ASSIGNMENT NO.07

TITLE: PL/SQL Stored Procedure and Stored Function

PROBLEM STATEMENT: 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 marks 899 and 825 category is Higher Second Class.
Write a PL/SQL block for using procedure created with above requirement. Stud_Marks(name, total_marks) Result (Roll,Name, Class)

OBJECTIVES:

- To use named PLSQL blocks and call functions & procedures
- To create stored procedures and functions, pass parameters and return variables

OUTCOMES:

- Students will be able to create named PLSQL blocks & call stored procedures and functions
- Students will be able understand how parameters are passed and values are returned

SOFTWARE & HARDWARE REQUIREMENTS:

1. 64-bit Open source Linux or its derivative
2. Oracle Database Server

THEORY:

PL/SQL stands for Procedural Language extension of SQL. PL/SQL is a combination of SQL along with the procedural features of programming languages. It was developed by Oracle Corporation in the early 90's to enhance the capabilities of SQL.

A Simple PL/SQL Block:

Each PL/SQL program consists of SQL and PL/SQL statements which from a PL/SQL block. PL/SQL Block consists of three sections:

- The Declaration section (optional).
- The Execution section (mandatory).
- The Exception Handling (or Error) section (optional).

Declaration Section: The Declaration section of a PL/SQL Block starts with the reserved keyword DECLARE. This section is optional and is used to declare any placeholders like variables, constants, records and cursors, which are used to manipulate data in the execution section. Placeholders may be any of Variables, Constants and Records, which stores data temporarily. Cursors are also declared in this section.

Execution Section: The Execution section of a PL/SQL Block starts with the reserved keyword BEGIN and ends with END. This is a mandatory section and is the section where the program logic is written to perform any task. The programmatic constructs like loops, conditional statement and SQL statements form the part of execution section.

Exception Section: The Exception section of a PL/SQL Block starts with the reserved keyword EXCEPTION. This section is optional. Any errors in the program can be handled in this section, so that the PL/SQL Blocks terminates gracefully. If the PL/SQL Block contains exceptions that cannot be handled, the Block terminates abruptly with errors.

Every statement in the above three sections must end with a semicolon ; . PL/SQL blocks can be nested within other PL/SQL blocks. Comments can be used to document code.

Advantages of PL/SQL

- **Block Structures:** PL SQL consists of blocks of code, which can be nested within each other. Each block forms a unit of a task or a logical module. PL/SQL Blocks can be stored in the database and reused.
- **Procedural Language Capability**: PL SQL consists of procedural language constructs such as conditional statements (if else statements) and loops like (FOR loops).
- **Better Performance:** PL SQL engine processes multiple SQL statements simultaneously as a single block, thereby reducing network traffic.
- **Error Handling:** PL/SQL handles errors or exceptions effectively during the execution of a PL/SQL program. Once an exception is caught, specific actions can be taken depending upon the type of the exception or it can be displayed to the user with a message.

Procedures and Functions are the subprograms which can be created and saved in the database as database objects. They can be called or referred inside the other blocks also.
A function is same as a procedure except that it returns a value. A PL/SQL function is also known as a subroutine or a subprogram.
When you create a function or procedure, you have to define IN/OUT/INOUT parameters.

An **IN parameter** is a **read-only** parameter. If the function or procedure tries to change the value of the IN parameters, the compiler will issue an error message. It indicates that the parameter will accept a value from the user.
An **OUT parameter** is a **write-only** parameter. The OUT parameters are used to return values back to the calling program. It indicates that the parameter will return a value to the user.
An **IN OUT parameter** is **read and write** parameter. It means the function or procedure reads the value from an IN OUT parameter, change its value and return it back to the calling

program. It indicates that the parameter will either accept a value from the user or return a value to the user.

STORED PROCEDURE

A procedure (often called a stored procedure) is a subroutine like a subprogram in a regular computing language, stored in database. A procedure has a name, a parameter list, and SQL statement(s). All most all relational database system supports stored procedure.

Syntax for creating Procedure

```
CREATE [OR REPLACE] PROCEDURE proc_name [list of parameters]
IS
  Declaration section
BEGIN
  Execution section
EXCEPTION
  Exception section
END;
```

Syntax for Executing Stored Procedure

```
EXEC proc_name;
```

Example:

```
CREATE OR REPLACE PROCEDURE adjust_salary(
   in_employee_id IN EMPLOYEES.EMP_ID%TYPE,
   in_percent IN NUMBER
) IS
BEGIN
  -- update employee's salary
  UPDATE employee
  SET salary = salary + salary * in_percent / 100
  WHERE emp_id = in_employee_id;
END;
```

TO EXECUTE THE PROCEDURE

```
EXEC ADJUST_SALARY(105,10)
```

PLSQL FUNCTIONS

A function is same as a procedure except that it returns a value. A PL/SQL function is also known as a subroutine or a subprogram.

PL/SQL Function Syntax

```
CREATE [OR REPLACE] FUNCTION function_name
        [ (parameter [,parameter]) ]
```

```
        RETURN return_datatype
IS | AS   [declaration_section]
        BEGIN
                executable_section
        [EXCEPTION
                        exception_section]
END [function_name];
```

CREATE FUNCTION instructs the compiler to create new procedure. Keyword 'OR REPLACE' instructs the compile to replace the existing procedure (if any) with the current one. Procedure name should be unique.

Keyword 'IS' will be used, when the procedure is nested into some other blocks. If the procedure is standalone then 'AS' will be used. Other than this coding standard, both have the same meaning.

PLSQL function can be called through another unnamed PLSQL block.

Example:

```
CREATE OR REPLACE FUNCTION totalCustomers
RETURN number IS
  total number(2) := 0;
BEGIN
  SELECT count(*) into total FROM customers;
  dbms_output.put_line('TOTAL CUSTOMERS ARE: ' ||total);
 END;
```

CALLING FUNCTION

```
DECLARE
  c number(2);
BEGIN
  c := TOTALCUSTOMERS();
  dbms_output.put_line('Total no. of Customers: ' || c);
END;
```

Dropping Function

Once you have created your function in Oracle, you might find that you need to remove it from the database.

The syntax to a drop a function in Oracle is:

```
DROP FUNCTION function_name;
```

ADVANTAGES OF USING PROCEDURE OR FUNCTION

Security: Enforce the data security by giving permissions to procedure or function.

Performance: Amount of information sent over the network is less. No compilation step is required to execute the code.

Memory Allocation: Only one copy of procedure needs to be loaded for execution by multiple users. Once a copy of a procedure or function is opened in Oracle engine's memory, other users may access it when required.

Productivity: By writing procedures and functions redundant coding can be avoided increasing the productivity.

Integrity: A procedure and function needs to be tested once to guarantee that it returns an accurate result.

CONCLUSION: