College Of Engineering Trivandrum

Application Software Development Lab

Abhishek Manoharan
S5 CSE Roll No:2

TVE17CS002

Department of Computer Science

August 23, 2019

Contents

List of Figures

Cycle 2

Exp No 11

PROCEDURES AND FUNCTIONS

1 Aim

To study PL/SQL trigger and exception handling.

2 Description

FUNCTIONS

A function is a subprogram that computes a value. Functions and procedures are structured alike, except that functions have a RETURN clause. You write functions using the syntax

```
FUNCTION name [(parameter[, parameter, ...])]
    RETURN datatype IS [local declarations]
BEGIN
    executable statements
[EXCEPTION
    exception handlers]
END
[name];
```

PROCEDURES

A procedure is a subprogram that performs a specific action. You write procedures using the syntax

```
  PROCEDURE name [(parameter[,parameter, ...])] IS [local declarations]
BEGIN
executable statements
[EXCEPTION
    exception handlers]
END
[name];
```

3 Questions

3.1 Factorial of a number

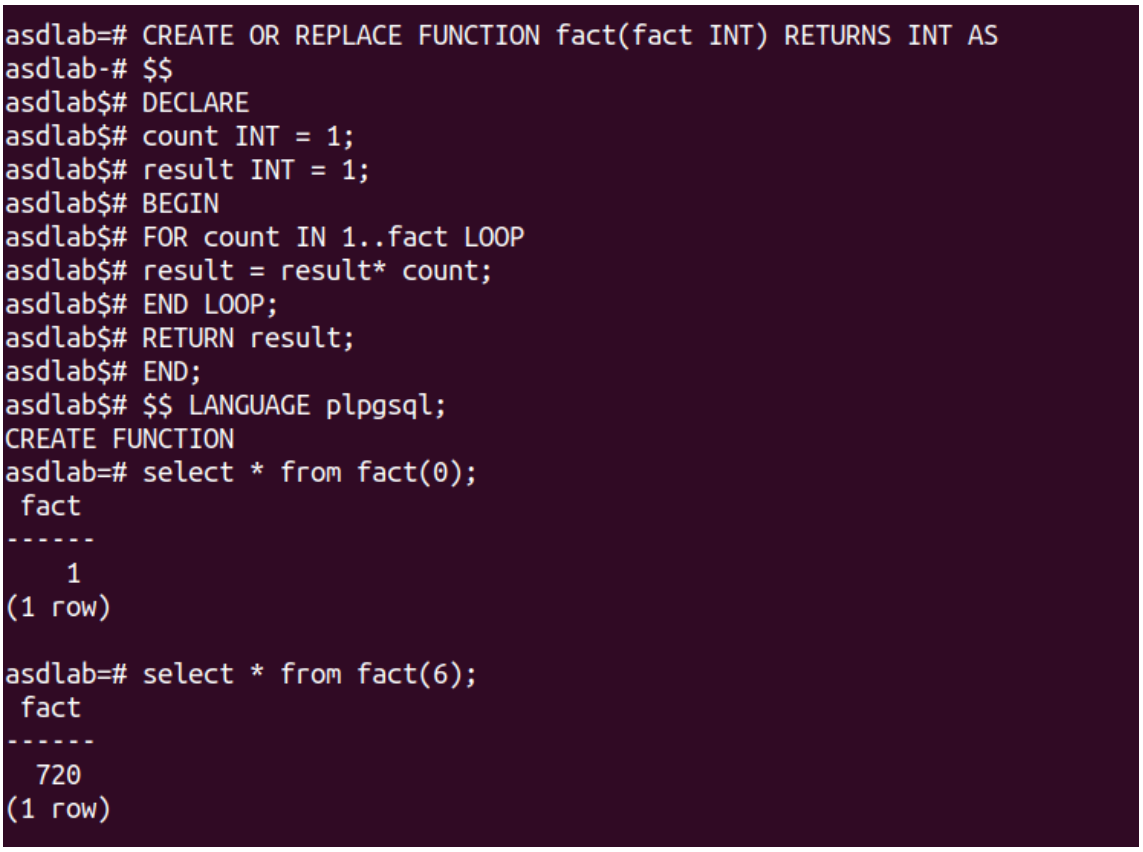
1. Create a function factorial to find the factorial of a number. Use this function in a PL/SQL Program to display the factorial of a number read from the user

