Roll No. :- COMPTEB1449

Name :- Paarth Kothari

PRN :- 72018337F

Assignment No. 5

Title :- Write PL/SQL block to implement Stored Procedure and Stored Function for categorization of student.

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)
Frame the separate problem statement for writing PL/SQL Stored Procedure and function, inline with above statement. The problem statement should clearly state the requirements.

Theory :-

PL/SQL (Procedural Language/Structured Query Language) :-

It is Oracle Corporation's proprietary procedural extension to the SQL database language, used in the Oracle database. Some other SQL database management systems offer similar extensions to the SQL language. PL/SQL's syntax strongly resembles that of Ada, and just like some Ada compilers of the 1980s, the PL/SQL runtime system uses Diana as intermediate representation. The key strength of PL/SQL is its tight integration with the Oracle database**.**

Basic code structure in PL/SQL :-

```
DECLARE
TYPE / item / FUNCTION / PROCEDURE declarations
BEGIN
Statements
EXCEPTION
EXCEPTION handlers
END;
```

The DECLARE and EXCEPTION sections are optional.

Simple Example:

```
DECLARE
number1 int;
number2 int:= 17; -- value default
text1 VARCHAR(12) := 'Hello world';
BEGIN
SELECT street_number
INTO number1
FROM address
WHERE name = 'Sahil';
END;
```

FUNCTIONS :-

Functions in PL/SQL are a collection of SQL and PL/SQL statements that perform a task and should return a value to the calling environment.

SYNTAX:

```
CREATE FUNCTION <function_name> [(input/output variable declarations)]
RETURN
return_type
<IS|AS>
BEGIN
[declaration block]
<PL/SQL block WITH RETURN statement>
[EXCEPTION
EXCEPTION block]
END;
```

PROCEDURES

Procedures are the same as Functions, in that they are also used to perform some task with the difference being that procedures cannot be used in a SQL statement and although they can have multiple out parameters they do not return a value.

SYNTAX:

```
CREATE PROCEDURE <procedure_name> [(input/output variable declarations)]
BEGIN
[declaration block]
<PL/SQL block statements>
[EXCEPTION
EXCEPTION block]
```

END;

Outcome :- Performed Stored Procedures and Functions in PL/SQL.

Code and Output :-

```
mysql> use Abhi;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed

mysql> create table marks(roll_no int,name varchar(20),total_marks
varchar(20));

Query OK, 0 rows affected (0.67 sec)

mysql> create table result(roll_no int,name varchar(20),class
varchar(20));

Query OK, 0 rows affected (0.41 sec)

mysql> insert into marks values('1','Abhi','1400');

Query OK, 1 row affected (0.04 sec)

mysql> insert into marks values('2','piyush','980');

Query OK, 1 row affected (0.08 sec)

mysql> insert into marks values('3','hitesh','880');

Query OK, 1 row affected (0.08 sec)

mysql> insert into marks values('4','ashley','820');

Query OK, 1 row affected (0.08 sec)
```

```
mysql> insert into marks values('5','partik','740');

Query OK, 1 row affected (0.03 sec)

mysql> insert into marks values('6','patil','640');

Query OK, 1 row affected (0.08 sec)

mysql> delimiter //

mysql> create procedure proc_result(in marks int,out class
char(20))
-> begin
-> if(marks<1500&&marks>990)
-> then
-> set class='Distincton';
-> end if;
-> if(marks<989&&marks>890)
-> then
-> set class='First Class';
-> end if;
-> if(marks<889&&marks>825)
-> then
-> set class='Higher Second Class';
-> end if;
-> if(marks<824&&marks>750)
-> then
-> set class='Second Class';-> end if;if(marks<749&&marks>650)
-> then
-> set class='Passed';
-> end if;
-> if(marks<649)
-> then
-> set class='Fail';
-> end if;
-> end;
-> //
Query OK, 0 rows affected (0.00 sec)
mysql> create function final_result3(R1 int)
```

