SQL + (프로그래밍) 언어적 성격 => PL/SQL CF) 오라클 DB에서만 적용

+ 종류

1. Anonymous Block 익명 블록

2. Proocedure

3. Function

4. Package

5. Trigger

1. Anonymous Block 기본 형태 (4가지 요소)

1) DECLARE (선택 요소임)

변수 선언부 : 변수를 선언

2) BEGIN (필수)

실행부 : 로직을 처리

3) EXCEPTION (선택)

예외처리부 : 예외사항을 처리

4) END (필수)

종료표시부

/

DEVELOPER에서는 SET SERVEROUTPUT ON를 한번만 해주면 됨

SET SERVEROUTPUT ON

PL/SQL 프로시저가 성공적으로 완료되었습니다.

BEGIN

DBMS\_OUTPUT.PUT\_LINE('HELLO PL/SQL WORLD');

END;

/

HELLO PL/SQL WORLD

PL/SQL 프로시저가 성공적으로 완료되었습니다.

CMD에서는 매번 껐다 킬때마다 SET 해야 나옴

SQL> SET SERVEROUTPUT ON

SQL>

SQL> BEGIN

2 DBMS\_OUTPUT.PUT\_LINE('HELLO PL/SQL WORLD');

3 END;

4 /

HELLO PL/SQL WORLD

PL/SQL procedure successfully completed.

-----------------------------------------------------------

+ 변수와 상수

변수 선언

cf) 자료형 변수명 (자바 언어)

변수명 자료형 (PL/SQL)

ex)

emp\_empno1 NUMBER(10);

grade CHAR(2);

변수 선언과 동시에 초기화

emp\_empno2 NUMBER(2):=10;

> %TYPE : 참조할 테이블에 있는 컬럼의 데이터 타입을 자동으로 인식

[형식]

변수명 테이블명.컬럼명%TYPE;

ex)

VSAL EMP.SAL%TYPE;

> %ROWTYPE

: %TYPE은 하나의 값에만 적용

%ROWTYPE은 하나 이상의 값에 대해 적용

[형식]

변수명 테이블명%ROWTYPE

> 상수

emp\_emp3 CONSTANT INTEGER:=20;

: 상수는 반드시 데이터를 할당하여 초기화 해야 함. 그렇지 않으면 오류가 발생

DECLARE

-- 변수 선언부

-- 변수명 자료형

VDATA NUMBER;

BEGIN

-- 실행부

-- PL/SQL은 := 이 대입연산자, = 은 비교연산자

VDATA:=10;

DBMS\_OUTPUT.PUT\_LINE(VDATA);

END;

/

10

PL/SQL 프로시저가 성공적으로 완료되었습니다.

--문자형 변수에 Hello PL/SQL WOW 문자열을 담고 콘솔에 출력

DECLARE

WOW CHAR(20);

BEGIN

WOW:='Hello PL/SQL WOW';

DBMS\_OUTPUT.PUT\_LINE(WOW);

END;

/

Hello PL/SQL WOW

PL/SQL 프로시저가 성공적으로 완료되었습니다.

-- 7900 사원의 사번, 이름, 급여를 구해서 출력해보세요

DECLARE

VEMPNO NUMBER(4);

VENAME VARCHAR2(10);

VSAL NUMBER(7);

BEGIN

--PL/SQL은 SQL문장이 가진 절차적 성격에 프로그래밍 언어적 성격을 가미한 언어

--SQL문을 그대로 사용할 수 있음

SELECT EMPNO, ENAME, SAL INTO VEMPNO, VENAME, VSAL

FROM EMP

WHERE EMPNO = 7900;

DBMS\_OUTPUT.PUT\_LINE(VEMPNO||' '||VENAME||' '||VSAL);

END;

/

7900 JAMES 950

PL/SQL 프로시저가 성공적으로 완료되었습니다.

-- 입력을 받아서 사원번호, 사원이름, 급여 출력

SELECT EMPNO, ENAME, SAL

FROM EMP

WHERE EMPNO = &EMPNO;

ACCEPT VNO PROMPT'검색할 사번 입력:'

DECLARE

-- 치환변수 : &변수명

VEMPNO NUMBER(4):=&VNO;

VENAME VARCHAR2(20);

VSAL NUMBER(7);

BEGIN

SELECT ENAME, SAL INTO VENAME, VSAL

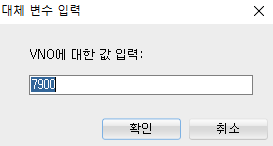
FROM EMP

WHERE EMPNO = VEMPNO;

DBMS\_OUTPUT.PUT\_LINE(VEMPNO||' '||VENAME||' '||VSAL);

END;

/



7900 JAMES 950

PL/SQL 프로시저가 성공적으로 완료되었습니다.

