/\* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

함수 (FUNCTION)

- 미리 컴파일되어 재사용 가능한 객체로 SQL문 또는 PL/SQL문 내에서 호출되어 사용되는 서브프로그램.

- 호출자로 부터 파라미터 값을 입력 받아 처리한뒤 그 결과를 호출자에게 반환한다.

- 파라미터는 안 받을 수 도 있다.

- 빌트인 함수

- 오라클에서 제공하는 내장함수

- 사용자 정의 함수이름

- 사용자 정의함수

- 사용자 정의 함수 구문

CREATE [OR REPLACE] FUNCTION 함수이름 [(파라미터1, 파라미터2, ..)]

RETURN datatype --하나의 값만 반환한다.

IS

-- 선언부: 변수, 상수등 선언

BEGIN

-- 실행 부 : 처리구문

RETURN value;

[EXCETPION]

-- 예외처리 부

END;

- RETURN 반환데이터타입

- 호출자에게 전달할 반환값의 타입 지정

- char/varchar2 나 number일 경우 size는 지정하지 않는다.

- 실행 부에서 `return 값` 구문을 이용해 처리결과를 반환한다.

- 파라미터

- 개수 : 0 ~ N개

- IN 모드만 가능.

- 구문

변수명 데이터타입 [:=기본값]

- 기본값이 없는 매개변수에는 호출자에서 반드시 값을 전달해야 한다.

함수 제거

DROP FUNCTION 함수명;

-수정(ALTER는 없음 replace를 통해 수정)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* \*/

-- 매개변수 없는 함수 생성 및 호출

create or replace function ex\_08\_01\_fn

return varchar2

is

begin

return '안녕하세요!';

end;

/

select ex\_08\_01\_fn() from dual; --매개변수가 없을 시 괄호( ) 생략 가능

select ex\_08\_01\_fn from dual;

select emp\_name , ex\_08\_01\_fn 메시지

from emp;

--매개변수가 1개 있는 함수

create or replace function ex\_08\_02\_fn(p\_num binary\_integer)

return varchar2

is

begin

return p\_num||'원';

end;

/

select ex\_08\_02\_fn(10000) from dual;

select ex\_08\_02\_fn from dual; --에러

select ex\_08\_02\_fn(salary) from emp;

--기본값이 있는 매개변수를 가진 함수

create or replace function ex\_08\_02\_fn(p\_num binary\_integer :=0)

--만약 매개변수로 받은 값이 없을 시 0을 default로

return varchar2

is

begin

return p\_num||'원';

end;

/

select ex\_08\_02\_fn(10000) from dual;

select ex\_08\_02\_fn from dual; --default출력

--매개변수가 여러개인 함수

create or replace function ex\_8\_3\_fn(p\_num1 binary\_double:=0,

p\_num2 binary\_double:=0,

p\_num3 binary\_double:=0,

p\_cur varchar2:='원')

return varchar2

is

v\_result varchar2(100);

begin

v\_result := (p\_num1 + p\_num2 + p\_num3)||p\_cur;

return v\_result;

end;

/

select ex\_8\_3\_fn from dual;

select ex\_8\_3\_fn(10,20,30) from dual;

select ex\_8\_3\_fn(10.5,20.5,30.5,'달러') from dual;

select ex\_8\_3\_fn(10.5,20.5,'달러') from dual;--에러: 매개변수는 반드시 순서대로 들어감

-- emp\_id를 받아서 그 직원의 커미션(salary \* comm\_pct)의 값을 반환하는 함수

select salary\*comm\_pct from emp;

create or replace function ex\_8\_5(p\_emp\_id emp.emp\_id%type)

return binary\_double

is

v\_result binary\_double;

begin

select salary \* comm\_pct

into v\_result

from emp

where emp\_id = p\_emp\_id;

return v\_result;

end;

/

select ex\_8\_5(145) from dual;

select emp\_id, ex\_8\_5(emp\_id) 계산값, comm\_pct, salary from emp;

select \* from emp;

-- TODO: 매개변수로 정수 두개를 받아 나눈 결과의 소숫점 이하 부분만 반환하는 함수 구현 (ex: 10/4 -> 0.5 반환)

create or replace function int\_divide(p\_num1 binary\_integer, p\_num2 binary\_integer)

return binary\_double

is

v\_result binary\_double;

begin

v\_result := p\_num1/p\_num2 - trunc(p\_num1/p\_num2);

return v\_result;

end;

/

select int\_divide(12,5) from dual;

-- TODO: 매개변수로 정수 두개를 받아서 나머지 연산을 처리하는 함수 구현. (10/4 -> 2를 반환)

-- 나머지계산: --피젯수 - (결과 정수값 \* 제수)

-- ex) 10/4의 나머지 : 10 - (10/4결과정수 \* 4)

create or replace function mod\_prac(p\_num1 binary\_integer, p\_num2 binary\_integer)

return binary\_integer

is

v\_result binary\_integer;

begin

v\_result := p\_num1 - (trunc(p\_num1/p\_num2) \* p\_num2);

return v\_result;

end;

/

select mod\_prac(10,4) from dual;

select mod(10,4) from dual;

-- TODO: 1. 매개변수로 사원 ID를 받아서 자신의 급여가 자신이 속한 부서의 평균급여 이상이면 TRUE를 미만이면 FALSE를 반환하는 함수 작성. 조회한 사원이 없는 경우 NULL을 반환

-- 2. 위의 함수를 호출하는 익명 프로시저에서 작성.

