Cours de bases de données, http://sql.bdpedia.fr

Triggers

Philippe Rigaux

le cnam

Les déclencheurs, ou *triggers*

Un déclencheur (*trigger*) est une procédure stockée qui se déclenche automatiquement sur certains événements.

Utilité :

- Gestion des redondances
- Enregistrement automatique de certains événements
- Gestion de contraintes complexes (exemple : le prix d'un produit ne peut qu'augmenter)
- Gestion de contraintes liées à l'environnement d'exécution (restrictions sur les horaires, les utilisateurs, etc.)

Inconvénient (fort) : les *triggers* s'exécutent de manière cachée, et peuvent mener à des cycles sans fin.

Etudions un exemple

Un logement est constitué de chambres ; sa capacité est la somme du nombre de lits.

On crée un *trigger* qui maintient cette valeur agrégative.

```
create trigger CumulCapacité
after update on Chambre
for each row
when (new.nbLits != old.nbLits)
begin
  update Logement
  set capacité = capacité - :old.nbLits  + :new.nbLits
  where  code = :new.codeLogement;
end;
```

Notez : les variables `old` et `new`, représentant le nuplet avant/après la mise à jour

Les composants d'un *trigger*

Modèle basé sur la séquence Evénement-Condition-Action (ECA) :

- Un *trigger* est déclenché par une insertion, destruction ou modification
- La condition est testée, et si elle n'est pas satisfaite, l'exécution du *trigger* s'arrête
- L'action est effectuée à l'aide du langage procédural (PL/SQL)

De plus : un *trigger* peut manipuler simultanément les valeurs ancienne et nouvelle du nuplet modifié.

Les *triggers* ont des inconvénients

On peut créer des *triggers* sources d'inefficacité. Exemple

```
create trigger CumulCapaciteGlobal
after update or insert or delete on Chambre
begin
  update Logement C
  set capacité = (select sum (nbLits)
                        from   Chambre
                        where  code = :new.codeLogement);
end;
```

L'exécution est invisible, donc problème difficile à détecter

Les *triggers* sont dangereux !

L'exécution combinée des *triggers* peut avoir des effets imprévisibles.

La création du *trigger* suivant bloque le programme qui l'exécute.

```
create trigger MAJChambre
after update or delete or insert on Logement
begin
  update chambre
  set nbLits = (select :new.capacité / count (*)
                from  Chambre
                where codeLogement = :new.code);
end;
```

Je modifie un logement ⇒ je modifie les chambres ⇒ je modifie le logement, etc.

À retenir

Les *triggers* : mécanisme puissant. Pratique, par exemple,

- pour contrôler ce qui se passe dans une base de données
- pour mémoriser les actions sensibles (destruction)

Mais dangereux, à utiliser avec précaution

- L'exécution d'un *trigger* est invisible : l'équivalent d'un effet de bord en programmation
- Un *trigger* peut être source (invisible) de lenteur
- La composition de plusieurs *triggers* est incontrôlable