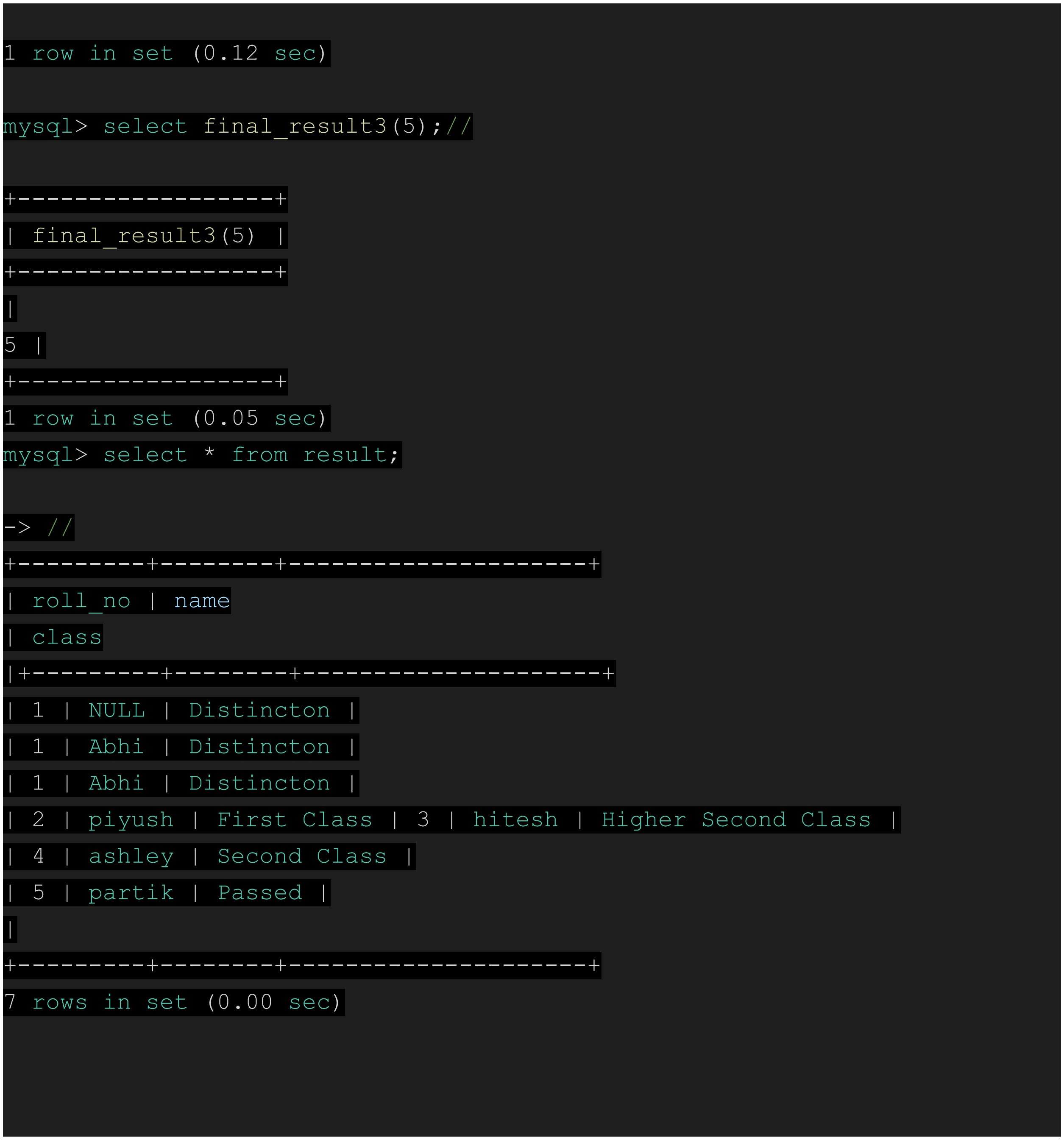
```
-> returns int
-> begin
-> declare fmarks integer;
-> declare grade varchar(20);
-> declare stud_name varchar(20);
-> select marks.total_marks,marks.name into
fmarks,stud_name from marks where marks.roll_no=R1;
-> call proc_grade(fmarks,@grade);
-> insert into result values(R1,stud_name,@grade);
-> return R1;
-> end;
-> //

Query OK, 0 rows affected (0.00 sec)

mysql> select final_result3(2);
-> //
+------------------+
| final_result3(2) |
+------------------+|
2 |
+------------------+
1 row in set (0.05 sec)

mysql> select final_result3(3);//
+------------------+
| final_result3(3) |
+------------------+
|
3 |
+------------------+
1 row in set (0.04 sec)

mysql> select final_result3(4);//
+------------------+
| final_result3(4) |
+------------------+
|
4 |
```



```
1 row in set (0.12 sec)

mysql> select final_result3(5);//

+-------------------+
| final_result3(5) |
+-------------------+
|
5 |
+-------------------+
1 row in set (0.05 sec)
mysql> select * from result;

-> //
+---------+--------+---------------------+
| roll_no | name
| class
|+---------+--------+---------------------+
| 1 | NULL | Distincton |
| 1 | Abhi | Distincton |
| 1 | Abhi | Distincton |
| 2 | piyush | First Class | 3 | hitesh | Higher Second Class |
| 4 | ashley | Second Class |
| 5 | partik | Passed |
|
+---------+--------+---------------------+
7 rows in set (0.00 sec)
```

Conclusion :- Thus successfully implemented procedures and function in PL/SQL.