Oracle (PL/SQL)

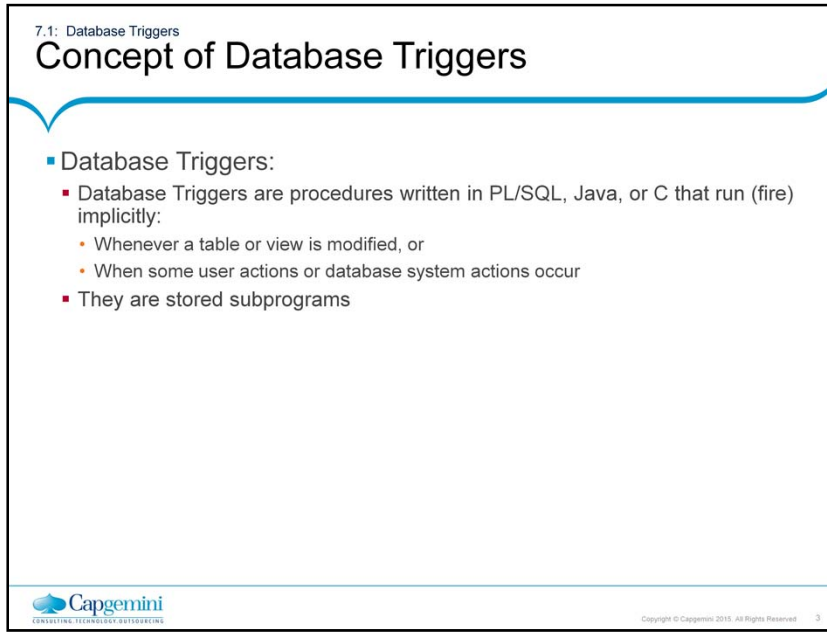
Lesson 7: Database Triggers

Lesson Objectives

- To understand the following topics:
  - Concept of Database Triggers
  - Types of Triggers
  - Disabling and Dropping Triggers
  - Restriction on Triggers
  - Order of Trigger firing
  - Using :Old and :New values, WHEN clause, Trigger predicates
  - Mutating and Constraining tables

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 2



Database Triggers:

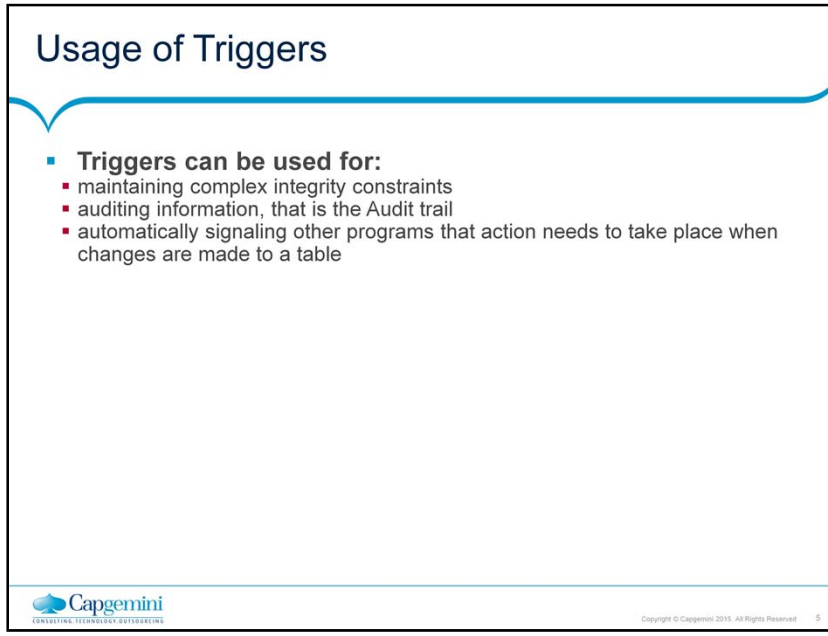
- A Trigger defines an action the database should take when some database related event occurs.
  - Anonymous blocks as well as stored subprograms need to be explicitly invoked by the user.
- Database Triggers are PL/SQL blocks, which are implicitly fired by ORACLE RDBMS whenever an event such as INSERT, UPDATE, or DELETE takes place on a table.
- Database triggers are also stored subprograms.

