



School of Computer Science, UPES, Dehradun.

A

LABORATORY FILE

On

DATABASE MANAGEMENT
SYSTEM (DBMS) LAB

B.TECH. -III Semester

AUG. – NOV.- 2024.

Submitted by:

Name: Harsh Vardhan Saini

SAP ID: 500120369

Roll No: R2142230056

Batch: 2

Experiment – 14

To understand the concepts of function and procedure in PL/SQL.

Objective:

Implement the above experiments in exp. 13th in PL/SQL using functions and procedures.

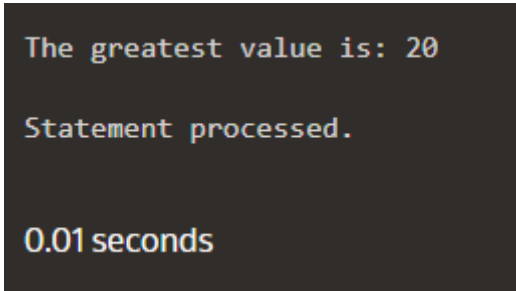
a) Write a PL/SQL code to accept the value of A, B & C display which is greater.

ii) Using Procedure

Code-

```
CREATE OR REPLACE PROCEDURE FindGreatest (A IN
NUMBER, B IN NUMBER, C IN NUMBER) IS
greatest_value NUMBER;
BEGIN
    IF A > B AND A > C THEN
        greatest_value := A;
    ELSIF B > C THEN
        greatest_value := B;
    ELSE
        greatest_value := C;
    END IF;
    DBMS_OUTPUT.PUT_LINE('The greatest value is: ' ||
greatest_value);
END;
BEGIN
    FindGreatest(10, 20, 15);
END;
```

Output-



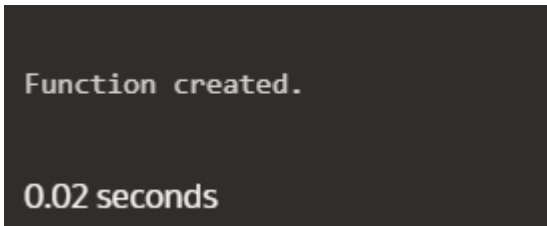
```
The greatest value is: 20
Statement processed.
0.01 seconds
```

ii) Using Function

Code-

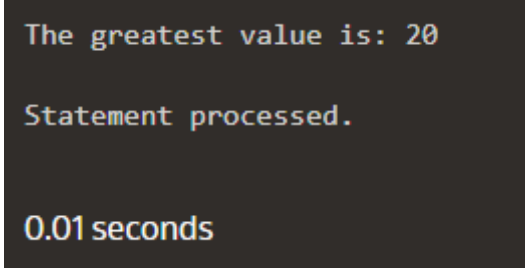
```
CREATE OR REPLACE FUNCTION GreatestValue (A IN
NUMBER, B IN NUMBER, C IN NUMBER) RETURN
NUMBER IS greatest_value NUMBER;
BEGIN
    IF A > B AND A > C THEN
        greatest_value := A;
    ELSIF B > C THEN
        greatest_value := B;
    ELSE
        greatest_value := C;
    END IF;
    RETURN greatest_value;
END;
DECLARE
    result NUMBER;
BEGIN
    result := GreatestValue(10, 20, 15);
    DBMS_OUTPUT.PUT_LINE('The greatest value is: ' || result);
END;
```

Output-



Function created.

0.02 seconds



The greatest value is: 20

Statement processed.

0.01 seconds

b) Using PL/SQL Statements create a simple loop that display message “Welcome to PL/SQL Programming” 20 times.

i) Using Procedure

Code-

```
CREATE OR REPLACE PROCEDURE DisplayMessage IS
    counter NUMBER := 1;
BEGIN
    WHILE counter <= 20 LOOP
```

```

        DBMS_OUTPUT.PUT_LINE('Welcome to PL/SQL
Programming');
        counter := counter + 1;
    END LOOP;
END;
BEGIN
    DisplayMessage;
END;
```

Procedure created.

0.02 seconds

Output-

[illegible]

ii) Using Function

Code-

```
CREATE OR REPLACE FUNCTION DisplayMessage20Times
RETURN VARCHAR2 IS
    result VARCHAR2(4000) := "";
    counter NUMBER := 1;
BEGIN
```

```
WHILE counter <= 20 LOOP
    result := result || 'Welcome to PL/SQL Programming' ||
CHR(10);
    counter := counter + 1;
END LOOP;
RETURN result;
END;
DECLARE
    message_output VARCHAR2(4000);
BEGIN
    message_output := DisplayMessage20Times;
    DBMS_OUTPUT.PUT_LINE(message_output);
END;
```

Function created.

