**Assignment No: 5**

**Title**:

PL/SQL Stored Procedure and Stored Function.

**Problem Statement:**

Create following tables.

* Stud\_Marks(name, total\_marks)
* Result(Roll,Name, Class)
* 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 otherwise fail

Insert data in both the tables and retrieve data from tables by calling above procedure

* Write a function which will returns total student in given class.

**Objective:**

* To understand the concept stored Procedure and function.
* To study syntax of creating and calling stored procedure.
* To create Function and different ways of calling it

**Theory**

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.

**Terminologies in PL/SQL Subprograms**

Before we learn about PL/SQL subprograms, we will discuss the various terminologies that are the part of these subprograms. Below are the terminologies that we are going to discuss.

**Parameter:**

The parameter is variable or placeholder of any valid PL/SQL datatype through which the PL/SQL subprogram exchange the values with the main code. This parameter allows to give input to the subprograms and to extract from these subprograms.

* These parameters should be defined along with the subprograms at the time of creation.
* These parameters are included in the calling statement of these subprograms to interact the values with the subprograms.
* The datatype of the parameter in the subprogram and in the calling statement should be same.
* The size of the datatype should not mention at the time of parameter declaration, as the size is dynamic for this type.

Based on their purpose parameters are classified as

**IN Parameter:**

* This parameter is used for giving input to the subprograms.
* It is a read-only variable inside the sub programs, their values cannot be changed inside the subprogram.
* In the calling statement these parameters can be a variable or a literal value or an expression, for example, it could be the arithmetic expression like '5\*8' or 'a/b' where 'a' and 'b' are variables.
* By default, the parameters are of IN type.

**OUT Parameter:**

* This parameter is used for getting output from the subprograms.
* It is a read-write variable inside the subprograms, their values can be changed inside the subprograms.
* In the calling statement, these parameters should always be a variable to hold the value from the current subprograms.

**IN OUT Parameter:**

* This parameter is used for both giving input and for getting output from the subprograms.
* It is a read-write variable inside the subprograms, their values can be changed inside the subprograms.
* In the calling statement, these parameters should always be a variable to hold the value from the subprograms.

These parameter types should be mentioned at the time of creating the subprograms.

**RETURN**

RETURN is the keyword that actually instructs the compiler to switch the control from the subprogram to the calling statement. In subprogram RETURN simply means that the control needs to exit from the subprogram. Once the controller finds RETURN keyword in the subprogram, the code after this will be skipped.

Normally, parent or main block will call the subprograms, and then the control will shift from those parent blocks to the called subprograms. RETURN in the subprogram will return the control back to their parent block. In the case of functions RETURN statement also returns the value. The datatype of this value is always mentioned at the time of function declaration. The datatype can be of any valid PL/SQL data type.

**Procedure**

Procedure is a subprogram unit that consists of a group of PL/SQL statements. Each procedure in Oracle has its own unique name by which it can be referred. This subprogram unit is stored as a database object. Below are the characteristics of this subprogram unit.

**Note:** Subprogram is nothing but a procedure, and it needs to be created manually as per the requirement. Once created they will be stored as database objects.

* Procedures are standalone blocks of a program that can be stored in the database.
* Call to these procedures can be made by referring to their name, to execute the PL/SQL statements.
* It is mainly used to execute a process in PL/SQL.
* It can have nested blocks, or it can be defined and nested inside the other blocks or packages.
* It contains declaration part (optional), execution part, exception handling part (optional).
* The values can be passed into the procedure or fetched from the procedure through parameters.
* These parameters should be included in the calling statement.
* Procedure can have a RETURN statement to return the control to the calling block, but it cannot return any values through the RETURN statement.
* Procedures cannot be called directly from SELECT statements; they can be called from another block or through EXEC keyword.

**Syntax:**

DELIMITER //

CREATE PROCEDURE Procedure\_Name (IN|OUT|INOUT variable name data type )

BEGIN

Statements 1;

Statements 2:

………..

END //

DELIMITER ;

call procedure\_name();

**Function**

Functions is a standalone PL/SQL subprogram. Like PL/SQL procedure, functions has a unique name by which it can be referred. These are stored as PL/SQL database objects. Below are some of the characteristics of functions.

* Functions are a standalone block that is mainly used for calculation purpose.
* Function use RETURN keyword to return the value, and the datatype of this is defined at the time of creation.
* Function should either return a value or raise the exception, i.e. return is mandatory in functions.
* Function with no DML statements can be directly called in SELECT query whereas the function with DML operation can only be called from other PL/SQL blocks.
* It can have nested blocks, or it can be defined and nested inside the other blocks or packages.
* It contains declaration part (optional), execution part, exception handling part (optional).
* The values can be passed into the function or fetched from the procedure through the parameters.
* These parameters should be included in the calling statement.
* Function can also return the value through OUT parameters other than using RETURN.
* Since it will always return the value, in calling statement it always accompany with assignment operator to populate the variables.

CREATE FUNCTION function\_name(param1,param2,…)

    RETURNS datatype

BEGIN

Statements

END

**Syntax Explanation:**

* CREATE FUNCTION instructs the compiler to create a new function.
* Function name should be unique.
* RETURN datatype should be mentioned.
* 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.

**Similarities between Procedure and Function**

* Both can be called from other PL/SQL blocks.
* If the exception raised in the subprogram is not handled in the subprogram exception handling section, then it will propagate to the calling block.
* Both can have as many parameters as required.
* Both are treated as database objects in PL/SQL.

**Difference between Procedure and Function**

|  |  |
| --- | --- |
| **Procedure** | **Function** |
| * Used mainly to execute certain process | * Used mainly to perform some calculation |
| * Cannot called in SELECT statement | * Function that contain no DML statements can be called in SELECT statement |
| * Use OUT parameter to return the value | * Use RETURN to return the value |
| * It is not mandatory to return the value | * It is mandatory to return the value |
| * RETURN will simply exit the control from subprogram. | * RETURN will exit the control from subprogram and also returns the value |
| * Return datatype will not be specified at the time of creation | * Return datatype is mandatory at the time of creation |

**Conclusion**

In this assignment we have studied how to create PL/SQL Functions and Procedures

and implemented the same to solve given problem statement.