SGBD - 2e

PL-SQL - Chapitre 10 - Les déclencheurs

Daniel Schreurs

12 février 2022

Haute École de Province de Liège

Table des matières du chapitre i

Introduction

Introduction

- Un déclencheur (trigger) permet de définir un ensemble d'actions qui sont déclenchées automatiquement par le SGBD lorsque certains phénomènes se produisent.
- Les actions sont enregistrées dans la base et non plus dans les programmes d'application.
- Cette notion n'est pas spécifiée dans SQL2, elle le sera dans SQL3.

Introduction

Les déclencheurs permettent de définir des contraintes dynamiques, ils peuvent être utilisés pour :

- Générer automatiquement une valeur de clé primaire ;
- Résoudre le problème des mises à jour en cascade ;
- Enregistrer les accès à une table ;
- Gérer automatiquement la redondance ;
- Empêcher la modification par des personnes non autorisées (dans le domaine de la confidentialité) ;
- Mettre en œuvre des règles de fonctionnement plus complexes ;

Introduction

- Les déclencheurs permettent de réaliser des opérations sophistiquées car ils constituent un bloc PL/SQL
- Depuis Oracle8i, il est possible de définir des déclencheurs s'activant suite à des commandes du LDD ou à certains événements systèmes
- Il existe 12 types de déclencheurs sensibles aux commandes LMD en fonction de l'instruction déclenchante (ajout, suppression, modification), du
- niveau du déclencheur (ligne ou table) et du moment du déclenchement (avant ou après)

Déclencheurs réagissant aux instructions LMD

Déclencheurs réagissant aux instructions LMD

Syntaxe

```
CREATE TRIGGER nom_déclencheur
  BEFORE | AFTER
    DELETE | INSERT | UPDATE [OF liste_colonne]
        ON nom_table | nom_vue
  [FOR EACH ROW]
  [WHEN condition]
[bloc PL/SQL];
```

Déclencheurs réagissant aux instructions LMD

Deux restrictions sur les commandes du bloc PL/SQL d'un déclencheur :

- Un déclencheur ne peut contenir de `COMMIT` ni de `ROLLBACK` ;
- Il est impossible d'exécuter une commande du LDD dans un déclencheur ;
- Un déclencheur de niveau de ligne ne peut pas :
  - Lire ou modifier le contenu d'une table mutante. [1]
  - Lire ou modifier les colonnes d'une clé primaire, unique ou étrangère d'une table contraignante. [2]

---

1. Table contraignante : table qui peut éventuellement être accédée en lecture afin de vérifier une contrainte de référence
2. Table Mutante : table en cours de modification.

Déclencheurs réagissant aux instructions LMD

Contrainte dynamique

```
1  CREATE TRIGGER upd_salaire_personnel
2      BEFORE UPDATE OF sal
3      ON Emp
4      FOR EACH ROW
5      WHEN (OLD.sal > NEW.sal)
6  DECLARE
7      salaire_diminue EXCEPTION;
8  BEGIN
9      RAISE salaire_diminue;
10 EXCEPTION
11     WHEN salaire_diminue THEN
12         raise_application_error(-20001,
13                                 'Le salaire ne peut
                                       diminuer');
14 END;
```

Déclencheurs réagissant aux instructions LMD i

Un chef de département doit être un employé attaché à ce département

```
1  CREATE OR REPLACE TRIGGER MajChefDept
2      BEFORE INSERT OR UPDATE OF numsecu
3      ON departements
4      FOR EACH ROW
5      WHEN (NEW.numsecu IS NOT NULL)
6  DECLARE
7      numdep Employes.numdep%TYPE;
8      ExcNumDep EXCEPTION;
9  BEGIN
10     SELECT numdep INTO numdep FROM employes WHERE
           numsecu = :NEW.numsecu;
11     IF numdep <> :NEW.numdep THEN RAISE ExcNumDep; END
           IF;
12 EXCEPTION
```

Déclencheurs réagissant aux instructions LMD ii

```
13      WHEN NO_DATA_FOUND THEN RAISE_APPLICATION_ERROR
14          ('-20000', 'Le nr ' || :New.numsecu || 'n''est
                pas un employé de la societé');
15      WHEN ExcNumDep THEN RAISE_APPLICATION_ERROR
16          ('-20001', 'Le nr ' || :New.numsecu || ' n''est
                pas attaché au dept  ' || :New.numdep);
17  end;
```