0.03 seconds

Output-

[illegible]

- c) Write a PL/SQL code block to find the factorial of a number.
- i) Using procedure

Code-

```
CREATE OR REPLACE PROCEDURE CalculateFactorial (num
IN NUMBER) IS
    factorial NUMBER := 1;
    counter NUMBER := 1;
BEGIN
    WHILE counter <= num LOOP
        factorial := factorial * counter;
        counter := counter + 1;
    END LOOP;
END;
```

```

END LOOP;
DBMS_OUTPUT.PUT_LINE('Factorial of ' || num || ' is: ' ||
factorial);
END;
BEGIN
  CalculateFactorial(5);
END;

```

Procedure created.

0.04 seconds

Factorial of 5 is: 120

Statement processed.

0.01 seconds

Output-

ii) Using function

Code-

```

CREATE OR REPLACE FUNCTION Factorial (num IN
NUMBER) RETURN NUMBER IS
  factorial NUMBER := 1;
  counter NUMBER := 1;
BEGIN
  WHILE counter <= num LOOP
    factorial := factorial * counter;
    counter := counter + 1;
  END LOOP;
  RETURN factorial;
END;
DECLARE
  result NUMBER;
BEGIN
  result := Factorial(5);
  DBMS_OUTPUT.PUT_LINE('Factorial is: ' || result);
END;

```

Function created.

0.03 seconds

Factorial is: 120

Statement processed.

0.02 seconds

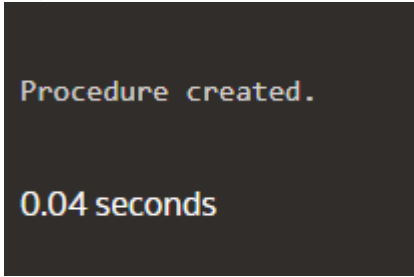
Output-

d) Write a PL/SQL program to generate Fibonacci series.

i) Using procedure

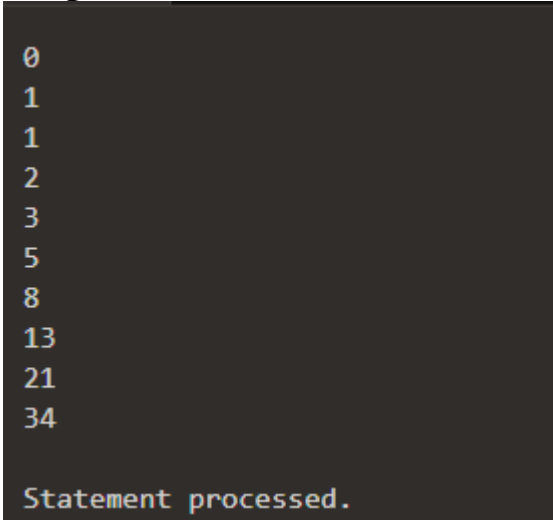
Code-

```
CREATE OR REPLACE PROCEDURE GenerateFibonacci (n IN
NUMBER) IS
    first NUMBER := 0;
    second NUMBER := 1;
    next NUMBER;
    counter NUMBER := 3;
BEGIN
    DBMS_OUTPUT.PUT_LINE(first);
    DBMS_OUTPUT.PUT_LINE(second);
    WHILE counter <= n LOOP
        next := first + second;
        DBMS_OUTPUT.PUT_LINE(next);
        first := second;
        second := next;
        counter := counter + 1;
    END LOOP;
END;
BEGIN
    GenerateFibonacci(10);
END;
```

A screenshot of a SQL*Plus terminal window. It shows the message 'Procedure created.' in a green font on a black background.

0.04 seconds

Output-

A screenshot of a SQL*Plus terminal window. It shows the first 10 Fibonacci numbers (0, 1, 1, 2, 3, 5, 8, 13, 21, 34) listed vertically in a green font on a black background. Below the numbers, it says 'Statement processed.' in a green font.

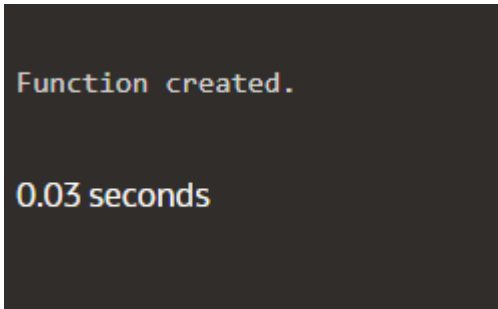
0
1
1
2
3
5
8
13
21
34

Statement processed.

