Experiment:6

- Develop a program that includes the features NESTED IF, CASE and CASE expression. The program can be extended using the NULLIF and COALESCE functions.

A. NESTED IF:

A nested if-then is an if statement that is the target of another if statement. Nested if-then statements mean an if statement inside another if statement

Syntax:-

if (condition1) then

-- Executes when condition1 is true if (condition2) then

-- Executes when condition2 is true end if;

end if;

PL/SQL CODE:

PL/SQL Program to find biggest of three number using nested if. SQL>ed 6a

Enter the following code into the text editor and save the file with .sql format

```
declare
a number:=10;
b number:=12;
c number:=5;
begin
dbms_output.put_line('a='||a||' b='||b||' c='||c);
if a>b AND a>c then
        dbms_output.put_line('a is greatest');
else
        if b>a AND b>c then
        dbms_output.put_line('b is greatest');
        else
        dbms_output.put_line('c is greatest');
end if;
```

```
end if;
end;
/
```

SQL> @6a;

```
a=10 b=12 c=5
b is greatest
PL/SQL procedure successfully completed.
```

B. CASE and CASE Expression : CASE statement selects one sequence of statements to execute. However, to select the sequence, the CASE statement uses a selector rather than multiple Boolean expressions. A selector is an expression, the value of which is used to select one of several alternatives. Syntax

```
CASE selector
WHEN 'value1' THEN S1;
 WHEN 'value2' THEN S2;
 WHEN 'value3' THEN S3;
...
ELSE Sn;
 -- default case END CASE;
```

SQL> create table emp(eno number(5), ename varchar2(10), loc varchar(10), salary number(10,2));

Table created.

SQL> insert into emp values(101,'ali','vja',15000);

1 row created.

SQL> insert into emp values(102,'ravi','hyd',25000);

1 row created.

SQL> insert into emp values(103,'raju','gnt',35000);

1 row created.

SQL> insert into emp values(104,'rakesh','vja',45000);

1 row created.

SQL> select *from emp;

ENO	ENAME	LOC	SALARY
101	ali	vja	15000
102	ravi	hyd	25000
103	raju	gnt	35000
104	rakesh	vja	45000

Example of CASE Expression:

SQL> select loc, case(loc) when 'vja' then salary+2000 when 'hyd' then salary+1000 else salary end "rev_salary" from emp;

LOC	rev_salary
vja	17000
hyd	26000
gnt	35000
vja	47000

PL/SQL CODE:

PL/SQL CODE to demonstrate CASE

SQL> ed 6b;

```
set serveroutput on;
declare
grade char(1);
begin
grade:='&grade';
case grade
when 'a' then
dbms_output.put_line('Excellent');
 when 'b' then
```

```
dbms_output.put_line('very good');
when 'c' then
dbms_output.put_line('good');
when 'd' then
dbms_output.put_line('fair');
when 'f' then
dbms_output.put_line('poor');
else
dbms_output.put_line('No such grade');
end case;
end;
/
```

SQL> @6b;

```
Enter value for grade: c
old 4: grade:='&grade';
new 4: grade:='c';
good
PL/SQL procedure successfully completed.
```

SQL> @6b;

```
Enter value for grade: g
old 4: grade:='&grade';
new 4: grade:='g';
No such grade
PL/SQL procedure successfully completed.
```

C. NULLIF: Takes two arguments. If the two arguments are equal, then NULL is returned. otherwise the first argument is returned.

Syntax: select column_name, NULLIF(argument1,arguement2) from table_name;

Example:

SQL> select ename, nullif('ali','ali1') from emp;

ENAME	NULLIF
ali	ali
ravi	ali
raju	ali
rakesh	ali

SQL> select ename, nullif('ali','ali') from emp;

ENAME	NULLIF
ali	
ravi	
raju	
rakesh	

D. COALESCE: COALESCE () function accepts a list of arguments and returns the first one that evaluates to a non-null value.

Syntax: coalesce("expression1","expression2",...);

Example:

SQL> select coalesce(NULL,'CRRCOE','IT') from dual;

COALESCE(NULL,'CRRCOE','IT')

CRRCOE