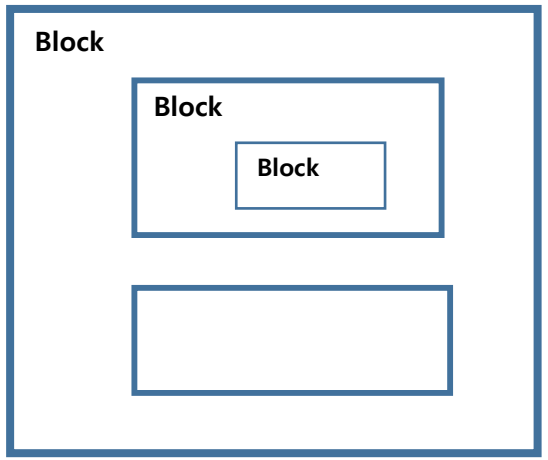
#Section1 기본문법

(기본구조, 제어문, 반복문 , Collections and Records)

PL/SQL 기본구조

PL/SQL block [익명 블록]

```
DECLARE
    -- 변수선언 , 서브 프로그램
BEGIN
    -- 실행구문
    dbms_output.put_line('Hello PL/SQL');
EXCEPTION when others then
     -- 예외처리
    null;
END;
```



익명 블록에 이름을 붙여서 사용

stored

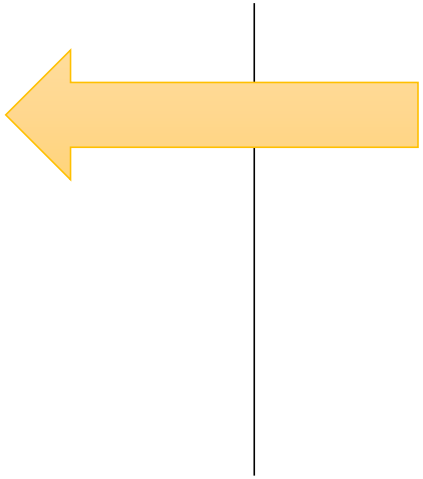
- FUNCTION
- PROCEDURE
- PACKAGE
- TRIGGER

변수선언

```
DECLARE
      -- 변수선언
      -- name varchar2(100);
      -- name sample.name%type;
       r_sample sample%rowtype;

BEGIN
     name := '홍길동';
     -- r_sample.name := '홍길순';

      -- 실행구문
      dbms_output.put_line('Hello '||name);
      -- dbms_output.put_line('Hello '||r_sample.name);
END;
```



```
create table sample(
    name varchar2(100),
     age number(10),
     reg_date date
 );


insert into sample(name, age)
values('홍길동',20);
insert into sample(name, age)
values('홍길순',30);
commit;
```

Table 3-1 Data Types with Different Maximum Sizes in PL/SQL and SQL

Data Type	Maximum Size in PL/SQL	Maximum Size in SQL
CHAR[Foot 1]	32,767 bytes	2,000 bytes
NCHAR[Foot 1]	32,767 bytes	2,000 bytes
RAW[Foot 1]	32,767 bytes	2,000 bytes[Foot 2]
VARCHAR2[Foot 1]	32,767 bytes	4,000 bytes[Foot 2]

https://docs.oracle.com/en/database/oracle/oracle-database/19/lnpls/plsql-data-types.html#GUID-C3B938C9-7B0B-4AAC-8323-FEB2ED0225D0

제어문 IF

```
IF  조건 Then
    dbms_output.put_line();

ELSIF 조건 Then
    dbms_output.put_line();

ELSE
    dbms_output.put_line();

END IF;
```

```
DECLARE
    --변수선언, 서브프로그램
    name varchar2(100):='홍길동';
    age number(10):=20;

BEGIN
    --실행구문
    name := '홍길동';
    age :=20;
    dbms_output.put_line('Hello '|| name|| to_char(age));

    IF age <13 Then
        dbms_output.put_line('초등학생');
    ELSIF age <16 Then
        dbms_output.put_line('중학생');
    ELSE
     dbms_output.put_line('고등학생이상');
    END IF;

END;
```

