# PL/SQL

Procedural Language extension to SQL의 약자로 오라클 자체에 내장되어 있는 절차적 언어 SQL의 단점을 보완하여 SQL문장 내에서 변수의 정의, 조건처리, 반복처리 등 지원

#### ✓ 구조

구조	설명
DECLARE SECTION	DECLARE로 시작
(선언부)	변수나 상수를 선언하는 부분
EXECUTABLE SECTION	BEGIN으로 시작
(실행부)	제어문, 반복문, 함수 정의 등 로직 기술
EXCEPTION SECTION	EXCEPTION으로 시작
(예외처리부)	예외사항 발생 시 해결하기 위한 문장 기술

#### √ 예시

# SET SERVEROUTPUT ON; \* 프로시저를 사용하여 출력하는 내용을 화면에 보여주도록 설정하는 환경변수로 기본 값은 OFF여서 ON으로 변경 BEGIN DBMS\_OUTPUT.PUT\_LINE('HELLO WORLD'); END; / \* PUT\_LINE이라는 프로시저를 이용하여 출력(DBMS\_OTUPUT패키지에 속해있음)

#### ✓ 타입 변수 선언 : 변수의 선언과 초기화, 변수 값 출력

```
DECLARE
       EMP_ID NUMBER;
       EMP_NAME VARCHAR2(30);
                                           PL/SQL 프로시저가 성공적으로 완료되었습니다.
BEGIN
       EMP_ID := 888;
                                           EMP ID: 888
       EMP_NAME := '배장남';
                                           EMP_NAME : 배장남
       DBMS_OUTPUT.PUT_LINE('EMP_ID : ' || EMP_ID);
       DBMS_OUTPUT.PUT_LINE('EMP_NAME : ' || EMP_NAME);
END;
```

# PL/SQL

#### ✓ 타입 변수 선언 : 레퍼런스 변수의 선언과 초기화, 변수 값 출력

#### **DECLARE** 대체 변수 입력 EMP\_ID EMPLOYEE.EMP\_ID%TYPE; EMPLID에 대한 값 입력:: EMP\_NAME EMPLOYEE.EMP\_NAME%TYPE; **BEGIN** 확인 **SELECT** EMP\_ID, EMP\_NAME PL/SQL 프로시저가 성공적으로 완료되었습니다. **INTO** EMP\_ID, EMP\_NAME FROM EMPLOYEE EMP ID : 214 EMP NAME : 방명수 **WHERE** EMP\_ID = '&EMP\_ID'; DBMS\_OUTPUT\_LINE('EMP\_ID : ' || EMP\_ID); DBMS\_OUTPUT.PUT\_LINE('EMP\_NAME : ' || EMP\_NAME); END;

