PL/SQL

OBJECTIVES:

- Introduction to PL/SQL
- Datatypes and their usage
- Control Structures
- Concept of Error Handling

A PL/SQL block can contain DML and TCL statements. A PL/SQL block can also contain any number of SQL statements integrated with flow of control statements. Using PL/SQL we can also trap runtime errors.

```
        E:\batch\PL/SQL
SQL> start e:\batch\PL/SQL
PL/SQL procedure successfully completed.
```

A PL/SQL block can be divided into three parts , namely, a declarative part, an executable part and an exception handling part.

~~~
Structure of PL/SQL
``````````````````````````````
DECLARE
declarations
BEGIN
executable statements
EXCEPTION
handlers
END;
~~~

Objects can be declared in the declarative part, which can be used in the executable part for further manipulations. All procedural statements are included in between the BEGIN and END statements. Errors that occur during execution are dealt in the exception handling part.

```
PL/SQL:1
        declare
        aaa number(5);
begin
        select qty_on_hand into aaa from masteritem where itemno=1;
 update masteritem set qty_on_hand = aaa+10 where itemno =1;
end;
Conditonal Control In PL/SQL:
        if <condition> then
               <action>
        elsIf <condition> Then
              <action>
        else
              <action>
        end if;
P2.SQL
        Declare
```

```
                aaa number(5);
  begin
    select qty_on_hand into aaa from masteritem where itemno =1;
 if aaa>1000 then
    update masteritem set qty_on_hand = aaa+20 where itemno =1;
 else
    update masteritem set qty_on_hand = aaa+10 where itemno =1;
end if
end;
```

P3.SQL

```
        declare
        aaa number(5);
begin
        select qty_on_hand into aaa from masteritem where itemno=1;
if aaa>1000
then
update masteritem set qty_on_hand = aaa+20 where itemno=1;
elsif aaa=1000
then
update masteritem set qty_on_hand = aaa+15 where itemno=1;
else
update masteritem set qty_on_hand = aaa+10 where itemno =1;
end if;
end;
```

P4.SQL:

Intializing Variables

```
        declare
         aaa number(5);
         bbb number(5):=1;
begin
select qty_on_hand into aaa from masteritem where itemno=bbb;
if aaa>1000 then
 update masteritem set qty_on_hand = aaa+1 where itemno = bbb;
else
 update masteritem set qty_on_hand = aaa+10 where itemno = bbb;
end if;
end;
/
```

p5.SQL

Dynamic Inialization

```
declare
 aaa number(5);
  bbb number(5);
  t number(5);
begin
 t = &bbb;
select qty_on_hand into aaa from masteritem where itemno = t;
if aaa>1000 then
```

```
update masteritem set qty_on_hand = aaa+20 where item no = t;
end if;
end;
```

P6.SQL

        Dynamic Data Types

```
declare
bbb number(5):=1;
aaa masteritem.itemno%type;
begin
 select qty_on_hand into aaa from master item where itemno=bbb;
if aaa>1000 then
update masteritem set qty_on_hand = aaa+20 where item no= bbb;
end if;
end;
```

P7.SQL

```
declare
aaa masteritem%rowtype;
begin
    select  qty_on_hand into aaa from masteritem where itemno=3;
if aaa > 1000
 then
update masteritem set qty_on_hand = aaa +20 where itemno = 3;
else
update masteritem set qty_on_hand = aaa + 10 where itemno = 3;
end if;
end;
```

```
create table tempp(val number, mesg varchar2(20));
```

P8.SQL

```
 declare
  str char(20);
begin
str:='&str';
insert into tempp values(10,str);
end;
```

For Loop:

        Syntax:

```
       for variable in [REVERSE] start..end
loop
      sequence_of_statements;
end loop;
```

P10.SQL

```
begin
for i in 1..5
loop
insert into tempp values(i,'infor');
end loop;
end;
```

P11.SQL

```
declare
aaa number:=10;
begin

for i in 1..aaa
loop
insert into tempp values(i,'infor');
end loop;
end;
/
```

P12.SQL

```
declare
aaa number:=10;
begin
for i in reverse 1..aaa
loop
dbms_output.put_line(i);
end loop;
end;
/
```

P13.SQL

```
 While loop
declare
 i number:=1;
begin
 while i<10
loop
if i=5
then
goto myproc;
end if;
dbms_output.put_line(i);
i:=i+1;
end loop;
<<myproc>>
dbms_output.put_line('myprog');
end;
/
```

p14:

```
begin
insert into tempp values(20,'before savepoint');
savepoint sp;
insert into tempp values(100, 'after savepoint');
 rollback to savepoint sp;
end;
/
```

p15

Loop & Lables:

```
/* set serveroutput on*/
begin
<<mainloop>>
for i in 1..10
loop
<<subloop>>
for j in 1..10
loop
dbms_output.put_line(j);
exit mainloop when j=5;
end loop subloop;
end loop mainloop;
end ;
/
```

p16

```
declare
aaa number :=10;
begin
insert into tempp values(aaa,'rollback');
rollback;
insert into tempp values(aaa+90,'commit');
commit;
end;
/
```