

# Base de données

## Cours 10 - Déclencheurs ("Triggers")

Steve Lévesque, Tous droits réservés © où applicables

# Table des matières

- 1 Caractéristiques
- 2 Syntaxe de base
  - Variables spéciales
- 3 Type de déclencheurs
  - DDL Triggers
  - DML Triggers
  - Logon Triggers
- 4 Appel de fonctions
- 5 Débogage et surveillance

# Déclencheurs (Triggers) - Caractéristiques

Un déclencheur ou “trigger” est une procédure stockée qui **s'exécute automatiquement** lorsque certains événements se produisent sur une table ou une vue.

- Associés à une table ou une vue spécifique.
- Configurés pour s'exécuter avant ou après des opérations telles que l'insertion (INSERT), la mise à jour (UPDATE) ou la suppression de données (DELETE).
- Utilisés pour valider les données, maintenir l'intégrité référentielle, ou effectuer des actions supplémentaires en réponse à des modifications de données.
- Utilisés pour automatiser des tâches répétitives ou pour appliquer des règles métier complexes.

# Déclencheurs (Triggers) - Syntaxe de base

Un déclencheur est créé à l'aide de la commande `CREATE TRIGGER` et comprend plusieurs composants clés :

- **trigger\_name** : Définit le nom du déclencheur.
- **BEFORE — AFTER** : Indique si le déclencheur doit s'exécuter avant ou après l'événement spécifié.
- **INSERT — UPDATE — DELETE** : Spécifie l'événement qui déclenche le déclencheur.
- **ON table\_name** : Spécifie la table ou la vue à laquelle le déclencheur est associé.
- **FOR EACH ROW — FOR EACH STATEMENT** : Indique si le déclencheur doit s'exécuter pour chaque ligne affectée ou une seule fois par instruction.
- **WHEN (condition)** : Permet de spécifier une condition pour l'exécution

# Déclencheurs (Triggers) - Syntaxe de base

Listing: <https://www.postgresql.org/docs/current/sql-createtrigger.html>

```
1 CREATE TRIGGER [trigger_name]
2 [BEFORE | AFTER] {INSERT | UPDATE | DELETE}
3 ON [table_name]
4 [FOR EACH ROW | FOR EACH STATEMENT]
5 [WHEN (condition)]
6 BEGIN
7 END;
```

# Déclencheurs (Triggers) - Variables spéciales

Dans les déclencheurs, certaines variables spéciales sont utilisées pour accéder aux données avant et après l'événement déclencheur :

- **NEW** : Représente la nouvelle ligne de données qui sera insérée ou mise à jour.  
Utilisé dans les déclencheurs BEFORE INSERT, AFTER INSERT, BEFORE UPDATE, et AFTER UPDATE.
- **OLD** : Représente l'ancienne ligne de données qui sera supprimée ou mise à jour.  
Utilisé dans les déclencheurs BEFORE DELETE, AFTER DELETE, BEFORE UPDATE, et AFTER UPDATE.

# Déclencheurs (Triggers) - Variables spéciales

Ici, le déclencheur vérifie que le solde du compte est changé de manière appropriée lors d'une mise à jour grâce aux variables NEW et OLD :

Listing: <https://www.postgresql.org/docs/current/sql-createtrigger.html>

```
1 CREATE TRIGGER check_update
2 BEFORE UPDATE ON accounts
3 FOR EACH ROW
4 WHEN (OLD.balance IS DISTINCT FROM NEW.balance)
5 EXECUTE FUNCTION check_account_update();
```

# Déclencheurs (Triggers) - Types

Les déclencheurs peuvent être classés en plusieurs types en fonction de leur objectif et de leur moment d'exécution :

- DDL Triggers
- DML Triggers
- Logon Triggers



# Déclencheurs (Triggers) - DDL Triggers

Les DDL Triggers sont déclenchés par des événements de définition de données (Data Definition Language) tels que la **création (CREATE)**, la **modification (ALTER)** ou la **suppression de tables ou de vues (DROP)**.

Ils permettent de surveiller et de contrôler les modifications apportées à la structure de la base de données.