#### ✓ 타입 변수 선언 : 한 행에 대한 ROWTYPE변수의 선언과 초기화, 값 출력

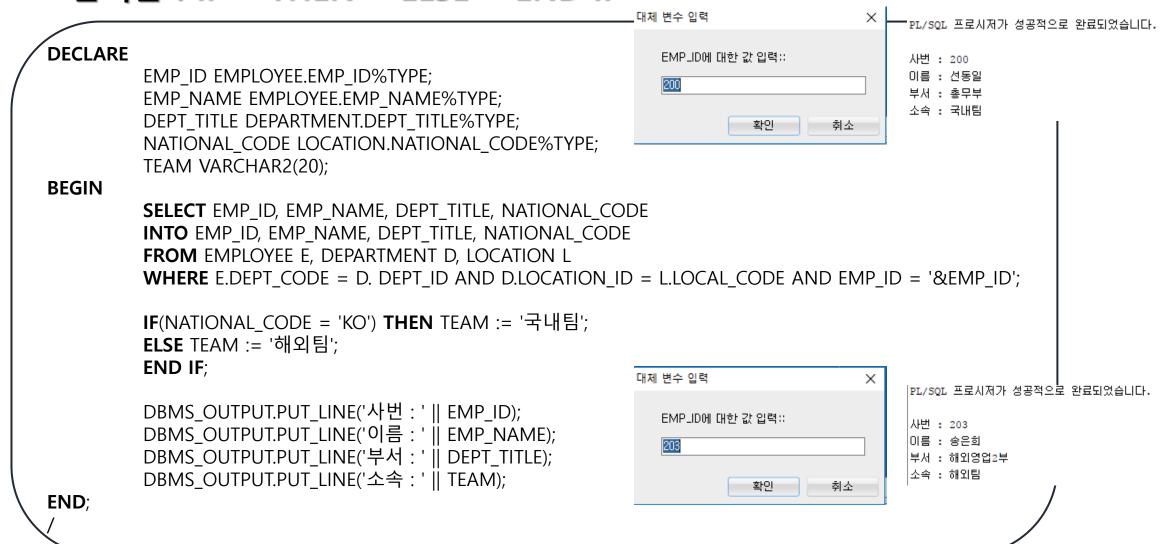
```
DECLARE
        E EMPLOYEE%ROWTYPE;
                                                                  대체 변수 입력
BEGIN
                                                                     EMPLID에 대한 값 입력::
        SELECT * INTO E
        FROM EMPLOYEE
                                                                              확인
                                                                                     취소
        WHERE EMP_ID = '&EMP_ID';
        DBMS_OUTPUT.PUT_LINE('EMP_ID : ' || E.EMP_ID);
                                                               PL/SQL 프로시저가 성공적으로 완료되었습니다.
        DBMS_OUTPUT_LINE('EMP_NAME : ' || E.EMP_NAME);
                                                               EMP ID: 200
        DBMS_OUTPUT.PUT_LINE('EMP_NO : ' || E.EMP_NO);
                                                               EMP NAME : 선동일
                                                               EMP NAME : 621235-1985634
        DBMS_OUTPUT_LINE('SALARY : ' || E.SALARY);
                                                               EMP NAME : 8000000
END;
```

#### ✓ 선택문 : IF ~ THEN ~ END IF

```
DECLARE
                                                                                        EMPLID에 대한 값 입력::
         EMP ID EMPLOYEE.EMP ID%TYPE;
         EMP NAME EMPLOYEE.EMP NAME%TYPE;
         SALARY EMPLOYEE.SALARY%TYPE;
                                                                                                  확인
                                                                                                           취소
         BONUS EMPLOYEE.BONUS%TYPE;
BEGIN
                                                                                      PL/SQL 프로시저가 성공적으로 완료되었습니다.
         SELECT EMP ID, EMP NAME, SALARY, NVL(BONUS, 0)
         INTO EMP_ID, EMP_NAME, SALARY, BONUS
                                                                                      사번 : 200
                                                                                      이름 : 선동일
         FROM EMPLOYEE
                                                                                      급여 : 8000000
         WHERE EMP ID = '&EMP ID';
                                                                                      보너스율 : 30%
                                                                                      대체 변수 입력
         DBMS_OUTPUT.PUT_LINE('사번:' || EMP_ID);
         DBMS_OUTPUT.PUT_LINE('이름:' || EMP_NAME);
                                                                                        EMPLID에 대한 값 입력::
         DBMS OUTPUT.PUT LINE('급여:' | SALARY);
         IF(BONUS = 0)
                                                                                                           취소
                   THEN DBMS_OUTPUT.PUT_LINE('보너스를 지급받지 않는 사원입니다.');
         END IF:
                                                                                      PL/SQL 프로시저가 성공적으로 완료되었습니다.
         DBMS_OUTPUT.PUT_LINE('보너스율: ' || BONUS * 100 || '%');
                                                                                      사번 : 201
                                                                                      이름 : 송종기
END;
                                                                                      급여 : 6000000
                                                                                      보너스를 지급받지 않는 사원입니다.
                                                                                      보너스율 : 0%
```

