PL/SQL Assignment 2020 [DBMS_ASS_4_19]

NAME: Soumitri Chattopadhyay
ROLL: 001911001083
DEPARTMENT: JU IT'23

---

1. Write a PL/SQL code to print Today is fall on weekend or weekdays using if else statement.

```
-- IS WEEKEND
set serveroutput on
declare
todays_date DATE;
current_day varchar(9);
begin
todays_date :=sysdate;
current_day:=to_char(todays_date,'day');
current_day:=initcap(current_day);
current_day:=rtrim(current_day);
if current_day='Sunday' or current_day='Saturday' then
dbms_output.put_line('Today is fall on weekend');
else
dbms_output.put_line('Today is fall on weekdays');
end if;
end;
```

```
SQL> -- IS WEEKEND
SQL> set serveroutput on
SQL>    declare
2     todays_date DATE;
3     current_day varchar(9);
4     begin
5     todays_date :=sysdate;
6     current_day:=to_char(todays_date,'day');
7     current_day:=initcap(current_day);
8     current_day:=rtrim(current_day);
9     if current_day='Sunday' or current_day='Saturday' then
10     dbms_output.put_line('Today is fall on weekend');
11     else
12     dbms_output.put_line('Today is fall on weekdays');
13     end if;
14     end;
15  /
Today is fall on weekdays
```

```
PL/SQL procedure successfully completed.
```

2.Write a PL/SQL code to check that an inputted a single character is vowel or not .If vowel then display which vowel it is.

```
-- IS VOWEL
DECLARE
c CHAR;
BEGIN
c := LOWER('&c');
-- IF/ELSE
IF c='a' OR c='e' OR c='i' OR c='o' OR c='u' THEN
DBMS_OUTPUT.PUT_LINE(c || ' is a vowel!');
ELSE
DBMS_OUTPUT.PUT_LINE('The entered charactered character is
not a vowel :(');
END IF;
END;
```

```
SQL> -- IS VOWEL
SQL> DECLARE
2    c CHAR;
3  BEGIN
4    c := LOWER('&c');
5    -- IF/ELSE
6    IF c='a' OR c='e' OR c='i' OR c='o' OR c='u' THEN
7      DBMS_OUTPUT.PUT_LINE(c || ' is a vowel!');
8    ELSE
9      DBMS_OUTPUT.PUT_LINE('The entered charactered character is
not a vowel :(');
10    END IF;
11  END;
12  /
Enter value for c: a
old   4:   c := LOWER('&c');
new   4:   c := LOWER('a');
a is a vowel!

PL/SQL procedure successfully completed.
```

3.Write a PL/SQL code block to find out the sum of first ten natural numbers (1+2+3+4+5+6+7+8+9+10 this series).

```
-- Natural number sum till 10
```

```
DECLARE
a NUMBER := 0;
BEGIN
FOR i IN 1..10 LOOP
a := a + i;
END LOOP;
DBMS_OUTPUT.PUT_LINE('SUM = ' || a);
END;

SQL> -- Natural number sum till 10
SQL> DECLARE
2    a NUMBER := 0;
3  BEGIN
4    FOR i IN 1..10 LOOP
5      a := a + i;
6    END LOOP;
7    DBMS_OUTPUT.PUT_LINE('SUM = ' || a);
8  END;
9  /
SUM = 55

PL/SQL procedure successfully completed.
```

