upGrad

#LifeKoKaroLift

TechM Full Stack Software Development

upGrad

Course: Foundation of Databases
Lecture On: Advanced PL/SQL - I
Instructor: Vishwa Mohan

upGrad

Today's Agenda

1. **Exception Handling**
2. **Records**
3. **Cursors**
4. **Triggers**

Exception

upGrad

An exception is an error condition or a warning that arises due to errors in the code. The mechanism to handle these exceptions is known as exception handling.

There are two ways to define exceptions in PL/SQL:

1. Internally defined: These exceptions are defined by the runtime system and raised implicitly.
2. Pre-defined: These exceptions have pre-defined names and are implicitly raised at by the runtime system.
3. User defined: These exceptions are explicitly defined by the users.

Exceptions can be handled by a PL/SQL block using an exception handling section that can have one or more exception handlers.

Exception

upGrad

The syntax used is as follows:

```
-- executable section
BEGIN
    ...
-- exception-handling section
EXCEPTION
        WHEN e1 THEN
            -- exception_handler1
        WHEN e2 THEN
            -- exception_handler1
        WHEN OTHERS THEN
            -- all_other_exceptions_handler
END;
```

Let's take a look at example 1.

Exception

upGrad

The RAISE statement

It is used for the following:

- To raise a user-defined exception
- To raise an internally defined exception
- To reraise the current exception

The syntax used for user-defined exception is as follows:

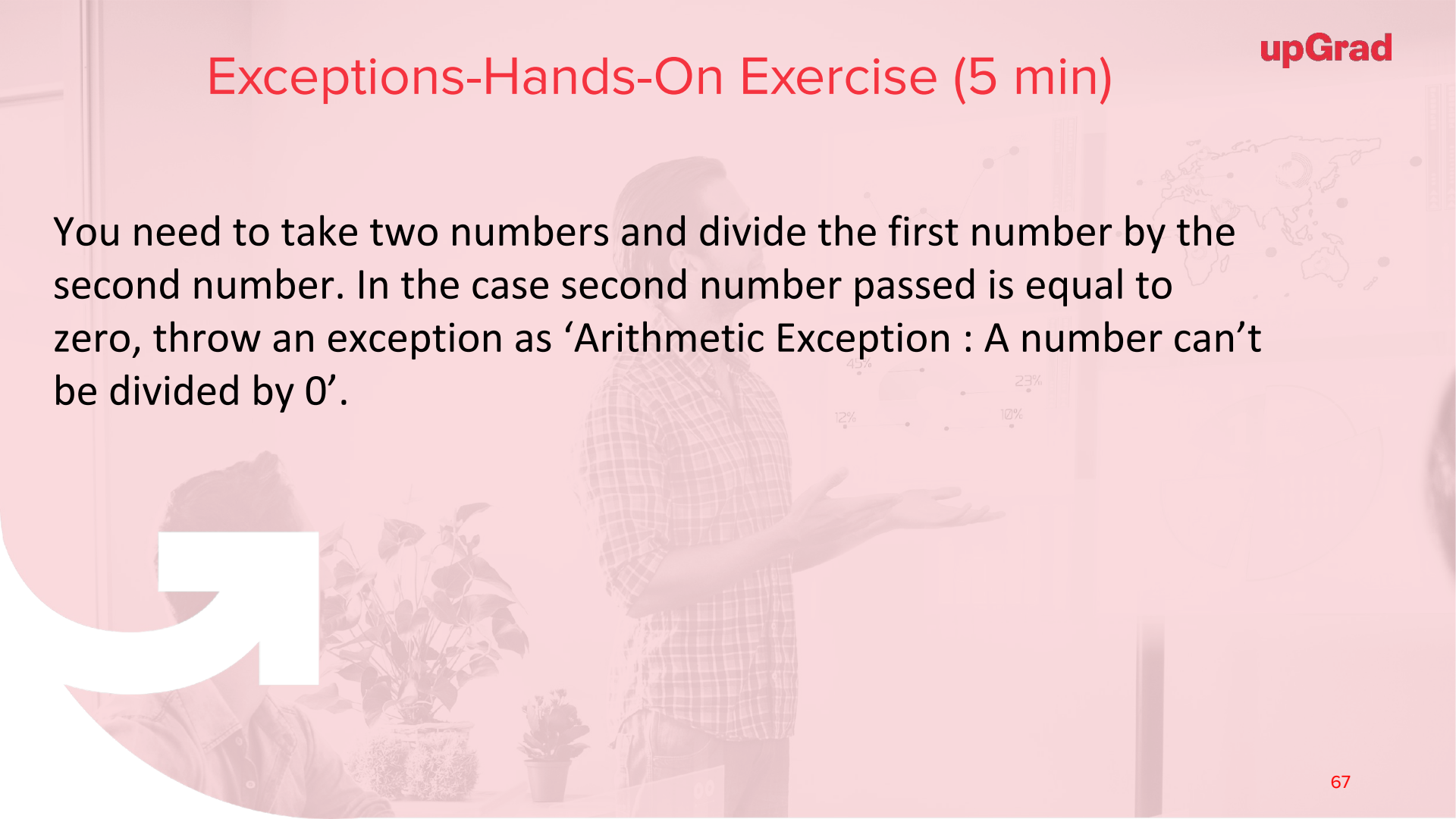
```
DECLARE
   exception_name EXCEPTION;
   PRAGMA EXCEPTION_INIT (exception_name, error_number);
```

