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

• Or, in order to understand the triggers that exist in the Data Dictionary, you may 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.

• To create triggers in other users' schemas, you need the CREATE ANY TRIGGER privilege.
Privileges Needed for Triggers

• Statements in the trigger body use the privileges of the trigger owner (Definer's Rights), NOT the privileges of the user executing the operation that fires the trigger (Invoker's Rights).

• You cannot specify Invoker’s Rights (AUTHID CURRENT_USER) for a trigger.

• The next slide shows an example.
Privileges Needed for Triggers Example

• User Monica 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;
```

• Monica needs the following privileges:
  – 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
- **Source code for triggers is in USER_TRIGGERS not USER_SOURCE.**
### USER_TRIGGERS Data Dictionary

<table>
<thead>
<tr>
<th>Column*</th>
<th>Column Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TRIGGER_NAME</td>
<td>Name of the trigger</td>
</tr>
<tr>
<td>TRIGGER_TYPE</td>
<td>When it fires - BEFORE, AFTER, ROW, etc.</td>
</tr>
<tr>
<td>TRIGGERING_EVENT</td>
<td>The DML operation firing the trigger</td>
</tr>
<tr>
<td>TABLE_NAME</td>
<td>Name of the associated table</td>
</tr>
<tr>
<td>REFERENCING_NAMES</td>
<td>Name used for :OLD and :NEW</td>
</tr>
<tr>
<td>WHEN_CLAUSE</td>
<td>The when_clause used</td>
</tr>
<tr>
<td>STATUS</td>
<td>The status of the trigger</td>
</tr>
<tr>
<td>TRIGGER_BODY</td>
<td>Action taken by the trigger</td>
</tr>
</tbody>
</table>

• 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';
```

<table>
<thead>
<tr>
<th>TRIGGER_NAME</th>
<th>TRIGGER_TYPE</th>
<th>TRIGGERING_EVENT</th>
<th>TABLE_NAME</th>
<th>STATUS</th>
<th>TRIGGER_BODY</th>
</tr>
</thead>
<tbody>
<tr>
<td>RESTRICT_SALARY</td>
<td>BEFORE EACH ROW</td>
<td>INSERT OR UPDATE</td>
<td>EMPLOYEES</td>
<td>ENABLED</td>
<td>BEGIN IF NOT (:NEW.job_id IN ('AD_PRES', 'AD_VP')) AND :NEW.salary &gt; 15000 THEN RAISE_APPLICATION_ERROR (-20202, 'Employee cannot earn more than $15,000'); END IF; END;</td>
</tr>
</tbody>
</table>
Changing the Status of Triggers

• Disable or re-enable a database trigger:
  ```sql
  ALTER TRIGGER trigger_name DISABLE | ENABLE;
  ```

• Disable or re-enable all triggers for a table:
  ```sql
  ALTER TABLE table_name DISABLE | ENABLE ALL TRIGGERS;
  ```

• Recompile a trigger for a table:
  ```sql
  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.
Terminology

Key terms used in this lesson included:

• ALL_TRIGGERS
• USER_TRIGGERS
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