Déclencheurs réagissant aux instructions LMD i

Gestion automatique de la redondance : Dans la table Service : colonne `Nombre_Emp` contient le nombre d'employés de chaque service => redondance

Déclencheurs réagissant aux instructions LMD i

On gère la redondance au moyen d'un déclencheur.

```
1  CREATE TRIGGER maj_nb_emp
2      BEFORE INSERT OR DELETE OR UPDATE OF num_service
3      ON personnel
4      FOR EACH ROW
5  BEGIN
6      IF inserting THEN
7          UPDATE service
8          SET nombre_emp = nombre_emp + 1
9          WHERE num_service = :NEW.num_service;
10     ELSIF deleting THEN
11         UPDATE service
12         SET nombre_emp = nombre_emp - 1
13         WHERE num_service = :OLD.num_service;
14     ELSIF updating THEN
```

Déclencheurs réagissant aux instructions LMD ii

```
15          UPDATE service
16          SET nombre_emp = nombre_emp + 1
17          WHERE num_service = :NEW.num_service;
18          UPDATE service
19          SET nombre_emp = nombre_emp - 1
20          WHERE num_service = :OLD.num_service;
21      END IF;
22  END;
```

Déclencheurs réagissant aux instructions LMD

Mise à jour en cascade

```
CREATE TRIGGER maj_cascade
    BEFORE DELETE OR UPDATE OF num_auteur
    ON auteurs
    FOR EACH ROW
BEGIN
    IF updating THEN
        UPDATE a_ecrit
        SET num_auteur = :NEW.num_auteur
        WHERE num_auteur = :OLD.num_auteur;
    ELSIF deleting THEN
        DELETE
        FROM a_ecrit
        WHERE num_auteur = :OLD.num_auteur;
    END IF;
END;
```

Déclencheurs réagissant aux instructions LMD i

Sécurité et enregistrement des accès

```
1  CREATE OR REPLACE TRIGGER TrAuditPersonnel
2      BEFORE INSERT OR DELETE OR UPDATE
3      ON personnel
4      FOR EACH ROW
5  BEGIN
6      IF INSERTING THEN
7          INSERT INTO auditPersonnel
8          VALUES (USER, SYSDATE, 'INSERT',
9                  :NEW.NUMERO, :NEW.NOM, :NEW.PRENOM,
10                 :NEW.SEXE, :NEW.ADRESSE, :NEW.
                       NUM_SERVICE,
11                 :NEW.SALAIRE, :NEW.TAXE);
12     ELSIF DELETING THEN
13         INSERT INTO auditPersonnel
```

Déclencheurs réagissant aux instructions LMD ii

```
14          VALUES (USER, SYSDATE, 'DELETE',
15                  :OLD.NUMERO, :OLD.NOM, :OLD.PRENOM,
16                  :OLD.SEXE, :OLD.ADRESSE,
17                  :OLD.NUM_SERVICE,
18                  :OLD.SALAIRE, :OLD.TAXE);
19      ELSIF UPDATING THEN
20          INSERT INTO auditpersonnel
21          VALUES (USER, SYSDATE, 'NOUVEAU',
22                  :NEW.NUMERO, :NEW.NOM, :NEW.PRENOM,
23                  :NEW.SEXE, :NEW.ADRESSE,
24                  :NEW.NUM_SERVICE,
25                  :NEW.SALAIRE, :NEW.TAXE);
26          INSERT INTO auditpersonnel
27          VALUES (USER, SYSDATE, 'ANCIEN',
28                  :OLD.NUMERO, :OLD.NOM, :OLD.PRENOM,
29                  :OLD.SEXE, :OLD.ADRESSE,
```

Déclencheurs réagissant aux instructions LMD iii

```
30                    :OLD.NUM_SERVICE,
31                    :OLD.SALAIRE, :OLD.TAXE);
32      END IF;
33  END;
```

Contourner les tables mutantes

Contourner les tables mutantes i

```
1  CREATE OR REPLACE TRIGGER departementsnumdep
2  BEFORE INSERT ON Departements -- Ici
3  FOR EACH ROW
4  BEGIN
5    SELECT 'd' ||
6        COALESCE(LPAD( MAX(SUBSTR(numdep,2,5))+1,5,'0'),'
             00001')
7        INTO :NEW.numdep   FROM departements; -- Ici
8  EXCEPTION
9    WHEN OTHERS THEN RAISE;
10 END;
```