4.Write a PL/SQL block that will ask for two numbers and one operand (+, -, *, /). Then it will calculate and display the result.

```
-- Operate on two numbers
DECLARE
a NUMBER;
b NUMBER;
res NUMBER;
op CHAR;
BEGIN
a := &a;
b := &b;
op := '&op';

DBMS_OUTPUT.PUT_LINE('Requested operation: ' || a || op ||
b);

-- IF/ELSE TO OPERATE
IF op = '+' THEN
res := a + b;
ELSIF op = '-' THEN
res := a - b;
```

```
ELSIF op = '*' THEN
res := a * b;
ELSIF op = '/' THEN
res := a / b;
ELSE
DBMS_OUTPUT.PUT_LINE('Operand not supported.');
END IF;

-- printing result if flag is set
DBMS_OUTPUT.PUT_LINE(a || op || b || ' = ' || res);

EXCEPTION
WHEN ZERO_DIVIDE THEN
DBMS_OUTPUT.PUT_LINE('Divide by 0 not possible.');
END;
```

```
SQL> -- Operate on two numbers
SQL> DECLARE
2    a NUMBER;
3    b NUMBER;
4    res NUMBER;
5    op CHAR;
6  BEGIN
7    a := &a;
8    b := &b;
9    op := '&op';
10
11    DBMS_OUTPUT.PUT_LINE('Requested operation: ' || a || op || b);
12
13    -- IF/ELSE TO OPERATE
14    IF op = '+' THEN
15      res := a + b;
16    ELSIF op = '-' THEN
17      res := a - b;
18    ELSIF op = '*' THEN
19      res := a * b;
20    ELSIF op = '/' THEN
21      res := a / b;
22    ELSE
23      DBMS_OUTPUT.PUT_LINE('Operand not supported.');
24    END IF;
25
26    -- printing result if flag is set
27    DBMS_OUTPUT.PUT_LINE(a || op || b || ' = ' || res);
28
29  EXCEPTION
30    WHEN ZERO_DIVIDE THEN
31      DBMS_OUTPUT.PUT_LINE('Divide by 0 not possible.');
32  END;
```

```
33  /
Enter value for a: 5
old   7:   a := &a;
new   7:   a := 5;
Enter value for b: 5
old   8:   b := &b;
new   8:   b := 5;
Enter value for op: +
old   9:   op := '&op';
new   9:   op := '+';
Requested operation: 5+5
5+5 = 10

PL/SQL procedure successfully completed.
```

5.Write a PL/SQL code block to display a number in a reverse way.

```
-- REVERSING A NUMBER
DECLARE
val BINARY_INTEGER;
rev BINARY_INTEGER := 0;
BEGIN
val := &val;
WHILE val <> 0 LOOP
rev := rev*10 + MOD(val, 10);
val := (val - MOD(val, 10)) / 10;
END LOOP;
DBMS_OUTPUT.PUT_LINE(rev);
END;
```

```
SQL> -- REVERSING A NUMBER
SQL> DECLARE
2    val BINARY_INTEGER;
3    rev BINARY_INTEGER := 0;
4  BEGIN
5    val := &val;
6    WHILE val <> 0 LOOP
7      rev := rev*10 + MOD(val, 10);
8      val := (val - MOD(val, 10)) / 10;
9    END LOOP;
10    DBMS_OUTPUT.PUT_LINE(rev);
11  END;
12  /
Enter value for val: 100
old   5:   val := &val;
new   5:   val := 100;
```

```
1

PL/SQL procedure successfully completed.
```

6.Write a PL/SQL block to display the dates of this month which are Tuesday.

```
-- TUESDAYS OF THIS MONTH
DECLARE
d DATE := '01-NOV-2019';
tuesday VARCHAR2(9) := 'Tuesday';
curr_day VARCHAR2(9);
BEGIN
WHILE d <> '1-DEC-2019' LOOP
curr_day := RTRIM(INITCAP(TO_CHAR(d, 'DAY')));
IF curr_day = 'Tuesday' THEN
DBMS_OUTPUT.PUT_LINE(d);
END IF;
d := d + 1;
END LOOP;
END;

SQL> -- TUESDAYS OF THIS MONTH
SQL> DECLARE
2    d DATE := '01-NOV-2019';
3    tuesday VARCHAR2(9) := 'Tuesday';
4    curr_day VARCHAR2(9);
5  BEGIN
6    WHILE d <> '1-DEC-2019' LOOP
7      curr_day := RTRIM(INITCAP(TO_CHAR(d, 'DAY')));
8      IF curr_day = 'Tuesday' THEN
9        DBMS_OUTPUT.PUT_LINE(d);
10      END IF;
11      d := d + 1;
12    END LOOP;
13  END;
14  /
05-NOV-19
12-NOV-19
19-NOV-19
26-NOV-19

PL/SQL procedure successfully completed.
```

7.Write a program in PL/SQL to print the prime numbers between 1 to 50.

```
-- PRIME BETWEEN 1 & 50
DECLARE
limit BINARY_INTEGER := 50;
f BINARY_INTEGER;
BEGIN
FOR i in 1..limit LOOP
f := 0;
IF i <> 1 THEN
FOR j in 2..(i-1) LOOP
IF MOD(i, j) = 0 THEN
f := 1;
EXIT;
END IF;
END LOOP;
IF f = 0 THEN
DBMS_OUTPUT.PUT_LINE(i);
END IF;
END IF;
END LOOP;
END;
```

```
SQL> -- PRIME BETWEEN 1 & 50
SQL> DECLARE
2    limit BINARY_INTEGER := 50;
3    f BINARY_INTEGER;
4  BEGIN
5    FOR i in 1..limit LOOP
6      f := 0;
7      IF i <> 1 THEN
8      FOR j in 2..(i-1) LOOP
9        IF MOD(i, j) = 0 THEN
10          f := 1;
11          EXIT;
12        END IF;
13      END LOOP;
14      IF f = 0 THEN
15        DBMS_OUTPUT.PUT_LINE(i);
16      END IF;
17      END IF;
18    END LOOP;
19  END;
20  /

PL/SQL procedure successfully completed.
```