--------------------------------------------------------------------------------------------

SET SERVEROUT ON

-- [LOOP]

DECLARE

i integer:=0;

BEGIN

-- basic loop

LOOP -- LOOP는 조건이 없을 시 계속해서 반복

i:=i+1;

EXIT WHEN i>9;

DBMS\_OUTPUT.PUT\_LINE('3 \* ' || i || ' = ' || 3\*i);

END LOOP;

END;

/

3 \* 1 = 3

3 \* 2 = 6

3 \* 3 = 9

3 \* 4 = 12

3 \* 5 = 15

3 \* 6 = 18

3 \* 7 = 21

3 \* 8 = 24

3 \* 9 = 27

PL/SQL 프로시저가 성공적으로 완료되었습니다.

-- [FOR LOOP]

-- FOR 카운터 변수 IN 최소값.. LOOP

-- 문장 1;

-- 문장 2;

-- .

-- .

-- END LOOP;

BEGIN

for i in 2 .. 9 loop

for j in 1 .. 9 loop

DBMS\_OUTPUT.PUT\_LINE(i || ' \* ' || j || ' = ' || j\*i);

end loop;

end loop;

END;

/

결과는 아래와 같이 2~9단까지 쭉 출력됨

.

.

8 \* 7 = 56

8 \* 8 = 64

8 \* 9 = 72

9 \* 1 = 9

9 \* 2 = 18

9 \* 3 = 27

9 \* 4 = 36

9 \* 5 = 45

9 \* 6 = 54

9 \* 7 = 63

9 \* 8 = 72

9 \* 9 = 81

PL/SQL 프로시저가 성공적으로 완료되었습니다.

-- basic loop, for loop로 각각 1부터 100까지의 합을 구해보세요

DECLARE

VSUM NUMBER;

BEGIN

VSUM:=0;

FOR I IN 1..100 LOOP

VSUM:=VSUM+I;

END LOOP;

DBMS\_OUTPUT.PUT\_LINE('1부터 100 합:'||VSUM);

END;

/

DECLARE

VSUM NUMBER;

i integer:=0;

BEGIN

VSUM:=0;

LOOP

i:=i+1;

VSUM:=VSUM+i;

EXIT WHEN i>=100;

END LOOP;

DBMS\_OUTPUT.PUT\_LINE('1부터 100 합:'||VSUM);

END;

/

1부터 100 합:5050

PL/SQL 프로시저가 성공적으로 완료되었습니다.

역순(REVERSE)

BEGIN

FOR I IN REVERSE 1..10 LOOP

DBMS\_OUTPUT.PUT\_LINE(I);

END LOOP;

END;

/

10

9

8

7

6

5

4

3

2

1

PL/SQL 프로시저가 성공적으로 완료되었습니다.

-- [WHILE LOOP]

-- WHILE 조건 LOOP

-- 필요한 로직;

-- END LOOP;

DECLARE

I NUMBER:=0;

BEGIN

WHILE I<=9 LOOP

I:=I+1;

DBMS\_OUTPUT.PUT\_LINE(I);

END LOOP;

END;

/

1 2 3 4 5 6 7 8 9 10

PL/SQL 프로시저가 성공적으로 완료되었습니다.

-- WHILE LOOP를 통해 1~100합 구하기

DECLARE

I INTEGER;

VSUM INTEGER;

BEGIN

I:=0;

VSUM:=0;

WHILE I<100 LOOP

I:=I+1;

VSUM:=VSUM+I;

END LOOP;

dbms\_output.put\_line(VSUM);

END;

/

5050

PL/SQL 프로시저가 성공적으로 완료되었습니다.

SET SERVEROUTPUT ON

--[제어문]

--[IF문]

-- - 처리 조건이 한개 일 경우

-- IF 조건 THEN

-- 문장...;

-- END IF;

--

-- - 처리 조건이 두개 일 경우

-- IF 조건 THEN

-- 문장...;

-- ELSE

-- 문장...;

-- END IF;

--

-- - 처리 조건이 여러개 일 경우

-- IF 조건1 THEN

-- 문장...;

-- ELSIF 조건2 THEN

-- 문장...;

-- ...

