Procedural Language / Structured Query Language (PL/SQL)

Block Structure of the PL/SQL program

```
DECLARE
        Declare section (optional), used for declaring variables
BEGIN
        Executable section (commands)
EXCEPTION
       Error handling section (optional)
END;
/
```

To create a PL/SQL program:

1. **Start Notepad**
2. **Type the code (not case sensitive)**
3. **Save the file (create a folder in C drive e,g, PLSQL) and select the above folder**
4. **Type the filename (e.g. P1.SQL)**
5. **Select Save as Type as "All Files"**
6. **Click on Save button**

To run the above PL/SQL program

1. **Start SQL *Plus**
2. **Login as scott/tiger**
3. **Type @C:\PLSQL\P1.SQL to run the program**

PL/SQL- Lab Cycle

--1. Write a PL/SQL program to display a welcome message

```
Set serveroutput on
Declare
Begin
        Dbms_output.put_line('Welcome to PL/SQL..');
End;
/
```

To run the script

```
SQL> @c:\plsql\p1.sql
```

--2. Write a PL/SQL program to display a welcome message along with a user name

```
Set serveroutput on
Declare
   Na varchar(15);
Begin
   Na:='Sachin';
   Dbms_output.put_line('Welcome to PL/SQL '|| na);
End;
/
```

To run the script

```
SQL> @c:\plsql\p2.sql
```

--3. Write a PL/SQL program to display a welcome message after reading a user name

```
Set serveroutput on
Declare
    Na varchar(20);
Begin
    Na:='&name';
Dbms_output.put_line('Welcome to PL SQL' || na);
End;
/
```

To run the script

```
SQL> @c:\plsql\p3.sql
```

--4. To calculate total and percentage after reading the student's name and two marks

```
Set serveroutput on
Declare
    Na varchar(20);
    M1 number(3);
    M2 number(3);
    Tot number(3);
    Per number(5,2);
Begin
    Na:='&name';
    M1:=&Mark1;
    M2:=&Mark2;
    Tot := m1+m2;
    Per :=tot/200*100;
    Dbms_output.put_line('Total ='|| tot);
    Dbms_output.put_line('Percentage = ' || per);
End;
/
```

To run the script

```
SQL> @c:\plsql\p4.sql
```

--5. Modify the above program, so that the results are inserted into the respective tables.
--Rollno and name into student table and Rollno, marks, total and perc into marks table.

Start SQL*Plus and type the following commands:

```
SQL>Drop table student;
SQL>Drop table marks;
SQL>Create Table Student (Rollno number(4) primary key, Sname varchar(20) not null);
SQL>Create Table Marks (Rollno number(4), mark1 number(3), Mark2 number(3), total
        number(3),perc number(5,2));
```

```
Set serveroutput on
Declare
     Roll number(4);
     Na varchar(20);
     M1 number(3);
     M2 number(3);
     Tot number(3);
     Per number(5,2);
Begin
     Roll := &rollno;
     Na:='&name';
     M1:=&Mark1;
     M2:=&Mark2;
     Tot := m1+m2;
     Per :=tot/200*100;
     Insert into student values(roll, na);
     Insert into marks values(roll,m1,m2,tot,per);
    Commit;
End;
/
Select * from student;
Select * from marks;
```

To run the script

```
SQL> @c:\plsql\p5.sql
```

--6. Write a PL/SQL script to update the NETSALARY
--Read values for empno, name and basic
--Calculate hra=30%*basic, da=20%*basic and pf=5%*basic, NETSAL=basic+hra+da-pf
--Insert all the values and save the changes to the table.

Start SQL*Plus and type the following commands:

```
SQL> Create table empl
     (empno number(4) primary key,
      Ename varchar(20),
      Basic number(9,2),
      Hra number(9,2),
      Da number(9,2),
      Pf number(9,2),
      Netsal number(9,2));
SQL> describe empl;
```

```
Declare
   Vempno empl.empno%TYPE;
   Vname  empl.ename%TYPE;
   Vbasic empl.basic%TYPE;
   Vhra empl.hra%TYPE;
   Vda  empl.da%TYPE;
   Vpf  empl.pf%TYPE;
   Vnetsal empl.netsal%TYPE;
Begin
   vempno := &empno;
   vename :='&name';
   vbasic :=&basic;
   vhra:=vbasic * 30/100;
   vda :=vbasic * 20/100;
   vpf :=vbasic * 5/100;
   vnetsal := vbasic + vhra + vda – vpf;
   insert into empl values(vempno,vename,vbasic,vhra,vda,vpf,vnetsal);
   commit;
End;
/
select * from empl;
```

