Database Programming with PL/SQL

Managing Triggers

Objectives

This lesson covers the following objectives:

- View trigger information in the Data Dictionary
- Disable and enable a database trigger
- Remove a trigger from the database

Purpose

There might be times when you want to turn off a trigger in order to perform some maintenance or debug some code.

Or, you might need to fill in for someone else and, in order to understand the triggers that exist in the Data Dictionary, you might need to view them.

You can do all of this by managing triggers.

Privileges Needed for Triggers

To create a trigger in your own schema, you need:

- `CREATE TRIGGER` system privilege
- Normal object privileges (`SELECT`, `UPDATE`, `EXECUTE`, and so on) on objects in other schemas that are referenced in your trigger body
- `ALTER` privilege on the table or view associated with the trigger.

Statements in the trigger body use the privileges of the trigger owner, not the privileges of the user executing the operation that fires the trigger. The next slide shows an example.

Privileges Needed for Triggers Example

User MOE needs to create the following trigger:

```
CREATE OR REPLACE TRIGGER upd_tom_emp
AFTER UPDATE ON tom.employees
BEGIN
  INSERT INTO mary.log_table VALUES(USER,SYSDATE);
  sharon.calledproc;
END;
```

Moe needs:

- CREATE TRIGGER
- ALTER on TOM.EMPLOYEES
- INSERT on MARY.LOG_TABLE
- EXECUTE on SHARON.CALLEDPROC.

Viewing Triggers in the Data Dictionary

You can see trigger information in the following Data Dictionary views:

- `USER_OBJECTS`: Object name and object type (as for all other object types in your schema)
- `USER_TRIGGERS`: Detailed code and status of the trigger
- `USER_ERRORS`: PL/SQL syntax errors (compilation errors) of the trigger

Using USER_TRIGGERS

Column*	Column Description
TRIGGER_NAME	Name of the trigger
TRIGGER_TYPE	The type is BEFORE, AFTER, INSTEAD OF
TRIGGERING_EVENT	The DML operation firing the trigger
TABLE_NAME	Name of the database table
REFERENCING_NAMES	Name used for :OLD and :NEW
WHEN_CLAUSE	The when_clause used
STATUS	The status of the trigger
TRIGGER_BODY	The action to take

* Not all columns are shown here

Viewing Trigger Information Using USER_TRIGGERS

This example shows the triggering event, timing, type of trigger, status, and detailed body code of the RESTRICT_SALARY trigger:

```
SELECT trigger_name, trigger_type, triggering_event,
       table_name, status, trigger_body
  FROM   USER_TRIGGERS
  WHERE  trigger_name = 'RESTRICT_SALARY';
```

TRIGGER_NAME	TRIGGER_TYPE	TRIGGERING_EVENT	TABLE_NAME	STATUS	TRIGGER_BODY
RESTRICT_SALARY	BEFORE EACH ROW	INSERT OR UPDATE	EMPLOYEES	ENABLED	BEGIN IF NOT (:NEW job_id IN ('AD_PRES', 'AD_VP')) AND :NEW.salary > 15000 THEN RAISE_APPLICATION_ERROR (-20202, 'Employee cannot earn more than $15,000'); END IF; END;

Changing the Status of Triggers

Disable or reenable a database trigger:

```
ALTER TRIGGER trigger_name DISABLE | ENABLE;
```

Disable or reenable all triggers for a table:

```
ALTER TABLE table_name DISABLE | ENABLE ALL TRIGGERS;
```

Recompile a trigger for a table:

```
ALTER TRIGGER trigger_name COMPILE;
```

Removing Triggers

To remove a trigger from the database, use the `DROP TRIGGER` statement:

```
DROP TRIGGER trigger_name;
```

Example:

```
DROP TRIGGER secure_emp;
```

Note: All triggers on a table are removed when the table is removed.

Summary

In this lesson, you should have learned how to:

- View trigger information in the data dictionary
- Disable and enable a database trigger
- Remove a trigger from the database