALUES ('cde1',to_date('20/12/2008','DD/MM/YYYY'),0,'c1');
```

.

Solution de la question

```
---- Le TRIGGER maj_lignecde
CREATE OR REPLACE TRIGGER maj_lignecde
before INSERT on lignecommande
FOR each row
```

```
declare
    ok number ;
    wdelta_stock produit.stockprod%type ;
    wprixprod produit.prixprod%type ;
    wstockprod produit.stockprod%type ;
    erreur_stock exception ;
BEGIN
    SELECT stockprod,stockprod-:new.quantiteprod ,prixprod
        INTO wstockprod,wdelta_stock, wprixprod FROM produit
    WHERE idproduit = :new.idproduit ;
    IF wdelta_stock < 0 THEN  raise erreur_stock ;
    END IF  ;
    UPDATE commande SET montanttotalcom=montanttotalcom
+(wprixprod*:new.quantiteprod)
    WHERE idcommande=:new.idcommande ;
    UPDATE produit SET stockprod = stockprod-:new.quantiteprod WHERE
idproduit=:new.idproduit ;
exception
    when no_data_found THEN
        raise_application_error(-20001,'erreur prod') ;
    when erreur_stock THEN
        ok:=ecriture_erreur_commande(:new.idcommande,:new.idlig,:new.idproduit,
        :new.quantiteprod,wstockprod);
        raise_application_error(-20001,'erreur stock') ;
    when others THEN  null ;
END;
/
CREATE OR REPLACE FUNCTION ecriture_erreur_commande(pidcommande
varchar2,plig varchar2,pidproduit varchar2,pquantiteprod
number,pstockprod number)
return number
is
PRAGMA AUTONOMOUS_TRANSACTION; -- transaction indépENDante
BEGIN
INSERT INTO ligneerreur VALUES
(pidcommande,plig,pidproduit,pquantiteprod,pstockprod) ;
commit;
return 1;
END;
/
```

.

Question 3)

Énoncé de la question

On désire maintenant afficher la facture correpondant à une commande. Sur cette facture doit apparaître : le client,le numéro de la commande et la date de la commande, le montant total ainsi que pour chaque produit commandé, la quantité, le prix et le montant à payer correspondant.

Ecrire la procédure qui acceptera en paramètre le numéro de la commande et qui permettra d'afficher la facture.

Solution de la question

```
---- la procédure reproting_facture
CREATE OR REPLACE PROCEDURE reporting_facture(pidcommande varchar2)
is
    -- déclaration des variables utiles
    vidcommande commande.idcommande%type ;
    vidclient commande.idclient%type ;
    vdatecde commande.datecde%type ;
    vmontanttot commande.montanttotalcom%type ;
    vmontant commande.montanttotalcom%type ;
    -- déclaration du curseur qui contiENDra les lignes de commande
    -- pour une commande (vidcommande) concernée
    CURSOR cur_lignecde(vidcommande varchar2) is
    SELECT idlig,quantiteprod,idproduit
    FROM ligne_commande
    WHERE idcommande = vidcommande ;
BEGIN
    -- on initialise la variable vidcommande avec la valeur passée en paramètre
    vidcommande := pidcommande ;
    -- on récupère les valeurs de la TABLE commande pour la commande -- --
concernée(pidcommande)
    SELECT idclient,datecde,montanttotalcom INTO vidclient,vdatecde,vmontanttot
    FROM commande
    WHERE idcommande = vidcommande ;
    -- on affiche l entête de la facture
    dbms_output.put_line('FACTURE client numéro '|| vidclient) ;
    dbms_output.put_line('Commande numéro : '|| vidcommande);
    dbms_output.put_line('Date de la commande : '|| vdatecde);
    dbms_output.put_line('-----------------------------------------') ;
    dbms_output.put_line('') ;
    -- on gère les lines de commande à l aide du curseur
    FOR vcur IN cur_lignecde(pidcommande) LOOP
        -- pour chaque ligne on récupère le prix et on calcul le montan pour le produit
commandé
        SELECT prixprod*vcur.quantiteprod INTO vmontant FROM produit WHERE
idproduit=vcur.idproduit ;
```

```
        -- on affiche les info demandés pour chaque produit commandé
        dbms_output.put_line
        (vcur.idlig|| ' ' ||vcur.idproduit||' '||vcur.quantiteprod||' '||vmontant);
    END LOOP ;
    -- on affiche le pied de la facture
    dbms_output.put_line('-----------------------------------------') ;
    dbms_output.put_line('Montant total de la commande : '|| vmontanttot );
exception
    when no_data_found THEN
    dbms_output.put_line('erreur dans le numéro de la commande') ;
END ;
/
```

.