Concept of Database Triggers

- **You can write triggers that fire whenever one of the following operations occur:**
  - User events:
    - DML statements on a particular schema object
    - DDL statements issued within a schema or database
    - user logon or logoff events
  - System events:
    - Server errors
    - Database startup
    - Instance shutdown

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 4

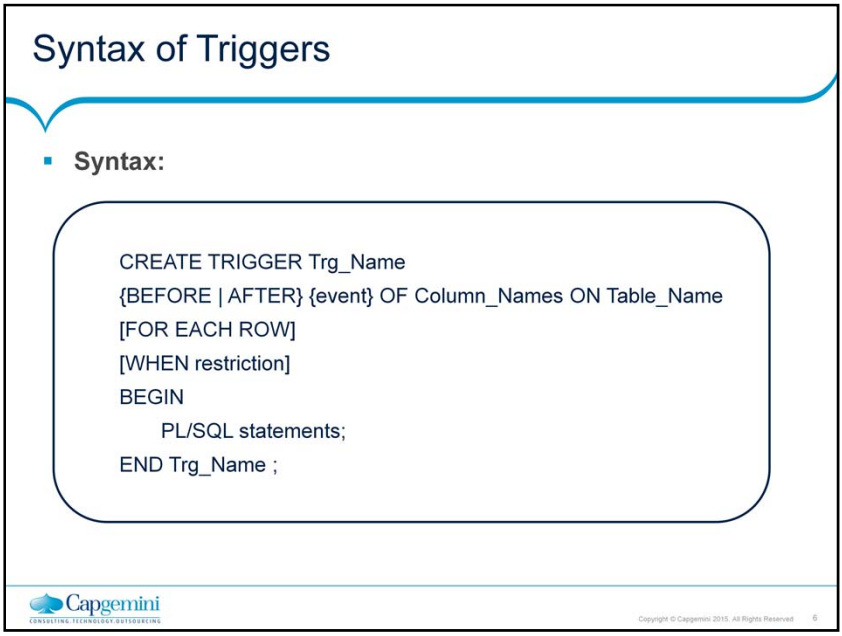


Database Triggers (contd.):

Triggers can be used for:

- Maintaining complex integrity constraints. This is not possible through declarative constraints that are enabled at table creation.
- Auditing information in a table by recording the changes and the identify of the person who made them. This is called as an audit trail.
- Automatically signaling other programs that action needs to take place when changes are made to a table.

contd.



Database Triggers (contd.):

To create a trigger on a table you must be able to alter that table.
The slide shows the syntax for creating a table.

contd.

Database Triggers (contd.):

Parts of a Trigger

A trigger has three basic parts:

- A triggering event or statement
- A trigger restriction
- A trigger action

Triggering Event or Statement

- A triggering event or statement is the SQL statement, database event, or user event that causes a trigger to fire. A triggering event can be one or more of the following:
  - An INSERT, UPDATE, or DELETE statement on a specific table (or view, in some cases)
  - A CREATE, ALTER, or DROP statement on any schema object
  - A database startup or instance shutdown
  - A specific error message or any error message
  - A user logon or logoff

Trigger Restriction

- A trigger restriction specifies a Boolean expression that must be true for the trigger to fire. The trigger action is not run if the trigger restriction evaluates to false or unknown.

