

PL/SQL





PL/SQL



PL/SQL

PL/SQL

- Procedural Language extension to SQL의 약자
- 오라클 자체에 내장되어 있는 절차적 언어
- SQL의 단점을 보완하여 SQL 문장내에서 변수의 정의, 조건처리, 반복처리 등을 지원

PL/SQL 블록 문법

DECLARE

[선언부]

BEGIN

[실행부]

EXCEPTION

[예외처리부]

END;

/

- 선언부는 선택사항,
- 변수나 상수를 선언
- 실행부는 필수사항,
- 제어문, 반복문, 함수 정의 등 로직 기술
- 예외처리부는 선택사항,
- 예외사항 발생 시 해결하기 위한 문장 기술
- 블록 종료
- PL/SQL 종료 및 실행



PL/SQL

PL/SQL

- 기본설정으로 PL/SQL을 사용하여 출력하는 내용을 화면에 보여주도록 설정하는 환경변수를 설정해야 함(DEFAULT OFF)

```
SET SERVEROUTPUT ON;
```

```
BEGIN
```

```
    DBMS_OUTPUT.PUT_LINE('HELLO WORLD');
```

```
END;
```

```
/
```

```
HELLO WORLD
```

PL/SQL 프로시저가 성공적으로 완료되었습니다.



PL/SQL

PL/SQL – 변수선언 및 초기화하여 출력

DECLARE

```
TEST_NO NUMBER := 999;  
TEST_STR VARCHAR2(30);
```

BEGIN

```
TEST_STR := 'HIHIHI';  
DBMS_OUTPUT.PUT_LINE('TEST_NO : '||TEST_NO);  
DBMS_OUTPUT.PUT_LINE('TEST_STR : '||TEST_STR);
```

END;

/

```
TEST_NO : 999  
TEST_STR : HIHIHI
```

PL/SQL 프로시저가 성공적으로 완료되었습니다.



PL/SQL

PL/SQL – 조회결과를 변수에 담아서 출력

DECLARE

```
E_ID NUMBER;  
E_NAME VARCHAR2(30);
```

BEGIN

```
SELECT EMP_ID, EMP_NAME  
  INTO E_ID, E_NAME  
FROM EMPLOYEE  
WHERE EMP_ID='&EMP_ID';  
DBMS_OUTPUT.PUT_LINE('EMP_ID : '||E_ID);  
DBMS_OUTPUT.PUT_LINE('E_NAME : '||E_NAME);
```

END;

/



PL/SQL

PL/SQL – 조회결과를 변수에 담아서 출력

대체 변수 입력 ×

EMP_ID에 대한 값 입력:

```
EMP_ID : 200  
E_NAME : 선동일
```

PL/SQL 프로시저가 성공적으로 완료되었습니다.

PL/SQL

PL/SQL 변수의 종류

1. 일반(스칼라 변수) 변수
 - 기존 SQL 자료형과 유사 값을 대입(:=)하고 변경하여 사용이 가능
2. 상수
 - 일반변수와 유사하나 CONSTANT 키워드가 자료형 앞에 붙고 선언 시 값을 할당해주어야 하며 변경 불가
3. %TYPE
 - 이전에 선언된 다른 변수 또는 테이블의 컬럼 자료형에 맞추어 선언하기 위한 변수
4. %ROWTYPE
 - %TYPE과 유사하게 참조할 테이블의 컬럼데이터 타입을 자동으로 가져오나 1개의 컬럼이 아니라 여러 개의 컬럼 값을 자동으로 가져옴
5. 레코드
 - %ROWTYPE이 참조할 테이블의 컬럼 데이터 타입을 자동으로 가져오는 반면 레코드는 직접적으로 컬럼타입 지정



PL/SQL

PL/SQL – 상수

DECLARE

```
T_NAME1 CONSTANT VARCHAR2(20) := '테스트1';  
T_NAME2 VARCHAR2(30);
```

BEGIN

```
T_NAME1 := '테스트11';           -- 상수는 선언 이후 변경 불가능  
T_NAME2 := '테스트22';  
DBMS_OUTPUT.PUT_LINE('T_NAME1: '||T_NAME1);  
DBMS_OUTPUT.PUT_LINE('T_NAME1: '||T_NAME2);
```

END;

/

오류 보고 -

ORA-06550: line 5, column 2:

PLS-00363: expression 'T_NAME1' cannot be used as an assignment target

ORA-06550: line 5, column 2:

PL/SQL: Statement ignored

06550. 00000 - "line %s, column %s:\n%s"

*Cause: Usually a PL/SQL compilation error.

