1. Addition of Two Numbers

```
 1  declare
 2    a number;
 3    b number;
 4    c number;
 5  begin
 6    a:= &a;
 7    b:= &b;
 8    c:= a+b;
 9    dbms_output.put_line('Sum of '|| a ||' and  ' || b ||' is ' || c);
10* end;
```

```
SQL> /
Enter value for a: 20
old   6: a:= &a;
new   6: a:= 20;
Enter value for b: 30
old   7: b:= &b;
new   7: b:= 30;

Sum of 20 and 30 is 50
```

2. Sum of 100 Numbers

```
 1  declare
 2     a number;
 3     s number default 0;
 4  begin
 5     a:=1;
 6     loop
 7        s:=s+a;
 8        exit when(a=100);
 9        a:=a+1;
10    end loop;
11  dbms_output.put_line('Sum of 100 numbers = '||s);
12* end;
```

```
SQL> /

Sum of 100 numbers = 5050

PL/SQL procedure successfully completed.
```

3. Sum of Odd Numbers using for loop

```
 1  declare
 2     n number;
 3     sum1 number default 0;
 4     endvalue number;
 5  begin
 6     endvalue:=&endvalue;
 7     n:=1;
 8     for n in 1..endvalue
 9     loop
10         if mod(n,2)=1
11          then
12          sum1:=sum1+n;
13         end if;
14      end loop;
15    dbms_output.put_line(' sum  =  '||sum1);
16* end;


SQL> /
Enter value for endvalue: 30
old   6: endvalue:=&endvalue;
new   6: endvalue:=30;

sum  =  225

PL/SQL procedure successfully completed.
```

4. Sum of 100 odd numbers using While loop

```
 1  declare
 2     n number;
 3     endvalue number;
 4     sum1 number default 0;
 5  begin
 6     endvalue:=&endvalue;
 7     n:=1;
 8     while(n<endvalue)
 9     loop
10       sum1:=sum1+n;
11       n:=n+2;
12    end loop;
13    dbms_output.put_line('Sum of Odd numbers between 1 and '|| endvalue   ||'
is '||sum1);
14  end;
15  /

Enter value for endvalue: 30
old   6: endvalue:=&endvalue;
new   6: endvalue:=30;
Sum of Odd numbers between 1 and 30  is 225

PL/SQL procedure successfully completed.
```

5. Greatest of Three Numbers

```
 1  declare
 2     a number;
 3     b number;
 4     c number;
```

```
 5    d number;
 6  begin
 7    a:=&a;
 8    b:=&b;
 9    c:=&c;
10    if(a>b) and (a>c) then
11      dbms_output.put_line('A is greater ');
12    elsif (b>a) and (b>c) then
13      dbms_output.put_line(' B is Greater ');
14    else
15      dbms_output.put_line('C is Greater');
16  end if;
17* end;
```

```
SQL> /
Enter value for a: 20
old   8: a:=&a;
new   8: a:=20;

Enter value for b: 15
old  10: b:=&b;
new  10: b:=15;

Enter value for c: 3
old  12: c:=&c;
new  12: c:=3;

A is greater

PL/SQL procedure successfully completed.
```

6. Calculation of Net Salary

```
 1  declare
 2    ename varchar2(15);
 3    basic number;
 4    da number;
 5    hra number;
 6    pf number;
 7    netsalary number;
 8  begin
 9    ename:=&ename;
```

```
10    basic:=&basic;
11   da:=basic*(65/100);
12   hra:=basic*(15/100);
13   if(basic<3000)
14      then
15      pf:=basic*(5/100);
16   elsif(basic >=3000 and basic <=5000)
17      then
18      pf:=basic*(7/100);
19    elsif (basic>=5000 and basic<=8000)
20       then
21      pf:=basic*(8/100);
22    else
23       pf:=basic*(10/100);
24    end if;
25       netsalary:=basic+da+hra+pf;
26      dbms_output.put_line('Employee Name :  '|| ename);
27      dbms_output.put_line('Providend Fund :  '|| pf);
28     dbms_output.put_line('Net Salary     :  '|| netsalary);
29* end;
SQL> /
Enter value for ename: 'Ram'
old   9: ename:=&ename;
new   9: ename:='Ram';
Enter value for basic: 15000
old  10: basic:=&basic;
new  10: basic:=15000;

Employee Name :  Ram
Providend Fund :  1500
Net Salary     :  28500
PL/SQL procedure successfully completed.
```