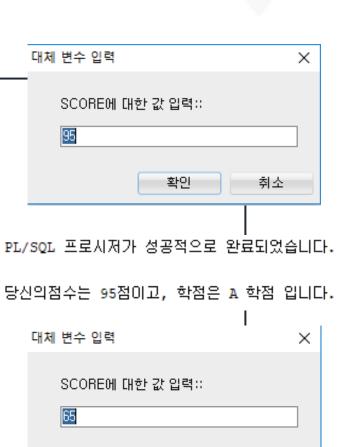
# PL/SQL

#### ✓ 선택문 : IF ~ THEN ~ ELSE ~ END IF



✓ 선택문 : IF ~ THEN ~ ELSIF ~ ELSE ~ END IF

```
DECLARE
       SCORE INT;
       GRADE VARCHAR2(2);
BEGIN
       SCORE := '&SCORE';
       IF SCORE >= 90 THEN GRADE := 'A';
       ELSIF SCORE >= 80 THEN GRADE := 'B';
       ELSIF SCORE >= 70 THEN GRADE := 'C';
       ELSIF SCORE >= 60 THEN GRADE := 'D';
       ELSE GRADE := 'F';
       END IF;
       DBMS_OUTPUT.PUT_LINE('당신의 점수는 ' || SCORE || '점이고,
                              학점은 ' || GRADE || '학점입니다.');
END;
```



PL/SQL 프로시저가 성공적으로 완료되었습니다.

확인

취소

당신의점수는 65점이고, 학점은 D 학점 입니다.

✓ 반복문 : BASIC LOOP

```
DECLARE
       N NUMBER := 1;
BEGIN
       LOOP
                                                   PL/SQL 프로시저가 성공적으로 완료되었습니다.
               DBMS_OUTPUT.PUT_LINE(N);
               N := N + 1;
               IF N > 5 THEN EXIT;
               END IF;
       END LOOP;
END;
```

#### ✓ 반복문 : WHILE LOOP

```
DECLARE
N NUMBER := 1;
BEGIN
WHILE N <= 5 LOOP
DBMS_OUTPUT.PUT_LINE(N);
N := N + 1;
END LOOP;
END;
/
```

✓ 반복문 : FOR LOOP

```
BEGIN
                                                   PL/SQL 프로시저가 성공적으로 완료되었습니다.
       FOR N IN 1..5 LOOP
               DBMS_OUTPUT.PUT_LINE(N);
       END LOOP;
END;
BEGIN
                                                   PL/SQL 프로시저가 성공적으로 완료되었습니다.
       FOR N IN REVERSE 1..5 LOOP
               DBMS_OUTPUT.PUT_LINE(N);
       END LOOP;
END;
```

#### ✓ 타입 변수 선언 : 테이블 타입의 변수 선언과 초기화, 변수 값 출력

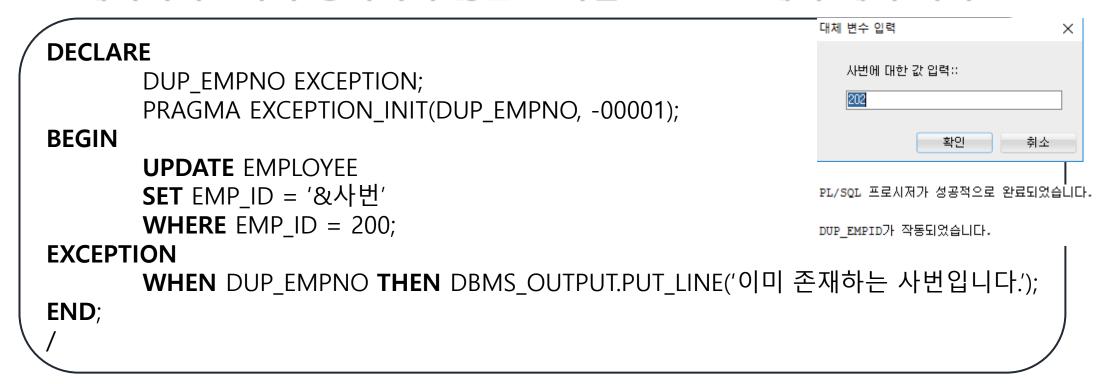
```
DECLARE
         TYPE EMP_ID_TABLE_TYPE IS TABLE OF EMPLOYEE.EMP_ID%TYPE
         INDEX BY BINARY INTEGER;
         TYPE EMP NAME TABLE TYPE IS TABLE OF EMPLOYEE.EMP NAME%TYPE
         INDEX BY BINARY INTEGER;
         EMP ID TABLE EMP ID TABLE TYPE;
         EMP NAME TABLE EMP NAME TABLE TYPE;
          I BINARY INTEGER := 0;
BEGIN
         FOR K IN (SELECT EMP ID, EMP NAME FROM EMPLOYEE) LOOP
                   | := | + 1;
                   EMP ID TABLE(I) := K.EMP ID;
                   EMP NAME TABLE(I) := K.EMP NAME;
         END LOOP:
         FOR J IN 1...I LOOP
                   DBMS OUTPUT.PUT LINE('EMP ID : ' || EMP ID TABLE(J) || ', EMP NAME : ' ||
EMP_NAME_TABLE(J));
         END LOOP:
END;
```

