Laboratory 5
SQL Triggers

Goal

The objective of this practice is to write some triggers in SQL to run on an Oracle database. To connect to a Database you can follow the guides related to the first lab practice. This lab can be done in Oracle SQL Developer or in Oracle APEX online without any difference. The triggers must be created using the functionality that allows the execution of SQL code.

Useful SQL statements

- Delete a trigger:

```
DROP TRIGGER triggerName;
DROP TRIGGER "triggerName";
```

- Update of an existing trigger (instead of delete and recreate it):

```
CREATE OR REPLACE TRIGGER triggerName …
```

- Display defined triggers:

```
SELECT trigger_name, triggering_event, table_name, status,
Description, action_type, trigger_body
FROM user_triggers
```

- Disable a trigger:

```
ALTER TRIGGER triggerName DISABLE;
```

- Display trigger errors:

```
SELECT * FROM USER_ERRORS;
```

Suggestions

Before doing each exercise, load the related tables by executing the scripts *script_db_es1.sql, script_db_es2.sql*.

To create a trigger, pay attention to the syntax and to the following issues:

- assign a proper name to the variables avoiding keywords like MIN, MAX, ...
- declare different variables on different lines and not on the same line delimited by a comma

```
MyVarOne NUMBER;
MyVarTwo NUMBER;
MyVarThree VARCHAR2(16)
```

- terminate the statements with ; character and assign new values to the variables with :=, e.g.

```
UPDATE tableName
SET varname=newvalue
WHERE column=:NEW.attribute;

IF A<3 OR A=3 THEN
        MyVar := 'Three';
ELSE
IF A>3 AND A<5 THEN
        MyVar := 'Four';
ELSE
        MyVar := 'Other';
END IF;
END IF;
```

Before starting this lab, we suggest to delete any existing trigger, which could affect the outcome of the exercises.

Remark. Pay attention because copying and pasting SQL code lines from the instruction text in PDF format into the Browser Web may generate errors due to invalid character encoding or conversion, such as *ora-00911: invalid character*.

1 Exercise 1

The following relations are given (primary keys are underlined; optional attributes are denoted with *):

- IMP(EMPNO, DEPTNO, ENAME, JOB, SAL)
- DIP(DEPTNO, DNAME, LOC, MINSAL, MAXSAL)

Write the trigger which manages the update of the DNAME attribute on the DIP table. When the DNAME attribute changes from 'ACCCOUNTING' to 'SALES', the wage (SAL attribute) for all employees, who work in the corresponding DEPTNO, is increased by 100.

Procedure:

- Create the database using script *create_db_es1.sql*.
- Create the trigger, eventually by means of a script.
- Verify the content of IMP and DIP tables.
- Modify the department name 'ACCOUNTING':

```
UPDATE DIP SET DNAME = 'SALES' WHERE DNAME = 'ACCOUNTING';
```

- Verify the content of the IMP and DIP tables.

2 Exercise 2

The following relations are given (primary keys are underlined; optional attributes are denoted with *):

- IMP(EMPNO, ENAME, JOB, SAL)
- SUMMARY(JOB, NUM)

In the SUMMARY table, the NUM attribute species the number of employees in the IMP table who perform the same job. Write the triggers to guarantee the consistency between IMP and SUMMARY tables when:

- A new record is inserted in the IMP table.
- The value of JOB in the IMP table is updated.

Create the database using script *create_db_es2.sql*.