

Introduction to Triggers

What Will I Learn?

In this lesson, you will learn to:

- Describe database triggers and their uses
- Define a database trigger
- Recognize the difference between a database trigger and an application trigger
- List two or more guidelines for using triggers
- Compare and contrast database triggers and stored procedures





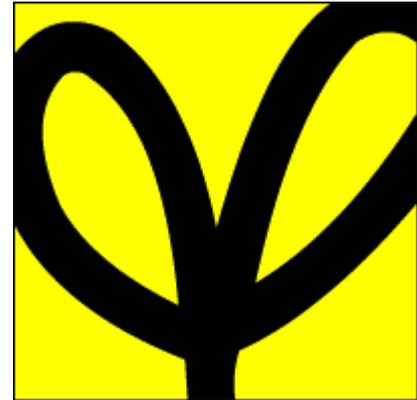
Why Learn It?

In this lesson, you learn about a database trigger.

Triggers allow specified actions to be performed automatically within the database, without having to write extra application code.

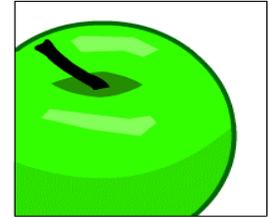
Triggers increase the power of the database, and the power of your application.

You will learn much more about triggers in the following lessons.



Tell Me / Show Me

Need For A Trigger



Let's start with an example: A business rule states that whenever an employee's salary is changed, the change must be recorded in a logging table.

You could create two procedures to do this: `UPD_EMP_SAL` to update the salary, and `LOG_SAL_CHANGE` to insert the row into the logging table. And you could invoke `LOG_SAL_CHANGE` from within `UPD_EMP_SAL`, or invoke `LOG_SAL_CHANGE` separately from the calling environment.

But you do not have to do this. Instead, you create a trigger. The next slide shows how.

Tell Me / Show Me

Example of a Simple Trigger

```
CREATE OR REPLACE TRIGGER log_sal_change_trigg
AFTER UPDATE OF salary ON employees
BEGIN
    INSERT INTO log_table (user_id, logon_date)
        VALUES (USER, SYSDATE);
END;
```

From now on, whenever an SQL statement updates a salary, this trigger executes automatically, inserting the row into the logging table.

You say that the trigger automatically fires (that is, executes) whenever the triggering event (updating a salary) occurs.

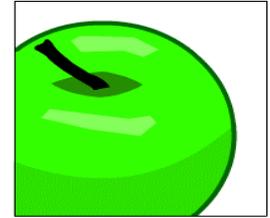
Cause and effect: The event occurs, and the trigger fires.

Tell Me / Show Me

What Is a Trigger?

A database trigger:

- Is a PL/SQL block associated with a specific action (an event) on a database object, such as a table or view.
- Is stored in the database
- Executes automatically whenever the associated action occurs.
- In the example on the previous slide, the trigger is associated with the action: `UPDATE OF salary ON employees.`



PL/SQL Block

|
association
|

Table, View,
etc.



Tell Me / Show Me

Application Compared to Database Triggers

- Database triggers execute automatically whenever a data event (such as DML or DDL) or a system event (such as a user connecting, or the DBA shutting down the database) occurs on a schema or database. Database triggers are created and stored in the database, just like PL/SQL procedures, functions, and packages.
- Application triggers execute automatically whenever a particular event occurs within an application. Application triggers are used extensively in applications developed with Oracle Forms Developer.

This course focuses on database triggers.

Tell Me / Show Me

Which Events Can Cause a Database Trigger to Fire?

The following events in the database can cause a trigger to fire:

- DML operations on a table
- DML operations on a view, with an `INSTEAD OF` trigger
- DDL statements, such as `CREATE` and `ALTER`
- Database system events, such as when a user logs on or the DBA shuts down the database

You will learn about each of these types of triggers in detail throughout the lessons in this section.

