Assignment No .6

Title of Assignment: Named PL/SQL Block: PL/SQL Stored Procedure and Stored Function.

Write a Stored Procedure namely proc_Grade for the categorization of student. If marks scored by students in examination is <=1500 and marks>=990 then student will be placed in distinction category if marks scored are between 989 and900 category is first class, if marks899and 825 category is Higher Second Class.

Write a PL/SQLblock to use procedure created with above requirement.

Stud_Marks(name, total_marks) Result(Roll,Name, Class)

Course Objective:

Implement PL/SQL Code block for given requirements

Course Outcome:

C306.4 Implement PL/SQL Code block for given requirements

Software Required: - Mysql

Theory: -

Stored Procedures : A stored procedure or in simple a proc is a named PL/SQL block which performs one or more specific task. This is similar to a procedure in other programming languages. A procedure has a header anda body. The header consists of the name of the procedure and the parameters or variables passed to the procedure. The body consists or declaration section, execution section and exception section similar to a general PL/SQL Block. A procedure is similar to an anonymous PL/SQL Block but it is named for repeated usage.

Procedures: Passing Parameters

We can pass parameters to procedures in three ways.

1) IN-parameters
2) OUT-parameters
3) IN OUT-parameters

A procedure may or may not return any

value.General Syntax to create a procedure

is:

CREATE [OR REPLACE] PROCEDURE proc_name [list of

parameters]IS

Declaration

sectionBEGIN

Execution

section

EXCEPTION

Exception

sectionEND;

IS - marks the beginning of the body of the procedure and is similar to DECLARE in anonymous PL/SQL Blocks. The code between IS and BEGIN forms the Declaration section.

The syntax within the brackets [ ] indicate they are optional. By using CREATE OR REPLACE together the procedure is created if no other procedure with the same name exists or the existing procedure is replaced with the current code.

Procedures: Example

The below example creates a procedure 'employer_details' which gives the details of the employee.

```
1> CREATE OR REPLACE PROCEDURE employer_details
2> IS
3> CURSOR emp_cur IS
4> SELECT first_name, last_name, salary FROM
emp_tbl;5> emp_rec emp_cur%rowtype;
6> BEGIN
7> FOR emp_rec in
sales_cur8> LOOP
9> dbms_output.put_line(emp_cur.first_name || ' '
||emp_cur.last_name10>    || ' ' ||emp_cur.salary);
11> END LOOP;
12>END;
13>
```

There are two ways to execute a procedure.

1) From the SQL prompt.

EXECUTE [or EXEC] procedure_name;

2) Within another procedure – simply use the procedure name.

procedure_name;

PL/SQL Functions

A function is a named PL/SQL Block which is similar to a procedure. The major difference between a procedureand a function is, a function must always return a value, but a procedure may or may not return a value.

General Syntax to create a function is

```
CREATE [OR REPLACE] FUNCTION function_name [parameters]
RETURN return_datatype;
IS
Declaration_section

BEGIN
Execution_section
```

Return

return_variable;

EXCEPTION
exception section
Return
return_variable;
END;

1) Return Type: The header section defines the return type of the function. The return datatype can be anyof the oracle datatype like varchar, number etc.

2) The execution and exception section both should return a value which is of the datatype defined in theheader section.

For example, let's create a frunction called "employer_details_func' similar to the one created in stored

proc1> CREATE OR REPLACE FUNCTION employer_details_func
2> RETURN VARCHAR(20);
3> IS
5> emp_name VARCHAR(20);
6> BEGIN
7> SELECT first_name INTO emp_name
8> FROM emp_tbl WHERE empID =
'100';9> RETURN emp_name;
10> END;

In the example we are retrieving the 'first_name' of employee with empID 100 to variable 'emp_name'.

The return type of the function is VARCHAR which is declared in line no 2.

The function returns the 'emp_name' which is of type VARCHAR as the return value in line no

9.A function can be executed in the following ways.

1) Since a function returns a value we can assign it to a variable.employee_name := employer_details_func;

If 'employee_name' is of datatype varchar we can store the name of the employee by assigning the return type ofthe function to it.

2) As a part of a SELECT statement

SELECT employer_details_func FROM dual;

3) In a PL/SQL Statements like,

dbms_output.put_line(employer_details_func);

This line displays the value returned by the function.

Conclusion:

Students are able to PL/SQL Stored Procedure and Stored Function.