College of Engineering, Trivandrum

Department of Computer Science and Engineering

CS333 APPLICATION SOFTWARE DEVELOPMENT LAB

LABORATORY REPORT 9

PL/SQL

Student Name	Student ID
1. Justine Biju(S5)	170445(Roll No:37)

Submission Date : 19/08/2019

1 Introduction

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. PL/SQL is one of three key programming languages embedded in the Oracle Database, along with SQL itself and Java. This tutorial will give you great understanding on PL/SQL to proceed with Oracle database and other advanced RDBMS concepts.

The advantages of PL/SQL are as follows:

1. Better performance, as SQL is executed in bulk rather than a single statement.
2. High Productivity
3. Tight integration with SQL
4. Full Portability
5. Support Object Oriented Programming concepts.
6. Tight Security

The PL/SQL architecture has mainly three components:

1. PL/SQL Block.
2. PL/SQL Engine.
3. Database Server.

The PL/SQL Block is the component which has the actual PL/SQL code. This consists of different sections to divide the code logically (declarative section for declaring purpose, execution section for processing statements, exception handling section for handling errors). It also contains the SQL instruction that used to interact with the database server. All the PL/SQL units are treated as PL/SQL blocks, and this is the starting stage of the architecture which serves as the primary input.

PL/SQL engine is the component where the actual processing of the codes takes place. The separated PL/SQL units will be handled by the PL/SQL engine itself.The SQL part will be sent to database server where the actual interaction with database takes place.

The Database Server is the most important component of Pl/SQL unit which stores the data. The PL/SQL engine uses the SQL from PL/SQL units to interact with the database server. It consists of SQL executor which parses the input SQL statements and execute the same.

All these are supported by PostgreSQL and can be implemented in a similar fashion.

2 Questions

1. To print the first 'n' prime numbers.

```
CREATE FUNCTION prime(n integer) RETURNS INTEGER AS $$
DECLARE
    num integer := n;
    c integer := 2;
    flag integer := 0;
    temp integer;
BEGIN
    LOOP
        EXIT WHEN c = n-1;
        temp := 2;
        flag = 0;
        LOOP
            EXIT WHEN temp = c/2+1;
            IF c % temp = 0 THEN
                flag := 1;
                EXIT;
            END IF;
            temp := temp +1;
        END LOOP;
        IF flag = 0 THEN
            RAISE NOTICE '% is prime',c;
        END IF;
        c := c+1;
    END LOOP;
    RETURN 0;
END;
$$ LANGUAGE plpgsql;
```

Figure 1: Question 1

2. Display the Fibonacci series upto ‘n’ terms.

```
CREATE FUNCTION fib(n integer) RETURNS INTEGER AS $$
DECLARE
    num INTEGER := n;
    cnt INTEGER := 0;
    a INTEGER := 0;
    b INTEGER := 1;
    c INTEGER := 0;
BEGIN
    RAISE NOTICE '%',a;
    RAISE NOTICE '%',b;
    LOOP
        EXIT WHEN cnt = n-1;
        c := a + b;
        RAISE NOTICE '%',c;
        a := b;
        b := c;
        cnt := cnt+1;
    END LOOP;
    RETURN 0;
END;
$$ LANGUAGE plpgsql;
```

Figure 2: Question 2

3. Create a table named student_grade with the given attributes: roll, name ,mark1,mark2,mark3, grade. Read the roll, name and marks from the user. Calculate the grade of the student and insert a tuple into the table using PL/SQL. (Grade= 'PASS' if AVG ¿40, Grade ='FAIL' otherwise.

```
CREATE OR REPLACE FUNCTION porf(rn integer, name varchar(5),
mark1 INTEGER, mark2 integer, mark3 integer)
RETURNS INTEGER AS $$
DECLARE
    r integer := rn;
    nm VARCHAR(5) := name;
    m1 INTEGER := mark1;
    m2 INTEGER := mark2;
    m3 INTEGER := mark3;
    grade CHAR(4);
    sum INTEGER;
BEGIN
    sum = (m1 + m2 + m3)/3;
    IF sum > 40 THEN
            grade := 'PASS';
        ELSE
            grade := 'FAIL';
    END IF;
    INSERT INTO student_marks VALUES (r,nm,m1,m2,m3,grade);
    RETURN 0;
END;
$$ LANGUAGE plpgsql;
```