- OK SI insertion d'une seule ligne
- Erreur SI insertion de plusieurs lignes(ORA-04091 : la table INFOSOFT.DEPARTEMENTS est en mutation)

Contourner les tables mutantes i

Syntaxe du COMPOUND TRIGGER

```
1  CREATE [OR REPLACE] TRIGGER nom_declencheur
2  FOR {INSERT | UPDATE | UPDATE OF liste_colonnes | DELETE
       }
3  ON nom_table | nom_vue
4  COMPOUND TRIGGER
5
6  [BEFORE STATEMENT IS
7  [instructions_déclarations;]
8  BEGIN
9  Instructions_execution;
10 END BEFORE STATEMENT;]
11
12 [BEFORE EACH ROW IS ...]
13
```

Contourner les tables mutantes ii

```
14 [AFTER EACH ROW IS ...]
15
16 [AFTER STATEMENT IS ...]
17
18 END nom_declencheur;
```

Contourner les tables mutantes i

Exemple COMPOUND TRIGGER

```
1  CREATE OR REPLACE TRIGGER GestionDesDept
2      FOR INSERT OR UPDATE OF numdep
3      ON employes
4      COMPOUND TRIGGER
5      TYPE ArrayOfRowid IS TABLE OF ROWID INDEX BY
           BINARY_INTEGER;
6      NewRowids ArrayOfRowid; Compt NUMBER; MaxEmployes
           NUMBER := 5;
7      NbreAtteint EXCEPTION;
8  AFTER EACH ROW IS
9  BEGIN
10     NewRowids(NewRowids.COUNT + 1) := :NEW.rowid;
11 End AFTER EACH ROW;
12     AFTER STATEMENT IS
```

Contourner les tables mutantes ii

```
13      BEGIN
14          FOR i in 1.. NewRowids.COUNT
15              LOOP
16                  SELECT COUNT(*)
17                  INTO Compt
18                  FROM employes
19                  WHERE numdep = (SELECT numdep FROM
                        employes WHERE ROWID = NewRowids(i));
20                  IF Compt > MaxEmployes THEN RAISE
                        NbreAtteint; END IF;
21              END LOOP;
22      EXCEPTION
23          WHEN NbreAtteint
24              THEN RAISE_APPLICATION_ERROR(-20002, '
                    Departement a atteint le max');
25      END AFTER STATEMENT;
```

Contourner les tables mutantes iii

```
26      END GestionDesDept;
```

Déclencheurs réagissant aux instr. du LDD

Déclencheurs réagissant aux instr. du LDD i

La syntaxe

```
CREATE TRIGGER nom_déclencheur
  BEFORE | AFTER
    instruction_DDL ON Schema | DATABASE
[bloc PL/SQL];
```

Déclencheurs réagissant aux instr. du LDD i

Chaque fois qu'un utilisateur crée un objet dans son schéma

```
1 CREATE TRIGGER auditCreation
2     AFTER CREATE
3     ON SCHEMA
4 BEGIN
5     INSERT INTO log
6     VALUES ('Objet créé le : ' || CURRENT_DATE);
7 END;
```

Déclencheurs réagissant aux événements syst

Déclencheurs réagissant aux événements syst i

La syntaxe

```
CREATE TRIGGER nom_déclencheur
  BEFORE | AFTER
    événement_système ON Schema | DATABASE
[bloc PL/SQL];
```

Déclencheurs réagissant aux événements syst i

- SERVERERROR : AFTER
- LOGON : est possible AFTER
- LOGOFF : est possible BEFORE
- STARTUP : est possible AFTER
- SHUTDOWN : est possible BEFORE

Déclencheurs réagissant aux événements syst i

Exemple déclencheur système

```
CREATE OR REPLACE TRIGGER tlogoff
    BEFORE LOGOFF
    ON SCHEMA
BEGIN
    INSERT INTO loginout
    VALUES (DBMS_STANDARD.SYSEVENT ||
            ' de ' || USER || ' le ' || CURRENT_DATE);
END;


CREATE OR REPLACE TRIGGER tlogon
    AFTER LOGON
    ON SCHEMA
BEGIN
```

Déclencheurs réagissant aux événements syst ii

```
15      INSERT INTO loginout
16      VALUES (DBMS_STANDARD.SYSEVENT ||
17              ' de ' || USER || ' le ' || CURRENT_DATE);
18  END;
```