# Déclencheurs (Triggers) - DDL Triggers

Exemple de création d'un déclencheur DDL qui prévient la création, la modification ou la suppression d'une table :

Listing: <https://www.postgresql.org/docs/current/sql-createtrigger.html>

```
1  -- First create the trigger function
2  CREATE OR REPLACE FUNCTION prevent_table_deletion()
3  RETURNS event_trigger
4  LANGUAGE plpgsql
5  AS $$
6  BEGIN
7      IF tg_tag = 'DROP TABLE' THEN
8          RAISE EXCEPTION 'You cannot drop tables in this database';
9      END IF;
10 END;
11 $$;
12
13 -- Then create the trigger
14 CREATE EVENT TRIGGER prevent_table_deletion_trigger
15     ON ddl_command_end
16     WHEN tag IN ('DROP TABLE')
17     EXECUTE FUNCTION prevent_table_deletion();
```

# Déclencheurs (Triggers) - DML Triggers

Les déclencheurs DML sont déclenchés par des événements de manipulation de données (Data Manipulation Language) tels que **l'insertion (INSERT), la mise à jour (UPDATE) ou la suppression de données (DELETE).**

Ils sont parfait pour vérifier l'intégrité des données avant ou après une modification, pour maintenir des journaux de modifications, ou pour la modification automatique ou en cascade des données dans d'autres tables.

# Déclencheurs (Triggers) - DML Triggers

Exemple de création d'un déclencheur DML qui prévient la modification non autorisée d'une table :

Listing: <https://www.postgresql.org/docs/current/sql-createtrigger.html>

```
1  -- First create the trigger function
2  CREATE OR REPLACE FUNCTION prevent_update_function()
3  RETURNS TRIGGER AS $$
4  BEGIN
5      RAISE EXCEPTION 'You cannot update the students table';
6      RETURN NULL;
7  END;
8  $$ LANGUAGE plpgsql;
9
10 -- Then create the trigger
11 CREATE TRIGGER prevent_update
12 BEFORE UPDATE ON student
13 FOR EACH ROW
14 EXECUTE FUNCTION prevent_update_function();
```

# Déclencheurs (Triggers) - Logon Triggers

Les Logon Triggers sont déclenchés lorsqu'un utilisateur se connecte à la base de données.

Ils sont utilisés pour appliquer des politiques de sécurité, enregistrer les tentatives de connexion, ou restreindre l'accès en fonction de divers critères.

# Déclencheurs (Triggers) - Logon Triggers

Exemple de création d'un déclencheur Logon qui affiche un message lorsqu'un utilisateur se connecte :

Listing: <https://www.postgresql.org/docs/current/sql-createtrigger.html>

```
1  -- First create the trigger function
2  CREATE OR REPLACE FUNCTION on_login_proc()
3      RETURNS event_trigger AS
4  $$
5      BEGIN
6          INSERT INTO user_login (who)
7              VALUES (SESSION_USER);
8      END;
9  $$ LANGUAGE plpgsql;
10
11  -- Then create the trigger
12  CREATE EVENT TRIGGER on_login_event
13      ON login
14      EXECUTE FUNCTION on_login_proc();
```

# Déclencheurs (Triggers) - Appel de fonctions

Les déclencheurs sont souvent utilisés pour appeler des fonctions stockées qui contiennent la logique à exécuter lorsque le déclencheur est activé.

Cela permet de séparer la logique de l'application de la logique de la base de données, facilitant ainsi la maintenance et la réutilisation du code.

# Déclencheurs (Triggers) - Appel de fonctions

Listing: <https://www.postgresql.org/docs/current/sql-createtrigger.html>

```
1  -- Trigger Function to validate employee data
2  CREATE OR REPLACE FUNCTION validate_employee()
3  RETURNS TRIGGER AS $$
4  BEGIN
5      IF NEW.salary < 0
6      THEN
7          RAISE EXCEPTION 'The salary cannot be negative';
8      END IF;
9      RETURN NEW;
10 END;
11 $$ LANGUAGE plpgsql;
12
13 -- Associated Trigger
14 CREATE TRIGGER trigger_validate_employee
15 BEFORE INSERT OR UPDATE ON employees
16 FOR EACH ROW
17 EXECUTE FUNCTION validate_employee();
```



# Déclencheurs (Triggers) - Surveillance

La surveillance des déclencheurs est essentielle pour garantir qu'ils fonctionnent comme prévu et pour diagnostiquer les problèmes potentiels.

Voici quelques méthodes courantes pour surveiller les déclencheurs :

Listing: <https://www.postgresql.org/docs/current/sql-createtrigger.html>

```
1  -- See all triggers in the database
2  SELECT * FROM information_schema.triggers;
3
4  -- See all triggers on a specific table
5  SELECT trigger_name, event_manipulation, action_statement
6  FROM information_schema.triggers
7  WHERE event_object_table = 'students';
8
9  -- Enable or disable a trigger
10 ALTER TABLE employees DISABLE TRIGGER trigger_validate_employee;
11 ALTER TABLE employees ENABLE TRIGGER trigger_validate_employee;
```

# Bibliographie

- <https://www.postgresql.org/docs/current/trigger-definition.html>
- <https://www.postgresql.org/docs/current/plpgsql-trigger.html>
- <https://www.postgresql.org/docs/current/sql-createtrigger.html>
- <https://www.geeksforgeeks.org/sql/sql-trigger-student-database/>