-- ELSE

-- 문장...;

-- END IF;

ACCEPT VNO NUMBER PROMPT'당신의 성적을 입력하세요:'

SET SERVEROUTPUT ON;

DECLARE

VSCORE NUMBER:=&VNO;

BEGIN

IF VSCORE>=90 THEN

DBMS\_OUTPUT.PUT\_LINE('A학점');

ELSIF VSCORE>=80 THEN

DBMS\_OUTPUT.PUT\_LINE('B학점');

ELSIF VSCORE>=70 THEN

DBMS\_OUTPUT.PUT\_LINE('C학점');

ELSIF VSCORE>=60 THEN

DBMS\_OUTPUT.PUT\_LINE('D학점');

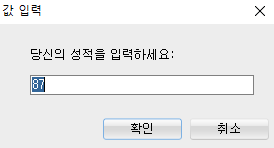
ELSE

DBMS\_OUTPUT.PUT\_LINE('F학점');

END IF;

END;

/



B학점

PL/SQL 프로시저가 성공적으로 완료되었습니다.

--[CASE 문]

ACCEPT VGRADE PROMPT'당신의 학점을 입력하세요' -- 따옴표 필수

DECLARE

--GRADE CHAR(1);

GRADE CHAR(2):=&VGRADE;

BEGIN

--GRADE:='B';

CASE GRADE

WHEN 'A' THEN

DBMS\_OUTPUT.PUT\_LINE('참 잘했어요');

WHEN 'B' THEN

DBMS\_OUTPUT.PUT\_LINE('잘했어요');

WHEN 'C' THEN

DBMS\_OUTPUT.PUT\_LINE('준수해요');

WHEN 'D' THEN

DBMS\_OUTPUT.PUT\_LINE('노력하세요');

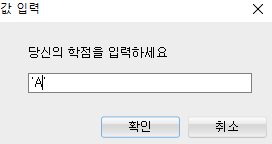
ELSE

DBMS\_OUTPUT.PUT\_LINE('돌아가세요');

END CASE;

END;

/

 문자형 넣어야 되므로 따옴표 꼭 해야 함

참 잘했어요

PL/SQL 프로시저가 성공적으로 완료되었습니다.

------------------------------------------------------

-- 사원번호가 7900인 사원의 급여 정보가 궁금

-- 이 사원의 급여가 3000 이상이면 급여를 출력

-- 급여가 3000 이하이면 급여를 미공개 - \*\*\*\*

DECLARE

VEMPNO NUMBER:=7900;

VENAME VARCHAR2(20);

VSAL NUMBER(4);

BEGIN

SELECT ENAME, SAL INTO VENAME, VSAL

FROM EMP

WHERE EMPNO = VEMPNO;

DBMS\_OUTPUT.PUT\_LINE('사번 '||'이름 '||'급여');

IF VSAL>=3000 THEN

DBMS\_OUTPUT.PUT\_LINE(VEMPNO||' '||VENAME||' '||VSAL);

ELSE

DBMS\_OUTPUT.PUT\_LINE(VEMPNO||' '||VENAME||' '||'\*\*\*\*');

END IF;

END;

/

사번 이름 급여

7900 JAMES \*\*\*\*

PL/SQL 프로시저가 성공적으로 완료되었습니다.

-----------------------------------------------------------------------------

ACCEPT VNO PROMPT'검색할 사번 입력'

DECLARE

-- VEMPNO NUMBER(5):=&VNO;

-- VEMPNO NUMBER(4):=&VNO;

-- VENAME VARCHAR2(20);

-- VSAL NUMBER(7);

-- VEMPNO EMP.EMPNO%TYPE:=&VNO;

-- VENAME EMP.ENAME%TYPE;

-- VSAL EMP.SAL%TYPE;

VEMPNO EMP.EMPNO%TYPE:=&VNO;

VEMP EMP%ROWTYPE;

BEGIN

-- SELECT EMPNO, ENAME, SAL INTO VEMPNO, VENAME, VSAL

-- SELECT \* INTO VEMPNO, VENAME, VSAL -- 일단 다 불러오고 그중 3개만 저장, 하지만 뭐가 어디에 매칭되는지 모름