7. Factorial of a Number

```
1  declare
```

```
 2     n number(2):=&dn;
 3    fact number:=1;
 4  begin
 5    for i in 1..n
 6    loop
 7      fact:=fact*i;
 8    end loop;
 9    dbms_output.put_line('Factorial of ' || n || ' is ' || fact);
10* end;

SQL> /
Enter value for dn: 5
old   2: n number(2):=&dn;
new   2: n number(2):=5;

Factorial of 5 is 120

PL/SQL procedure successfully completed.
```

8. Fibonacci Series

```
 1  declare
 2      n number(2):=&dn;
 3      a number:=-1;
 4      b number:=1;
 5      c number:=0;
 6  begin
 7    dbms_output.put_line('Fibonacci Series is : ');
 8    for i in 1..n
 9     loop
```

```
10       c:=a+b;
11       a:=b;
12       b:=c;
13      dbms_output.put_line('   '||c);
14    end loop;
15* end;
```

```
SQL> /
Enter value for dn: 5
old   2: n number(2):=&dn;
new   2: n number(2):=5;

Fibonacci Series is :
0
1
1
2
3

PL/SQL procedure successfully completed.
```

9. To find out whether the given number is a prime number or not.

```
 1  declare
 2    n number(3):=&dn;
 3    prime number:=1;
 4    r number;
 5  begin
 6    for i in 2..n-1
 7    loop
 8          r:=mod(n,i);
 9          if r=0 then
10                prime:=0;
```

```
11          end if;
12          exit when prime=0;
13    end loop;
14    if prime=1 then
15          dbms_output.put_line(n ||'is a prime number');
16    else
17          dbms_output.put_line(n ||'is not a prime number');
18    end if;
19* end;
```

```
SQL> /
Enter value for dn: 6
old   2:       n number(3):=&dn;
new   2:       n number(3):=6;
6is not a prime number

PL/SQL procedure successfully completed.

SQL> set verify off
SQL> /
Enter value for dn: 8
8is not a prime number

PL/SQL procedure successfully completed.
```

10. To generate prime numbers upto N.

```
 1  declare
 2    n number(3):=&dn;
 3    prime number;
 4    r number;
 5  begin
 6    if n=0 then raise_application_error(-20002,'input zero_error');
 7    end if;
 8    <<outer_loop>>
 9    for i in 1..n
10     loop
11          prime:=1;
12          <<inner_loop>>
13          for j in 2..i-1
14          loop
15                r:=mod(i,j);
16                if r=0 then
17                      prime:=0;
```

```
18                 end if;
19                 exit when prime=0;
20              end loop inner_loop;
21              if prime=1 then
22                   dbms_output.put_line(i);
23              end if;
24     end loop outer_loop;
25* end;
26  .
SQL> /
Enter value for dn: 9
1
2
3
5
7
PL/SQL procedure successfully completed.
SQL> /
Enter value for dn: 0
declare
*
ERROR at line 1:
ORA-20002: input zero_error
ORA-06512: at line 6
```