*Action:



PL/SQL

PL/SQL – %TYPE

DECLARE

```
E_ID EMPLOYEE.EMP_ID%TYPE;  
E_NAME EMPLOYEE.EMP_NAME%TYPE;
```

BEGIN

```
SELECT EMP_ID, EMP_NAME  
INTO E_ID, E_NAME  
FROM EMPLOYEE  
WHERE EMP_ID='&EMP_ID';  
DBMS_OUTPUT.PUT_LINE('EMP_ID : '||E_ID);  
DBMS_OUTPUT.PUT_LINE('E_NAME : '||E_NAME);
```

END;

/



PL/SQL

PL/SQL – %TYPE

대체 변수 입력 ×

EMP_ID에 대한 값 입력:

```
EMP_ID : 200  
E_NAME : 선동일
```

PL/SQL 프로시저가 성공적으로 완료되었습니다.

PL/SQL

PL/SQL – %ROWTYPE

DECLARE

```
MYROW EMPLOYEE%ROWTYPE;
```

BEGIN

```
SELECT EMP_ID, EMP_NAME, EMAIL, SALARY
```

```
INTO
```

```
MYROW.EMP_ID, MYROW.EMP_NAME,  
MYROW.EMAIL, MYROW.SALARY
```

```
FROM EMPLOYEE
```

```
WHERE EMP_ID='&EMP_ID';
```

```
DBMS_OUTPUT.PUT_LINE('ID : '||MYROW.EMP_ID);
```

```
DBMS_OUTPUT.PUT_LINE('NAME : '||MYROW.EMP_NAME);
```

```
DBMS_OUTPUT.PUT_LINE('EMAIL : '||MYROW.EMAIL);
```

```
DBMS_OUTPUT.PUT_LINE('SALARY : '||MYROW.SALARY);
```

END;

/



PL/SQL

PL/SQL – %ROWTYPE

대체 변수 입력 ×

EMP_ID에 대한 값 입력:

ID : 200

NAME : 선동일

EMAIL : sun_di@kh.or.kr

SALARY : 8000000

PL/SQL 프로시저가 성공적으로 완료되었습니다.

PL/SQL

PL/SQL – 레코드

DECLARE

```
TYPE REC IS RECORD(  
    REC_ID EMPLOYEE.EMP_ID%TYPE,  
    REC_NAME EMPLOYEE.EMP_NAME%TYPE,  
    REC_SAL EMPLOYEE.SALARY%TYPE  
);
```

```
MYREC REC;
```

BEGIN

```
SELECT EMP_ID, EMP_NAME, SALARY
```

```
INTO MYREC
```

```
FROM EMPLOYEE
```

```
WHERE EMP_ID='&EMP_ID';
```

```
DBMS_OUTPUT.PUT_LINE('ID : '||MYREC.REC_ID);
```

```
DBMS_OUTPUT.PUT_LINE('NAME : '|| MYREC.REC_NAME);
```

```
DBMS_OUTPUT.PUT_LINE('SALARY : '|| MYREC.REC_SAL);
```

END;

/



PL/SQL

PL/SQL – 레코드

대체 변수 입력 ×

EMP_ID에 대한 값 입력:

ID : 200

NAME : 선동일

SALARY : 8000000

PL/SQL 프로시저가 성공적으로 완료되었습니다.



PL/SQL 선택문



PL/SQL 선택문

PL/SQL 선택문

- PL/SQL의 모든 문장들은 기술한 순서대로 순차적으로 수행
- 문장을 선택적으로 수행하려면 선택문을 사용

PL/SQL 선택문 종류

1. IF ~ THEN ~ END IF 문 (자바의 if문)
2. IF ~ THEN ~ ELSE ~ END IF 문 (자바의 if ~ else 문)
3. IF ~ THEN ~ ELSIF ~ ELSE ~ END IF 문 (자바의 if ~ else if 문)
4. CASE 문 (자바의 switch)



PL/SQL 선택문

PL/SQL 선택문 – IF ~ THEN ~ END IF

DECLARE

```
E_ID EMPLOYEE.EMP_ID%TYPE;  
E_NAME EMPLOYEE.EMP_NAME%TYPE;  
SAL EMPLOYEE.SALARY%TYPE;  
BONUS EMPLOYEE.BONUS%TYPE;
```