SELECT \* INTO VEMP -- 그래서 1:1만 되는 TYPE 대신 전부 가져오는 ROWTYPE 씀

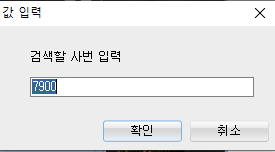
FROM EMP

WHERE EMPNO = VEMPNO;

DBMS\_OUTPUT.PUT\_LINE('=========================');

--DBMS\_OUTPUT.PUT\_LINE(VEMP.EMPNO||' '||VEMP.ENAME||' '||VEMP.SAL);

DBMS\_OUTPUT.PUT\_LINE(VEMP.EMPNO||' '||VEMP.ENAME||' '||VEMP.SAL||' '||VEMP.TEL);/



7900 JAMES 950

PL/SQL 프로시저가 성공적으로 완료되었습니다.

-- 변경1

-- 만 번 대 자리 수를 입력하면 오류가 난다.

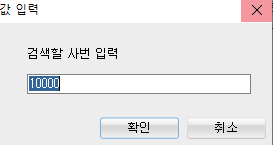
-- 현재 회사가 잘 성장해서 드디어 사원번호가 10000 번이 생김

ALTER TABLE EMP

MODIFY EMPNO NUMBER(5); -- 만 번 대 자리까지 확보

INSERT INTO EMP (EMPNO, ENAME, SAL, DEPTNO)

VALUES(10000, 'GOOTT', 500, 10); -- 하나 삽입해줌 확인용임



10000 GOOTT 500

PL/SQL 프로시저가 성공적으로 완료되었습니다. 이제 잘 나옴

-- 변경2

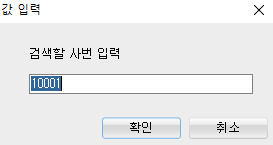
-- 회사 정보에 수정이 생김(전화번호 항목 추가 후 사원 한명 더 추가)

ALTER TABLE EMP

ADD TEL VARCHAR2(16);

INSERT INTO EMP (EMPNO, ENAME, SAL, DEPTNO, TEL)

VALUES (10001,'GOOTT2',600,20,'01011112222');



10001 GOOTT2 600 01011112222

PL/SQL 프로시저가 성공적으로 완료되었습니다.

--SET SERVEROUTPUT ON

--

--+ PL/SQL 서브 프로그램

-- : PL/SQL 서브 프로그램은 파라미터와 고유의 이름을 가진 PL/SQL 블럭

-- : 데이터베이스 객체로 존재

-- : 호출하여 사용할 목적

--

-- : 종류에는 함수(FUNCTION), 내장 프로시저(Stored Procedure)가 있음

--+ 함수(FUNCTION)

-- [형식]

-- CREATE OR REPLACE FUNCTION 함수이름 (parameter)

-- RETURN 자료형 IS

-- 변수 선언;

-- BEGIN

-- 수행 로직;

-- RETURN 변수;

-- END;

-- /

CREATE OR REPLACE FUNCTION HELLO

RETURN VARCHAR2 IS

MSG VARCHAR2(30);

BEGIN

MSG:='HELLO PL/SQL FUNCTION';

RETURN MSG;

END;

/

Function HELLO이(가) 컴파일되었습니다.

VARIABLE HI VARCHAR2(30); -- 순서대로 실행해야됨

EXECUTE :HI :=HELLO; -- 컬럼에 함수 실행

PRINT HI; -- 컬럼/내용 으로 출력됨(SELECT와 비슷한 출력방식)

SELECT HELLO FROM DUAL;

-- parameter

CREATE OR REPLACE FUNCTION TAX(P\_VALUE IN NUMBER)

RETURN NUMBER IS

RESULT NUMBER;

BEGIN

RESULT := (P\_VALUE\*0.1);

RETURN RESULT;

END;

/

SELECT TAX(800) FROM DUAL;

-- SAL\_DESCRIBE : 사원번호를 입력하면 그 사원의 급여정보를 출력

CREATE OR REPLACE FUNCTION SAL\_DESCRIBE(VEMPNO IN NUMBER)

RETURN NUMBER

IS

VSAL EMP.SAL%TYPE;

BEGIN

SELECT SAL INTO VSAL

FROM EMP

WHERE EMPNO = VEMPNO;

RETURN VSAL;

END;

/

VAR SALARY NUMBER;

EXECUTE :SALARY := SAL\_DESCRIBE(7900);

PRINT SALARY;

