Sameer Dehadrai Page: 1

PL*SQL

Exercise 1

1. Write a program that computes the perimeter and the area of a rectangle. Define your own values for the length and width. (Assuming that L and W are the length and width of the rectangle, Perimeter = 2*(L+W) and Area = L*W. Display the output on the screen using dbms_output.put_line.

```
  1  create or replace procedure rectangle(l number,w number)
  2  as
  3  area number;
  4  peri number;
  5  begin
  6     area:=l*w;
  7     peri:=2*(l+w);
  8     DBMS_OUTPUT.PUT_LINE('area ' || area);
  9     DBMS_OUTPUT.PUT_LINE('area ' || peri);
 10* end;
SQL> /

Procedure created.

SQL> variable l number;
SQL> variable w number;
SQL> execute rectangle(10,20);

PL/SQL procedure successfully completed.

SQL> set serveroutput on
SQL> execute rectangle(10,20);
area 200
area 60
```

2. Write a program that declares an integer variable called num, assigns a value to it, and computes and inserts into the tempp table the value of the variable itself, its square, and its cube.

```
  1  create or replace procedure sqrt1(num number)
  2  as
  3  square number;
  4  cube number;
  5  begin
  6     square:=num*num;
  7          cube:=num*num*num;
  8          insert into temp values(num,square,cube);
  9* end;
SQL> /

Procedure created.
```

```
SQL> execute sqrt1(7);

PL/SQL procedure successfully completed.

SQL> select * from temp;

      NUMB     SQUARE       CUBE
---------- ---------- ----------
         5         25        125
         8         64        512
         8         64        512
         7         49        343
```

3. Convert a temperature in Fahrenheit (F) to its equivalent in Celsius (C) and vice versa. The required formulae are:-

C= (F-32)*5/9

F= 9/5*C + 32

Display the output on the screen using dbms_output.put_line. Data has to be input by the user.

```
  1  create or replace procedure temperature(c in number,f in number)
  2  as
  3  newc number;
  4  newf number;
  5  begin
  6     newc:=round(((f-32)*5)/9,2);
  7     newf:=round((9/5*c)+32,2);
  8     DBMS_OUTPUT.PUT_LINE('changing into f :' || newf);
  9     DBMS_OUTPUT.PUT_LINE('changing into c :' || newc);
 10* end;
SQL> /

Procedure created.

SQL> execute temperature(50,150)
changing into f :122
changing into c :65.56

PL/SQL procedure successfully completed.
```

4. Convert a number of inches into yards, feet, and inches. For example, 124 inches equals 3 yards, 1 foot, and 4 inches. Display the output on the screen using dbms_output.put_line. Data has to be input by the user.

```
  1  create or replace procedure yards(n number)
  2  as
  3  yard number;
  4  foot number;
  5  inch number;
  6  temp number;
  7  begin
  8     yard:=round(n/36);
  9     temp:=mod(n,36);
 10     foot:=round(temp/12);
 11     temp:=mod(n,12);
 12     DBMS_OUTPUT.PUT_LINE(n || 'inches :' );
 13     DBMS_OUTPUT.PUT_LINE(yard || 'yards, '||foot || 'foots, '|| temp || 'inches' );
 14* end;
SQL>
SQL> /

Procedure created.

SQL> execute yards(124)
124inches :
3yards, 1foots, 4inches

PL/SQL procedure successfully completed.

SQL>
```

5. Write a program that enables a user to input an integer. The program should then

state whether the integer is evenly divisible by 5. (Use decode instead of IF statement where required). Display the output on the screen using dbms_output.put_line. Data has to be input by the user.

```
  1  create or replace procedure divitest
  2  as
  3  num number:=&n;
  4  begin
  5     if mod(num,5)=0 then
  6     DBMS_OUTPUT.PUT_LINE('divisible');
  7     else
  8     DBMS_OUTPUT.PUT_LINE(' not divisible');
  9  end if;
 10* end;
SQL> /
Enter value for n: 56
old   3: num number:=&n;
new   3: num number:=56;

Procedure created.

SQL> execute divitest()
not divisible

PL/SQL procedure successfully completed.

SQL>
```

```
Enter value for n: 15
old   3: num number:=&n;
new   3: num number:=15;

Procedure created.

SQL> execute divitest()
divisible
```

WITH DECODE

```
SQL> ed
Wrote file afiedt.buf

  1  create or replace procedure fiveb
  2  as
  3  num number:=&n;
  4  result varchar2(20);
  5  begin
  6  select DECODE(mod(num,5), 0, 'DIVISIBLE', 'NOT DIVISIBLE') into result
  7  from dual;
  8  DBMS_OUTPUT.PUT_LINE(result);
  9* end;
SQL> /
Enter value for n: 56
old   3: num number:=&n;
new   3: num number:=56;

Procedure created.

SQL> execute fiveb();
NOT DIVISIBLE

PL/SQL procedure successfully completed.
```

```
Enter value for n: 50
old   3: num number:=&n;
new   3: num number:=50;

Procedure created.

SQL> execute fiveb();
DIVISIBLE
```

6. Your block should read in two real numbers and tell whether the product of the two numbers is equal to or greater than 100. Display the output on the screen using dbms_output.put_line. (Use decode instead of IF statement where required). Data has to be input by the user

```
 1  create or replace procedure productnum
 2  as
 3  n1 number:=&n;
 4  n2 number:=&n;
 5  product number;
 6  str varchar2(20);
 7  begin
 8     product:=n1*n2;
 9     if product>=100 then
10     select decode(product,100,'hundred','greater') into str
11     from dual;
12     DBMS_OUTPUT.PUT_LINE(str);
13     else
14     DBMS_OUTPUT.PUT_LINE('lower than 100');
15     end if;
16* end;
```

```
PL/SQL procedure successfully completed.

SQL> /
Enter value for n: 20
old   3: n1 number:=&n;
new   3: n1 number:=20;
Enter value for n: 5
old   4: n2 number:=&n;
new   4: n2 number:=5;

Procedure created.

SQL> execute productnum();
hundred

PL/SQL procedure successfully completed.

SQL> /
Enter value for n: 600
old   3: n1 number:=&n;
new   3: n1 number:=600;
Enter value for n: 2
old   4: n2 number:=&n;
new   4: n2 number:=2;

Procedure created.

SQL> execute productnum();
greater
```