ii) Using Function

Code-

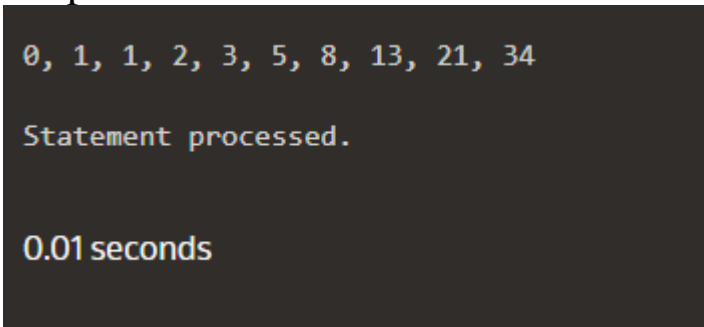
```
CREATE OR REPLACE FUNCTION GenerateFibonacciSeries(n
IN NUMBER) RETURN VARCHAR2 IS
    first NUMBER := 0;
    second NUMBER := 1;
    next NUMBER;
    counter NUMBER := 3;
    result VARCHAR2(4000) := '';
BEGIN
    result := result || first || ', ' || second;
    WHILE counter <= n LOOP
        next := first + second;
        result := result || ', ' || next;
        first := second;
        second := next;
        counter := counter + 1;
    END LOOP;
    RETURN result;
END;
DECLARE
    fibonacci_output VARCHAR2(4000);
BEGIN
    fibonacci_output := GenerateFibonacciSeries(10); -- Replace 10
with the desired number of terms
    DBMS_OUTPUT.PUT_LINE(fibonacci_output);
END;
```



Function created.

0.03 seconds

Output-



0, 1, 1, 2, 3, 5, 8, 13, 21, 34

Statement processed.

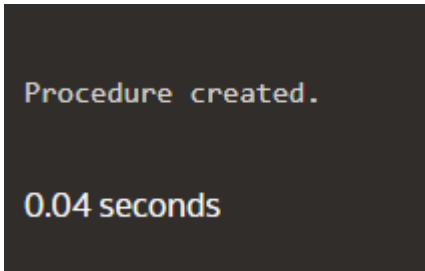
0.01 seconds

e) Write a PL/SQL code to find the sum of first N numbers

i) Using Procedure

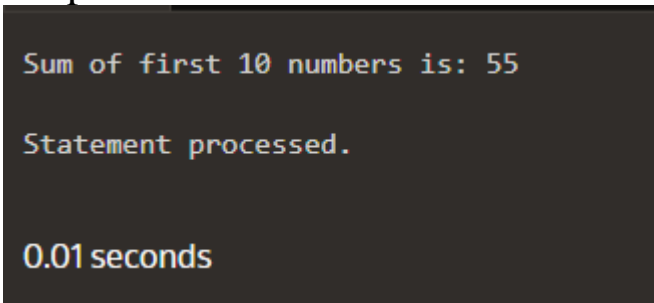
Code-

```
CREATE OR REPLACE PROCEDURE SumFirstN (n IN
NUMBER) IS
    sum_result NUMBER := 0;
    counter NUMBER := 1;
BEGIN
    WHILE counter <= n LOOP
        sum_result := sum_result + counter;
        counter := counter + 1;
    END LOOP;
    DBMS_OUTPUT.PUT_LINE('Sum of first ' || n || ' numbers is: '
|| sum_result);
END;
BEGIN
    SumFirstN(10);
END;
```

Procedure created.

0.04 seconds

Output-

Sum of first 10 numbers is: 55

Statement processed.

0.01 seconds

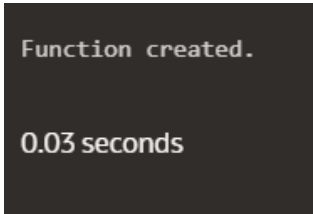
ii) Using Function

Code-

```
CREATE OR REPLACE FUNCTION SumOfN (n IN NUMBER)
RETURN NUMBER IS
    sum_result NUMBER := 0;
    counter NUMBER := 1;
BEGIN
    WHILE counter <= n LOOP
        sum_result := sum_result + counter;
        counter := counter + 1;
    END LOOP;
    RETURN sum_result;
END;
```

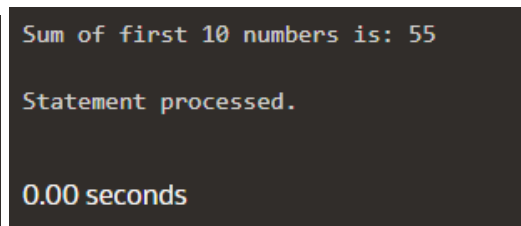
```
END;  
DECLARE  
    result NUMBER;  
BEGIN  
    result := SumOfN(10);  
    DBMS_OUTPUT.PUT_LINE('Sum of first 10 numbers is: ' ||  
result);  
END;
```

Output-

A screenshot of a terminal window with a dark background. It displays the text "Function created." in a light-colored monospace font. Below it, "0.03 seconds" is shown, indicating the execution time for the previous command.

Function created.

0.03 seconds

A screenshot of a terminal window with a dark background. It displays the text "Sum of first 10 numbers is: 55" in a light-colored monospace font. Below it, "Statement processed." is shown, and at the bottom, "0.00 seconds" is displayed.

Sum of first 10 numbers is: 55

Statement processed.

0.00 seconds