**Tut 9**

1

DECLARE

a number;

b number;

temp number;

BEGIN

a:=:enter\_a;

b:=:enter\_b;

DBMS\_OUTPUT.PUT\_LINE('BEFORE value of a:' ||a);

DBMS\_OUTPUT.PUT\_LINE('BEFORE value of b:' ||b);

temp:=a;

a:=b;

b:=temp;

DBMS\_OUTPUT.PUT\_LINE('AFTER value of a:' ||a);

DBMS\_OUTPUT.PUT\_LINE('AFTER value of b:' ||b);

END;

2

DECLARE

en\_rol number(10);

name varchar(10);

ph\_no number(10);

quali varchar(10);

BEGIN

en\_rol:=:enter\_enrolment;

name:=:enter\_name;

ph\_no:=:enter\_phno;

quali:=:enter\_qualification;

DBMS\_OUTPUT.PUT\_LINE('enrollment no:' ||en\_rol);

DBMS\_OUTPUT.PUT\_LINE('name:' ||name);

DBMS\_OUTPUT.PUT\_LINE('phone no:' ||ph\_no);

DBMS\_OUTPUT.PUT\_LINE('qualification:'||quali);

END;

3

declare

n number(10);

begin

n:=:enter\_year;

if(mod(n,4)=0) then

DBMS\_OUTPUT.PUT\_LINE('year is leap year');

else

DBMS\_OUTPUT.PUT\_LINE('year is not leap year');

end if;

end;

4

Declare

n number;

fac number:=1;

i number;

begin

n:=:enter\_n;

for i in 1..n

loop

fac:=fac\*i;

end loop;

dbms\_output.put\_line('factorial='||fac);

end;

5

declare

n number(10);

temp int;

remainder int;

rev int:=0;

begin

n:=:enter\_num;

temp:=n;

while n>1

loop

remainder:=mod(n,10);

rev:=rev\*10+remainder;

n:=n/10;

end loop;

DBMS\_OUTPUT.PUT\_LINE('your number'||temp);

DBMS\_OUTPUT.PUT\_LINE('reverse number'||rev);

end;

6

declare

n int;

i int:=2;

f int:=0;

begin

n:=:enter\_num;

for i in 2..n/2

loop

if mod(n,i)=0 then f:=1;

exit;

end if;

end loop;

if f=0 then DBMS\_OUTPUT.PUT\_LINE('number is prime:' ||n);

else DBMS\_OUTPUT.PUT\_LINE('number is not prime:' ||n);

end if;

end;

**tut 10:**

1-explicit

DECLARE

CURSOR ABC IS SELECT SUPPLIER\_NO,NAME FROM SUPPLIER;

A SUPPLIER.SUPPLIER\_NO%TYPE;

B SUPPLIER.NAME%TYPE;

BEGIN

OPEN ABC;

LOOP

FETCH ABC INTO A,B;

DBMS\_OUTPUT.PUT\_LINE(A||' '||B);

exit when abc%notfound;

END LOOP;

CLOSE ABC;

END;

2

declare

cursor inc is select supplier\_no,price from product;

A product.SUPPLIER\_NO%TYPE;

B product.price%TYPE;

begin

open inc;

loop

fetch inc into a,b;

if b<15000 then b:=b+b\*0.20;

else b:=b+b\*0.12;

exit when inc%notfound;

update product set price=b where supplier\_no=a;

end if;

end loop;

end;

3-implicit

declare

a customer.name%type;

b depot.location%type;

c customer.customer\_no%type;

begin

c:=:enter\_customer\_no;

select name,location into a,b from customer

inner join depot on customer.depot\_no=depot.dept\_no where customer\_no=c;

DBMS\_OUTPUT.PUT\_LINE(a||' '||b);

end;

**tut 11:**

1

create or replace procedure p1(x in number,y in char,z in number)

as

begin

if z=1 then

insert into employe values (x,y);

dbms\_output.put\_line('inserted successfully');

elsif z=2 then

update employe set name=y where emp\_no=x;

dbms\_output.put\_line('updated successfully');

elsif z=3 then

delete from employe where emp\_no=x;

dbms\_output.put\_line('deleted successfully');

else

dbms\_output.put\_line('select correct choice from 1,2,3');

end if;

end;

begin

p1(10,'shreya',1);

end;

begin

p1(10,'shreya',2);

end;

begin

p1(10,'shreya',3);

end;

2

create or replace procedure p2(x in number,y in number,z in number)

as

cursor inc is select emp\_no,salary from employe2;

a number;

b number;

begin

open inc;

loop

fetch inc into a,b;

exit when inc%notfound;

if x<b and b<y then b:=b+b\*(z\*0.01);

update employe2 set salary=b where emp\_no=a;

dbms\_output.put\_line('salary increased'||b);

end if;

end loop;

end p2;

declare

sal number;

sal1 number;

sal2 number;

begin

sal:=:increment;

sal1:=:enter\_range1;

sal2:=:enter\_range2;

p2(sal1,sal2,sal);

dbms\_output.put\_line('salary increased');

end;

3

create or replace procedure p3(x out number)

as

cursor large is select product\_no,price from product;

a number;

b number;

max1 number:=0;

begin

open large;

loop

fetch large into a,b;

exit when large%notfound;

if max1<b then max1:=b;

end if;

end loop;

x:=max1;

close large;

select product\_no,price into a,b from product where price=x;

dbms\_output.put\_line('LARGEST PRICE:');

dbms\_output.put\_line('product\_no:'|| a);

dbms\_output.put\_line('price:'|| b);

end p3;

declare

x int;

begin

p3(x);

end;

**tut 12:**

1

create or replace trigger sal\_diff

after update

on employe2

for each row

declare

s number;

begin

s:=:old.salary-:new.salary;

dbms\_output.put\_line('salary difference'||s);

end;

update employe2 set salary=50000 where emp\_no=2;

2

create or replace trigger stop\_trans

before insert or update

on employe2

for each row when(new.salary>10000)

declare

x number;

begin

raise\_application\_error(-20001,'you cant perform the transaction');

end;

update employe2 set salary=8000 where emp\_no=1;

3

create or replace trigger log\_tab

before update

on employe2

for each row

begin

insert into employe2 values(:old.emp\_no,:old.salary);

end;

insert into employe2 values(6,9999);

select \* from employe2

3a

create table emp\_log(emp\_no number,salary varchar(10),datime TIMESTAMP)

select \* from emp\_log1

create or replace trigger log

before delete or update

on employe2

for each row

begin

insert into emp\_log values(:old.emp\_no,:old.salary,SYSTIMESTAMP);

end;

update employe2 set salary=4000 where emp\_no=2;

select \* from emp\_log

3b

create table emp\_log1(emp\_no number,salary varchar(10),datime TIMESTAMP, typ varchar2(50))

create or replace trigger log2

before delete or update

on employe2

for each row

declare

a varchar2(50);

begin

IF DELETING THEN

a:='delete';

elsif UPDATING then

a:='update';

end if;

insert into emp\_log1 values(:old.emp\_no,:old.salary,SYSTIMESTAMP,a);

end;

update employe2 set salary=8000 where emp\_no=1;

delete from employe2 where emp\_no=1;

select \* from emp\_log1

SELECT SESSIONTIMEZONE, CURRENT\_TIMESTAMP FROM DUAL;

select SYSTIMESTAMP from dual