6:**Write a PL/ SQL to add two numbers**

declare

    a number :=5;

    b number :=6;

    c number;

begin

    c :=a+b;

    dbms\_output.put\_line('sum of' ||a|| 'and' ||b|| 'is' ||c);

end;

7:**Write a PL/ SQL to check whether the given number is even or odd**

declare

a number :=5;

b number :=6;

c number;

begin

c :=a+b;

dbms\_output.put\_line('sum of' ||a|| 'and' ||b|| 'is' ||c);

end;

8: **Write a PL/ SQL to generate natural number using various loops**

using while loop

declare

a number :=8;

i number :=1;

begin

if(a>0) then

while(i<=a)

loop

dbms\_output.put\_line(i);

i:=i+1;

end loop;

end if;

end;

using for loop

declare

a number :=8;

i number :=1;

begin

if(a>0) then

for i in 1..a

loop

dbms\_output.put\_line(i);

end loop;

end if;

end;

9:**Write a PL/ SQL to find the roots of quadratic equation**

declare

a number :=5;

b number :=6;

c number :=2;

d number ;

r1 number ;

r2 number ;

begin

d:=b\*b-4\*a\*c;

if(d>0) then

r1:=(-b+sqrt(d))/(2\*a);

r2:=(-b-sqrt(d))/(2\*a);

dbms\_output.put\_line('the roots are' ||r1|| 'and' ||r2);

else if(d=0) then

r1:=-b/(2\*a);

dbms\_output.put\_line('the roots are' ||r1|| 'and' ||r1);

else

r1:=-b/(2\*a);

r2:=sqrt(-d)/(2\*a);

dbms\_output.put\_line('the roots are' ||r1|| 'and' ||'+i'||r2||r1||'-i'||r2);

end if;

end if;

end;

10 **write a PL/ SQL to check whether the given number is prime or not**

declare

n number :=2;

i number :=1;

cnt number :=0;

begin

for i in 1..n

loop

if(mod(n,i)=0) then

cnt :=cnt+1;

end if;

end loop;

if(cnt=2) then

dbms\_output.put\_line(n ||'is prime');

else

dbms\_output.put\_line(n ||'is not prime');

end if;

end;

11**.Write a PL/ SQL to generate prime numbers upto m**

declare

m number :=10;

j number;

n number;

c number;

i number;

begin

i :=2;

for i in 1..m

loop

c :=0;

j :=1;

for j in 1..i

loop

if(mod(i,j)=0) then

c:=c+1;

end if;

end loop;

if(c=2) then

dbms\_output.put\_line(i);

end if;

end loop;

end;

12 **for a given numberWrite a PL/ SQL to generate mathematical product table**

declare

n number :=5;

i number :=1;

begin

for i in 1..10

loop

dbms\_output.put\_line(n ||'\*'||i||'='||n\*i);

end loop;

end;

13 **Write a PL/ SQL to check whether the given number is palindrome or not**

declare

n number :=65;

rem number :=0;

tot number :=0;

k number :=0;

begin

k :=n;

while(n>0)

loop

rem :=mod(n,10);

tot :=tot\*10+rem;

n :=trunc(n/10);

end loop;

if(tot=k) then

dbms\_output.put\_line(k || 'is palindrome');

else

dbms\_output.put\_line(k || 'is not palindrome');

end if;

end;

14 **Write a PL/ SQL to check whether the given string is palindrome or not**

declare

g varchar2(20);

r varchar2(20);

i number(4);

begin

g:='mam';

for i in reverse 1..length(g)

loop

r:=r||substr(g,i,1);

end loop;

dbms\_output.put\_line('reverse string is' ||r);

if r=g then

dbms\_output.put\_line('string is palindrome');

else

dbms\_output.put\_line('string is not palindrome');

end if;

end;

15 **Write a PL/ SQL to find the date birth of a given programmer**

declare

pn programmer.pname%type:='Mary';

db programmer.dob%type;

begin

select dob into db from programmer where pname=pn;

dbms\_output.put\_line('Date of birth is '||db);

exception

when no\_data\_found then

dbms\_output.put\_line('No data');

end;

