

```

1  REM SUBPROGRAM
2  --1. 잘 정리된 논리적 코드분할이름
3  --2. 컴파일된 상태에서 데이터베이스에 저장되기 때문에 성능이 향상된다.
4  --3. 테이블이름이나 컬럼의 이름을 명시하지 않기 때문에 보안에 도움이 된다.
5  --4. 모듈화를 통한 관리 용이
6  --5. 종류
7      --1) Stored Procedure    <-----Java 에서 사용
8      --2) Stored Function
9
10 REM Stored PROCEDURE(저장프로시저)
11 --1. 목적 : 속도, 보안
12 --2. compile(pcode로 변환)상태로 RDBMS 에 저장
13 --3. 나중에 실행될 일련의 명령의 집합
14 --4. 리턴하는 행의 갯수가 많으면 에러난다.
15 --5. PL/SQL에서 SELECT는 반드시 SELECT INTO 를 사용해야
16 --6. Syntax
17 CREATE OR REPLACE PROCEDURE procedure_name
18 (
19     매개변수 영역
20     변수이름    변수모드    변수타입
21     v_empno      IN(OUT, IN OUT)    emp.EMPNO%TYPE
22 )
23 IS
24     내부변수
25 BEGIN
26     PL/SQL 문장들
27 END;
28
29 --OR REPLACE :이미 존재할 경우 procedure 의 내용을 지우고 다시 생성
30 --변수 Mode : 3가지
31 --IN : 입력 매개변수
32 --OUT : 출력 매개변수
33 --IN OUT : 입력, 출력 매개변수
34
35 --7. Guide Lines
36 --1) 어떤 parameter도 가능
37 --2) IS 로 시작
38 --3) local 변수 선언은 IS와 BEGIN 사이에 선언
39 --4) PL/SQL 에서 SELECT시 반드시 INTO 사용
40 --5) SELECT시 여러 개의 행을 리턴한다면 에러 발생
41
42 CREATE OR REPLACE PROCEDURE test proc
43 IS
44     v_name    VARCHAR2(30);
45 BEGIN
46     v name := '한지민';
47     DBMS_OUTPUT.PUT_LINE('My name is ' || v_name);
48 END;
49 /
50
51 EXEC test_proc;
52
53 SET SERVEROUTPUT ON
54 CREATE OR REPLACE PROCEDURE sp_HelloWorld
55 IS
56 BEGIN
57     DBMS_OUTPUT.PUT LINE('Hello, World');
58 END;
59 /
60
61 EXEC sp HelloWorld;
62
63 -- emp 테이블이 모든 데이터를 삭제하는 Stored Procedure 를 작성하시오.
64 CREATE OR REPLACE PROCEDURE del all
65 IS
66 BEGIN
67     DELETE FROM emp;
68 END;
69 /
70
71 EXEC del all
72
73 --8. 오류 발생시 오류 보기
74 DESC USER ERRORS
75 SHOW ERROR
76
77 --9. 확인하기
78 DESC USER PROCEDURES;
79 SELECT OBJECT_NAME, PROCEDURE NAME FROM USER PROCEDURES;
80 DESC USER_SOURCE;
81 COL NAME FOR A10
82 COL TEXT FOR A40
83 SELECT NAME, TEXT FROM USER SOURCE;
84
85 --10. IN 매개변수
86 CREATE OR REPLACE PROCEDURE test proc
87 (
88     p_name    IN VARCHAR2
89 )
90 IS
91 BEGIN
92     DBMS_OUTPUT.PUT_LINE('My name is ' || p_name);
93 END;
94 /
95

```

```

96 EXEC test proc('설운도');
97
98 SET SERVEROUTPUT ON
99 CREATE OR REPLACE PROCEDURE sp_OneInParameter
100 (param1 IN VARCHAR2)
101 IS
102 BEGIN
103     DBMS_OUTPUT.PUT_LINE('Hello, World IN parameter ' || param1);
104 END;
105 /
106
107 EXEC sp_OneInParameter('Michael');
108 Hello, World IN parameter Michael
109