Let's take a look at Example 2.

Exceptions-Hands-On Exercise (5 min)

upGrad

You need to take two numbers and divide the first number by the second number. In the case second number passed is equal to zero, throw an exception as ‘Arithmetic Exception : A number can’t be divided by 0’.



PL/SQL Records

upGrad

Record - A composite data structure comprising multiple fields

It helps in moving from field level to record level.

Three type of records:

1. Table-based records
2. Cursor-based records
3. Programmer-defined records

PL/SQL Records

upGrad

Table-Based Record

Syntax:

```
DECLARE
  record_name table_name%ROWTYPE;
```

Example

```
DECLARE
  movie_rec movie%ROWTYPE;
```

movie_rec will have the same columns as movie table.

*The **%ROWTYPE** attribute provides a record type that represents a row in a database table.*

Let's take a look at Example 3

PL/SQL Records

upGrad

Cursor-Based Record

Syntax:

```
DECLARE
   record_name cursor_name%ROWTYPE;
```

Continuing Example 3

PL/SQL Records

upGrad

Programmer-Defined Record

Syntax:

```
TYPE record_type IS RECORD (
    field_name1 data_type1 [[NOT NULL] := | DEFAULT
default_value],
    field_name2 data_type2 [[NOT NULL] := | DEFAULT
default_value],
    ...
);
```

Continuing Example 3

Poll 1

Similar to a row of a database table, a record consists of multiple fields (or columns).

1. True
2. False

Poll 1 (Answer)

Similar to a row of a database table, a record consists of multiple fields (or columns).

1. **True**
2. False

Cursors

upGrad

Cursor is a pointer that points to a result of a query.

Cursors are of two types:

1. Implicit cursors
2. Explicit cursors

Implicit Cursors

Created automatically by Oracle whenever any SQL query is executed.

SELECT INTO - It creates an implicit cursor while fetching the record(s).

INSERT - The cursor holds the data that needs to be inserted.

UPDATE - The cursor identifies the rows that would be affected.

DELETE - The cursor identifies the rows that would be affected.

Explicit Cursors

These are user-defined cursors. They give the developer more control. They have to be defined in the declaration section. They should be used for SELECT statements that return more than one row.

Syntax:

```
CURSOR cursor_name IS
     select_statement;
```

Explicit Cursors

Using an explicit cursor requires performing following steps:

1. Declaring the cursor in the declaration section
2. Opening the cursor (allocates memory). The **OPEN** command is used for opening a cursor
3. Retrieving data from the cursor. The **FETCH** command is used for opening a cursor
4. Releasing the allocated memory by closing the cursor. The **CLOSE** command is used for opening a cursor

Let's take a look at Example 4

Attributes of Cursors

upGrad

Following are the attributes of cursors:

1. **%FOUND**

Returns true if an INSERT/UPDATE/DELETE or a SELECT INTO statement results in one or more rows. Otherwise, returns false

2. **%NOTFOUND**

Logically opposite of %FOUND

3. **%ISOPEN**

Always false for a cursor, as Oracle closes an SQL cursor automatically after execution

4. **%ROWCOUNT**

Returns the number of rows that are affected by an INSERT/UPDATE/DELETE or a SELECT INTO statement

Continuing Example 4

Cursors: Hands-On Exercise (5 min)

upGrad

Create an employee table with the following columns:
id, name, gender, salary, rating.