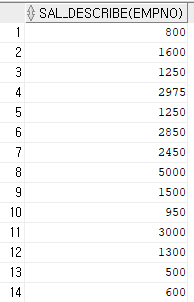
SELECT SAL\_DESCRIBE(7900) FROM DUAL;

SALARY

----------

950

SELECT SAL\_DESCRIBE(EMPNO) FROM EMP;



-- [실습] 전체 사원의 급여를 10% 인상한 결과를 출력해보세요 (FUNCTION - SAL\_UP)

CREATE OR REPLACE FUNCTION SAL\_UP(VEMPNO IN NUMBER)

RETURN NUMBER

IS

VSAL EMP.SAL%TYPE;

BEGIN

SELECT SAL INTO VSAL

FROM EMP

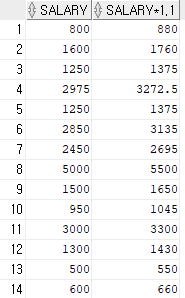
WHERE EMPNO = VEMPNO;

RETURN VSAL\*1.1;

END;

/

SELECT SAL\_DESCRIBE(EMPNO) AS "SALARY", SAL\_UP(EMPNO) AS "SALARY\*1.1" FROM EMP;



--PL/SQL : SQL + 프로그래밍 성격

--

--1. Anonymous Block

--2. Procedure

--3. Function

--4. Package

--5. Trigger

--

--Anonymous Blcok 기본 구조

--DECLARE (선택)

-- 변수 선언부 : 변수를 선언

--BEGIN (필수)

-- 실행부 : 로직 처리

--EXCEPTION (선택)

-- 예외처리부 : 예외사항을 처리

--END (필수)

-- 종료표시부

--/

SET SERVEROUTPUT ON

DECLARE

COUNTER INTEGER;

BEGIN

COUNTER := COUNTER + 1;

IF COUNTER IS NULL THEN

DBMS\_OUTPUT.PUT\_LINE('COUNTER IS NULL');

ELSE

DBMS\_OUTPUT.PUT\_LINE('COUNTER ' || COUNTER);

END IF;

END;

/

DECLARE

COUNTER INTEGER;

I INTEGER;

BEGIN

FOR I IN 1..9 LOOP

COUNTER := (2\*I);

DBMS\_OUTPUT.PUT\_LINE('2 \* ' || I || ' = ' || COUNTER);

END LOOP;

END;

/

CREATE OR REPLACE FUNCTION HELLO

RETURN VARCHAR2 IS

MSG VARCHAR2(30);

BEGIN

MSG := 'HELLO PL/SQL';

RETURN MSG;

END;

/

VARIABLE HI VARCHAR2(30);

EXECUTE :HI := HELLO;

PRINT HI;

--------------------------------------------------------------------------------

--CREATE TABLE CEMP

--AS

--SELECT \* FROM EMP;

--새로운 테이블 생성 -> EMP 배껴서 CEMP로

--------------------------------------------------------------------------------

--사원번호를 입력받아 해당 사원의 급여 출력 (10% 인상된 결과)

CREATE OR REPLACE FUNCTION CEMP\_UP(VEMPNO IN NUMBER)

RETURN NUMBER

IS

VSAL CEMP.SAL%TYPE;

BEGIN

UPDATE CEMP

SET SAL = SAL\*1.1

WHERE EMPNO = VEMPNO;

COMMIT;

SELECT SAL INTO VSAL

FROM CEMP

WHERE EMPNO = VEMPNO;

RETURN VSAL;

END;

/

VARIABLE SALARY NUMBER;

EXECUTE :SALARY := CEMP\_UP(10000);

PRINT SALARY;

Function CEMP\_UP이(가) 컴파일되었습니다.

PL/SQL 프로시저가 성공적으로 완료되었습니다.

SALARY

----------

550

VARIABLE SALARY2 NUMBER;

EXECUTE :SALARY2 := CEMP\_UP(10001);

PRINT SALARY2;

PL/SQL 프로시저가 성공적으로 완료되었습니다.