Trigger Action

- A trigger action is the procedure (PL/SQL block, Java program, or C callout) that contains the SQL statements and code to be run when the following events occur:
  - a triggering statement is issued
  - the trigger restriction evaluates to true
- Like stored procedures, a trigger action can:
  - contain SQL, PL/SQL, or Java statements
  - define PL/SQL language constructs such as variables, constants, cursors, exceptions
  - define Java language constructs
  - call stored procedures

Types of Triggers

- **Types of Triggers:**
  - Type of Trigger is determined by the triggering event, namely:
    - INSERT
    - UPDATE
    - DELETE
  - Triggers can be fired:
    - before or after the operation
    - on row or statement operations
  - Trigger can be fired for more than one type of triggering statement

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 8

Types of Triggers

Category	Values	Comments
Statement	INSERT, DELETE, UPDATE	Defines which kind of DML statement causes the trigger to fire.
Timing	BEFORE, AFTER	Defines whether the trigger fires before the statement is executed or after the statement is executed.
Level	Row or Statement	• If the trigger is a row-level trigger, it fires once for each row affected by the triggering statement. • If the trigger is a statement-level trigger, it fires once, either before or after the statement. • A row-level trigger is identified by the FOR EACH ROW clause in the trigger definition.

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 9

Note:

- A trigger can be fired for more than one type of triggering statement. In case of multiple events, OR separates the events.
- From ORACLE 7.1 there is no limit on number of triggers you can write for a table.
  - Earlier you could write maximum of 12 triggers per table, one of each type.

Types of Triggers (contd.)

Row Triggers and Statement Triggers

- When you define a trigger, you can specify the number of times the trigger action has to be run:
  - Once for every row affected by the triggering statement, such as a trigger fired by an UPDATE statement that updates many rows.
  - Once for the triggering statement, no matter how many rows it affects.

Row Triggers

- A Row Trigger is fired each time the table is affected by the triggering statement. For example: If an UPDATE statement updates multiple rows of a table, a row trigger is fired once for each row affected by the UPDATE statement.
- If a triggering statement affects no rows, a row trigger is not run.
- Row triggers are useful if the code in the trigger action depends on data provided by the triggering.

INSTEAD OF Triggers

- INSTEAD OF triggers provide a transparent way of modifying views that cannot be directly modified through DML statements (INSERT, UPDATE, and DELETE). These triggers are called INSTEAD OF triggers because, unlike other types of triggers, Oracle fires the trigger instead of executing the triggering statement.
- You can write normal INSERT, UPDATE, and DELETE statements against the view and the INSTEAD OF trigger is fired to update the underlying tables appropriately. INSTEAD OF triggers are activated for each row of the view that gets modified.

Modify Views

- Modifying views can have ambiguous results:
  - Deleting a row in a view could either mean deleting it from the base table or updating some values so that it is no longer selected by the view.
  - Inserting a row in a view could either mean inserting a new row into the base table or updating an existing row so that it is projected by the view.
  - Updating a column in a view that involves joins might change the semantics of other columns that are not projected by the view.
- Object views present additional problems. For example, a key use of object views is to represent master/detail relationships. This operation inevitably involves joins, but modifying joins is inherently ambiguous.
- As a result of these ambiguities, there are many restrictions on which views are modifiable. An INSTEAD OF trigger can be used on object views as well as relational views that are not otherwise modifiable.

contd.

Types of Triggers (contd.)

Modify Views (contd.)

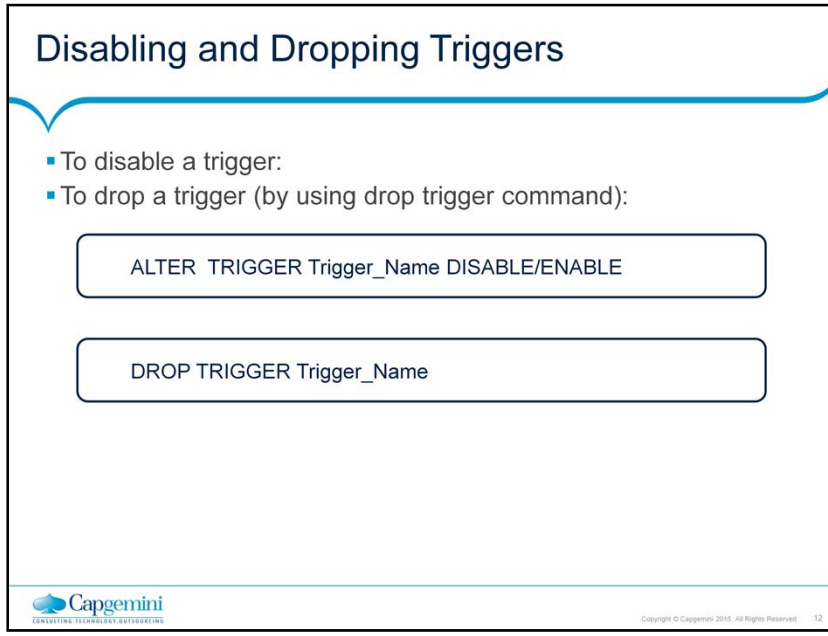
- A view is inherently modifiable if data can be inserted, updated, or deleted without using INSTEAD OF triggers and if it conforms to the restrictions. Even if the view is inherently modifiable, you might want to perform validations on the values being inserted, updated or deleted. INSTEAD OF triggers can also be used in this case. Here the trigger code performs the validation on the rows being modified and if valid, propagates the changes to the underlying tables, statement or rows that are affected.