Insert five records into the table.

Perform the following operations on the table:

1. Use an implicit cursor to increase the salary of every employee who has received a rating of greater than or equal to 3 by 10%

2. Use an explicit cursor to print all the records of the updated table after the salary increments

Poll 2

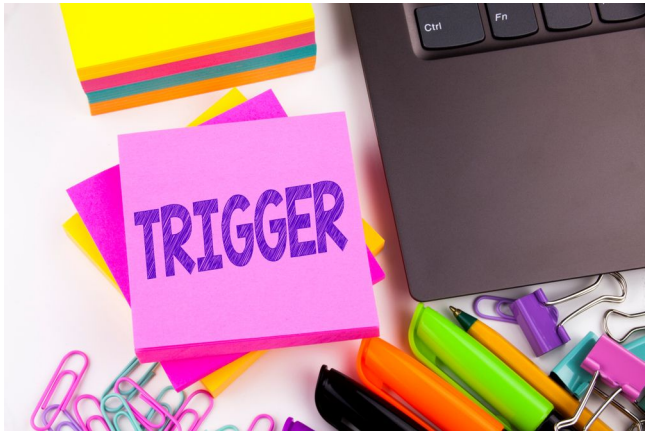
Which of the following statements about cursors is correct?

1. Implicit cursors are created automatically by Oracle
2. Explicit cursors have to be defined manually
3. Both statement 1 and statement 2
4. None of the above

Poll 2 (Answer)

Which of the following statements about cursors is correct?

1. Implicit cursors are created automatically by Oracle
2. Explicit cursors have to be defined manually
3. **Both statement 1 and statement 2**
4. None of the above



Triggers

A trigger is a named PL/SQL unit stored in the database that can be invoked repeatedly. A trigger can be **Enabled or Disabled**, but it cannot be invoked explicitly. It is invoked automatically by the database when the event occurs in case the trigger is enabled.

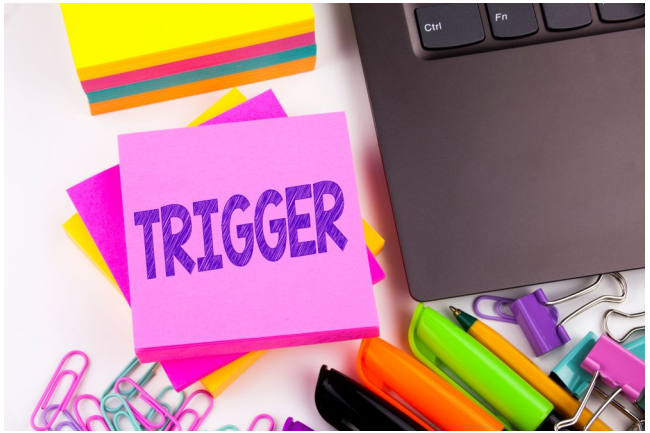
Triggers are executed in response to one of the following events:

DML (DELETE, INSERT or UPDATE)

DDL (CREATE, ALTER, DROP, etc.)

Database Events (SERVERERROR, LOGON, LOGOFF, STARTUP or SHUTDOWN)

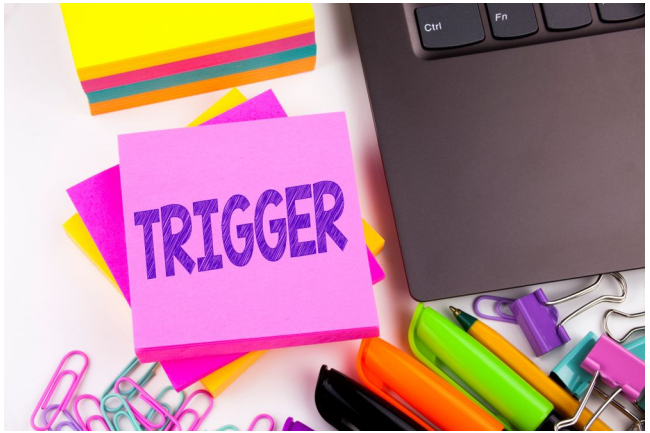
Triggers



Triggers

They can be defined on:

1. Tables
1. Views
1. Schemas
1. Databases

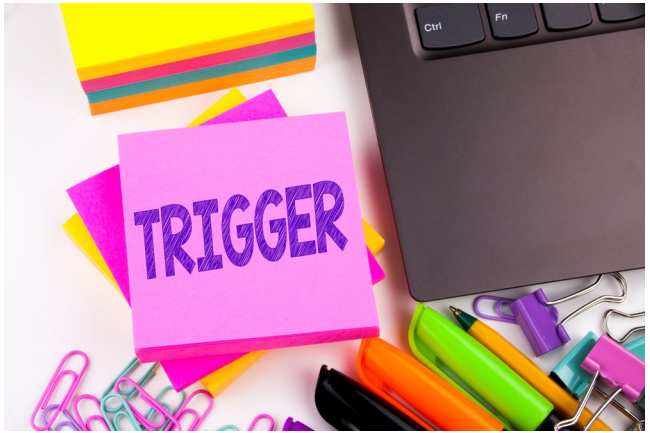


Triggers

Use Cases for Triggers

1. Generate virtual column values automatically
2. Log events
3. Gather statistics on table access
4. Modify table data when DML statements are issued against views
5. Enforce referential integrity when child and parent tables are on different nodes of a distributed database
6. Publish information about database events, user events and SQL statements to subscribing applications
7. Prevent DML operations on a table after regular business hours
8. Prevent invalid transactions
9. Enforce complex business or referential integrity rules that you cannot define with constraints

Reference: Official Documentation



Triggers

Syntax:

```
CREATE [OR REPLACE ] TRIGGER trigger_name
{BEFORE | AFTER | INSTEAD OF }
{INSERT [OR] | UPDATE [OR] | DELETE}
[OF col_name]
ON table_name
[REFERENCING OLD AS o NEW AS n]
[FOR EACH ROW]
WHEN (condition)
DECLARE
  Declaration-statements
BEGIN
  Executable-statements
EXCEPTION
  Exception-handling-statements

END;
```

Let's take a look at Example 5

upGrad

Triggers

Syntax:

```
CREATE [OR REPLACE ] TRIGGER trigger_name
{BEFORE | AFTER | INSTEAD OF }
{INSERT [OR] | UPDATE [OR] | DELETE}
[OF col_name]
ON table_name
[REFERENCING OLD AS o NEW AS n]
[FOR EACH ROW]
WHEN (condition)
DECLARE
  Declaration-statements
BEGIN
  Executable-statements
EXCEPTION
  Exception-handling-statements

END;
```

CREATE [OR REPLACE] TRIGGER - It will create a new trigger if not present already. Otherwise, it will update the already existing trigger.

BEFORE | AFTER | INSTEAD OF - This indicates when a trigger has to be applied.

INSERT [OR] | UPDATE [OR] | DELETE - This indicates the DML statements on which it will be triggered.

[OF col_name] - This indicates the column name that will be updated.

ON table_name - This represents the table on which the trigger has been applied.

REFERENCING OLD AS o NEW AS n - It is used for relating the old and new values for different DML statements

[FOR EACH ROW] - It specifies that the trigger will be executed for each row of records.

