



Objectif : le but de ce TD est d'assurer des contraintes via des déclencheurs

Aide-mémoire : Définition d'un déclencheur

```
CREATE [OR REPLACE] TRIGGER nom_declencheur
BEFORE|AFTER
INSERT|DELETE|UPDATE | INSERT [[OR] DELETE] [[OR] UPDATE]
ON nom_table
[FOR EACH ROW [WHEN <condition>]]
Bloc PL/SQL
```

Exercice 1 : Transformation, vérification et mise à jour automatique des données

Soit la table "Citoyen" qui présente les informations d'une personne et le numéro son conjoint. **Citoyen** (NumC, Nom, Prenom, NomJeuneF, numConjoint*)

| CITOYEN | | |
|---------------------------|-------------|--------------------|
| P | NUMC | INTEGER |
| | NOM | VARCHAR2 (20 BYTE) |
| | PRENOM | VARCHAR2 (20 BYTE) |
| | NOMJEUNEF | VARCHAR2 (20 BYTE) |
| F | NUMCONJOINT | INTEGER |
| PK_CITOYEN (NUMC) | | |
| FK_CONJOINT (NUMCONJOINT) | | |

1.Transformer, au moment de l'insertion en majuscule, la valeur du nom du citoyen quel que soit son format d'origine (utilisation de UPPER()).

```
CREATE OR REPLACE TRIGGER 'UPPER_NOMP'
BEFORE INSERT ON CITOYEN
FOR EACH ROW
BEGIN
:NEW.NOM := upper (:NEW.NOM) ;
END;
```

2.Vérifier, au moment de l'insertion ou modification, que le nom des deux conjoints est le même.

```
CREATE OR REPLACE TRIGGER 'CHECK_NOM_CONJ'
BEFORE INSERT OR UPDATE ON CITOYEN
FOR EACH ROW
DECLARE nomconj VARCHAR2( 20 ) ;
BEGIN
Select Nom Into Nomconj From CITOYEN
Where NumC = :NEW.Num conjoint ;
I f( Nomconj <> :NEW. Nom)Then
RAISE_APPLICATION_ERROR( 20300 , Oops! Le nom de la personne et
Le nom de son conjoint ne sont pas identiques' ) ;
END IF;
END;
```

3- incrémenter la valeur de la clé NumC automatiquement

```
CREATE OR REPLACE TRIGGER 'AUTO_INC_NUMP'
BEFORE INSERT ON CITOYEN
FOR EACH ROW
DECLARE
MAX_VAL NUMBER(5);
BEGIN
  Select Max(NumC) Into Max_Val From CITOYEN;
  If Max_Val is not null then
    :NEW.NumC := Max_Val + 1 ;
  Else
    :NEW.NumC := 1;
  END IF;
END;
```

```
CREATE SEQUENCE "SeqCitoyen" MINVALUE 1 MAXVALUE
1000000
INCREMENT BY 1 ;
CREATE OR REPLACE TRIGGER "TRIG_CLE_PRIM_PER"
BEFORE INSERT on "personne"
FOR EACH ROW
BEGIN
  Select Seq Citoyen.nextval into :NEW.NumC from
  dual;
END;
```

Exercice 2 : Problème d'interblocage

Soient les deux triggers "T1" et "T2" sur la table "tab1" et la table "tab2" de la même base de données

1. Expliquer le problème que pose le déclenchement du Trigger T1.

```
CREATE TRIGGER "T1"
AFTER DELETE ON tab1
FOR EACH ROW
BEGIN
  UPDATE tab2
  SET attribut1='A'
END ;
```

```
CREATE TRIGGER "T2"
AFTER UPDATE of attribut1 on tab2
FOR EACH ROW
BEGIN
  INSERT INTO tab1 Values (1,'A');
END ;
```

Le déclenchement du trigger T1, en premier, provoque un blocage car il fait appel au déclencheur "T2" qui exécute une insertion sur la table 'tab1' alors que l'opération de suppression sur cette table ne s'est pas encore terminée. Si le déclencheur "T2" avait été déclenché en premier, le problème ne se pose pas car il ne déclenche pas le deuxième trigger en cascade. En effet, il lance une insertion alors que T1 est déclenché lors d'une suppression

2- Corriger les erreurs du déclencheur suivant

```
CREATE OR REPLACE TRIGGER "T2"  
AFTER UPDATE OF attribut1 on tableT2  
FOR EACH ROW  
BEGIN  
    :NEW.attribute1 := 'A';  
END ;
```

On ne peut pas affecter (changer) des valeurs dans un déclencheur After

Exercice 3 :

Soit la base de données pour la gestion d'évènements sportifs "DZZumbaDays". Cet Évènement est organisé le 15 octobre de chaque année dans un lieu différent, en deux sessions : session "matin" et "après-midi"

EVENEMENT (NumEven, LieuEven, adresse, capacité).

PARTICIPATION (NumPart, numPersonne*, NumEven*, Session)

PERSONNE (numPersonne, nom, âge, téléphone)

capacité : nombre de participants que peut accueillir le lieu où est organisé l'évènement : par exemple l'évènement "DZZumbaDays2022" sera organisé dans le stade "Frère Zerga Tlemcen". La capacité est de 100 participants ; deux sessions seront organisées : matin et après-midi donc le champ "session" peut prendre deux valeurs : matin , après-midi.

- Définir un déclencheur qui permet d'interdire une participation si l'évènement affiche complet (la capacité est atteinte).

```
CREATE OR REPLACE TRIGGER Evn  
BEFORE INSERT ON Participation  
for each row  
DECLARE  
    NombreParticip INTEGER; NbCapacité INTEGER;  
BEGIN  
    NombreParticip:= 0 ;  
    Select count(*) into NombreParticip from Participation  
    Where NumEven = :new.NumEven  
    and Session = :new.Session;  
  
    select capacité into NbCapacité from Evenement  
    where NumEven = :new.NumEven;  
  
    if NombreParticip + 1 > capacité then RAISE_APPLICATION_ERROR(-20006,  
'l'évènement est au complet');  
End if;  
END;
```