```

--사원번호와 봉급을 입력받아 업데이트하는 PL/SQL 문장을 완성하시오.

```

111
112 SET SERVEROUTPUT ON
113 ACCEPT p_empno PROMPT 'Enter a Number : '
114 ACCEPT p_sal PROMPT 'Enter a Salary : '
115 DECLARE
116     v_empno emp.empno%TYPE := &p_empno;
117     v_sal emp.sal%TYPE := &p_sal;
118 BEGIN
119     UPDATE emp
120     SET sal = v_sal
121     WHERE empno = v_empno;
122     COMMIT;
123 END;
124 /
125 SET SERVEROUTPUT OFF
126
127
128 CREATE OR REPLACE PROCEDURE emp_sal_update
129 (
130     v_empno IN emp.empno%TYPE,
131     v_sal IN emp.sal%TYPE
132 )
133 IS
134 BEGIN
135     UPDATE emp
136     SET sal = v_sal
137     WHERE empno = v_empno;
138     COMMIT;
139 END;
140 /
141
142 -- 실행 EXEC[UTE] Procedure_name(parameter1, parameter2,...);
143 EXEC emp_sal_update(7788, 8000);
144

```

----사원을 받아 삭제하는 프로시저

```

145
146 CREATE OR REPLACE PROCEDURE emp_del
147 (
148     v_empno IN emp.empno%TYPE
149 )
150 IS
151 BEGIN
152     DELETE FROM emp
153     WHERE empno = v_empno;
154 END;
155 /
156 --실행할 때
157 EXEC emp_del(7900);
158

```

--부서번호, 부서이름, 지역을 받아 삽입하는 프로시저

```

159
160 CREATE OR REPLACE PROCEDURE sp_insert_dept
161 (
162     v_deptno IN dept.deptno%TYPE,
163     v_dname IN dept.dname%TYPE,
164     v_loc IN dept.loc%TYPE
165 )
166 IS
167 BEGIN
168     INSERT INTO DEPT
169     VALUES (v_deptno, UPPER(v_dname), UPPER(v_loc));
170 END;
171 /
172
173 EXEC sp_insert_dept(50, 'marketing', 'yatap');
174

```

--emp table에서 새로운 사원의 정보를 이름, 업무, 매니저, 급여를 입력받아 등록하는 emp\_input 프로시저를 생성하여라. 단, 부서번호는 매니저의 부서 번호와 동일하게 하고 보너스는 SALESMAN은 0을 그 외는 NULL을 입력하라.

```

175
176 CREATE SEQUENCE emp_empno_seq
177 START WITH 1
178 INCREMENT BY 1
179 MAXVALUE 999
180 NOCYCLE
181 CACHE 10;
182
183
184 CREATE PROCEDURE emp_input
185 (
186     v_name IN emp.ename%TYPE,
187     v_job IN emp.job%TYPE,
188     v_mgr IN emp.mgr%TYPE,
189     v_sal IN emp.sal%TYPE