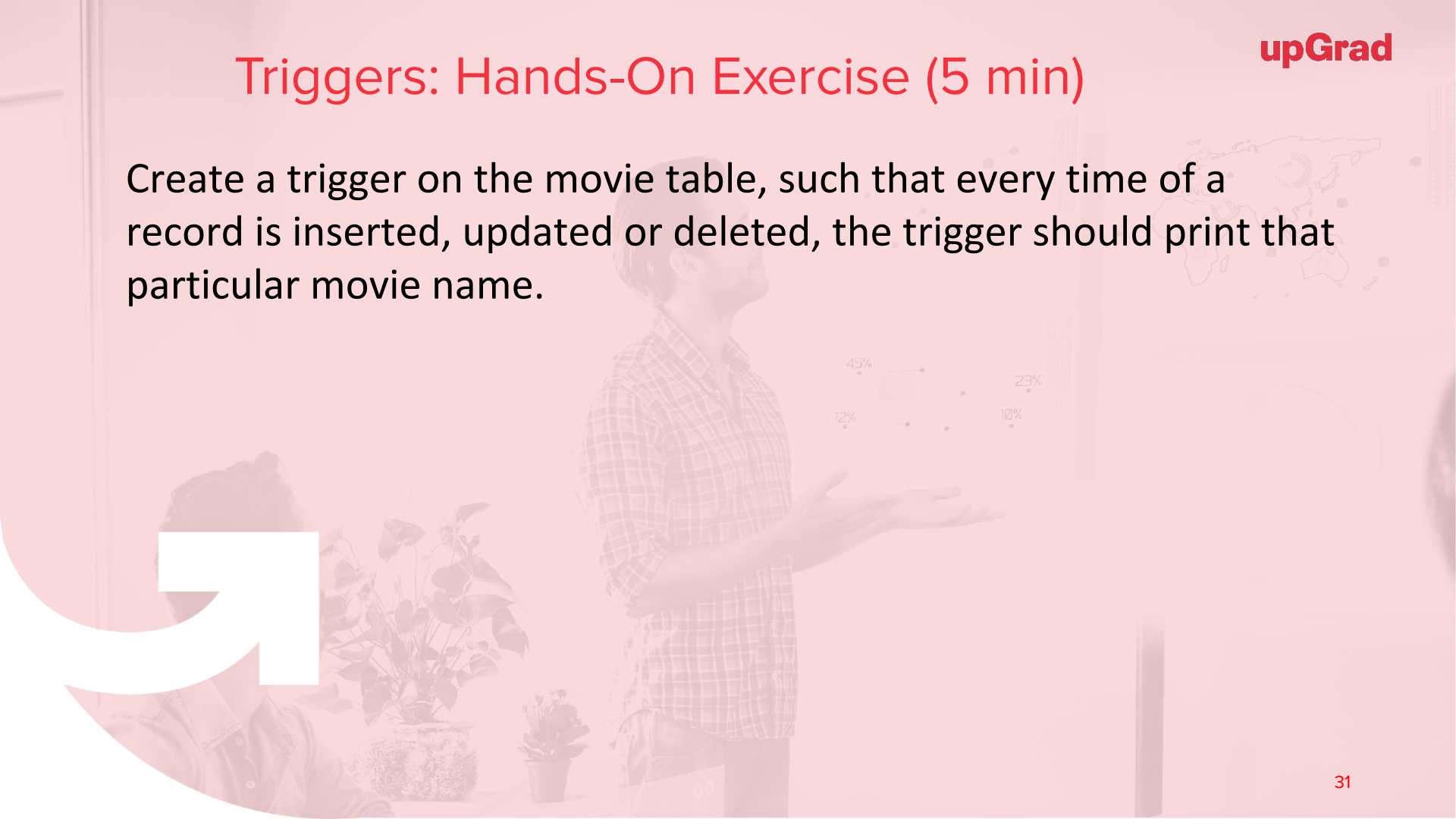
When the **WHEN** (condition) condition is satisfied by a row, the trigger will be executed.

Let's take a look at Example 6

Triggers: Hands-On Exercise (5 min)

upGrad

Create a trigger on the movie table, such that every time of a record is inserted, updated or deleted, the trigger should print that particular movie name.



Poll 4

A trigger is a reactive response to an event in an Oracle database.

1. True
2. False

Poll 4 (Answer)

A trigger is a reactive response to an event in an Oracle database.

1. **True**
2. False

Poll 5

In which of the following events are triggers executed?

1. On execution of DDL statements

2. On execution of DML statements

1. In database events such as STARTUP or SHUTDOWN

1. All of the above

Poll 5 (Answer)

In which of the following events are triggers executed?

1. On execution of DDL statements

2. On execution of DML statements

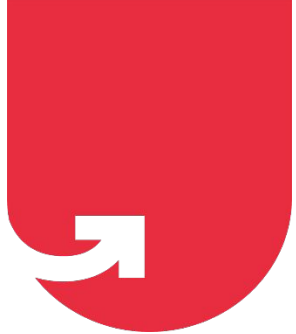
1. In database events such as STARTUP or SHUTDOWN

1. **All of the above**


DISCUSSION TIME

upGrad

#LifeKoKaroLift



Thank You!