**PL-SQL**

Declare

i Number(5);

Begin

i:=&i;

If i>=30 Then

Dbms\_Output.Put\_Line('i greater than or equal to 30');

ElsIf i>=20 Then

Dbms\_Output.Put\_Line('i greater than or equal to 20 and less than 30');

ElsIf i>=10 Then

Dbms\_Output.Put\_Line('i greater than or equal to 10 and less than 20');

Else

Dbms\_Output.Put\_Line('i greater than or equal to 0 and less than 10');

End If;

End;

/

Declare

i Number(5);

Begin

i:=1;

Loop

Dbms\_Output.Put\_Line(i);

Exit When i>=10;

i:=i+1;

End Loop;

End;

/

Declare

i Number(5);

Begin

i:=1;

While i<=10 Loop

Dbms\_Output.Put\_Line(i);

i:=i+1;

End Loop;

End;

/

Begin

For i in 1..10 Loop

Dbms\_Output.Put\_Line(i);

End Loop;

End;

/

Begin

For i in Reverse 1..10 Loop

Dbms\_Output.Put\_Line(i);

End Loop;

End;

/

Declare

var\_accno Account.accno%Type;

var\_accholder Account.accholder%Type;

var\_balance Account.balance%Type;

Begin

var\_accno:=&var\_accno;

Select accholder,balance INTO var\_accholder,var\_balance

from account where accno=var\_accno;

Dbms\_Output.Put\_Line('Name: ' || var\_accholder || ' Balance: ' || var\_balance);

End;

/

Declare

var\_accno Account.accno%Type;

var\_accholder Account.accholder%Type;

var\_balance Account.balance%Type;

Begin

var\_accno:=&var\_accno;

Select accholder,balance INTO var\_accholder,var\_balance

from account where accno=var\_accno;

/\*

Using single quote in string

\*/

Dbms\_Output.Put\_Line( var\_accholder || '''s Balance is: ' || var\_balance);

End;

/

Declare

var\_accno Account.accno%Type;

var\_transtype varchar2(1);

var\_amt Account.balance%Type;

var\_balance Account.balance%Type;

Begin

var\_accno:=&var\_accno;

var\_transtype:='&var\_transtype';

var\_amt:=&var\_amt;

Select balance INTO var\_balance from account

where accno=var\_accno;

If var\_transtype='W' Then

var\_balance:=var\_balance - var\_amt;

Else

var\_balance:=var\_balance + var\_amt;

End If;

Update account set balance=var\_balance

where accno=var\_accno;

End;

/

**Function**

**Function accepting accno & returning balance:--**

Create Or Replace Function myfun(var\_accno IN Number)

Return Number IS

var\_balance Account.balance%Type;

Begin

Select balance INTO var\_balance from account

where accno=var\_accno;

Return var\_balance;

End;

/

**Calling above function through PL-SQL Block:--**

Declare

var\_accno Account.accno%Type;

var\_balance Account.balance%Type;

Begin

var\_balance:=myfun(&var\_accno);

Dbms\_Output.Put\_Line('Balance=' || var\_balance);

End;

/

**Procedures**

**Procedure which doesn’t return value:--**

Create Or Replace Procedure myproc(var\_accno IN Number,

var\_transtype IN Varchar2, var\_amt IN Number) IS

var\_balance Account.balance%Type;

Begin

Select balance INTO var\_balance from account

where accno=var\_accno;

If var\_transtype='W' Then

var\_balance:=var\_balance - var\_amt;

Else

var\_balance:=var\_balance + var\_amt;

End If;

Update account set balance=var\_balance

where accno=var\_accno;

End;

/

**Calling above procedure by using execute command:--**

exec myproc(1,'W',1000);

**Calling above procedure by PL-SQL Block:--**

Declare

var\_accno Account.accno%Type;

var\_transtype Varchar2(1);

var\_amt Account.balance%Type;

Begin

myproc(&var\_accno, '&var\_transtype', &var\_amt);

End;

/

**Procedure accepting accno & returning accholder & balance:--**

Create Or Replace Procedure myproc(var\_accno IN Number,

var\_accholder OUT Varchar2, var\_balance OUT Number) IS

Begin

Select accholder,balance INTO var\_accholder,var\_balance

from account where accno=var\_accno;

End;

/

**Calling above Procedure through PL-SQL Block:--**

Declare

var\_accno Account.accno%Type;

var\_accholder Account.accholder%Type;

var\_balance Account.balance%Type;

Begin

myproc(&var\_accno,var\_accholder,var\_balance);

Dbms\_Output.Put\_Line('Accholder: ' || var\_accholder || ' Balance:' || var\_balance);

End;

/

**Triggers**

**Example1: Trigger creating backup of account table after updation or deletion**

Create Or Replace Trigger MyTrigger

After Update Or Delete ON Account

For Each Row

Declare

var\_oper Varchar2(10);

begin

If UPDATING Then

var\_oper:='Update';

End If;

If DELETING Then

var\_oper:='Delete';

End If;

Insert into BackUp values(:OLD.accno, :OLD.accholder, :OLD.balance, var\_oper);

End;

/

**Example2:--**

**Trigger which doesn’t allow balance value to be less than 500**

Create Or Replace Trigger MyTrigger

Before Update Or Insert ON Account

For Each Row

begin

If :NEW.balance<500 Then

Raise\_Application\_Error(-20001, 'Balance less than 500');

End If;

End;

/

**Example3:**

**Trigger auto generating account no**

Create Or Replace Trigger MyTrigger

Before Insert ON Account

For Each Row

Declare

maxnum Account.accno%Type;

begin

Select max(accno) INTO maxnum from account;

maxnum:=maxnum + 1;

:NEW.accno:=maxnum;

End;

/

**Cursor**

**Example1:**

Declare

Cursor MyCursor IS

Select \* from account;

var\_accno Account.accno%Type;

var\_accholder Account.accholder%Type;

var\_balance Account.balance%Type;

Begin

Open MyCursor;

If MyCursor%ISOPEN Then

Loop

Fetch MyCursor INTO var\_accno,var\_accholder,var\_balance;

Exit When MyCursor%NOTFOUND;

Dbms\_Output.Put\_Line('Acc No: ' || var\_accno || ' Holder:' || var\_accholder || ' Balance:' || var\_balance);

If var\_balance<1000 Then

Update account set balance=1000 where accno=var\_accno;

End If;

End Loop;

Else

Dbms\_Output.Put\_Line('Cursor can''t be opened');

End If;

Close MyCursor;

End;

/

**Example2:-**

**Parameterized Cursor:--**

Declare

Cursor MyCursor(accnum Number) IS

Select \* from account where accno<accnum;

var\_accno Account.accno%Type;

var\_accholder Account.accholder%Type;

var\_balance Account.balance%Type;

Begin

Open MyCursor(&accnum);

If MyCursor%ISOPEN Then

Loop

Fetch MyCursor INTO var\_accno,var\_accholder,var\_balance;

Exit When MyCursor%NOTFOUND;

Dbms\_Output.Put\_Line('Acc No: ' || var\_accno || ' Holder:' || var\_accholder || ' Balance:' || var\_balance);

End Loop;

Else

Dbms\_Output.Put\_Line('Cursor can''t be opened');

End If;

Close MyCursor;

End;

/