Département Informatique et Statistique, ICOM, Université Lumière Lyon 2
M1 Informatique – Year 2015-2016
**Database programming – Labwork #4: Subprograms**
J. Darmont (*http://eric.univ-lyon2.fr/~jdarmont/*), 16/09/16

## Exercise #1: Functions and procedures

1. In an anonymous PL/SQL block, define a function named *MyMax* that inputs two real parameters *n1* and *n2* and returns a real that equals to the greatest number among *n1* and *n2*.

2. In the main program, declare two real variables *a* and *b* and initialize them with any value. Call function *MyMax* for *a* and *b* and display the result. Test!

3. In the same PL/SQL block, define a type TABLE of reals named *TabR*, and a variable *tab* of this type. Initialize *tab* with several values.

4. Define a new function named *MultiMax* that inputs a *TabR* collection and returns the greatest number in the collection. Use function *MyMax* in function *MultiMax*. In the main program, call function *MultiMax* for *tab* and display the result. Test!

5. Complement function *MultiMax* with an exception that is raised when the input collection is empty (fatal error). Test by initializing *tab* to empty.

6. Still in the same PL/SQL block, define a procedure named *PSort* that sorts by ascending order the contents of a *TabR* collection passed in parameter. Implement a simple permutation sort.

7. Write another procedure named *Display* that displays (astonishing, isn't it?) all elements of a *TabR* collection passed in parameter.

8. In the main program, call procedures *PSort* and *Display* for *tab*. What must the parameter mode be in each case? Test!

## Exercise #2: Stored procedure

**Memo: Debugging stored procedures**

If a stored procedure (or a package or package body) definition is incorrect, Oracle only indicates that it has been created "with compilation errors". To visualize these errors, use the following statement.

```
SHOW ERRORS
```

1. Write an <u>anonymous</u> PL/SQL block that displays the *n* first employees in table EMP (you can copy table DARMONT.EMP again if you dropped it). Number *n* can be stored in a variable. Deal with the case where *n* is greater than the number of rows in table EMP (then, display all employees). Test!

2. Transform the anonymous block in to a stored procedure named *empnames*, with variable `n` becoming an input parameter. Test by successively using the `EXECUTE empnames(3)` and `EXECUTE empnames(45)` statements, for instance.

3. Quit *SQL Developer*, launch it again and execute procedure *empnames* again. Conclusion?

4. Write an anonymous PL/SQL block that includes the declaration and initialization of two integer variables *n1* and *n2*, and calls procedure *empnames* with *n1* and *n2* in parameters, successively. Test!

## Exercise #3: Stored function

1. Transform function *MyMax* into a stored function (cut and paste the code).

2. Write an anonymous PL/SQL block that includes the declaration and initialization of two real variables *n1* and *n2*, and displays the result of function procedure *MyMax(n1, n2).* Test!

3. Is it possible to call *MyMax* directly through an `EXECUTE` statement out of a PL/SQL block? Find a way to do so.