Statement Triggers

- A statement trigger is fired once on behalf of the triggering statement, regardless of the number of rows in the table that the triggering statement affects, even if no rows are affected.
  For example: If a DELETE statement deletes several rows from a table, a statement-level DELETE trigger is fired only once.
- Statement triggers are useful if the code in the trigger action does not depend on the data provided by the triggering statement or the rows affected.
- For example: Use a statement trigger to:
  - make a complex security check on the current time or user
  - generate a single audit record

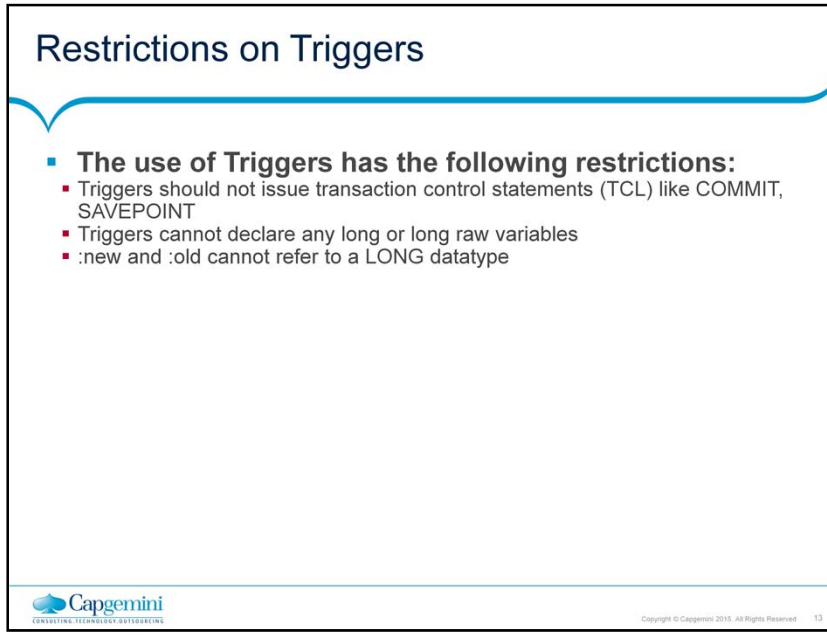
BEFORE and AFTER Triggers

- When defining a trigger, you can specify the trigger timing — whether the trigger action has to be run before or after the triggering statement.
- BEFORE and AFTER apply to both statement and row triggers.
- BEFORE and AFTER triggers fired by DML statements can be defined only on tables, not on views. However, triggers on the base tables of a view are fired if an INSERT, UPDATE, or DELETE statement is issued against the view. BEFORE and AFTER triggers fired by DDL statements can be defined only on the database or a schema, and not on particular tables.