BEGIN

```
SELECT EMP_ID, EMP_NAME, SALARY, NVL(BONUS, 0)  
INTO E_ID, E_NAME, SAL, BONUS  
FROM EMPLOYEE  
WHERE EMP_ID = '&EMP_ID';  
DBMS_OUTPUT.PUT_LINE('사번 : ' || E_ID);  
DBMS_OUTPUT.PUT_LINE('이름 : ' || E_NAME);  
DBMS_OUTPUT.PUT_LINE('급여 : ' || SAL);  
IF (BONUS = 0)  
THEN DBMS_OUTPUT.PUT_LINE('보너스를 받지 않는 사원입니다.');
```

END;

/



PL/SQL 선택문

PL/SQL 선택문 – IF ~ THEN ~ END IF


사번 : 200
이름 : 선동일
급여 : 8000000

PL/SQL 프로시저가 성공적으로 완료되었습니다.

사번 : 205
이름 : 정중하
급여 : 3900000

보너스를 받지 않는 사원입니다.

PL/SQL 프로시저가 성공적으로 완료되었습니다.



PL/SQL 선택문

PL/SQL 선택문 – IF ~ THEN ~ ELSE ~ END IF

DECLARE

```
E_ID EMPLOYEE.EMP_ID%TYPE;  
E_NAME EMPLOYEE.EMP_NAME%TYPE;  
SAL EMPLOYEE.SALARY%TYPE;  
J_CODE EMPLOYEE.JOB_CODE%TYPE;
```

BEGIN

```
SELECT EMP_ID, EMP_NAME, SALARY, JOB_CODE  
INTO E_ID, E_NAME, SAL, J_CODE  
FROM EMPLOYEE  
WHERE EMP_ID = '&EMP_ID';  
DBMS_OUTPUT.PUT_LINE('사번 : ' || E_ID);  
DBMS_OUTPUT.PUT_LINE('이름 : ' || E_NAME);  
DBMS_OUTPUT.PUT_LINE('급여 : ' || SAL);  
IF (J_CODE IN ('J1', 'J2'))  
THEN DBMS_OUTPUT.PUT_LINE('임원진 입니다.');
```

```
ELSE DBMS_OUTPUT.PUT_LINE('일반직원 입니다.');
```

```
END IF;
```

END;

/



PL/SQL 선택문

PL/SQL 선택문 – IF ~ THEN ~ ELSE ~ END IF

사번 : 200

이름 : 선동일

급여 : 8000000

임원진 입니다.

PL/SQL 프로시저가 성공적으로 완료되었습니다.

사번 : 211

이름 : 전형돈

급여 : 2000000

일반직원 입니다.

PL/SQL 프로시저가 성공적으로 완료되었습니다.



PL/SQL 선택문

PL/SQL 선택문 – IF ~ THEN ~ ELSIF ~ ELSE ~ END IF

DECLARE

```
E_ID EMPLOYEE.EMP_ID%TYPE;  
E_NAME EMPLOYEE.EMP_NAME%TYPE;  
SAL EMPLOYEE.SALARY%TYPE;  
SALGRADE CHAR(1);
```

BEGIN

```
SELECT EMP_ID, EMP_NAME, SALARY  
INTO E_ID, E_NAME, SAL  
FROM EMPLOYEE  
WHERE EMP_ID = '&EMP_ID';  
SAL := SAL/10000;  
IF 0 <= SAL AND SAL <= 99 THEN SALGRADE := 'F';  
ELSIF 100 <= SAL AND SAL <= 199 THEN SALGRADE := 'E';  
ELSIF 200 <= SAL AND SAL <= 299 THEN SALGRADE := 'D';  
ELSIF 300 <= SAL AND SAL <= 399 THEN SALGRADE := 'C';  
ELSIF 400 <= SAL AND SAL <= 499 THEN SALGRADE := 'B';  
ELSE SALGRADE := 'A';  
END IF;  
DBMS_OUTPUT.PUT_LINE('사번 : ' || E_ID);  
DBMS_OUTPUT.PUT_LINE('이름 : ' || E_NAME);  
DBMS_OUTPUT.PUT_LINE('급여등급 : ' || SALGRADE);
```

END;

/



PL/SQL 선택문

PL/SQL 선택문 – IF ~ THEN ~ ELSIF ~ ELSE ~ END IF

사번 : 200
이름 : 선동일
급여등급 : A

PL/SQL 프로시저가 성공적으로 완료되었습니다.

사번 : 203
이름 : 송은희
급여등급 : D

PL/SQL 프로시저가 성공적으로 완료되었습니다.

사번 : 217
이름 : 전지연
급여등급 : C

PL/SQL 프로시저가 성공적으로 완료되었습니다.

