Triggers

> Attention, ceci est uniquement valable in SQL Server

- Créer une trigger qui renvois une erreur dans un cas précis

```
-- On donne un nom à la trigger et on lui dit sur quelle table il opère
-- Ici le trigger opère sur la table "offre"
CREATE TRIGGER date_offre_posterieure_date_vente ON offre

-- On lui dit l'action pour laquelle il doit s'exécuter (ici après un insert)
-- On pourrait aussi avoir "after delete" ou "after update"
AFTER INSERT

AS BEGIN
	-- On déclare les variables avec leur types
	-- Ces déclarations doivent toujours être au début du trigger
	declare @dateOffre SMALLDATETIME
	declare @idBien INTEGER
	declare @dateMiseVente SMALLDATETIME
	
	-- "inserted" est une table temporaire qui renvois le tuple inséré
	-- On pourrait aussi avoir "deleted" à la place de inserted, et dans le cas d'un UPDATE
	-- On a les deux (deleted et inserted)
	
	-- La syntaxe de @variable = attribut va attribuer la valeur de l'attribut du tuple
	-- dans la variable définie plus tot
	select @idBien = id_bien, @dateOffre = date_offre from inserted
	
	-- Ensuite on peut simplement utiliser la variable n'importe où dans une query
	select @dateMiseVente = date_mise_en_vente from bien where id_bien = @idBien
	
	-- on crée une condition (dans lequel il faut annuler l'insert)
	if @dateOffre < @dateMiseVente
	begin
```

```
    -- on génère une erreur
    raiserror('La date de l''offre doit être postérieur à la date de la mise en vente.', 7, 1)
    -- On annule l'insert
    rollback transaction
    -- On met fin à la trigger
    return
end

    -- si la condition ne corresponds aps, alors l'insert est bien inséré
END
```

- On peut aussi faire un curseur pour itérer sur plusieurs lignes d'une requète SQL

```
-- Ici on déclare une variable curseur pour une certaine query (que l'on va itérer)
DECLARE crsrClientA_F CURSOR
FOR SELECT nom, adresse
FROM Client
WHERE UPPER(SUBSTRING(nom,1,1)) IN ('A','B','C','D','E','F')

-- On ouvre le curseur et on récupère le premier élément dans des variables
OPEN crsrClientA_F
FETCH crsrClientA_F INTO @nom, @adresse

-- Tant que le dernier fetch a trouvé quelque chose, on va print le nom et l'addresse
WHILE @@FETCH_STATUS = 0
BEGIN
    PRINT @nom + ' habite à ' + @adresse

    -- A la fin de la boucle, on refait un fetch pour récupérer la valeur suivante
    -- La boucle s'arretera quand ce fetch rattera
    FETCH crsrClientA_F INTO @nom, @adresse
END

-- Une fois terminé on peut fermer le curseur et le désallouer
CLOSE crsrClientA_F
DEALLOCATE crsrClientA_F
```

Revision #3
Created 6 June 2023 09:49:33 by SnowCode
Updated 7 June 2023 11:59:54 by SnowCode