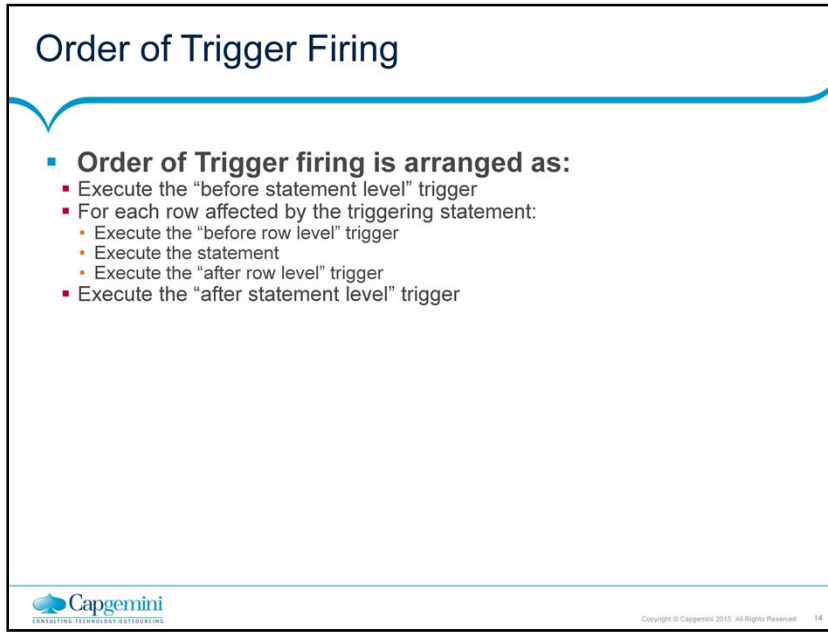
Note:

An UPDATE or DELETE statement affects multiple rows of a table.
If the trigger is fired once for each affected row, then FOR EACH ROW clause is required. Such a trigger is called a ROW trigger.
If the FOR EACH ROW clause is absent, then the trigger is fired only once for the entire statement. Such triggers are called STATEMENT triggers



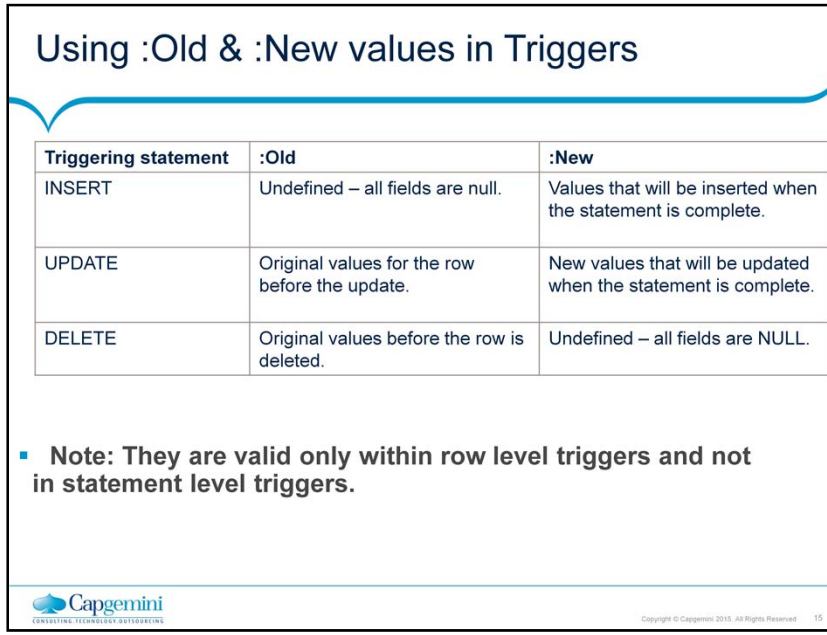
Restrictions on Triggers:

- A Trigger may not issue any transaction control statements (TCL) like COMMIT, SAVEPOINT.
  - Since the triggering statement and the trigger are part of the same transaction, whenever triggering statement is committed or rolled back the work done in the trigger gets committed or rolled back, as well.
- Any procedures or functions that are called by the Trigger cannot have any transaction control statements.
- Trigger body cannot declare any long or long raw variables. Also :new and :old cannot refer to a LONG datatype.
- There are restrictions on the tables that a trigger body can access depending on type of “triggers” and “constraints” on the table.