11.To count and print all the prime numbers between m and n.

```
 1  declare
 2       m number(3):=&dm;
 3       n number(3):=&dn;
 4       prime number;
 5       r number;
 6      pcount number:=0;
 7  begin
 8    <<outer_loop>>
 9    for i in m..n
10     loop
11          prime:=1;
12          <<inner_loop>>
13          for j in 2..i-1
14          loop
15               r:=mod(i,j);
16               if r=0 then
17                    prime:=0;
18               end if;
19               exit when prime=0;
```

```
 20          end loop inner_loop;
 21          if prime=1 then
 22                dbms_output.put_line(i);
 23                pcount:=pcount+1;
 24          end if;
 25     end loop outer_loop;
 26      if pcount=0 then
 27          dbms_output.put_line('no prime number between '||m||' and '||n);
 28      else
 29          dbms_output.put_line('total number of prime numbers between
'||m||' and '||n ||'is ' ||pcount);
 30       end if;
 31* end;
 32  .
SQL> /
Enter value for dm: 5
Enter value for dn: 16
5
7
11
13
total number of prime numbers between  5 and 16is 4
PL/SQL procedure successfully completed.
```

12. Creation of Simple PL/SQL Program which includes Declaration Section, Executable Section and Exception Handling Section.

```
SQL> desc student;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 STUNO                                     NOT NULL NUMBER(5)
 STUNAME                                            VARCHAR2(10)
 MARK1                                              NUMBER(5)
 MARK2                                              NUMBER(5)
 MARK3                                              NUMBER(5)

SQL> select *from student;

    STUNO STUNAME    MARK1    MARK2    MARK3
    ---------- ---------------    ----------    ----------  ----------
      100        imran            52          63          30
      200        bhaskar          45          60          88
      300        basha            45          66          32
      400        ashok            54          65          55
```

```
       500        priyanka        74        66        50
       600        harika          54        88        30
       700        gowtham         98        66        31
       800        shankar         66        35        88
       900        vikram          36        35        31
      1001        bhanu           68        35        32

10 rows selected.

 1  declare
 2               sno student.stuno %type:=&no;
 3               sname student.stuname %type;
 4               smark1 student.mark1 %type;
 5               smark2 student.mark2 %type;
 6               smark3 student.mark3 %type;
 7               stotal number;
 8               savg number;
 9  begin
10               select stuno,stuname,mark1,mark2,mark3
                  into sno,sname,smark1,smark2,smark3
                  from  student
11               where stuno=sno;
12               stotal:=smark1+smark2+smark3;
13               savg:=stotal/3;
14               if savg>=60 then
15                        dbms_output.put_line('first class');
16               elsif savg >=40 then
17                        dbms_output.put_line('second class');
18               else
19                        dbms_output.put_line('fail');
20               end if;
21  exception
22               when no_data_found then
23                        dbms_output.put_line('No records found');
24 end;
SQL> /
Enter value for no: 200
old   2: sno student.stuno %type:=&no;
new   2: sno student.stuno %type:=200;
first class

PL/SQL procedure successfully completed.

SQL> /
Enter value for no: 1001
```

```
old   2: sno student.stuno %type:=&no;
new   2: sno student.stuno %type:=1001;
second class

PL/SQL procedure successfully completed.
Enter value for no: 900
old   2:  sno student.stuno %type:=&no;
new   2:  sno student.stuno %type:=900;
fail

PL/SQL procedure successfully completed.
```

13. Finding Even or Odd using CASE (Selector)

```
 1  declare
 2     num number:=&dn;
 3     r number;
 4  begin
 5     r:=mod(num,2);
 6    case r
 7     when 0 then
 8          dbms_output.put_line(num||' is EVEN number');
 9     else
10          dbms_output.put_line(num||' is ODD number');
11    end case;
12* end;

SQL> /
Enter value for dn: 7
old   2: num number:=&dn;
new   2: num number:=7;
```

```
7 is ODD number

PL/SQL procedure successfully completed.

SQL> /
Enter value for dn: 2
old   2: num number:=&dn;
new   2: num number:=2;
2 is EVEN number
```

