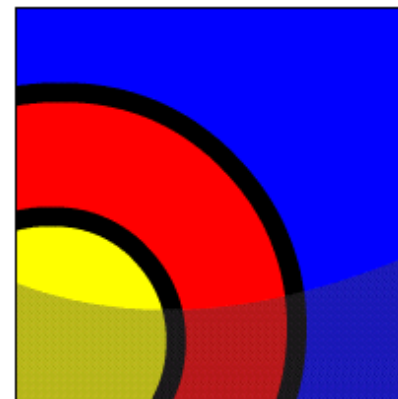


Creating DDL and Database Event Triggers

What Will I Learn?

In this lesson, you will learn to:

- Describe events that cause DDL and database event triggers to fire
- Create a trigger for a DDL statement
- Create a trigger for a database event
- Describe the functionality of the `CALL` statement
- Describe the cause of a mutating table





Why Learn It?

What if you accidentally drop an important table? If you have a backup copy of the table data, you can retrieve the lost data. But it might be important to know exactly when the table was dropped.

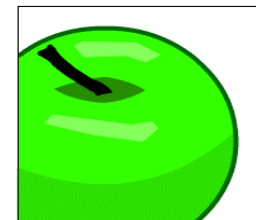


For security reasons, a Database Administrator might want to keep an automatic record of who has logged into a database, and when.

These are two examples of the uses of DDL and Database Event triggers.

Tell Me / Show Me

What are DDL and Database Event Triggers?



DDL triggers are fired by DDL statements: CREATE, ALTER, or DROP.

Database Event triggers are fired by non-SQL events in the database, for example:

- A user connects to, or disconnects from, the database.
- The DBA starts up, or shuts down, the database.
- A specific exception is raised in a user session.



Tell Me / Show Me

Creating Triggers on DDL Statements

Syntax:

```
CREATE [OR REPLACE] TRIGGER trigger_name  
Timing  
[ddl_event1 [OR ddl_event2 OR ...]]  
ON {DATABASE|SCHEMA}  
trigger_body
```

- ON DATABASE fires the trigger for DDL on all schemas in the database
- ON SCHEMA fires the trigger only for DDL on objects in your own schema



Tell Me / Show Me

Example of a DDL Trigger:

You want to write a log record every time a new database object is created in your schema:

```
CREATE OR REPLACE TRIGGER log_create_trigg  
AFTER CREATE ON SCHEMA  
BEGIN  
    INSERT INTO log_table  
        VALUES (USER, SYSDATE);  
END;
```

The trigger fires whenever any (type of) object is created. You cannot create a DDL trigger that refers to a specific database object.



Tell Me / Show Me

A Second Example of a DDL Trigger

You want to prevent any objects being dropped from your schema.

```
CREATE OR REPLACE TRIGGER prevent_drop_trigg  
BEFORE DROP ON SCHEMA  
BEGIN  
    RAISE_APPLICATION_ERROR  
        (-20203, 'Attempted drop - failed');  
END;
```

The trigger fires whenever any (type of) object is dropped. Again, you cannot create a DDL trigger that refers to a specific database object.



Tell Me / Show Me

Creating Triggers on Database Events

Syntax:

```
CREATE [OR REPLACE] TRIGGER trigger_name
    timing
    [database_event1 [OR database_event2 OR ...]]
ON {DATABASE|SCHEMA}
    trigger_body
```

- ON DATABASE fires the trigger for events on all sessions in the database.
- ON SCHEMA fires the trigger only for your own sessions.