```

```

190 )
191 IS
192     v_comm emp.comm%TYPE;
193     v_deptno emp.deptno%TYPE;
194 BEGIN
195     IF UPPER(v_job) = 'SALESMAN' THEN
196         v_comm := 0;
197     ELSE
198         v_comm := NULL;
199     END IF;
200
201     SELECT deptno
202     INTO v_deptno
203     FROM emp
204     WHERE empno = v_mgr;
205
206     INSERT INTO emp
207     VALUES(emp_empno_seq.NEXTVAL, v_name, UPPER(v_job), v_mgr, SYSDATE, v_sal, v_comm, v_deptno);
208 END;
209 /
210
211 --11. OUT 매개변수
212 CREATE OR REPLACE PROCEDURE test_proc
213 (
214     p_name OUT VARCHAR2
215 )
216 IS
217 BEGIN
218     p_name := 'My name is 한지민.';
219 END;
220 /
221
222 --실행
223 --OUT 파라미터는 Binding 변수가 필요하다.
224 --VAR[TABLE] 변수이름 변수타입
225 --OUT 변수값 출력
226 --PRINT 변수이름
227 VAR g_ename VARCHAR2(20);
228 EXEC test_proc(:g_ename); --binding 변수
229 PRINT g_ename;
230
231 OR
232
233 DECLARE
234     v_name VARCHAR2(100);
235 BEGIN
236     test_proc(v_name);
237     DBMS_OUTPUT.PUT_LINE(v_name);
238 END;
239 /
240
241 -----
242 SET SERVEROUTPUT ON
243 CREATE OR REPLACE PROCEDURE sp_OneOutParameter
244 (outParam1 OUT VARCHAR2)
245 IS
246 BEGIN
247     outParam1 := 'Hello World OUT parameter';
248 END;
249 /
250
251 SQL> variable a VARCHAR2(30);
252 SQL> EXEC sp_OneOutParameter(:a);
253
254 PL/SQL procedure successfully completed.
255
256 SQL> print a;
257
258 A
259 -----
260 Hello World OUT parameter
261
262 OR
263
264 SET SERVEROUTPUT ON
265 DECLARE
266     outParam1 VARCHAR2(100);
267 BEGIN
268     sp_OneOutParameter(outParam1);
269     DBMS_OUTPUT.PUT_LINE(outParam1);
270 END;
271 /
272
273 SQL> @demo.sql
274 Hello World OUT parameter
275
276 PL/SQL procedure successfully completed.
277
278 OR
279
280 --굳이 Stored Procedure를 별도로 생성할 필요가 없을때는
281 SET SERVEROUTPUT ON
282 DECLARE
283     outParam1 VARCHAR2(100);
284

```

```

285 PROCEDURE procOneOUTParameter
286 (outParam1 OUT VARCHAR2)
287 IS
288 BEGIN
289     outParam1 := 'Hello World OUT parameter';
290 END;
291
292 BEGIN
293     procOneOUTParameter(outParam1);
294     DBMS_OUTPUT.PUT_LINE(outParam1);
295 END;
296 /
297
298 -----
299 SEE SERVEROUTPUT ON
300 DECLARE
301     a NUMBER;
302     b NUMBER;
303     c NUMBER;
304
305 PROCEDURE findMin(x IN number, y IN number, z OUT NUMBER) IS
306 BEGIN
307     IF x < y THEN
308         z := x;
309     ELSE
310         z := y;
311     END IF;
312 END;
313
314 BEGIN
315     a := 23;
316     b := 45;
317     findMin(a, b, c);
318     DBMS_OUTPUT.PUT_LINE(' Minimum of (23, 45) : ' || c);
319 END;
320 /
321
322 SQL> @demo.sql
323 Minimum of (23, 45) : 23
324
325 PL/SQL procedure successfully completed.
326
327 -----

```

--사번을 받아 사원이름과 봉급 검색

```

330 CREATE OR REPLACE PROCEDURE sp_emp_select
331 (
332     v_empno IN emp.EMPNO%TYPE,
333     v_ename OUT emp.ENAME%TYPE,
334     v_sal OUT emp.sal%TYPE
335 )
336 IS
337 BEGIN
338     SELECT ename, sal
339     INTO v_ename, v_sal
340     FROM EMP
341     WHERE empno = v_empno;
342 END;
343 /
344
345 SQL> var g_ename VARCHAR2(20);
346 SQL> var g_sal NUMBER;
347 SQL> EXEC sp_emp_select(7566, :g_ename, :g_sal);
348
349 PL/SQL 처리가 정상적으로 완료되었습니다.

```

SQL> print g\_ename;

G\_ENAME

-----

JONES

SQL> print g\_sal;

G SAL

-----

2975

--이름을 입력받아서 그 사원의 정보 중 부서명과 급여를 검색하는 프로시저를 완성하시오.

```

364 CREATE OR REPLACE PROCEDURE emp_dept_sal_select
365 (
366     v_ename IN emp.ename%TYPE,
367     v_dname OUT dept.dname%TYPE,
368     v_sal OUT emp.sal%TYPE
369 )
370 IS
371     v_deptno emp.deptno%TYPE;
372 BEGIN
373     SELECT deptno, sal
374     INTO v_deptno, v_sal
375     FROM emp
376     WHERE ename = UPPER(v_ename);
377
378     SELECT dname
379     INTO v_dname