14. Finding Even or Odd using CASE (without selector)

```
 1  declare
 2      num number:=&dn;
 3      r number;
 4  begin
 5    case
 6       when mod(num,2)=0 then
 7            dbms_output.put_line(num||' is EVEN number');
 8       else
 9            dbms_output.put_line(num||' is ODD number');
10    end case;
11*  end;
SQL> /
Enter value for dn: 4
old   2:  num number:=&dn;
```

```
new   2:  num number:=4;
4 is EVEN number

PL/SQL procedure successfully completed.

SQL> /
Enter value for dn: 5
old   2:  num number:=&dn;
new   2:  num number:=5;
5is ODD number

PL/SQL procedure successfully completed.
```

15. Arithmetic Calculator using CASE

```
 1  declare
 2      a number:=&a;
 3      b number:=&b;
 4      c number;
 5      choice number:=&ch;
 6     wrong_choice exception;
 7  begin
 8    case choice
 9        when 1 then c:=a+b;
10          dbms_output.put_line('Addition of two numbers is '||c);
11        when 2 then c:=a-b;
12          dbms_output.put_line('Subtraction of two numbers is '||c);
13        when 3 then c:=a*b;
14          dbms_output.put_line('Multiplication is '||c);
15        when 4 then c:=a/b;
16          dbms_output.put_line('Division is '||c);
17        else
18           raise wrong_choice;
19    end case;
20  exception
21     when wrong_choice then
```

```
 22          dbms_output.put_line('Please enter valid choice');
 23* end;

SQL> /
Enter value for a: 1
old   2:   a number:=&a;
new   2:   a number:=1;
Enter value for b: 2
old   3:   b number:=&b;
new   3:   b number:=2;
Enter value for ch: 3
old   5:   choice number:=&ch;
new   5:   choice number:=3;

Multiplication is  2

PL/SQL procedure successfully completed.



SQL> /
Enter value for a: 4
old   2:   a number:=&a;
new   2:   a number:=4;
Enter value for b: 5
old   3:   b number:=&b;
new   3:   b number:=5;
Enter value for ch: 4
old   5:   choice number:=&ch;
new   5:   choice number:=4;

Division is 8


SQL> /
Enter value for a: 20
old   2:   a number:=&a;
new   2:   a number:=20;
Enter value for b: 30
old   3:   b number:=&b;
new   3:   b number:=30;
Enter value for ch: 5
old   5:   choice number:=&ch;
new   5:   choice number:=5;
```

Please enter valid choice

PL/SQL procedure successfully completed.

16. In-Built Exception and User Defined Exception

```
declare
        sno student.stuno %type:=&no;
        sname student.stuname %type;
        smark1 student.mark1 %type;
        smark2 student.mark2 %type;
        smark3 student.mark3 %type;
        stotal number;
        savg number;
        sres varchar(10);
        fail_exc exception;
begin
        select stuno,stuname,mark1,mark2,mark3
        into sno,sname,smark1,smark2,smark3 from  student
        where stuno=sno;
        stotal:=smark1+smark2+smark3;
        savg:=stotal/3;
        if savg>=60 then
                sres:='First Class';
        elsif savg>=50 then
```

```
20                      sres:='Second Class';
21              elsif savg>=40 then
22                      sres:='Third Class';
23              else
24                      raise fail_exc;
25              end if;
26        insert into sresult values(sno,sname,smark1,smark2,smark3,stotal,sres);
27        commit;
28    exception
29         when no_data_found then
30              dbms_output.put_line('No records found');
31         when fail_exc then
32              dbms_output.put_line('Fail');
33* end;
```

```
SQL> /
Enter value for no: 001
old   2:  sno student.stuno %type:=&no;
new   2: sno student.stuno %type:=001;

First Class
PL/SQL procedure successfully completed.

SQL> select * from sresult;

Sno   sname     smark1   smark2    smark3   stotal        sres
-----  ----------   ----------   ---------   ----------  ---------      --------
001   imran          70         85          70         225       First Class
```