Tell Me / Show Me

Example 1: LOGON and LOGOFF Triggers:

```
CREATE OR REPLACE TRIGGER logon_trig
AFTER LOGON ON SCHEMA
BEGIN
    INSERT INTO log_trig_table(user_id,log_date,action)
        VALUES (USER, SYSDATE, 'Logging on');
END;
```

```
CREATE OR REPLACE TRIGGER logoff_trig
BEFORE LOGOFF ON SCHEMA
BEGIN
    INSERT INTO log_trig_table(user_id,log_date,action)
        VALUES (USER, SYSDATE, 'Logging off');
END;
```



Tell Me / Show Me

Example 2: A SERVERERROR Trigger:

You want to keep a log of any ORA-00942 errors that occur in your sessions:

```
CREATE OR REPLACE TRIGGER servererror_trig
AFTER SERVERERROR ON SCHEMA
BEGIN
    IF (IS_SERVERERROR (942)) THEN
        INSERT INTO error_log_table ...
    END IF;
END;
```



Tell Me / Show Me

CALL Statements in a Trigger:

```
CREATE [OR REPLACE] TRIGGER trigger_name
timing
event1 [OR event2 OR event3]
ON table_name
[REFERENCING OLD AS old | NEW AS new]
[FOR EACH ROW]
[WHEN condition]
CALL procedure_name
```

```
CREATE OR REPLACE TRIGGER log_employee
BEFORE INSERT ON EMPLOYEES
CALL log_execution
```

There is no `END;` statement, and no semicolon at the end of the `CALL` statement.



Tell Me / Show Me

Mutating Tables and Row Triggers

A mutating table is a table that is currently being modified by a DML statement.

A row trigger cannot `SELECT` from a mutating table, because it would see an inconsistent set of data (the data in the table would be changing while the trigger was trying to read it).

However, a row trigger can `SELECT` from a different table if needed.

This restriction does not apply to DML statement triggers, only to DML row triggers.



Tell Me / Show Me

Mutating Table: Example

```
CREATE OR REPLACE TRIGGER check_salary
  BEFORE INSERT OR UPDATE OF salary, job_id ON employees
  FOR EACH ROW
DECLARE
  v_minsalary employees.salary%TYPE;
  v_maxsalary employees.salary%TYPE;
BEGIN
  SELECT MIN(salary), MAX(salary)
    INTO v_minsalary, v_maxsalary
   FROM employees
  WHERE job_id = :NEW.job_id;
  IF :NEW.salary < v_minsalary OR
     :NEW.salary > v_maxsalary THEN
    RAISE_APPLICATION_ERROR(-20505,'Out of range');
  END IF;
END;
```



Tell Me / Show Me

Mutating Table: Example (continued)

```
UPDATE employees
  SET salary = 3400
  WHERE last_name = 'Davies';
```

```
ORA-04091: table USVA_TEST_SQL01_T01.EMPLOYEES is mutating, trigger/function may not
see it
ORA-06512: at "USVA_TEST_SQL01_T01.CHECK_SALARY", line 5
ORA-04088: error during execution of trigger 'USVA_TEST_SQL01_T01.CHECK_SALARY'
3.  WHERE last_name = 'Davies';
```

Tell Me / Show Me

Terminology

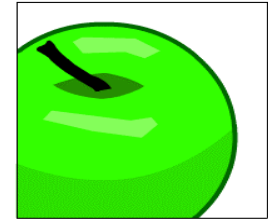
Key terms used in this lesson include:

DDL trigger

Database Event trigger

CALL statement

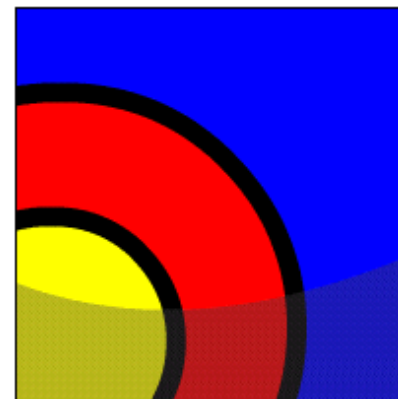
Mutating table



Summary

In this lesson, you learned to:

- Describe events that cause DDL and database event triggers to fire
- Create a trigger for a DDL statement
- Create a trigger for a database event
- Describe the functionality of the `CALL` statement
- Describe the cause of a mutating table





Try It / Solve It

The exercises in this lesson cover the following topics:

- Describing events that cause DDL and database event triggers to fire
- Creating a trigger for a DDL statement
- Creating a trigger for a database event
- Describing the functionality of the `CALL` statement
- Describing the cause of a mutating table