PL/SQL 프로시저가 성공적으로 완료되었습니다. EMP\_ID : 200, EMP\_NAME : 선동일 EMP\_ID : 201, EMP\_NAME : 송종기 EMP\_ID : 202, EMP\_NAME : 노옹철 EMP\_ID : 203, EMP\_NAME : 송은희 EMP\_ID : 204, EMP\_NAME : 유재식 EMP ID : 205, EMP NAME : 정중하 EMP ID : 206, EMP NAME : 박나라 EMP ID : 207, EMP NAME : 하이유 EMP\_ID : 208, EMP\_NAME : 김해술 EMP ID : 209, EMP NAME : 심봉선 EMP ID : 210, EMP NAME : 윤은해 EMP ID : 211, EMP NAME : 전형돈 EMP\_ID : 212, EMP\_NAME : 장쯔위 EMP\_ID : 213, EMP\_NAME : 하동운 EMP\_ID : 214, EMP\_NAME : 방명수 EMP\_ID : 215, EMP\_NAME : 대북혼 EMP ID : 216, EMP\_NAME : 차태연 EMP\_ID : 217, EMP\_NAME : 전지연 EMP\_ID : 218, EMP\_NAME : 이오리 EMP\_ID : 219, EMP\_NAME : 임시환 EMP\_ID : 220, EMP\_NAME : 이중석 EMP\_ID : 221, EMP\_NAME : 유하진 EMP\_ID : 222, EMP\_NAME : 이태림 EMP ID : 900, EMP NAME : 장채현

✓ 타입 변수 선언 : 레코드 타입의 변수 선언과 초기화, 변수 값 출력

```
DECLARE
         TYPE EMP RECORD TYPE IS RECORD (
                  EMP ID EMPLOYEE.EMP ID%TYPE,
                                                                     대체 변수 입력
                  EMP NAME EMPLOYEE.EMP NAME%TYPE,
                  DEPT_TITLE DEPARTMENT.DEPT_TITLE%TITLE,
                                                                        EMP_NAME에 대한 값 입력::
                  JOB NAME JOB.JOB NAME%TYPE
                                                                        노옹철
                                                                                  확인
                                                                                          취소
         EMP RECORD EMP RECORD TYPE;
BEGIN
         SELECT EMP ID, EMP NAME, DEPT TITLE, JOB NAME INTO EMP RECORD
         FROM EMPLOYEE E, DEPARTMENT D, JOB J
         WHERE E.DEPT CODE = D.DEPT ID
                                                                     PL/SQL 프로시저가 성공적으로 완료되었습니다.
                AND EJOB CODE = JJOB CODE
                                                                     사번 : 202
                AND EMP NAME = '&EMP NAME';
                                                                     이름 : 노옹철
                                                                     부서 : 총무부
         DBMS_OUTPUT.PUT_LINE('사번:' || EMP_RECORD.EMP_ID);
                                                                     직급 : 부사장
         DBMS_OUTPUT.PUT_LINE('이름:' || EMP_RECORD.EMP_NAME);
         DBMS_OUTPUT.PUT_LINE('부서:' || EMP_RECORD.DEPT_TITLE);
         DBMS_OUTPUT.PUT_LINE('직급:' || EMP_RECORD.JOB_NAME);
END;
```

#### ✓ 예외처리 : 미리 정의되지 않은 오라클 SERVER 에러 예외 처리



#### ✓ 시스템 예외: 오라클에서 미리 정의한 예외

예외처리명	발생 예외 내용
NO_DATA_FOUND	조회한 결과가 없을 경우 예외 발생 시
TOO_MANY_ROWS	조회한 결과가 여러 행일 때 예외 발생 시
ZERO_DIVIDE	0으로 값을 나누려고 할 때 예외 발생 시
DUP_VAL_ON_INDEX	UNIQUE 조건에 위배되는 경우 예외 발생 시
OTHERS	예외 발생 시