TP BDD avancée

Syntaxe de création d'un trigger

CREATE [OR REPLACE] TRIGGER trigger_name
{BEFORE | AFTER }
{INSERT [OR] | UPDATE [OR] | DELETE}
[OF col_name]
ON table_name
[FOR EACH ROW]
WHEN (condition)
BEGIN
--- Instruction PLSQL
END;

• ***CREATE [OR REPLACE] TRIGGER trigger_name*** *: pour créer ou écraser un trigger existant.*
• ***{BEFORE | AFTER | INSTEAD OF }*** : le moment du déclenchement du trigger (avant ou après l'opération de mise à jour).
• ***{INSERT [OR] | UPDATE [OR] | DELETE}*** : l'événement de mise à jour qui provoquera le déclenchement du trigger. Plusieurs événements séparés par OR sont possibles.
• ***[OF col_name]*** *: utilisé dans le cas de l'opération Update appliquée sur une colonne particulière.*
• ***[ON table_name]*** *: le nom de la table sur laquelle le trigger est défini.*
• ***[FOR EACH ROW]*** *: spécifie si le trigger est lancé pour chaque ligne affecté ou une seule fois.*
• ***WHEN (condition)*** *: le trigger est lance seulement lorsque la ligne affectée vérifie la condition.*

Remarque *: pour générer une exception et empêcher le programme de continuer, l'utilisateur peut lancer la procédure* ***raise_application_error (-Num_Message, 'Message à Afficher') ;*** *Num_Message est compris entre 20000 et 20999.*

- Exemple de création des trigger sur phpmyadmin
  https://www.youtube.com/watch?v=L5RpqspNAuc&t=338s&ab_channel=learnWebCoding

exemple corrigé trigger

Considérez le schéma relationnel suivant. Un employé peut travailler dans plus d'un département; le champ pct_temps de la relation Travail indique le pourcentage de temps pendant lequel un employé donné travaille dans un service donné.

- Employe(**eid:integer,** enom:string, age:integer, salaire:real)
- Travail(***#eid:integer, #depid:integer***, pct_temps:integer)
- Departement(**depid:integer**, budget:real, ***#responsable_id:integer***)

Exprimez les exigences suivantes en utilisant des déclencheurs :

Les employés doivent gagner un salaire minimum de 1000$.

```
1. CREATE OR REPLACE TRIGGER quest_1
2. BEFORE  INSERT OR UPDATE ON Employe  FOR EACH  ROW
3. BEGIN
4.     IF :NEW.salaire < 1000 THEN
5.         RAISE_APPLICATION_ERROR(-20005,'Les employés
   doivent gagner un salaire minimum de 1000$.');
6.     END IF;
7. END;
```

Le pourcentage total du temp de travail pour un employé doit être inférieur à 100%.

```
1  CREATE OR REPLACE TRIGGER quest_2
2  AFTER INSERT OR UPDATE ON Travail  FOR EACH  ROW
   DECLARE
3      pct NUMBER;
4  BEGIN
5      SELECT SUM(pct_temps)  INTO pct  FROM Travail  WHERE
   eid=:NEW.eid;
6      IF pct >=100 THEN
7          RAISE_APPLICATION_ERROR(-20005,'Le pourcentage total
   du temp de travail pour un employé doit être inférieur à
8  100');
9      END IF;
10 END;
```

Un responsable doit toujours avoir un salaire plus élevé que n'importe quel employé qu'il ou elle dirige.

```
CREATE OR REPLACE TRIGGER quest_3
AFTER INSERT OR UPDATE ON Travail  FOR EACH  ROW
DECLARE
```

```
    SAL REAL;
BEGIN
    SELECT E.salaire INTO SAL FROM Departement D INNER JOIN
EMPLOYE E ON E.eid=D.responsable_id WHERE D.depid=:NEW.depid;
    IF :NEW.salaire > SAL THEN
        RAISE_APPLICATION_ERROR(-20005,'Un responsable doit
toujours avoir un salaire plus élevé');
    END IF;
END;
```

Chaque fois qu'un employé reçoit une augmentation, le salaire du responsable doit être augmenté du même montant.

```
CREATE OR REPLACE TRIGGER quest_4
AFTER UPDATE ON EMPLOYE FOR EACH ROW
WHEN NEW.salaire>OLD.salaire
DECLARE
    DEP NUMBER;
    RESP NUMBER;
    DIFF REAL;
BEGIN
    DIFF:=:NEW.salaire-:OLD.salaire;
    SELECT T.depid INTO DEP FROM Travail T INNER JOIN Employe E
ON E.eid=T.eid WHERE E.eid=:NEW.eid;
    SELECT responsable_id INTO RESP FROM Departement WHERE
depid=DEP;
    UPDATE Employe SET salaire=salaire+DIFF WHERE eid = RESP;
END;
```

Chaque fois qu'un employé reçoit une augmentation, le salaire du responsable doit être augmenté au moins autant. De plus, chaque fois qu'un employé reçoit une augmentation, le budget du département doit être augmenté pour être supérieur à la somme des salaires des employés du département

