

# **SGBD - 2<sup>e</sup>**

## PL-SQL - Chapitre 10 - Les déclencheurs

---

Anne Léonard

6 septembre 2023

Haute École de la Province de Liège

# Table des matières du chapitre i

1. Introduction
2. Déclencheurs réagissant aux instructions LMD
3. Contourner les tables mutantes
4. Déclencheurs réagissant aux instr. du LDD
5. Déclencheurs réagissant aux événements syst

# 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.

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 ;

- 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

```
1 CREATE TRIGGER nom_déclencheur
2   BEFORE | AFTER
3     DELETE | INSERT | UPDATE [OF liste_colonne]
4     ON nom_table | nom_vue
5   [FOR EACH ROW]
6   [WHEN condition]
7 [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.<sup>1</sup>
  - Lire ou modifier les colonnes d'une clé primaire, unique ou étrangère d'une table contraignante.<sup>2</sup>

---

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
14                                diminuer');
15 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
11    numsecu = :NEW.numsecu;
12  IF numdep <> :NEW.numdep THEN RAISE ExcNumDep; END
13  IF;
14 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 société');
15  WHEN ExcNumDep THEN RAISE_APPLICATION_ERROR
16      ('-20001', 'Le nr ' || :New.numsecu || ' n''est
      pas attaché au dept ' || :New.numdep);
17  end;
```

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

```
1 CREATE TRIGGER maj_cascade
2     BEFORE DELETE OR UPDATE OF num_auteur
3     ON auteurs
4     FOR EACH ROW
5 BEGIN
6     IF updating THEN
7         UPDATE a_ecrit
8         SET num_auteur = :NEW.num_auteur
9         WHERE num_auteur = :OLD.num_auteur;
10    ELSIF deleting THEN
11        DELETE
12        FROM a_ecrit
13        WHERE num_auteur = :OLD.num_auteur;
14    END IF;
15 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.
11              NUM_SERVICE,
12              :NEW.SALAIRE, :NEW.TAXE);
13   ELSIF DELETING THEN
14     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'),
7             '00001')
8     INTO :NEW.numdep FROM departements; -- Ici
9 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
```

```
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;
```



# Contourner les tables mutantes ii

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

26 **END** AFTER STATEMENT;

27 **END** GestionDesDept;

## **Déclencheurs réagissant aux instr. du LDD**

---

# Déclencheurs réagissant aux instr. du LDD i

La syntaxe

```
1 CREATE TRIGGER nom_déclencheur
2   BEFORE | AFTER
3     instruction_DDL ON Schema | DATABASE
4 [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

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

- 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

```
1 CREATE OR REPLACE TRIGGER tlogoff
2     BEFORE LOGOFF
3     ON SCHEMA
4 BEGIN
5     INSERT INTO loginout
6     VALUES (DBMS_STANDARD.SYSEVENT ||
7             ' de ' || USER || ' le ' || CURRENT_DATE);
8 END;
```

  

```
9
10
11 CREATE OR REPLACE TRIGGER tlogon
12     AFTER LOGON
13     ON SCHEMA
14 BEGIN
```

## Déclencheurs réagissant aux événements syst ii

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

- Les présents diaporamas constituent un résumé du livre : DELMAL, P, *SQL2 - SQL3*, Bruxelles, De Boeck Université, 2004, 512 pages.
- Ce résumé a été étoffé d'éléments extraits de la documentation officielle d'Oracle : [Oracle Database Documentation](#), 12/02/2021.
- Ainsi que de l'ouvrage : SOUTOU, Ch, *SQL pour Oracle*, Paris, Editions Eyrolles, 2013, 642 pages.