Page No.
Date

Assignment - 07

* Date of Completion : 28/09/2020

* Date of Submission : 12/10/2020

* Title : PL/SQL stored Procedure and stored Function

* Problem Statement :-
Write a stored procedure namely proc_grade for the categorization of customer. If purchase by customer in year is <= 20000 and >= 100000 then customer will be placed in platinum category. If purchase by customer is between 9999 and 5000 category is gold, if purchase between 4999 and 2000 category is silver. Write a PL/SQL block for using procedure created with above requirement.
Customer (cust_id, name, total_purchase)
Category (Cust_id, Name, class)

* Learning Objective:
- To understand the PL/SQL stored procedure.
- To understand the PL/SQL stored function.

* Learning Outcome:
Students will be able to:
- write PL/SQL stored procedure
- write PL/SQL stored function

for the categorization of customers.

Teacher's Sign.: _____

Page No.
Date

* S/W and H/W Requirements:
  MySQL, Windows 10 (64 bit), i5 processor, PL/SQL

* Theory:

* MySQL

MySQL supports two kinds of routines:

(1) Stored procedures:
A procedure is a subroutine in a regular scripting language, stored in a database.

- create a stored procedure

```
create procedure proc_name ( parameters )
begin
    --- declaration statements
    --- execution statements
end;
```

- calling a procedure
  Syntax:

```
call proc_name (parameters);
```

- dropping a procedure
  Syntax: drop procedure proc_name;

Teacher's Sign.: _____

- Parameter modes:

IN - you must pass the value when you call the procedure. Its original value is retained after the stored procedure has been executed

OUT - the value of an out parameters can change within stored procedure and its value is returned to calling application.

INOUT - you can pass the initial value, the stored procedure can change it and it will change & return the new value to the calling application.

(2) Stored function:
A stored function is a set of SQL statements that perform some operation and return a single value

- creating a stored function:
Syntax:

```
create function function_name (parameters)
return datatype [characteristics]
[NOT] DETERMINISTIC
begin
    MySQL statements
    RETURN expression;
end;
```

Teacher's Sign.: _____

- calling a stored function:
  - a stored function can be called inside a procedure or select statement

- Deterministic function:
  - they always return the same result any time they are called with a specific set of input values

- Non deterministic function
  - returns different result each time

- PL/SQL:

i) Stored procedure:
A stored procedure is a named PL/SQL block which performs a specific task.

Passing parameters
- IN parameters
- OUT parameters
- INOUT parameters
- Creating a stored procedure:

Syntax:

```
create [or replace] procedure proc_name [parameter
is
declare
    declaration section.
```

Teacher's Sign.:

```
begin
    execution section
exception
    exception section
end;
```

- How to execute a stored procedure?

(1) from the SQL prompt

Syntax: execute [or exec] procedure_name (parameters);

(2) within another procedure - simply use the procedure name

Syntax: procedure_name;

iii) Stored function:

A function is a named PL/SQL block which is similar to a procedure. The major difference is a function must always return a value.

- Creating a stored function:

Syntax:

```
create [or replace] function funcname [parameter]
return data_type;
is
declaration section
begin
    execution section
RETURN return variable;
exception
    ---
return return_variable
end;
```

Teacher's Sign.: _____

i) the return type can be any oracle data type
ii) the execution & exception value section both should return a value which is of datatype defined in the header section.

- Stored function for given problem statement

```
create function getclass ( purchase int)
return varchar (20)
is
declare class varchar (20) ;
begin
    if (purchase <= 20000 and purchase >= 10000) then
        set class = 'platinum';
    elsif ( purchase <= 9999 and procedure >= 5000) then
        set class = 'gold';
    elsif (purchase <= 4999 and purchase >= 2000) then
        set class = 'silver';
    end if;
    return (class);
end;
```

- stored procedure without parameters

```
create procedure proc_1()
is
declare
id int ;
name1 varchar(20);
```

Teacher's Sign.: _____

```
purchase1 int;
class varchar(20);
cursor cur1 is select cust_id, name, total_purchase
    from customer;
begin
  open cur_1;
  loop
    fetch cur1 into id, name1, purchase1;
      exit when cur1%. not_found;
    set class = getclass(purchase1);
    insert into category values( id, name1, class);
  end loop;
end;
```

- stored procedure with parameters

```
create procedure update_category (in id, int,
  in purchase int, out class1 varchar(20))
is
begin
  update customer set total_procedure = purchase
      where cust_id = id;
  update category set class = getclass(purchase)
      where cust_id = id;
  select class into class1 from category where
      cust_id = id;
end;
```

Teacher's Sign.: _____

Page No.
Date

* CONCLUSION:

Thus, we implemented

i) stored procedure without parameters.
ii) stored procedure with parameters
iii) stored functions.

Teacher's Sign.: _____