EISTI Ecole Internationale des Sciences du Traitement de l'Information

M1 Adeo 2013-2014

Databases – PL/SQL

N. Nguyen – B. George – R. Chelouah

Exercises – Series 1 (Introduction to PL/SQL)

Introduction

Here, we'll take time to discover how PL/SQL language works. How to manipulate its syntax and its concepts. How to combine SQL's data manipulation power with PL/SQL's data processing power.

0 – The very beginning

Exercise 0.1
Write a program that says "goodbye".

Exercise 0.2
Write a program that assigns to variables *a* and *b* values 10 and 20, then assigns to variable *c* the sum of *a* and *b* before displaying it, and finally assigns to variable *f* the division between 70 and 30 before displaying the result.

Your code must produce the following result :
Value of c: 30
Value of f: 23.333333333333333333
PL/SQL procedure successfully completed.

Exercise 0.3
Here, we're going to see the difference between *local* variables and *global* variables. Write a program that assigns to variables *num1* and *num2* values 95 and 85, then displays them. Inside the program, write a sub-block that assigns to local variables *num1* and *num2* values 195 and 185, then displays them.

Your code must produce the following result :
Outer variable num1 : 95
Outer variable num2 : 85
Inner variable num1 : 195
Inner variable num2 : 185
PL/SQL procedure successfully completer

Exercise 0.4
Here, we're going to learn how to use reference types and the SELECT INTO clause.

First, create the following table :

```
CREATE TABLE CUSTOMERS(
        ID INT NOT NULL,
        NAME VARCHAR (20) NOT NULL,
        AGE INT NOT NULL,
        ADDRESS CHAR (25),
        SALARY
        DECIMAL (18, 2),
        PRIMARY KEY (ID)
);
```

Next, insert some values into the table :

```
INSERT INTO CUSTOMERS (ID,NAME,AGE,ADDRESS,SALARY)
VALUES (1, 'Ramesh', 32, 'Ahmedabad', 2000.00 );

INSERT INTO CUSTOMERS (ID,NAME,AGE,ADDRESS,SALARY)
VALUES (2, 'Khilan', 25, 'Delhi', 1500.00 );

INSERT INTO CUSTOMERS (ID,NAME,AGE,ADDRESS,SALARY)
VALUES (3, 'kaushik', 23, 'Kota', 2000.00 );

INSERT INTO CUSTOMERS (ID,NAME,AGE,ADDRESS,SALARY)
VALUES (4, 'Chaitali', 25, 'Mumbai', 6500.00 );

INSERT INTO CUSTOMERS (ID,NAME,AGE,ADDRESS,SALARY)
VALUES (5, 'Hardik', 27, 'Bhopal', 8500.00 );

INSERT INTO CUSTOMERS (ID,NAME,AGE,ADDRESS,SALARY)
VALUES (6, 'Komal', 22, 'MP', 4500.00 );
```

Then, write a program that assigns values from the first entry of the table to 4 PL/SQL variables before displaying them (try second entry next).

Your code must produce the following result :

```
Customer Ramesh from Ahmedabad earns 2000
PL/SQL procedure completed successfully
```

Exercise 0.5
Here, we're going to learn how to use the IF conditional. Write a program that assigns to variable *a* value 100 (you can try several values), then checks whether *a* is inferior or superior to 1. In the first case, it displays that a is less than 20. In the second case, it displays that a is not less than 20. Finally, it displays the value of a.

Your code must produce the following result :

```
a is not less than 20
value of a is : 100
PL/SQL procedure successfully completed.
```

Exercise 0.6
Here, we're going to learn how to use the CASE conditional. Write a program that assigns to variable *grade* a character value between 'A' and 'F' (try several values), then displays a message according to the grade. An 'A' will result in displaying an 'Excellent' message, while 'B' will result in a 'Very Good', etc...

With value 'A' assigned to variable *grade*, your code must produce the following result :
Excellent
PL/SQL procedure successfully completed.

Exercise 0.7
Here, we're going to learn how to use the simple LOOP. Write a program that assigns to variable *x* value 10, then adds 10 to *x* and displays the result until x becomes superior to 60. Try to stop the loop with an IF conditional, then a EXIT WHEN clause.

In any case, your code must produce the following result :
10
20
30
40
50
After Exit x is: 60
PL/SQL procedure successfully completed.

Exercise 0.8
Here, we're going to learn how to use the WHILE and FOR loops. Write a program that assigns to variable *a* value 10, then increments it and displays the result until a equals to 20. Try with a WHILE loop first, then with a FOR loop.

In any case, your code must produce the following result :
value of a : 10
value of a : 11
value of a : 12
value of a : 13
value of a : 14
value of a : 15
value of a : 16
value of a : 17
value of a : 18
value of a : 19
PL/SQL procedure successfully completed.

Exercise 0.9
Write a program that reverses the previous loop.

Exercise 0.10
Here, we're going to learn how to use Varrays. Write a program that declares two varrays, one of characters (the students' names), another of integers (the students' grades), then fills them with 5 values each, and finally displays both array entries together.

Example :
Student: Aziz Marks: 18

1 – Food for thought

Exercise 1
Write a program that assigns to variables *a* and *b* values 1 and 2 before exchanging them.

Exercise 2
Write a program putting value 10 in variable *a* before displaying its factorial.

Exercise 3
Write a program putting values 48 and 84 in two variables *a* and *b*, before calculating and displaying their GCD (Greatest Common Divisor).

2 – Arrays

Exercise 1
1/ Create an array that can contain until 50 integers.
2/ Resize the array to 20 integers.
3/ Put the 20 first square numbers into this array
4/ Revert the order of elements
5/ Display the final array

Exercise 2
Sort the previous array with an algorithm of your choice.

Exercise 3
Search the number 15 into the previous array. Do the same with number 225.