사번 : 219
이름 : 임시환
급여등급 : E

PL/SQL 프로시저가 성공적으로 완료되었습니다.



PL/SQL 선택문

PL/SQL 선택문 – CASE

DECLARE

```
E_ID EMPLOYEE.EMP_ID%TYPE;  
E_NAME EMPLOYEE.EMP_NAME%TYPE;  
SAL EMPLOYEE.SALARY%TYPE;  
SALGRADE CHAR(1);
```

BEGIN

```
SELECT EMP_ID, EMP_NAME, SALARY  
INTO E_ID, E_NAME, SAL  
FROM EMPLOYEE  
WHERE EMP_ID = '&EMP_ID';  
SAL := SAL/10000;  
CASE FLOOR(SAL/100)  
  WHEN 0 THEN SALGRADE := 'F';  
  WHEN 1 THEN SALGRADE := 'E';  
  WHEN 2 THEN SALGRADE := 'D';  
  WHEN 3 THEN SALGRADE := 'C';  
  WHEN 4 THEN SALGRADE := 'B';  
  ELSE SALGRADE := 'A';  
END CASE;  
DBMS_OUTPUT.PUT_LINE('사번 : ' || E_ID);  
DBMS_OUTPUT.PUT_LINE('이름 : ' || E_NAME);  
DBMS_OUTPUT.PUT_LINE('급여등급 : ' || SALGRADE);
```

END;

/



PL/SQL 선택문

PL/SQL 선택문 – CASE

사번 : 200
이름 : 선동일
급여등급 : A

PL/SQL 프로시저가 성공적으로 완료되었습니다.

사번 : 203
이름 : 송은희
급여등급 : D

PL/SQL 프로시저가 성공적으로 완료되었습니다.

사번 : 217
이름 : 전지연
급여등급 : C

PL/SQL 프로시저가 성공적으로 완료되었습니다.

사번 : 219
이름 : 임시환
급여등급 : E

PL/SQL 프로시저가 성공적으로 완료되었습니다.





PL/SQL 반복문



PL/SQL 반복문

PL/SQL 반복문

- 문장을 반복적으로 수행하기 위해 사용하는 구문

PL/SQL 반복 종류

1. BASIC LOOP : 조건 없이 무한 반복
2. FOR LOOP : 지정한 반복 횟수 만큼 반복
3. WHILE LOOP : 제어 조건이 TRUE인 동안만 반복



PL/SQL 반복문

PL/SQL 선택문 – BASIC LOOP

```
DECLARE
```

```
    N NUMBER := 1;          -- 초기값 1 대입
```

```
BEGIN
```

```
    LOOP                    -- 반복 시작
```

```
        DBMS_OUTPUT.PUT_LINE(N);
```

```
        N := N+1;
```

```
        IF N > 5
```

```
        THEN EXIT;
```

```
        END IF;
```

```
    END LOOP; -- 반복 끝
```

```
END;
```

```
/
```

1

2

3

4

5

PL/SQL 프로시저가 성공적으로 완료되었습니다.

PL/SQL 반복문

PL/SQL 선택문 – FOR LOOP

DECLARE

BEGIN

```
FOR N IN 1..5 LOOP                -- 카운트용 변수 N은 자동 선언
    DBMS_OUTPUT.PUT_LINE(N);
END LOOP; -- 반복 끝
```

END;

/

```
1
2
3
4
5
```

PL/SQL 프로시저가 성공적으로 완료되었습니다.

PL/SQL 반복문

PL/SQL 선택문 – FOR LOOP(REVERSE)

```
DECLARE
```

```
BEGIN
```

```
    FOR N IN REVERSE 1..5 LOOP          -- REVERSE는 감소  
        DBMS_OUTPUT.PUT_LINE(N);  
    END LOOP; -- 반복 끝
```

```
END;
```

```
/
```

```
5  
4  
3  
2  
1
```

PL/SQL 프로시저가 성공적으로 완료되었습니다.

PL/SQL 반복문

PL/SQL 선택문 – WHILE LOOP

```
DECLARE
    N NUMBER := 1;

BEGIN
    WHILE N <= 5 LOOP                -- 반복여부를 체크할 조건식
        DBMS_OUTPUT.PUT_LINE(N);
        N := N + 1;
    END LOOP; -- 반복 끝

END;
/
```

```
1
2
3
4
5
```

PL/SQL 프로시저가 성공적으로 완료되었습니다.