SALARY2

----------

660

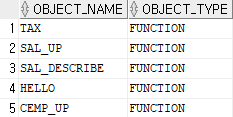
--사용자가 생성한 함수 찾기

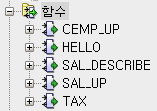
DESC USER\_PROCEDURES

SELECT OBJECT\_NAME, OBJECT\_TYPE

FROM USER\_PROCEDURES

WHERE OBJECT\_TYPE = 'FUNCTION';





--함수 삭제

--DROP FUNCTION [함수명]

--실습 : 10번 부서에 근무하고 있는 사원의 사번, 이름, 급여를 조회하시오

-- : 함수명 - EMP\_SAL

-- : 사원번호는 7900

CREATE OR REPLACE FUNCTION EMP\_SAL (VEMPNO IN NUMBER)

RETURN NUMBER

IS

VSAL EMP.SAL%TYPE;

BEGIN

SELECT SAL INTO VSAL

FROM EMP

WHERE EMPNO = VEMPNO;

RETURN VSAL;

END;

/

VARIABLE SALARY3 NUMBER;

EXECUTE :SALARY3 := EMP\_SAL(7900);

PRINT SALARY3;

Function EMP\_SAL이(가) 컴파일되었습니다.

PL/SQL 프로시저가 성공적으로 완료되었습니다.

SALARY3

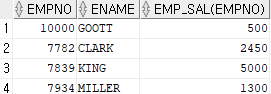
----------

950

SELECT EMPNO, ENAME, EMP\_SAL(EMPNO)

FROM EMP

WHERE DEPTNO = 10;



--실습 : 전체 사원의 사번, 이름, 부서명을 출력하시오

-- : 주어진 데이터는 부서 번호

-- : 함수명 = getDname()

create or replace function getDname(vdeptno EMP.DEPTNO%TYPE)

return varchar2

is

vdname VARCHAR2(30);

begin

select dname into vdname

from dept

where deptno = vdeptno;

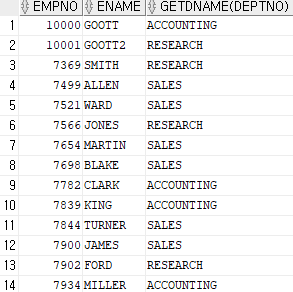
return vdname;

end;

/

SELECT EMPNO, ENAME, getDname(DEPTNO)

FROM EMP;



--전체 사원의 사원번호, 사원이름, 급여, 수당, 연봉, 세금(10%)를 조회

CREATE OR REPLACE FUNCTION FULL\_SAL

(VSAL EMP.SAL%TYPE, VCOMM EMP.COMM%TYPE)

RETURN NUMBER

IS

BEGIN

RETURN (VSAL\*12 + NVL(VCOMM,0));

END;

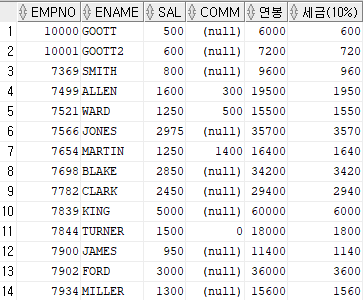
/

SELECT EMPNO, ENAME, SAL, COMM, SAL\*12+NVL(COMM,0) "연봉", (SAL\*12+NVL(COMM,0))\*0.1 "세금(10%)"

FROM EMP;

SELECT EMPNO, ENAME, SAL, COMM, FULL\_SAL(SAL, COMM) "연봉", FULL\_SAL(SAL, COMM)\*0.1 "세금(10%)"

FROM EMP;



--실습 : 전체 사원의 사원번호, 사원이름, 급여, 수당, 연봉, 세금을 조회

-- (단, 급여 수준에 따라 세율을 달리 적용)

-- : 급여가 1000미만이면 5%, 1000이상2000미만 10%, 2000이상 20% 적용

-- : 세금 적용은 연봉에 적용 구분은 급여로

-- 함수명 : DIF\_TAX

CREATE OR REPLACE FUNCTION DIF\_TAX(VSAL EMP.SAL%TYPE, VCOMM EMP.COMM%TYPE)

RETURN NUMBER

IS

RSUM NUMBER;

RTAX NUMBER;

BEGIN

RSUM := VSAL\*12 + NVL(VCOMM, 0);

IF VSAL<1000 THEN

RTAX := RSUM \* 0.05;

ELSIF VSAL<2000 THEN

RTAX := RSUM \* 0.1;

ELSE

RTAX := RSUM \* 0.2;

END IF;

RETURN RTAX;

END;

/

SELECT EMPNO, ENAME, SAL, COMM, FULL\_SAL(SAL, COMM) "연봉", DIF\_TAX(SAL, COMM) "세금"

FROM EMP;