3.1.1 Code

```
CREATE OR REPLACE FUNCTION fact(fact INT) RETURNS INT AS
$$
DECLARE
    count INT = 1;
    result INT = 1;
BEGIN
    FOR count IN 1..fact LOOP
    result = result* count;
    END LOOP;
RETURN result;
END;
$$ LANGUAGE plpgsql;
```

3.1.2 Output

```
SELECT * FROM fact(n);
```



```
asdlab=# CREATE OR REPLACE FUNCTION fact(fact INT) RETURNS INT AS
asdlab-# $$
asdlab$# DECLARE
asdlab$# count INT = 1;
asdlab$# result INT = 1;
asdlab$# BEGIN
asdlab$# FOR count IN 1..fact LOOP
asdlab$# result = result* count;
asdlab$# END LOOP;
asdlab$# RETURN result;
asdlab$# END;
asdlab$# $$ LANGUAGE plpgsql;
CREATE FUNCTION
asdlab=# select * from fact(0);
 fact
------
    1
(1 row)

asdlab=# select * from fact(6);
 fact
------
  720
(1 row)
```

Figure 1: Factorial of a number

3.2 Boost Marks

2. Create a table student_details(roll int,marks int, phone int). Create a procedure pr1 to update all rows in the database. Boost the marks of all students by 5%..

3.2.1 Table Creation

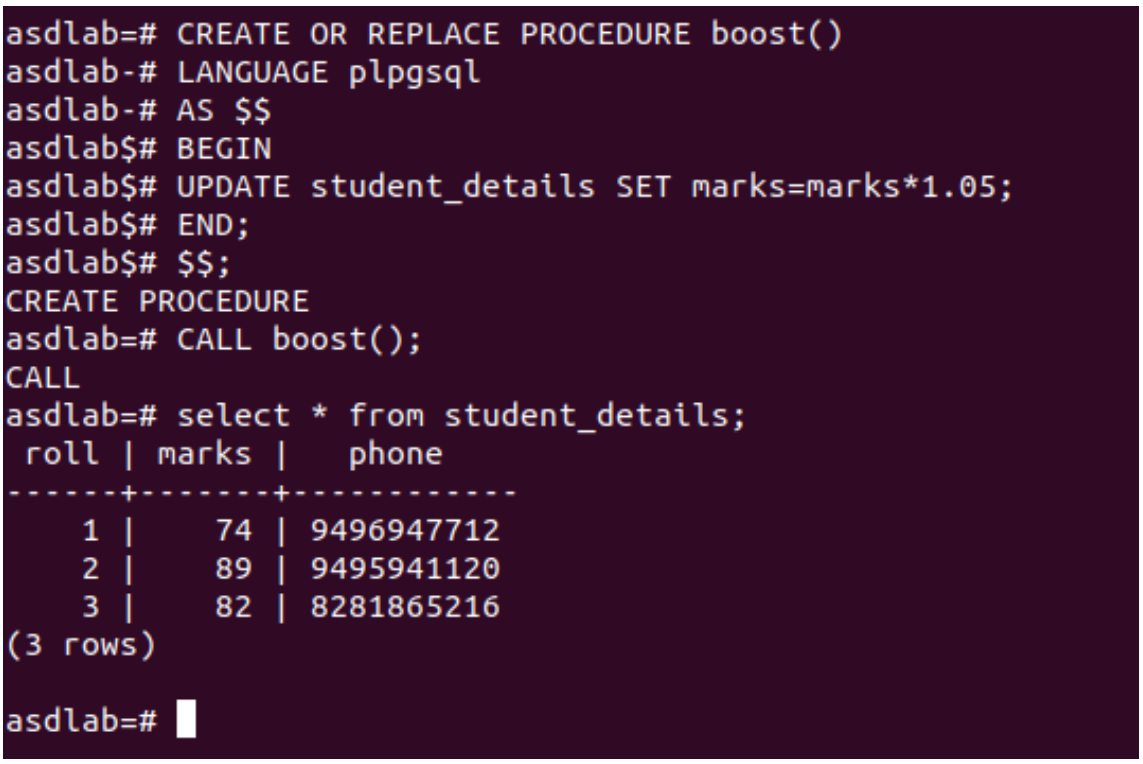
```
TABLE student_details(roll int,marks int, phone int);

INSERT INTO student_details VALUES(1,70,9496947423);
INSERT INTO student_details VALUES(2,85,9495941358);
INSERT INTO student_details VALUES(3,78,8281865009);
```