16 **Write a PL/ SQL to display the names and date of birth of programmers**

declare

cursor s is select \* from programmer;

t s%rowtype;

begin

open s;

loop

fetch s into t;

exit when s%notfound;

dbms\_output.put\_line('Pname '||t.pname||' '||'Date of Birth '||t.dob);

end loop;

close s;

end;

17 **Write a PL/ SQL to find the titles of projects done by a given programmer**

declare

pn software.pname%type:='Mary';

cursor s is select \* from software where pname=pn;

t s%rowtype;

begin

open s;

loop

fetch s into t;

exit when s%notfound;

dbms\_output.put\_line(t.title);

end loop;

close s;

end;

18 **Write a PL/ SQL to find the name of programmer for a given project**

declare

t software.title%type:='Read Me';

pn software.pname%type;

begin

select pname into pn from software where title=t;

dbms\_output.put\_line('Name '||pn);

exception

when no\_data\_found then

dbms\_output.put\_line('No Data');

end;

19 **Write a PL/ SQL to calculate area and perimeter of radii present in the table Radius and insert the radius, area and perimeter into another table circle**

First create radius and circle table then execute the following code:

declare

cursor s is select \* from radius;

t s%rowtype;

begin

open s;

loop

fetch s into t;

exit when s%notfound;

insert into circle values(t.radius,3.14\*t.radius\*t.radius,2\*3.14\*t.radius);

end loop;

end;

20 **Write a PL/SQL for a procedure to calculate the product two numbers**

A

create or replace procedure product(a in number,b in number)as c number;

begin

c:=a\*b;

dbms\_output.put\_line('Product of'||a||'and'||b||'is'||c);

end;

b

declare

x number:=100;

y number:=6;

begin

product(x,y);

end;

21 **Write a PL/SQL for a procedure to get the date of birth for a given programmer**

a

create or replace procedure db(p in programmer.pname%type) as

d programmer.dob%type;

begin

select dob into d from programmer where pname=p;

dbms\_output.put\_line('Dob is '||d);

exception

when no\_data\_found then

dbms\_output.put\_line('NO DATA');

end;

b

declare

x programmer.PNAME%type:='Altaf';

begin

db(x);

end;

22 **Write a PL/SQL for a function to return the sum of two numbers**

a

create or replace function add2(a in number,b in number)return number as c number;

begin

c:=a+b;

return(c);

end;

b

declare

x number:=5;

y number:=12;

r number;

begin

r:=add2(x,y);

dbms\_output.put\_line('Sum is :'||r);

end;

23 **Write a PL/SQL for a function to return the date of birth for a given programmer**

A

create or replace function getdb(p in programmer.pname%type) return date as

d programmer.dob%type;

begin

select dob into d from programmer where pname=p;

return(d);

end;

b

declare

x programmer.pname%type:='Altaf';

r programmer.dob%type;

begin

r:=getdb(x);

dbms\_output.put\_line('Date of birth is '||r);

end;

24 **Write a PL/SQL for a procedure to display the names of programmers studied in a given institute**

A

create or replace procedure getname(sp in studies.splace%type) as

cursor s is select \* from studies where splace=sp;

t s%rowtype;

begin

open s;

loop

fetch s into t;

exit when s%notfound;

dbms\_output.put\_line(t.pname);

end loop;

close s;

end;

b

declare

x studies.splace%type:='BITS';

begin

getname(x);

end;

25 **Write a PL/SQL for a function to calculate the total development cost for a given programmer using cursors**

**A**

create or replace function gettot(p in software.pname%type) return number as

r number;

cursor s is select \* from software where pname=p;

t s%rowtype;

begin

r:=0;

open s;

loop

fetch s into t;

exit when s%notfound;

r:=r+t.dcost;

end loop;

close s;

return(r);

end;

**b**

declare

res number;

p software.pname%type:='Vijaya';

begin

res:=gettot(p);

dbms\_output.put\_line('Total development cost is '||res);

end;