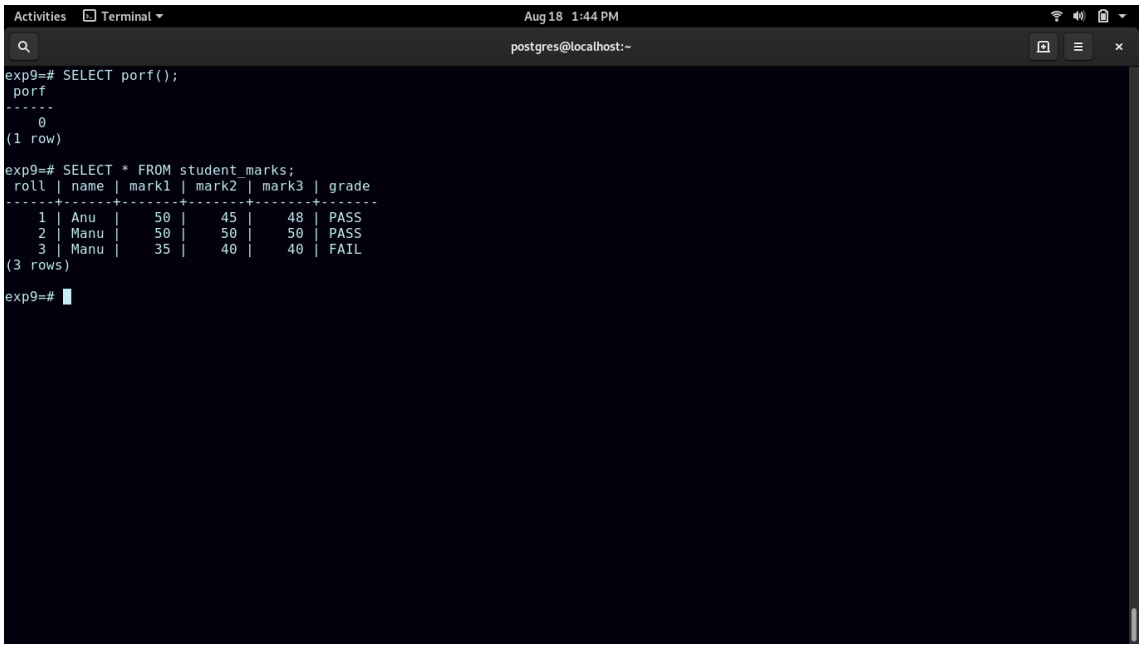


Figure 3: Question 3

4. Create table circle_area (rad,area). For radius 5,10,15,20 25., find the area and insert the corresponding values into the table by using loop structure in PL/SQL.

```
CREATE OR REPLACE FUNCTION calc_area(n integer) RETURNS INTEGER AS
DECLARE
    nu integer := n;
    count INTEGER := 0;
    r INTEGER;
BEGIN
    LOOP
        EXIT WHEN count = n;
        r := 5*(count+1);
        INSERT INTO circle_area VALUES(r, 3.14*r*r);
        count := count +1;
    END LOOP;
    RETURN 0;
END;
$$ LANGUAGE plpgsql;
```