17. Raise Application Error (1)

```
1  declare
 2        dbalance bank.balance%type;
 3        daccno bank.accno%type:=&ano;
 4        wamount number(5):=&wamt;
 5        temp number(5);
 6        minamount exception;
 7  begin
 8  select accno,balance into daccno,dbalance from bank where accno=daccno;
 9        temp:=dbalance-wamount;
10        if temp<500 then
11           raise minamount;
12       else
13          dbms_output.put_line('Eligible to Withdraw');
14       end if;
15  exception
16        when minamount then
17             raise_application_error(-20001,'Need Minimum Balance');
18* end;
```

```
SQL> /
Enter value for ano: 102
old   3: daccno bank.accno%type:=&ano;
new   3: daccno bank.accno%type:=102;
Enter value for wamt: 500
old   4: wamount number(5):=&wamt;
new   4: wamount number(5):=500;

Eligible to Withdraw

PL/SQL procedure successfully completed.

SQL> /
Enter value for ano: 103
old   3: daccno bank.accno%type:=&ano;
new   3: daccno bank.accno%type:=103;
Enter value for wamt: 600
old   4: wamount number(5):=&wamt;
new   4: wamount number(5):=600;
declare
*
ERROR at line 1:
ORA-20001: Need minimum balance
ORA-06512: at line 17
```

18. Raise Application Error (2)

```
 1  declare
 2          dbalance bank.balance%type;
 3          daccno bank.accno%type:=&ano;
 4          wamount number(5):=&wamt;
 5          temp number(5);
 6   begin
 7   select accno,balance into daccno,dbalance from bank where accno=daccno;
 8         temp:=dbalance-wamount;
 9         if temp<500 then
10               raise_application_error(-20001,'Need Minimum Balance');
11        else
12             dbms_output.put_line('Eligible to Withdraw');
13        end if;
14    exception
15            when no_data_found then
16                  dbms_output.put_line('Invalid Account Number');
17   end;
18   /

SQL>/
Enter value for ano: 103
old   3:  daccno bank.accno%type:=&ano;
new   3:  daccno bank.accno%type:=103;
Enter value for wamt: 600
old   4:  wamount number(5):=&wamt;
new   4:  wamount number(5):=600;
```

```
 declare
*
ERROR at line 1:
ORA-20001: Need minimum balance
ORA-06512: at line 10
```

19. Cursor

```
SQL> declare
  2          cursor c_stu is
  3              select *from stu;
  4                 stdno stu.dno%type;
  5                 stdname stu.dname%type;
  6                 stdm1 stu.dm1%type;
  7                 stdm2 stu.dm2%type;
  8                stdm3 stu.dm3%type;
  9                stdtotal number;
 10                stdavg number;
 11                stdgrade varchar(5);
 12  begin
 13        open c_stu;
 14           loop
 15              fetch c_stu into stdno,stdname,stdm1,stdm2,stdm3;
 16              exit when c_stu%notfound;
 17           stdtotal:=stdm1+stdm2+stdm3;
 18           stdavg:=stdtotal/3;
 19         If stdavg>=70 then stdgrade:='Distinction';
 20            elsif stdavg>=60 then stdgrade:='First Class';
 21            elsif stdavg>=40 then stdgrade:='Second Class';
 22            else stdgrade:='Fail';
 23         end if;
 24  insert into result
values(stdno,stdname,stdm1,stdm2,stdm3,stdtotal,stdavg,stdgrade);
 25      end loop;
 26      close c_stu;
 27      commit;
 28  end;
 29  /

PL/SQL procedure successfully completed.
```

SQL> select *from result;

STDNO	STDNAME	STDM1	STDM2	STDM3	STDTOTAL	STDAVG	STDGR
101	rakesh	89	68	75	232	77	Distinction
102	ramu	58	64	55	177	59	Second Class
103	rohit	68	95	84	247	82	Distinction
104	ravi	68	75	74	217	72	Distinction
105	rajesh	40	42	28	110	37	Fail