---

8. Write a PL/SQL code block that will accept an Employee number from the user and allows a Commission of Rs.200 only if the net salary is less than Rs.2000.
Write a PL/SQL block that will display the name, department and the salary of the first 4 employees getting lowest salary.

```
DECLARE
eno emp.empno%type;
s emp.sal%type;
c integer;
BEGIN
eno := &eno;
c:=0;
SELECT sal into s from emp where empno = eno;
IF s<2000 THEN
UPDATE emp SET comm = 200 where empno = eno;
END IF;
FOR cursor1 IN (SELECT ename, dname, sal
from emp, dept
where emp.deptno = dept.deptno
order by (12*sal+nvl(comm, 0)))
LOOP
c:=c+1;
IF  c>4 then
exit;
END IF;
DBMS_OUTPUT.PUT_LINE('Name = ' || cursor1.ename ||
', Department = ' || cursor1.dname||
',Salary = ' || cursor1.sal );
END LOOP;
EXCEPTION
when no_data_found then
DBMS_OUTPUT.PUT_LINE('Wrong employee id');
END;
```

```
SQL> DECLARE
2    eno emp.empno%type;
3    s emp.sal%type;
4    c integer;
5     BEGIN
6     eno := &eno;
7     c:=0;
8     SELECT sal into s from emp where empno = eno;
9     IF s<2000 THEN
10     UPDATE emp SET comm = 200 where empno = eno;
11     END IF;
12     FOR cursor1 IN (SELECT ename, dname, sal
13     from emp, dept
```

```
14     where emp.deptno = dept.deptno
15     order by (12*sal+nvl(comm, 0)))
16        LOOP
17          c:=c+1;
18          IF  c>4 then
19          exit;
20          END IF;
21          DBMS_OUTPUT.PUT_LINE('Name = ' || cursor1.ename ||
22                                    ', Department = ' ||
cursor1.dname||
23                                    ',Salary = ' || cursor1.sal );
24        END LOOP;
25     EXCEPTION
26     when no_data_found then
27     DBMS_OUTPUT.PUT_LINE('Wrong employee id');
28     END;
29  /
Enter value for eno: 7900
old   6:    eno := &eno;
new   6:    eno := 7900;
Name = SMITH, Department = RESEARCH,Salary = 800
Name = JAMES, Department = SALES,Salary = 950
Name = ADAMS, Department = RESEARCH,Salary = 1100
Name = WARD, Department = SALES,Salary = 1250

PL/SQL procedure successfully completed.
```

---

9. Write a PL/SQL block to display the name of the department and their employee name only who got the highest net salary.

```
DECLARE
BEGIN
FOR cursor1 IN
(select ename, dname from
emp inner join dept on
emp.deptno = dept.deptno
where 12*sal+nvl(comm, 0) in
(select  max(12*sal+nvl(comm, 0))
from emp group by deptno ))
LOOP
DBMS_OUTPUT.PUT_LINE('Name = ' || cursor1.ename ||
', Department = ' || cursor1.dname);
END LOOP;
END;

SQL> DECLARE
2  BEGIN
```

```
3  FOR cursor1 IN
4  (select ename, dname from
5  emp inner join dept on
6  emp.deptno = dept.deptno
7  where 12*sal+nvl(comm, 0) in
8  (select  max(12*sal+nvl(comm, 0))
9  from emp group by deptno ))
10            LOOP
11              DBMS_OUTPUT.PUT_LINE('Name = ' || cursor1.ename ||
12                                   ', Department = ' ||
cursor1.dname);
13            END LOOP;
14  END;
15  /
Name = BLAKE, Department = SALES
Name = FORD, Department = RESEARCH
Name = SCOTT, Department = RESEARCH
Name = KING, Department = ACCOUNTING

PL/SQL procedure successfully completed.
```

---

10. Write a PL/SQL procedure that will calculate and print the incentive of the employees those who worked under a specific Job.