3.2.2 Code

```
CREATE OR REPLACE PROCEDURE boost()
LANGUAGE plpgsql
AS $$
BEGIN
    UPDATE student_details SET marks=marks*1.05;
END;
$$;
```

3.2.3 Output

```
CALL boost();
```



```
asdlab=# CREATE OR REPLACE PROCEDURE boost()
asdlab-# LANGUAGE plpgsql
asdlab-# AS $$
asdlab$# BEGIN
asdlab$# UPDATE student_details SET marks=marks*1.05;
asdlab$# END;
asdlab$# $$;
CREATE PROCEDURE
asdlab=# CALL boost();
CALL
asdlab=# select * from student_details;
 roll | marks |   phone
------+-------+------------
    1 |    74 | 9496947712
    2 |    89 | 9495941120
    3 |    82 | 8281865216
(3 rows)

asdlab=#
```

Figure 2: Boost Marks

3.3 Finding Total and grade

3. Create table student (id, name, m1, m2, m3, total, grade).Create a function f1 to calculate grade. Create a procedure p1 to update the total and grade.

3.3.1 Table creation

CREATE TABLE studentmark(id int, name varchar(10), m1 int, m2 int, m3 int, total int, grade varchar(1));

3.3.2 Code

```
CREATE OR REPLACE FUNCTION insert_stud(id INT ,name varchar(20),m1 INT, m2 INT, m3 INT)
RETURNS VOID AS
$$
DECLARE
total INT;
grade CHAR;
BEGIN
total=m1+m2+m3;
INSERT INTO studentmark VALUES(id,name,m1,m2,m3,total);
IF  total >=240 THEN
grade='A';
ELSIF total >=180 THEN
grade='B';
ELSIF total>=120 THEN
grade='C';
ELSIF  total>=60 THEN
grade = 'D' ;
ELSE
grade ='F';
END IF;
CALL insert_grade(id,grade);
END;
$$
LANGUAGE plpgsql;

CREATE OR REPLACE PROCEDURE insert_grade(sid INT ,sgrade CHAR)
LANGUAGE plpgsql
AS $$
BEGIN
UPDATE studentmark SET grade=sgrade WHERE id=sid;
END;
$$;
```

3.3.3 Output

```
asdlab=# CREATE OR REPLACE FUNCTION insert_stud(id INT ,name varchar(20),m1 INT, m2 INT, m3 INT)
asdlab-# RETURNS VOID AS
asdlab-# $$
asdlab$# DECLARE
asdlab$#  total INT;
asdlab$#  grade CHAR;
asdlab$# BEGIN
asdlab$#  total=m1+m2+m3;
asdlab$#  INSERT INTO studentmark VALUES(id,name,m1,m2,m3,total);
asdlab$#  IF  total >=240 THEN
asdlab$#   grade='A';
asdlab$#  ELSIF total >=180 THEN
asdlab$#   grade='B';
asdlab$#  ELSIF total>=120 THEN
asdlab$#   grade='C';
asdlab$#  ELSIF  total>=60 THEN
asdlab$#   grade = 'D' ;
asdlab$#  ELSE
asdlab$#   grade ='F';
asdlab$#  END IF;
asdlab$#  CALL insert_grade(id,grade);
asdlab$# END;
asdlab$# $$
asdlab-# LANGUAGE plpgsql;
CREATE FUNCTION
asdlab=# CREATE OR REPLACE PROCEDURE insert_grade(sid INT ,sgrade CHAR)
asdlab-# LANGUAGE plpgsql
asdlab-# AS $$
asdlab$# BEGIN
asdlab$#   UPDATE studentmark SET grade=sgrade WHERE id=sid;
asdlab$# END;
asdlab$# $$;
CREATE PROCEDURE
asdlab=# select from insert_stud(1,'raju',90,90,90);
--
(1 row)

asdlab=# select * from studentmark;
 id | name | m1 | m2 | m3 | total | grade
----+------+----+----+----+-------+-------
  1 | raju | 90 | 90 | 90 |   270 | A
(1 row)

asdlab=#
```

Figure 3: Total Marks and Grade

4 Result

The PL/SQL program was executed successfully and the output was obtained.