Note:

- As each trigger is fired, it will see the changes made by the earlier triggers, as well as the database changes made by the statement.
- The order in which triggers of same type are fired is not defined.
- If the order is important, combine all the operations into one trigger.

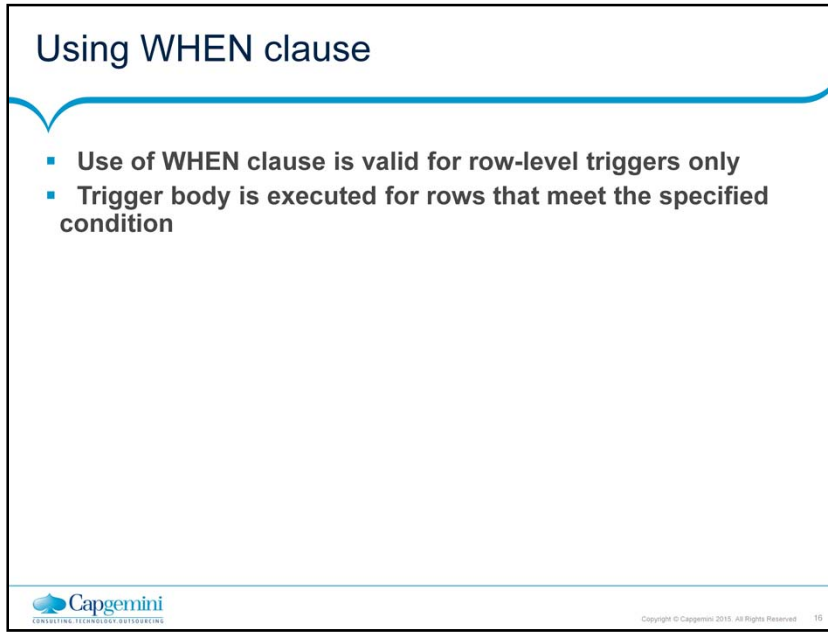


Using :old and :new values in Row Level Triggers:

- Row level trigger fires once per row that is being processed by the triggering statement.
- Inside the trigger, you can access the row that is currently being processed. This is done through keywords :new and :old (they are called as pseudo records). The meaning of the terms is as shown in the slide.

Note:

- The pseudo records are valid only within row level triggers and not in statement level triggers.
  - :old values are not available if the triggering statement is INSERT.
  - :new values are not available if the triggering statement is DELETE.
- Each column is referenced by using the notation :old.Column_Name or :new.Column_Name.
- If a column is not updated by the triggering update statement, then :old and :new values remain the same.



Using WHEN Clause:

- The WHEN clause is valid for row-level triggers only. If present, the trigger body will be executed only for those rows that meet the condition specified by the WHEN clause.
- The :new and :old records can be used here without colon.

contd.

Summary

- **In this lesson, you have learnt:**
  - Database Triggers are procedures written in PL/SQL, Java, or C that run (fire) implicitly:
    - whenever a table or view is modified, or
    - when some user actions or database system actions occur
- **There are three types of triggers:**
  - Statement based triggers
  - Timing based triggers
  - Level based triggers

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 17

Summary

- Disabling and Dropping triggers can be done instead of actually removing the triggers
- Order of trigger firing is decided depending on the type of triggers used in the sequence

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 18

Review Question

- **Question 1: Triggers should not issue Transaction Control Statements (TCL)**
  - True / False
- **Question 2: BEFORE DROP and AFTER DROP triggers are fired when a schema object is dropped**
  - True / False
- **Question 3: The :new and :old records must be used in WHEN clause with a colon**
  - True / False

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 19

Review Question

- Question 4: A ___ is a table that is currently being modified by a DML statement
- Question 5: A ___ is fired once on behalf of the triggering statement, regardless of the number of rows in the table that the triggering statement affects

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING
Copyright © Capgemini 2015. All Rights Reserved