ENSIA 2023-2024

Oracle LAB (5)

PLSQL

PL/SQL is a Procedural Language extension for the SQL Language. The features of PL/SQL are:

- Declaration of variables,
- Conditional processing,
- Repetitive processing,
- Cursor processing,
- Exceptions handling.

PL/SQL programs are organized and interpreted in blocks. A block is a set of commands, it is structured in three sections as follows:

```
-- PLSQL block
DECLARE
/* Declaration of variables, types, cursors, functions, and procedures */
BEGIN
/* PLSQL statements. Every PL/SQL statement should be followed by semicolon (;) */
EXCEPTION
/* Exception handling statements */
END;
- - End of PL/ SQL block
```

Note:

To handle errors, declare a variable of type **EXCEPTION** and use it in the EXCEPTION block.

Example:

Display the number and the names of employees of category “Assistant” by rank:

```
DECLARE
cursor cr is SELECT LASTNAME_EMP FROM EMPLOYEE WHERE CATEGORY = 'Assistant';  -- PL/SQL cursor definition
c_rec cr%rowtype;                                                             -- c_rec takes the same type as cr
i binary_integer;                                                             -- basically an integer
empty EXCEPTION;
BEGIN
    i := 1;
    for c_rec in cr loop                                                      -- put cr in c_rec
        dbms_output.put_line('Employee' ||i||' is '|| c_rec.LASTNAME_EMP);
        i := i+1;
        exit when cr%notfound;
    end loop;

    if (i < 2) then RAISE empty;
    else
        i := i-1;
        dbms_output.put_line('Assistant category contains '||i||' employee ');
    end if;
EXCEPTION WHEN empty THEN
    dbms_output.put_line('The category Assistant does not contain any employees');
END;
```

To display a text in the console, you can use the **DBMS_OUTPUT** package.

To make the display visible in SQL*Plus, use the following command: **SET SERVEROUTPUT ON**

Functions and procedures

PL/SQL code can be saved in a procedure or a function, with or without parameters.

```
CREATE [OR REPLACE] PROCEDURE Procedure_name (arg1 type, arg2 type, ...) IS
Declaration of local variables
BEGIN
Statements;
END;
```

The following command is used to execute a procedure:

```
EXECUTE Procedure_name (argument values);
```

The following command is used to see the syntactic errors raised when declaring a procedure:

```
show errors procedure Procedure_name.
```

Questions

Let's assume that the tables from the previous labs are created and filled.

1. Write a PLSQL code that displays the number of models for each brand.

   Example: The brand " **TOYOTA**" has **2 models**.

2. Add the following constraint: the salary of an employee must be between 10000DA and 30000DA. The maintenance center decides to increase the salary of employees of category Assistant by 30% and Mechanic by 50%.
   Write a procedure that increases each employee's salary. Disable the integrity constraint to perform the updates. Now display each employee with his new salary.
   Example: The salary of the employee **OUSSEDIK Hakim** of category **Mechanic** has switched from **20000DA** to **30000DA**

3. Write a verification procedure that displays "Positive verification" if the start date of the intervention is less than the end date of the intervention. Otherwise, it displays "Negative verification". Test the procedure for all interventions where the repaired vehicles are from the year 1998**.**

4. Write a function that returns, for each given employee, the number of interventions performed. Run the function for several employees.
   Example: The employee **IGOUDJIL Redouane** performed 3 interventions.

5. Create a procedure that allows to add an **Intervention** from all the necessary attributes. Do not forget to check the uniqueness of the key and the existence of a foreign key to the table **Vehicle**. Display proper error messages in case of problem.