제어문 Case

```
CASE    selector(변수)
WHEN   selector_value_1 THEN
            statements_1
WHEN selector_value_2 THEN
            statements_2
WHEN selector_value_n THEN
            statements_n
ELSE
            else_statements
END CASE;
```

```
DECLARE
    --변수선언, 서브프로그램
    name varchar2(100):='홍길동';
    age number(10):=20;

BEGIN
    --실행구문
    name := '홍길동';
    age :=20;
    dbms_output.put_line('Hello '|| name|| to_char(age));


    Case age
    When 13 Then
        dbms_output.put_line('초등학생');
    When 16 Then
        dbms_output.put_line('중학생');
    When 20 Then
        dbms_output.put_line('대학생');
    ELSE
     dbms_output.put_line('고등학생이상');
    END Case;

END;
```

반복문 Loop

LOOP

statements

IF 조건 Then

statements

-- continue

Else

statements

exit

End IF;

END LOOP

```
DECLARE
    --변수선언, 서브프로그램
    name varchar2(100):='홍길동';
    age number(10):=20;
    x number := 0;

BEGIN
    --실행구문
    name := '홍길동';
    age :=20;
    dbms_output.put_line('Hello '|| name|| to_char(age));

   Loop
    x := x+1;
    IF x < age Then
        dbms_output.put_line('x 카운트 '||x);
        --continue;
    Else
        dbms_output.put_line('x 카운트 '||x);
        --Exit;
    End If;
        dbms_output.put_line('마지막 Loop Line');
        --Exit;
   End Loop;

END;
```

반복문 For Loop

```
FOR i IN 1..3
LOOP
    DBMS_OUTPUT.PUT_LINE (i);
END LOOP;
```

```
DECLARE
    --변수선언, 서브프로그램
    name varchar2(100):='홍길동';
    age number(10):=20;
    x number :=0;

BEGIN
    --실행구문
    name := '홍길동';
    age :=20;
    dbms_output.put_line('Hello '|| name|| to_char(age));

   For x in 1..100
   Loop
      IF x < age Then
        dbms_output.put_line('x 카운트 '||x);
        continue;
      Else
        dbms_output.put_line('x 카운트 '||x);
        Exit;
      End If;
        dbms_output.put_line('마지막 Loop Line');
   End Loop;

END;
```

반복문 While Loop

WHILE *condition*

LOOP *statements*

END LOOP

LOOP *statements*

EXIT WHEN *condition*;

END LOOP;

```
DECLARE
    --변수선언, 서브프로그램
    name varchar2(100):='홍길동';
    age number(10):=20;
    x number :=0;
BEGIN
    --실행구문
    name := '홍길동';
    age :=20;
    dbms_output.put_line('Hello '|| name|| to_char(age));

    While x<age
    Loop
        x := x+1;
        dbms_output.put_line('x Count'||x);
    End Loop;

    x:=0;

    Loop
        x := x+1;
        dbms_output.put_line('x Count'||x);
    Exit When x=age;
    End Loop;

END;
```

반복문 커서 FOR 루프

FOR 커서명 IN (DB Table Select)

LOOP

statements;

END LOOP;

```
BEGIN
  FOR fc IN (SELECT * FROM TAB)
    LOOP
      DBMS_OUTPUT.PUT_LINE(fc.tname);
    END LOOP;
END;
```

GOTO 구문

GOTO label

<<label>>

```
BEGIN
  FOR fc IN (SELECT * FROM TAB)
    LOOP
        DBMS_OUTPUT.PUT_LINE(fc.tname);

        IF fc.tname='COFFEE_MENU' Then
            goto last_mission;
        End If;

    END LOOP;

    <<last_mission>>
      DBMS_OUTPUT.PUT_LINE('Goto move');
END;
```

NULL Statement

The NULL statement only passes control to the next statement. Some languages refer to such an instruction as a no-op (no operation).

NULL

```
BEGIN
  FOR fc IN (SELECT * FROM TAB)
    LOOP
        DBMS_OUTPUT.PUT_LINE(fc.tname);

        IF fc.tname='COFFEE_MENU' Then
            goto last_mission;
        Else
            null;
        End If;

    END LOOP;

    <<last_mission>>
      DBMS_OUTPUT.PUT_LINE('Goto move');
END;
```