JOB	Incentive
Clerk	10% of their Salary
Salesman	12% of their Salary
Manager	5% of their Salary
Analyst	1000

```
declare
cursor c is
select ename,job,sal
from emp;
name emp.ename%type;
ejob emp.job%type;
esal emp.sal%type;
begin
open c;
loop
fetch c into name,ejob,esal;
exit when c%notfound;
if(ejob ='clerk') then
esal:=esal*1.1;
elsif(ejob ='salesman') then
esal:=esal*1.12;
elsif(ejob ='manager') then
```

```
esal:=esal*1.15;
else
esal:=esal+1000;
end if;
dbms_output.put_line(name||' '||ejob||' '||esal);
end loop;
close c;
end;
```

```
SQL> declare
2  cursor c is
3  select ename,job,sal
4  from emp;
5  name emp.ename%type;
6  ejob emp.job%type;
7  esal emp.sal%type;
8  begin
9  open c;
10  loop
11  fetch c into name,ejob,esal;
12  exit when c%notfound;
13  if(ejob ='clerk') then
14  esal:=esal*1.1;
15  elsif(ejob ='salesman') then
16  esal:=esal*1.12;
17  elsif(ejob ='manager') then
18  esal:=esal*1.15;
19  else
20  esal:=esal+1000;
21  end if;
22  dbms_output.put_line(name||' '||ejob||' '||esal);
23  end loop;
24  close c;
25  end;
26  /
FORD ANALYST 4000
MILLER CLERK 2300
SMITH CLERK 1800
ALLEN SALESMAN 2600
WARD SALESMAN 2250
JONES MANAGER 3975
MARTIN SALESMAN 2250
BLAKE MANAGER 3850
CLARK MANAGER 3450
SCOTT ANALYST 4000
KING PRESIDENT 6000
TURNER SALESMAN 2500
ADAMS CLERK 2100
JAMES CLERK 1950
```

```
PL/SQL procedure successfully completed.
```

---

11. Write a PL/SQL Cursor that will accept an employee number from the user and allows an increment of 20% if the salary is less than 1500 otherwise increment of Rs.1200. It should print old and new salary for all employees.

```
declare
empno1 emp.empno%type;
cursor c is
select sal
from emp
where empno=empno1;
oldsal emp.sal%type;
esal emp.sal%type;
begin
empno1:=&empno1;
open c;
fetch c into oldsal;
if(oldsal<1500) then
esal:=oldsal*1.2;
else
esal:=oldsal+1200;
end if;
dbms_output.put_line('oldsal = '||oldsal||',newsal =
'||esal);
update emp
set sal=esal
where empno=empno1;
close c;
end;
```

```
SQL> declare
2  empno1 emp.empno%type;
3  cursor c is
4  select sal
5  from emp
6  where empno=empno1;
7  oldsal emp.sal%type;
8  esal emp.sal%type;
9  begin
10  empno1:=&empno1;
11  open c;
12  fetch c into oldsal;
13  if(oldsal<1500) then
```

```
14  esal:=oldsal*1.2;
15  else
16  esal:=oldsal+1200;
17  end if;
18  dbms_output.put_line('oldsal = '||oldsal||',newsal = '||esal);
19  update emp
20  set sal=esal
21  where empno=empno1;
22  close c;
23  end;
24  /
Enter value for empno1: 2001
old  10: empno1:=&empno1;
new  10: empno1:=2001;
oldsal = ,newsal =

PL/SQL procedure successfully completed.
```

---

12. When any new record will be inserted in EMP_XX table, a PL/SQL trigger will fired that will check if the department no is available or not. If available then the tuple will be inserted otherwise it will undo the operation and display appropriate messages.

```
CREATE OR REPLACE TRIGGER trg
BEFORE INSERT
on emp
for each row
DECLARE
e exception;
BEGIN
if(:new.deptno not in (10,20,30,40)) then
raise e;
END IF;
EXCEPTION
when e then
Raise_application_error(-20000,'department no wrong');
END;

SQL> CREATE OR REPLACE TRIGGER trg
2  BEFORE INSERT
3  on emp
4  for each row
5  DECLARE
6  e exception;
7  BEGIN
8  if(:new.deptno not in (10,20,30,40)) then
```

```
9  raise e;
10  END IF;
11  EXCEPTION
12  when e then
13  Raise_application_error(-20000,'department no wrong');
14  END;
15  /

Trigger created.
```