-- 함수를 호출하여 반환되는 boolean 값이

-- TRUE이면 '직원의 급여가 부서평균 이상' 을 FALSE이면 '직원의 급여가 부서평균 미만' 을 NULL이면 '없는 직원' 을 출력한다.

CREATE OR REPLACE FUNCTION todo\_03\_fn(p\_emp\_id emp.emp\_id%TYPE)

RETURN boolean

IS

v\_cnt binary\_integer;

v\_salary emp.salary%type; --개인 급여

v\_dept\_id dept.dept\_id%type;--조회할 부서 id

v\_sal\_avg binary\_double; --부서 급여 평균

ex\_found\_null exception; --예외처리(null 일때)

BEGIN

-- 조회 직원이 없으면 null을 반환

select count(\*) into v\_cnt from emp where emp\_id = p\_emp\_id;

if v\_cnt = 0 then

return null;

end if;

-- salary 와 부서 id를 조회

select salary, dept\_id into v\_salary, v\_dept\_id from emp where emp\_id = p\_emp\_id;

-- 부서의 평균 급여를 조회

select avg(salary) into v\_sal\_avg from emp

where dept\_id = v\_dept\_id;

-- 직원의 salary와 부서 평균급여를 비교해서 boolean 값을 반환

/\*

if v\_salary >= v\_sal\_avg then

return TRUE;

else

return FALSE;

end if;

\*/

RETURN v\_salary >= v\_sal\_avg;

END;

/

SELECT todo\_03\_fn(emp\_id) FROM emp; --SQL에서는 boolean타입이 존재하지 않기 때문에 사용 불가

-- 170(이상) 180(이하)

-- ------위의 함수 사용하는 프로시저. ---------

-- TRUE이면 '직원의 급여가 부서평균 이상' 을 FALSE이면 '직원의 급여가 부서평균 미만' 을 NULL이면 '없는 직원' 을 출력한다.

DECLARE

v\_result boolean;

BEGIN

for idx in 170..180

loop

v\_result := todo\_03\_fn(idx);

if v\_result is null then --이 때 null도 false로 취급하기 때문에 조건문에서 제일 먼저 쓴다.

dbms\_output.put\_line(idx||'없는 직원');

elsif v\_result then --v\_result 자체가 true이기 때문에 v\_result = true라고 적을 필요 없다.

dbms\_output.put\_line(idx||'직원의 급여가 부서평균 이상');

else

dbms\_output.put\_line(idx||'직원의 급여가 부서평균 미만');

end if;

end loop;

END;

/

--확인 sql문

select emp\_id,salary, dept\_id from emp where emp\_id = 175;

select avg(salary) from emp where dept\_id = (select dept\_id from emp where emp\_id = 175);

--TODO 매개변수로 5개의 'y' 또는 'n'을 받아서 몇번째 매개변수가 y인지를 하나의 문자열로 묶어서 반환하는 함수 구현.

-- select todo\_03\_fn('y','n', 'n', 'y', 'y') from dual; => 1, 4, 5

create or replace function todo\_04\_fn ( p\_yn1 varchar2,

p\_yn2 varchar2,

p\_yn3 varchar2,

p\_yn4 varchar2,

p\_yn5 varchar2)

return varchar2

is

v\_return varchar2(100); -- 최종결과 문자열을 저장할 변수.

begin

--p\_yn1 처리: y 이면 v\_return 문자열에 연결.

if(p\_yn1 = 'y') then

if nvl(length(v\_return),0) = 0 then

v\_return := '1';

else

v\_return := v\_return||', 1';

end if;

end if;

--p\_yn2 처리

if(p\_yn2 = 'y') then

if nvl(length(v\_return),0) = 0 then

v\_return := '2';

else

v\_return := v\_return||', 2';

end if;

end if;

--p\_yn3 처리

if(p\_yn3 = 'y') then

if nvl(length(v\_return),0) = 0 then

v\_return := '3';

else

v\_return := v\_return||', 3';

end if;

end if;

--p\_yn4 처리

if(p\_yn4 = 'y') then

if nvl(length(v\_return),0) = 0 then

v\_return := '4';

else

v\_return := v\_return||', 4';

end if;

end if;

--p\_yn5 처리

if(p\_yn5 = 'y') then

if nvl(length(v\_return),0) = 0 then

v\_return := '5';

else

v\_return := v\_return||', 5';

end if;

end if;

-- 결과값 반환

return v\_return;

end;

/

select todo\_04\_fn('y','y', 'n', 'y', 'n') from dual;

select todo\_04\_fn('n','n', 'n', 'y', 'y') from dual;

create table choice\_tb(

ch1 char check(ch1 in ('y', 'n')),

ch2 char check(ch2 in ('y', 'n')),

ch3 char check(ch3 in ('y', 'n')),

ch4 char check(ch4 in ('y', 'n')),

ch5 char check(ch5 in ('y', 'n'))

);

insert into choice\_tb values ('n' ,'y', 'n', 'y', 'y');

insert into choice\_tb values ('y' ,'y', 'y', 'y', 'y');

insert into choice\_tb values ('y' ,'y', 'n', 'n', 'y');

insert into choice\_tb values ('y' ,'y', 'n', 'n', 'n');

insert into choice\_tb values ('n' ,'y', 'n', 'y', 'n');

commit;

select todo\_04\_fn(ch1, ch2, ch3,ch4,ch5) 선택사항 from choice\_tb;