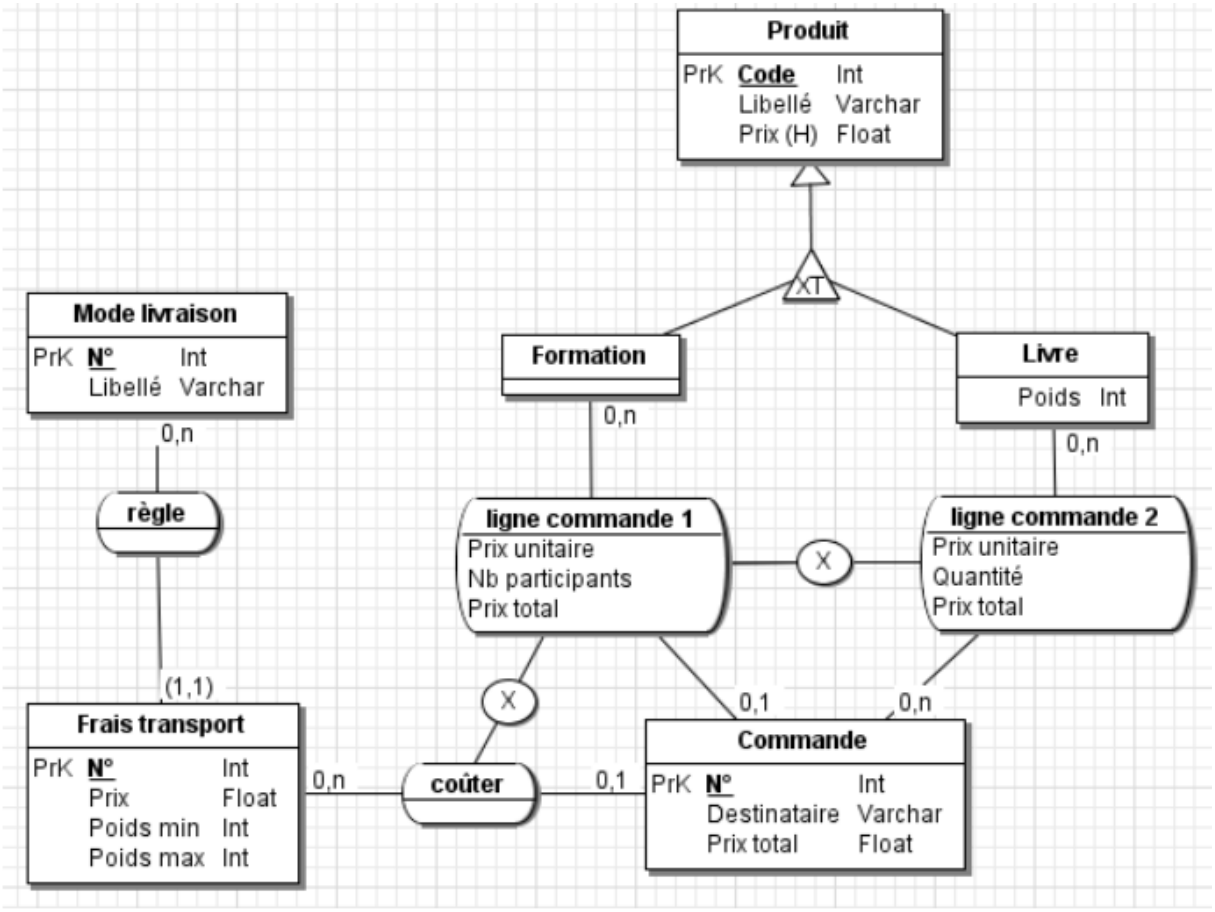
```
CREATE OR REPLACE TRIGGER quest_5
AFTER UPDATE ON EMPLOYE FOR EACH ROW
FOLLOWS quest_4
WHEN NEW.salaire>OLD.salaire

DECLARE
    DEP NUMBER;
    BUDG REAL;
    DIFF REAL;
    SOMME REAL;
BEGIN
    DIFF:=:NEW.salaire-:OLD.salaire;
```

```
    SELECT T.depid INTO DEP FROM Travail T INNER JOIN Employe E
ON E.eid=T.eid WHERE E.eid=:NEW.eid;
    SELECT budget INTO BUDG FROM Departement WHERE depid=DEP;
    SELECT SUM(E.salaire) INTO SOMME FROM Travail T INNER JOIN
Employe E ON E.eid=T.eid WHERE T.depid=DEP;
    IF SOMME>BUDG THEN
        UPDATE Employe SET salaire=salaire+(SOMME-BUDG) WHERE
eid = RESP;
    END IF;
END;
```

Travail à faire

Soit le modèle de données suivant :

Produit(code, libelle, prix, formation, poids)
Clef primaire : code
Commentaire : le champ discrimination formation prend la valeur 1 (vrai) ou 0 (faux)
Mode_Livraison(num, libelle)
Clef primaire : num
Frais_Transport(#livraison, num, prix, poidsMin, poidsMax)
Clef primaire : livraison, num
Clefs étrangères : livraison en référence à Mode_Livraison(num)
Commande(num, destinataire, prixTotal, #(livraion, transport))
Clef primaire : num
Clef étrangère : (livraison, transport) en référence à Frais_Transport(livraison, num)
Ligne_Commande(#commande, #produit, prixu, quantite, prixt)
Clef primaire : commande, produit
Clef étrangères :
- commande en référence à Commande(num)
- produit en référence à Produit(code)



1. Rédiger le script de création de table relatif à la base de données ci-avant décrite.
2. Rédiger le trigger permettant de vérifier la contrainte suivante : si une commande porte sur une formation, alors elle ne peut porter sur un ou plusieurs livres.
3. Rédiger le trigger permettant de vérifier la contrainte réciproque : si une commande porte sur un ou plusieurs livres, alors elle ne peut porter sur une formation.
4. Adapter le trigger de la question 2 ou 3 afin de vérifier la contrainte suivante : si la commande porte sur une formation, alors il ne doit pas y avoir de frais de transport.
5. Adapter le trigger de la question 2 ou 3 afin que le prix total d'une commande soit automatiquement mis à jour à lorsqu'une ligne de commande est insérée.
6. Préciser les cas que nous n'avons pas traités au travers des questions 1 à 5