Triggers in PL/SQL

Triggers in oracle are blocks of PL/SQL code which the oracle engine can execute automatically based on some action or event.

These events can be:

* DDL statements (CREATE, ALTER, DROP, TRUNCATE)
* DML statements (INSERT, SELECT, UPDATE, DELETE)
* Database operation like connecting or disconnecting to oracle (LOGON, LOGOFF, SHUTDOWN)

Triggers are automatically and repeatedly called upon by oracle engine on satisfying certain condition.

Triggers can be activated or deactivated depending on the requirements.

If triggers are activated then they are executed implicitly by oracle engine and if triggers are deactivated then they are executed explicitly by oracle engine.

## **PL/SQL: Uses of Triggers**

Here we have mentioned a few use cases where using triggers proves very helpful:

* Maintaining complex constraints which is either impossible or very difficult via normal constraint(like primary, foreign, unique etc) applying technique.
* Recording the changes made on the table.
* Automatically generating primary key values.
* Prevent invalid transactions to occur.
* Granting authorization and providing security to database.
* Enforcing referential integrity.

## **PL/SQL: Parts of a Trigger**

Whenever a trigger is created, it contains the following three sequential parts:

* **Triggering Event or Statement:** The statements due to which a trigger occurs is called triggering event or statement. Such statements can be DDL statements, DML statements or any database operation, executing which gives rise to a trigger.
* **Trigger Restriction:** The condition or any limitation applied on the trigger is called trigger restriction. Thus, if such a condition is **TRUE** then trigger occurs otherwise it does not occur.
* **Trigger Action:** The body containing the executable statements that is to be executed when trigger occurs that is with the execution of Triggering statement and upon evaluation of Trigger restriction as **True** is called Trigger Action.

## **PL/SQL: Types of Triggers**

The above diagram clearly indicated that Triggers can be classified into three categories:

1. Level Triggers
2. Event Triggers
3. Timing Triggers

**which are further divided into different parts.**

### **Level Triggers**

There are 2 different types of level triggers, they are:

1. **ROW LEVEL TRIGGERS**
   * It fires for every record that got affected with the execution of DML statements like INSERT, UPDATE, DELETE etc.
   * It always use a FOR EACH ROW clause in a triggering statement.
2. **STATEMENT LEVEL TRIGGERS**
   * It fires once for each statement that is executed.

### **Event Triggers**

There are 3 different types of event triggers, they are:

1. **DDL EVENT TRIGGER**
   * It fires with the execution of every DDL statement(CREATE, ALTER, DROP, TRUNCATE).
2. **DML EVENT TRIGGER**
   * It fires with the execution of every DML statement(INSERT, UPDATE, DELETE).
3. **DATABASE EVENT TRIGGER**
   * It fires with the execution of every database operation which can be LOGON, LOGOFF, SHUTDOWN, SERVERERROR etc.

### **Timing Triggers**

There are 2 different types of timing triggers, they are:

* **BEFORE TRIGGER**
  + It fires before executing DML statements.
  + Triggering statement may or may not executed depending upon the before condition miblock.
* **AFTER TRIGGER**
  + It fires after executing DML statement.

### **Syntax for creating Triggers**

CREATE OR REPLACE TRIGGER <trigger\_name>

BEFORE/AFTER/INSTEAD OF

INSERT/DELETE/UPDATE ON <table\_name>

REFERENCING (OLD AS O, NEW AS N)

FOR EACH ROW WHEN (test\_condition)

DECLARE

-- Variable declaration;

BEGIN

-- Executable statements;

EXCEPTION

-- Error handling statements;

END <trigger\_name>;

END;

where,

CREATE OR REPLACE TRIGGER is a keyword used to create a trigger and **<trigger\_name>** is user-defined where a trigger can be given a name.

BEFORE/AFTER/INSTEAD OF specify the timing of the trigger's occurance. INSTEAD OF is used when a view is created.

**INSERT/UPDATE/DELETE** specify the DML statement.

**<table\_name>** specify the name of the table on which DML statement is to be applied.

REFERENCING is a keyword used to provide reference to old and new values for DML statements.

FOR EACH ROW is the clause used to specify row level trigger.

WHEN is a clause used to specify condition to be applied and is only applicable for row-level trigger.

DECLARE, BEGIN, EXCEPTION, END are the different sections of PL/SQL code block containing variable declaration, executable statements, error handling statements and marking end of PL/SQL block respectively where DECLARE and EXCEPTION part are optional.

CREATE OR REPLACE TRIGGER CheckAge

BEFORE

INSERT OR UPDATE ON student

FOR EACH ROW

BEGIN

IF :new.Age>30 THE N

raise\_application\_error(-20001, 'Age should not be greater than 30');

END IF;

END;

After initializing the trigger CheckAge, whenever we will insert any new values or update the existing values in the above table STUDENT our trigger will check the **age** before executing INSERT or UPDATE statements and according to the result of triggering restriction or condition it will execute the statement.

INSERT into STUDENT values(16, 'Saina', 32, 'BCOM');

Age should not be greater than 30