20. Nested Cursor

```
 1  declare
 2        id number;
 3        s number;
 4         no number;
 5         sal number;
 6         cursor c1 is select empid,sum(hours) from works
 7               group by empid;
 8         cursor c2 is select empid,salary from empl;
 9       begin
10         open c1;
11         loop
12            fetch c1 into id,s;
13            exit when c1%notfound;
14                  for r_c2 in c2
15                   loop
16                           if id=r_c2.empid then
17                               sal:=s*100;
18                update empl set salary=r_c2.salary +sal where empid=id;
19                           dbms_output.put_line('Record Successfully Updated');
20                           end if;
21                    end loop;
22          end loop;
23         close c1;
24*    end;
```

```
SQL> /
Record Successfully Updated
Record Successfully Updated
Record Successfully Updated
Record Successfully Updated
Record Successfully Updated
Record Successfully Updated
Record Successfully Updated
```

PL/SQL procedure successfully completed.

SQL> select * from empl;

EMPID	NAME	ADDRESS	SALARY	AGE
100	imran	bangalore	5000	24
101	basha	kuppam	6000	25
102	ashok	tirupathi	8000	23
103	gowtham	chittoor	7750	25
104	harika	nellore	7300	24
105	krithika	bangalore	11200	26
106	priyanka	hyderabad	6660	22
107	manoj	kuppam	7450	23
108	arun	chennai	6500	24
109	mouli	bangalore	6540	22

10 rows selected.

Updated Records

SQL> select * from empl;

EMPID	NAME	ADDRESS	SALARY	AGE
100	imran	bangalore	5000	24
101	basha	kuppam	7500	25
102	ashok	tirupathi	9500	23
103	gowtham	chittoor	8950	25
104	harika	nellore	8100	24
105	krithika	bangalore	12800	26
106	priyanka	hyderabad	8760	22
107	manoj	kuppam	8450	23
108	arun	chennai	6500	24
109	mouli	bangalore	6540	22

10 rows selected.

21. Finding Even or Odd using Case Expression

```
 1  declare
 2           num number:=&da;
 3           num_type varchar(20);
 4   begin
 5            num_type:=
 6       case
 7          when mod(num,2)=0 then num||' is Even'
 8          else num||' is Odd'
 9          end;
10          dbms_output.put_line('The Given Number  '||num_type);
11  end;
```

```
SQL> /
Enter value for da: 4
old   2:  num number:=&da;
new   2:  num number:=4;

The Given Number  4 is Even

PL/SQL procedure successfully completed.

SQL> /
Enter value for da: 3
old   2:  num number:=&da;
new   2:  num number:=3;


The Given Number  3 is Odd

PL/SQL procedure successfully completed
```

22. Displaying the Result using Case Expression

```
SQL> declare
  2                 sno student.stuno %type:=&no;
  3                 sname student.stuname %type;
  4                 smark1 student.mark1 %type;
  5                 smark2 student.mark2 %type;
  6                 smark3 student.mark3 %type;
  7                 stotal number;
  8                 savg number;
  9                 sres varchar(10);
 10                 fail_exc exception;
 11        begin
 12                 select stuno,stuname,mark1,mark2,mark3
 13                 into sno,sname,smark1,smark2,smark3 from  student
 14                 where stuno=sno;
 15                 stotal:=smark1+smark2+smark3;
 16                 savg:=stotal/3;
 17       sres:=
 18                 case
 19                 when  savg>=90 then 'HONOUR'
 20                  when savg>=70 then 'DISTINCTION'
 21                 when savg>=60 then 'FIRST'
 22       when savg>=40 then 'SECOND'
 23                 else 'NOT ELIGIBLE'
 24                 end;
 25                 insert into sresult
values(sno,sname,smark1,smark2,smark3,stotal,sres);
 26              dbms_output.put_line('Record Inserted Successfully');
 27                 commit;
 28          exception
 29                      when no_data_found then
 30                        dbms_output.put_line('No records found');
 31                      when fail_exc then
 32                        dbms_output.put_line('he/she is fail');
 33          end;
```

```
SQL> /
Enter value for no: 200
Record Inserted Successfully

PL/SQL procedure successfully completed.

SQL> select * from sresult;

    STUNO STUNAME      MARK1    MARK2     MARK3     TOTAL RESULT
---------- ---------- ---------- ---------- ---------- ---------- ----------
      200 satish            45         60         88        193 FIRST
```