PL/SQL 예외처리



PL/SQL 예외처리

PL/SQL 예외처리

- JAVA의 예외처리와 동일하게 예외상황에 대한 처리
- 예외의 이름을 아는 경우와 모르는 경우에 대하여 사용방법이 다름

PL/SQL 예외처리

DECLARE

[선언 영역]

BEGIN

실행 영역

EXCEPTION

[예외처리 영역]

END;

/



PL/SQL 예외처리

PL/SQL 예외처리 – EXCEPTION 이름을 아는 경우

```
DECLARE
    USERINFO EMPLOYEE%ROWTYPE;
    N NUMBER := 0;          -- 초기값 0 대입
BEGIN
    LOOP
        SELECT *
        INTO USERINFO
        FROM EMPLOYEE WHERE EMP_ID = 200 + N;
        DBMS_OUTPUT.PUT_LINE('사번 : '||USERINFO.EMP_ID);
        DBMS_OUTPUT.PUT_LINE('이름 : '||USERINFO.EMP_NAME);
        DBMS_OUTPUT.PUT_LINE('입사일 : '||USERINFO.HIRE_DATE);
        DBMS_OUTPUT.PUT_LINE('-----');
        N := N+1;
    END LOOP;
EXCEPTION
    WHEN NO_DATA_FOUND
    THEN DBMS_OUTPUT.PUT_LINE('데이터가 없습니다.');
```

END;
/



PL/SQL 예외처리

PL/SQL 예외처리 – EXCEPTION 이름을 아는 경우

사번 : 219
이름 : 임시환
입사일 : 99/09/09

사번 : 220
이름 : 이중석
입사일 : 14/09/18

사번 : 221
이름 : 유하진
입사일 : 94/01/20

사번 : 222
이름 : 이태림
입사일 : 97/09/12

데이터가 없습니다.

PL/SQL 예외처리

PL/SQL 예외처리 – 오라클에 정의된 EXCEPTION

EXCEPTION	설명
ACCESS_INTO_NULL	초기화 되지 않은 오브젝트에 값을 할당하려고 할 때
CASE_NOT_FOUND	CASE 문장에서 ELSE 구문도 없고 WHEN절에 명시된 조건을 만족하는 것이 없는 경우
COLLECTION_IS_NULL	초기화되지 않은 중첩 테이블이나 VARRAY같은 컬렉션을 EXISTS외의 다른 메소드로 접근을 시도하는 경우
CURSOR_ALREADY_OPEN	이미 오픈 된 커서를 다시 오픈하려고 시도하는 경우
DUP_VAL_ON_INDEX	UNIQUE가 설정된 컬럼에 중복데이터를 입력할 경우
INVALID_CURSOR	허용되지 않은 커서에 접근할 경우
INVALID_NUMBER	문자형 데이터를 숫자형으로 변환할 때 제대로 된 숫자가 아닌 경우
LOGIN_DENIED	잘못된 사용자명이나 비밀번호로 접속을 시도할 때
NO_DATA_FOUND	SELECT INTO 문장의 결과로 선택된 행이 하나도 없는 경우



PL/SQL 예외처리

PL/SQL 예외처리 – EXCEPTION 이름을 모르는 경우

DECLARE

```
USERINFO EMPLOYEE%ROWTYPE;  
N NUMBER := 0;           -- 초기값 0 대입  
NO_DATA EXCEPTION;      --EXCEPTION 변수 선언  
PRAGMA EXCEPTION_INIT(NO_DATA,+100)  
-- NO_DATA_FOUND에러의 에러번호 +100
```

BEGIN

```
LOOP  
SELECT *  
INTO USERINFO  
FROM EMPLOYEE WHERE EMP_ID = 200 + N;  
DBMS_OUTPUT.PUT_LINE('사번 : '||USERINFO.EMP_ID);  
DBMS_OUTPUT.PUT_LINE('이름 : '||USERINFO.EMP_NAME);  
DBMS_OUTPUT.PUT_LINE('입사일 : '||USERINFO.HIRE_DATE);  
DBMS_OUTPUT.PUT_LINE('-----');  
N := N+1;  
END LOOP;
```

EXCEPTION

```
WHEN NO_DATA  
THEN DBMS_OUTPUT.PUT_LINE('데이터가 없습니다.');
```

END;

/



PL/SQL 예외처리

PL/SQL 예외처리 – EXCEPTION 이름을 모르는 경우

사번 : 219
이름 : 임시환
입사일 : 99/09/09

사번 : 220
이름 : 이중석
입사일 : 14/09/18

사번 : 221
이름 : 유하진
입사일 : 94/01/20

사번 : 222
이름 : 이태림
입사일 : 97/09/12

데이터가 없습니다.