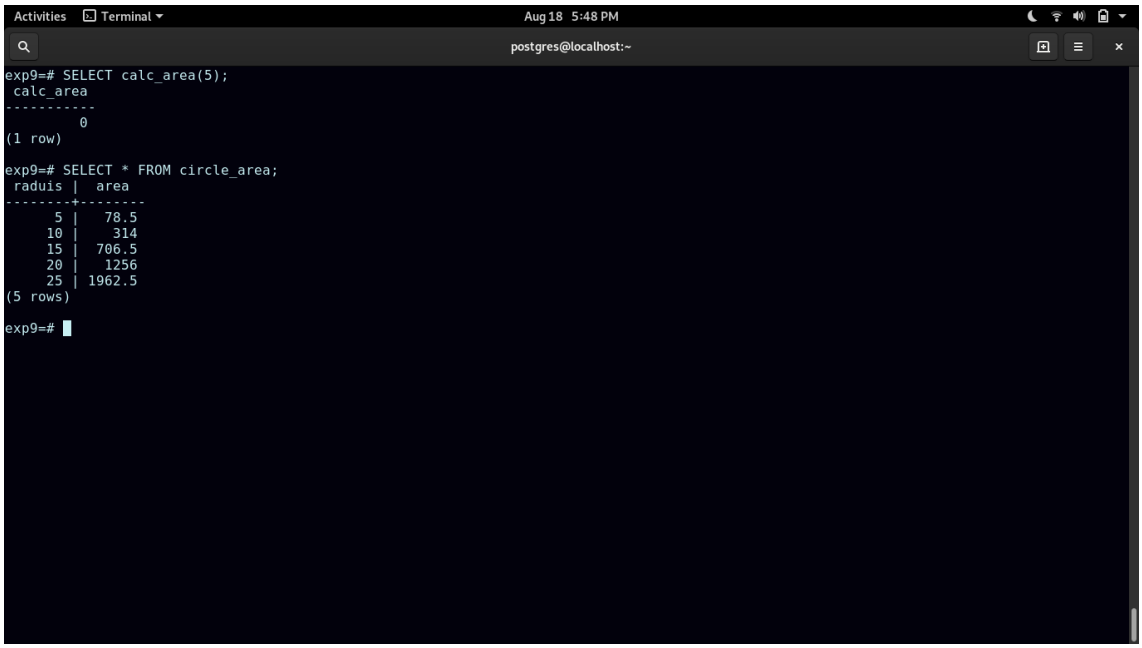
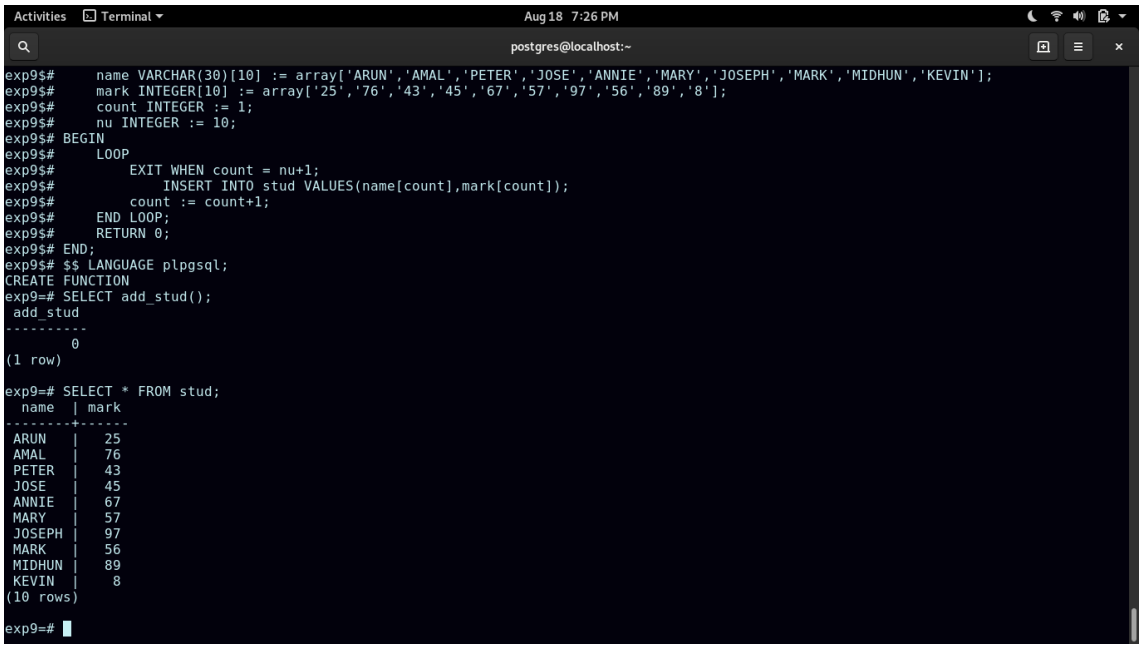


Figure 4: Question 4

5. Use an array to store the names, marks of 10 students in a class. Using Loop structures in PL/SQL insert the ten tuples to a table named stud.

```
CREATE OR REPLACE FUNCTION add_stud() RETURNS INTEGER AS $$
DECLARE
    name VARCHAR(30)[10] :=
    array['ARUN','AMAL','PETER','JOSE','ANNIE',
    'MARY','JOSEPH','MARK','MIDHUN','KEVIN'];
    mark INTEGER[10] :=
    array['25','76','43','45','67','57','97','56','89','8'];
    count INTEGER := 1;
    nu INTEGER := 10;
BEGIN
    LOOP
        EXIT WHEN count = nu+1;
            INSERT INTO stud VALUES(name[count],mark[count]);
        count := count+1;
    END LOOP;
    RETURN 0;
END;
$$ LANGUAGE plpgsql;
```



```
exp9$#     name VARCHAR(30)[10] := array['ARUN','AMAL','PETER','JOSE','ANNIE','MARY','JOSEPH','MARK','MIDHUN','KEVIN'];
exp9$#     mark INTEGER[10] := array['25','76','43','45','67','57','97','56','89','8'];
exp9$#     count INTEGER := 1;
exp9$#     nu INTEGER := 10;
exp9$# BEGIN
exp9$#     LOOP
exp9$#         EXIT WHEN count = nu+1;
exp9$#             INSERT INTO stud VALUES(name[count],mark[count]);
exp9$#         count := count+1;
exp9$#     END LOOP;
exp9$#     RETURN 0;
exp9$# END;
exp9$# $$ LANGUAGE plpgsql;
CREATE FUNCTION
exp9=# SELECT add_stud();
 add_stud
----------
        0
(1 row)

exp9=# SELECT * FROM stud;
  name  | mark
--------+------
 ARUN   |   25
 AMAL   |   76
 PETER  |   43
 JOSE   |   45
 ANNIE  |   67
 MARY   |   57
 JOSEPH |   97
 MARK   |   56
 MIDHUN |   89
 KEVIN  |    8
(10 rows)

exp9=#
```