26 **Write a PL/SQL for a package with one procedure and one function. Procedure displays mathematical product table for a given number. Function return product of two numbers**

**A**

create or replace package my\_pack1 is

procedure product\_table(a in number);

function product(a in number,b in number)return number;

end;

**b**

create or replace package body my\_pack1 as

procedure product\_table(a in number)as

i number;

begin

i:=1;

while(i<=10)

loop

dbms\_output.put\_line(a||'\*'||i||'='||a\*i);

i:=i+1;

end loop;

end product\_table;

function product(a in number,b in number)return number as c number;

begin

c:=a\*b;

return c;

end product;

end;

**c**

declare

x number:=12;

y number:=2;

z number;

begin

my\_pack1.product\_table(x);

z:=my\_pack1.product(x,y);

dbms\_output.put\_line('Product of'||x||'and'||y||'is:'||z);

end;

27 **Write a PL/SQL for a package with one procedure and one function. Procedure displays the salary of given programmer. Function returns the name of programmer for a given project**

a

create or replace package pack1 is

procedure p1(p in programmer.pname%type);

function f1(t in software.TITLE%type)return software.PNAME%type;

end;

b

create or replace package body pack1 as

procedure p1(p in programmer.pname%type)as

s programmer.SALARY%type;

begin

select SALARY into s from programmer where PNAME=P;

dbms\_output.put\_line('Salary is :'||s);

end p1;

function f1(t in software.TITLE%type)return

software.PNAME%type as

x software.PNAME%type;

begin

select PNAME into x from software where TITLE=t;

return x;

end f1;

end;

c

declare

a programmer.PNAME%type:='Mary';

b software.TITLE%type:='Dead Lee';

r software.PNAME%type;

begin

pack1.p1(a);

r:=pack1.f1(b);

dbms\_output.put\_line('PNAME og given project :' ||r);

end;

28 **Write a PL/SQL for a package with one procedure and one function. Procedure displays the tiles of projects done in a given language. Function returns the name of institute in which the programmer studied**

A

create or replace package pack2 is

procedure p2(x in software.DEV\_D%type);

function f2(y in studies.PNAME%type)return studies.SPLACE%type;

end;

b

create or replace package body pack2 is

procedure p2(x in software.DEV\_D%type) as

cursor s is select\*from software where DEV\_D=x;

t s%rowtype;

begin

open s;

loop

fetch s into t;

exit when s%notfound;

dbms\_output.put\_line(t.TITLE);

end loop;

close s;

end p2;

function f2(y in studies.PNAME%type)return studies.SPLACE%type as

r studies.SPLACE%type;

begin

select SPLACE into r from studies where PNAME=r;

return r;

end f2;

end;

**c**

declare

a programmer.pname%type:='Anand';

b software.title%type:='Read Me';

r software.pname%type;

begin

pack1.p1(a);

r:=pack1.f1(b);

dbms\_output.put\_line('pname of given project:'||r);

end;

30 **Write a PL/SQL for a trigger to calculate the total of a student (tuple) before insert operation**

A

create table student1(rollno number(3) primary key,

sname varchar2(15),

marks1 number(3),

marks2 number(3),

total number(4));

b

create or replace trigger auto\_cal before insert on student1 for each row

declare

begin

:new.total := :new.marks1 + :new.marks2;

end;

c

select \* from student1;

31 **Write a PL/SQL for a trigger to store the details of updated salary of a programmer into another table**

A

create table prog as select pname, salary from programmer;

b

create table update\_prog

(pname varchar2(20),

old\_salary number(7,2),

new\_salary number(7,2),

dt date,

time varchar2(10));

c

create or replace trigger update\_status after update on prog for each row

begin

insert into update\_prog values(:old.pname, :old.salary, :new.salary, sysdate, substr(current\_timestamp,11,8));

end;

d

select \* from update\_prog;

32 **Write a PL/SQL for a trigger to know latest and oldest tuples in a table**

A

create table student

(roll number(3) primary key,

sname varchar2(15),

age number(3));

b

create or replace trigger age before insert on student

declare

begin

update student set age=age+1;

end;

c

select \* from student;