**TRIGGER**

Un déclencheur (TRIGGER) spécifie que la base de données doit exécuter automatiquement une fonction donnée chaque fois qu'un certain type d'opération est exécuté. Les fonctions déclencheurs peuvent être définies pour s'exécuter avant ou après une commande INSERT, UPDATE ou DELETE, soit une fois par ligne modifiée, soit une fois par expression SQL.

**Principe de fonctionnement.**

Deux tables virtuelles sont créées au moment de la MAJ sur la table:

**INSERTED**

**DELETED.**

Elles sont destinées à contenir les lignes de la table sur lesquelles ont été effectuées des opérations.

Les tables **INSERTED** et **DELETED** peuvent être utilisées par le trigger pour déterminer comment le traitement doit se dérouler.

Ce traitement est à écrire par le développeur.

Les mécanismes d’utilisation par SQL Server de ces deux tables diffèrent en fonction de l’opération effectuée sur la table contenant le ou les triggers.

**Cas de suppression d’une ligne de table (delete)**

La/les lignes supprimées sont placées dans la table temporaire DELETED et supprimées de la table réelle; la table DELETED et les tables de la base ne peuvent pas avoir de lignes en commun.

**Cas de création d’une ligne de table (insert)**

La/les lignes nouvelles sont placées dans la table temporaire INSERTED et dans la table réelle; toutes les lignes de la table INSERTED apparaissent dans la table de la base.

**Cas de modification d’une ligne de table (update)**

La/les lignes avant modification sont placées dans la table temporaire DELETED et la/les lignes après modification sont placées dans la table temporaire INSERTED et dans la table réelle.

**Exemple de TRIGGER**

**Soustraction des quantités de stocks**

**Il faut déduire du stock les quantités d’articles commandés par les clients.**

USE [gestioncdexx];

CREATE TRIGGER Soustraction\_Stock\_Articles

ON lignescommande

AFTER INSERT

AS

BEGIN

DECLARE @id\_article int, @Qte\_stock int

SELECT @id\_article = [codearticle] FROM INSERTED

SELECT @Qte\_stock = [Quantite] FROM INSERTED

UPDATE [ARTICLES]

SET [STOCK] = [STOCK]-@Qte\_stock

WHERE @id\_article = [codearticle]

END;

**Tout nouveau salarié devra avoir un numéro supérieur ou égal à 10000, Il faut donc interdire toute insertion qui ne reflète pas cette nouvelle directive**

USE gestioncdeXX;

CREATE TRIGGER controle\_Salarié

ON achat.salarie

FOR INSERT

AS

BEGIN

DECLARE @nsal int

SELECT @nsal = [num\_salarie] FROM INSERTED

IF @nsal< 10000

BEGIN

PRINT 'Le numéro employé ' PRINT @nsal PRINT 'est inférieur à 10000'

ROLLBACK TRANSACTION

END;

End;

**Contrôle du nom du salarié au minimum 5 caractères.**

CREATE TRIGGER longueur\_nom

ON [achat].[SALARIE]

AFTER INSERT

AS

BEGIN

DECLARE @name nvarchar(50)

SELECT @name = Nom\_salarie

FROM inserted

IF len(@name) < 5

BEGIN

PRINT 'nom trop cours, corrigez SVP'

ROLLBACK TRANSACTION

END

END;

**Passer en majuscule la colonne NOM et DESCRIPTION de la table *CATEGORIE* lors d’une nouvelle saisie ou d’une modification du contenu des colonnes**

use gestioncdexx;

CREATE TRIGGER majuscule\_catégorie

ON CATEGORIES

INSTEAD OF INSERT

AS

BEGIN

INSERT INTO CATEGORIES(NOM,DESCRIPTION)

SELECT

UPPER(nom),UPPER(DESCRIPTION)

FROM inserted

IF @@ERROR <>0

ROLLBACK TRANSACTION

END

GO

**Vérification si la catégorie d’article existe avant d’inserer l’article.**

use gestioncdeXX;

CREATE TRIGGER TInsert\_Article

ON ARTICLES FOR INSERT

AS

BEGIN

IF (SELECT count(\*) FROM CATEGORIES, INSERTED

WHERE CATEGORIES.IDCATEGORIE = INSERTED.IDCATEGORIE) = 0

BEGIN

PRINT 'Déclencheur Insert sur ARTICLES. Catégorie d article inexistante. Création impossible'

ROLLBACK TRANSACTION

RETURN

END

END