Tell Me / Show Me

Possible Uses for Triggers

You can use triggers to:

- Enhance complex database security rules
- Create auditing records automatically
- Enforce complex data integrity rules
- Create logging records automatically
- Prevent tables from being accidentally dropped
- Prevent invalid DML transactions from occurring

- And many other purposes!

Tell Me / Show Me

More Uses for Triggers

You can use triggers to:

- Generate derived column values automatically
- Maintain synchronous table replication
- Gather statistics on table access
- Modify table data when DML statements are issued against views

- And many other purposes!

Tell Me / Show Me

Example 1: Creating Logging Records Automatically

The Database Administrator wants to keep an automatic record (in a database table) of who logs onto the database, and when. He/she could create the log table and a suitable trigger as follows:

```
CREATE TABLE log_table (  
  user_id          VARCHAR2(30),  
  logon_date       DATE);  
  
CREATE OR REPLACE TRIGGER logon_trigg  
AFTER LOGON ON DATABASE  
BEGIN  
  INSERT INTO log_table (user_id, logon_date)  
    VALUES (USER, SYSDATE);  
END;
```



Tell Me / Show Me

Example 2: Enforcing Complex Data Integrity Rules

Imagine a rule that states that no employee's job can be changed to a job that the employee has already done in the past.

```
CREATE OR REPLACE TRIGGER check_sal_trigg
BEFORE UPDATE OF job_id ON employees
FOR EACH ROW
DECLARE
    v_job_count      INTEGER;
BEGIN
    SELECT COUNT(*) INTO v_job_count
    FROM job_history
    WHERE employee_id = :OLD.employee_id
    AND job_id = :NEW.job_id;
    IF v_job_count > 0 THEN
        RAISE_APPLICATION_ERROR
            (-20201,'This employee has already done this job');
    END IF;
END;
```

Tell Me / Show Me

Guidelines for Triggers

Do not define triggers to duplicate or replace actions you can do easily in other ways. For example, implement simple data integrity rules using constraints, not triggers.

Excessive use of triggers can result in complex interdependencies, which can be difficult to maintain. Use triggers only when necessary, and be aware of recursive and cascading effects.

Avoid lengthy trigger logic by creating stored procedures or packaged procedures that are invoked in the trigger body.

 **Tell Me / Show Me****Comparison of Database Triggers and Stored Procedures**

Triggers	Procedures
Defined with <code>CREATE TRIGGER</code>	Defined with <code>CREATE PROCEDURE</code>
Data Dictionary contains source code in <code>USER_TRIGGERS</code>	Data Dictionary contains source code in <code>USER_SOURCE</code>
Implicitly invoked	Explicitly invoked
<code>COMMIT</code>, <code>SAVEPOINT</code>, and <code>ROLLBACK</code> are not allowed	<code>COMMIT</code>, <code>SAVEPOINT</code>, and <code>ROLLBACK</code> are allowed

Tell Me / Show Me

Terminology

Key terms used in this lesson include:

Triggers

Database triggers

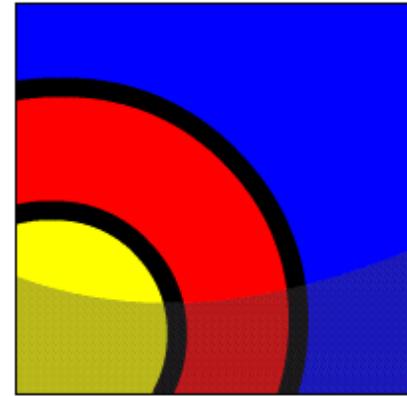
Application triggers



Summary

In this lesson, you learned to:

- Describe database triggers and their uses
- Define a database trigger
- Recognize the difference between a database trigger and an application trigger
- List two or more guidelines for using triggers
- Compare and contrast database triggers and stored procedures





Try It / Solve It

The exercises in this lesson cover the following topics:

- Describing database triggers and their uses
- Defining a database trigger
- Recognizing the difference between a database trigger and an application trigger
- Listing two or more guidelines for using triggers
- Comparing and contrasting database triggers and stored procedures