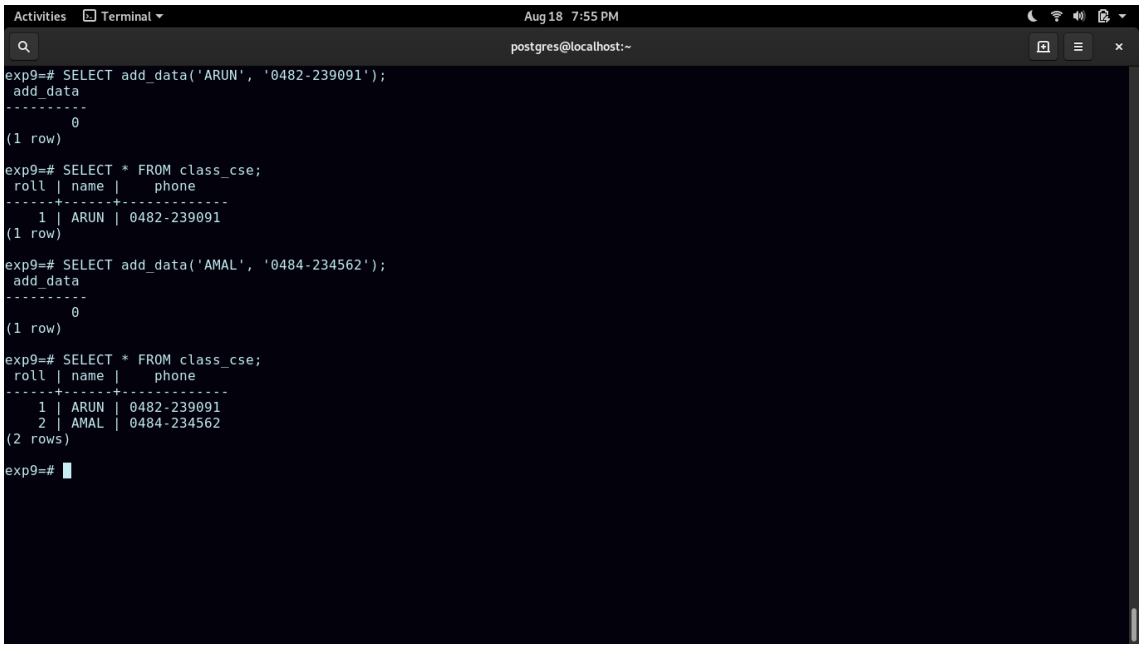
Figure 5: Question 5

6. Create a sequence using PL/SQL. Use this sequence to generate the primary key values for a table named class_cse with attributes roll,name and phone. Insert some tuples using PL/SQL programming.

```
CREATE SEQUENCE ai
START WITH 1
INCREMENT BY 1;

CREATE OR REPLACE FUNCTION add_data(name VARCHAR(30), ph VARCHAR(30
DECLARE

BEGIN
    INSERT INTO class_cse VALUES(nextval('ai'),name, ph);
    RETURN 0;
END;
$$ LANGUAGE plpgsql;
```



```
exp9=# SELECT add_data('ARUN', '0482-239091');
 add_data
----------
        0
(1 row)

exp9=# SELECT * FROM class_cse;
 roll | name |    phone
------+------+------------
    1 | ARUN | 0482-239091
(1 row)

exp9=# SELECT add_data('AMAL', '0484-234562');
 add_data
----------
        0
(1 row)

exp9=# SELECT * FROM class_cse;
 roll | name |    phone
------+------+------------
    1 | ARUN | 0482-239091
    2 | AMAL | 0484-234562
(2 rows)

exp9=#
```

Figure 6: Question 6

3 Result

- Successfully implemented PL/SQL in PostgreSQL.