23. Displaying the result using Case Statement

```
declare
        sno student.stuno %type:=&no;
        sname student.stuname %type;
        smark1 student.mark1 %type;
        smark2 student.mark2 %type;
        smark3 student.mark3 %type;
        stotal number;
        savg number;
        sres varchar(10);
        fail_exc exception;
 begin
        select stuno,stuname,mark1,mark2,mark3
          into sno,sname,smark1,smark2,smark3 from  student
           where stuno=sno;
        stotal:=smark1+smark2+smark3;
        savg:=stotal/3;
 case
      when  savg>=60 then
                sres:='FIRST';
      when savg>=50 then
                 sres:='SECOND';
      when savg>=40 then
                sres:='THIRD';
      else
                    raise fail_exc;
        end case;
    insert into sresult  values(sno,sname,smark1,smark2,smark3,stotal,sres);
       dbms_output.put_line('Record Inserted Successfully');
        commit;
 exception
            when no_data_found then
              dbms_output.put_line('No Records Found');
            when fail_exc then
              dbms_output.put_line('He/She is Fail');
 end;
```

```
SQL> /
Enter value for no: 200
old   2:                sno student.stuno %type:=&no;
new   2:                 sno student.stuno %type:=200;

Record Inserted Successfully

PL/SQL procedure successfully completed.

SQL> /
Enter value for no: 700
old   2:                sno student.stuno %type:=&no;
new   2:                 sno student.stuno %type:=700;

Record Inserted Successfully

PL/SQL procedure successfully completed.

SQL> SELECT * FROM SRESULT;

STUNO STUNAME       MARK1     MARK2     MARK3    TOTAL   RESULT
---------- ---------------      -----------     -----------     -----------    ----------
-----------
     200   satish              45          60          88          193      FIRST
     700   vishnu              98          66          31          195      FIRST
```

24. Gcd Of Two Numbers Using While

```
1 declare
 2     u number:=&du;
 3     v number:=&dv;
 4     temp number;
 5  begin
 6     if u<v then
 7          temp:=u;
 8          u:=v;
 9          v:=temp;
10  end if;
11  while v!=0
12     loop
13       temp:=mod(u,v);
14       u:=v;
15       v:=temp;
16    end loop;
17     dbms_output.put_line('The GCD of Two numbers is '||u);
18  end;
```

```
SQL> /
Enter value for du: 8
old   2: u number:=&du;
new   2: u number:=8;
Enter value for dv: 6
old   3: v number:=&dv;
new   3: v number:=6;

The GCD of Two numbers is 2

PL/SQL procedure successfully completed.
```

25. Find the Biggest of Three Numbers Using NESTED IF

```
SQL> declare
```

```
 2       a number:=&a;
 3       b number:=&b;
 4       c number:=&c;
 5  begin
 6       if a>b and a>c then
 7       dbms_output.put_line('A is Big');
 8  else
 9       if b>c and b>a then
10       dbms_output.put_line('B is Big');
11  else
12       dbms_output.put_line('C is Big');
13       end if;
14  end if;
15  end;
16  /
Enter value for a: 10
old   2: a number:=&a;
new   2: a number:=10;
Enter value for b: 20
old   3: b number:=&b;
new   3: b number:=20;
Enter value for c: 30
old   4: c number:=&c;
new   4: c number:=30;
C is Big

PL/SQL procedure successfully completed.

SQL> /
Enter value for a: 50
old   2: a number:=&a;
new   2: a number:=50;
Enter value for b: 10
old   3: b number:=&b;
new   3: b number:=10;
Enter value for c: 20
old   4: c number:=&c;
new   4: c number:=20;
A is Big

PL/SQL procedure successfully completed.
```