To run the script

```
SQL> @c:\plsql\p6.sql
```

--7. Write a PL/SQL script to update the item details

Start SQL*Plus and type the following commands:

```
SQL> drop table STOCK;
SQL> create table STOCK (code number(4), name varchar(20),  Qty number(4), price number
(9,2));
SQL> describe stock;
SQL> insert into stock values(100,'Pen',200,10);
SQL> insert into stock values(101,'Pencil',500,5);
SQL>Commit;
```

```
--P7.SQL
Select * from stock;
Declare
        Vcode number(4):=&code;
        Vqty number(4) :=&qty;
Begin
        Update stock set qty=qty+vqty where code=vcode;
        Commit;
End;
/
Select * from stock;
```

To run the script

```
SQL> @c:\plsql\p7.sql
```

--8. Write a program to delete an item based on the itemcode from the STOK table.

```
Select * from STOCK;
Declare
        Vcode number(4):=&code;
Begin
        Delete from stock where code = vcode;
        Commit;
End;
/
Select * from STOCK;
To run the script
SQL> @c:\plsql\p8.sql
```

using IF command in PL/SQL

Simple IF command:

```
If(condition) then
      Statements...
Else
         Statements...
End if;
```

--9. To read a number and to check, whether it is positive or negative?

```
Set serveroutput on
Declare
    N number(3) := &n;
Begin
    If (n>0) then
       Dbms_output.put_line('Positive..');
    Else
           Dbms_output.put_line('Negative..');
    End if;
End;
/

To run the script
SQL> @c:\plsql\p9.sql
```

Multiple IF command:

```
If(condition1) then
      Statements...
elsif (condition2)
         Statements...
else
         Statements...
End if;
```

--10. To read a number and to check, whether it is positive, negative or Zero?

```
Set serveroutput on
Declare
     N number(3) := &n;
Begin
     If (n>0) then
        Dbms_output.put_line(‘Positive..’);
     Elsif (n<0) then
           Dbms_output.put_line(‘Negative..’);
     Else
          Dbms_output.put_line(‘Zero..’);
     End if;
End;
/
```

To run the script

```
SQL> @c:\plsql\p10.sql
```

--11. To read two numbers and to print the larger number?

```
Set serveroutput on
Declare
     A number(3) := &a;
     B number(3) := &b;
Begin
     If (a > b) then
        Dbms_output.put_line(‘Larger number is  = ‘||a);
     Else
        Dbms_output.put_line(‘Larger number is  = ‘||b);
     End if;
End;
/
```

To run the script

```
SQL> @c:\plsql\p11.sql
```

12. Write a PL/SQL program to read three numbers and to print the largest number?

```
Set serveroutput on
Declare
     A number(3) := &a;
     B number(3) := &b;
     C number(3) := &c;
Begin
     If (a > b) and (a > c) then
        Dbms_output.put_line(‘Largest = ‘||a);
     Elsif (b>c) then
           Dbms_output.put_line(‘Largest = ‘||b);
     Else
        Dbms_output.put_line(‘Largest = ‘||c);
     End if;
End;
/
```

To run the script
SQL> @c:\plsql\p12.sql

--13. To read an item number and to print the item name and price (STOCK table)

```
Select * from STOCK;
Declare
        Vcode number(4):=&code;
        Vname   varchar(20);
        Vprice     number(9,2);
Begin
        Select name,price into vname,vprice from stock where code=vcode;
        Dbms_output.put_line('Name =  '||vname);
        Dbms_output.put_line('Price=  '||vprice);
End;
/
```

To run the script
SQL> @c:\plsql\p13.sql

--14. Bank Transaction (To withdraw amount if sufficient balance is there in the account)
Start SQL*Plus and type the following commands:

```
SQL> drop table bank;
SQL> create table bank (acno number(4) primary key, name varchar(20),balance number(9,2));
SQL> insert into bank values(101,'Scott',4500);
SQL> insert into bank values(102,'Blake',6500);
SQL> insert into bank values(103,'Smith',7500);
SQL> Commit;
SQL> select * from bank;
```

```
--Program
Set serveroutput on
Select * from bank;
Declare
        Vacno number(4):=&acno;
        Vbalance number(9,2);
        Vamount number(9,2):=&amount;
Begin
        Select balance into vbalance from bank where acno=vacno;
        If(vamount < vbalance) then
            Update bank set balance=balance-vamount where acno=vacno;
            Commit;
        Else
            Dbms_output.put_line('Insufficient Fund');
End;
/
Select * from bank;
```

To run the script
SQL> @c:\plsql\p14.sql

--15. Program to Deposit an amount into the above bank account

```
Set serveroutput on
Select * from bank;
Declare
        Vacno number(4):=&acno;
        Vbalance number(9,2);
        Vamount number(9,2):=&amount;
Begin
        Select balance into vbalance from bank where acno=vacno;
        If(vbalance>0) then
            Update bank set balance=balance+vamount where acno=vacno;
            Commit;
        Else
            Dbms_output.put_line('Account not active');
End;
/
Select * from bank;
```

To run the script

```
SQL> @c:\plsql\p15.sql
```