```

```

380         FROM dept
381         WHERE deptno = v deptno;
382     END;
383 /
384
385     VAR g_dname VARCHAR2(20);
386     VAR g_sal NUMBER;
387     EXEC emp_dept_sal_select('scott', :g_dname, :g_sal);
388
389     PL/SQL PROCEDURE successfully completed.
390
391     PRINT g_dname;
392     PRINT g_sal;
393
394     OR
395
396     SET SERVEROUTPUT ON
397     DECLARE
398         v_ename      emp.ename%TYPE;
399         v_dname      dept.dname%TYPE;
400         v_sal        emp.sal%TYPE;
401     BEGIN
402         v_ename := 'scott';
403         emp_dept_sal_select(v_ename, v_dname, v_sal);
404         DBMS_OUTPUT.PUT_LINE('Name : ' || v_ename);
405         DBMS_OUTPUT.PUT_LINE('Dept Name : ' || v_dname);
406         DBMS_OUTPUT.PUT_LINE('Salary : ' || v_sal);
407     END;
408 /
409
410     SQL> @demo.sql
411     Name : scott
412     Dept Name : RESEARCH
413     Salary : 3000
414
415     PL/SQL procedure successfully completed.
416
417 -----
418 --우편번호검색, 단 동이름은 역삼동으로만 할 것
419     CREATE OR REPLACE PROCEDURE sp_zipcode_select
420     (
421         v_dong1 IN zipcode.DONG%TYPE,
422         v_zipcode OUT zipcode.ZIPCODE%TYPE,
423         v_sido OUT zipcode.SIDO%TYPE,
424         v_gugun OUT zipcode.GUGUN%TYPE,
425         v_dong OUT zipcode.DONG%TYPE,
426         v_bunji OUT zipcode.BUNJI%TYPE
427     )
428     IS
429     BEGIN
430         SELECT zipcode.ZIPCODE, sido, gugun, dong, bunji
431         INTO v_zipcode, v_sido, v_gugun, v_dong, v_bunji
432         FROM ZIPCODE
433         WHERE dong LIKE CONCAT(CONCAT('%', v_dong1), '%');
434     END;
435 /
436
437 --12. IN OUT 파라미터
438     CREATE OR REPLACE PROCEDURE test_proc
439     (
440         p_name IN OUT VARCHAR2
441     )
442     IS
443     BEGIN
444         p_name := 'My name is ' || p_name;
445     END;
446 /
447
448     DECLARE
449         v_name VARCHAR2(100);
450     BEGIN
451         v_name := '나훈아';
452         test_proc(v_name);
453         DBMS_OUTPUT.PUT_LINE(v_name);
454     END;
455 /
456
457 -----
458     CREATE OR REPLACE PROCEDURE sp_OneINOUTParameter
459     (genericParam IN OUT VARCHAR2)
460     IS
461     BEGIN
462         genericParam := 'Hello World INOUT parameter ' || genericParam;
463     END;
464 /
465
466     SET SERVEROUTPUT ON
467     DECLARE
468         genericParam VARCHAR2(100) := 'Jimin';
469     BEGIN
470         sp_OneINOUTParameter(genericParam);
471         DBMS_OUTPUT.PUT_LINE(genericParam);
472     END;
473 /
474

```

```

475 SQL> @demo.sql
476 Hello World INOUT parameter Jimin
477
478 PL/SQL procedure successfully completed.
479
480 OR
481
482 SET SERVEROUTPUT ON
483 DECLARE
484     genericParam VARCHAR2(100) := 'Jimin';
485
486 PROCEDURE sp_OneINOUTParameter
487 (genericParam IN OUT VARCHAR2)
488 IS
489 BEGIN
490     genericParam := 'Hello World INOUT parameter ' || genericParam;
491 END;
492
493 BEGIN
494     sp_OneINOUTParameter(genericParam);
495     DBMS_OUTPUT.PUT_LINE(genericParam);
496 END;
497 /
498
499 SQL> @demo.sql
500 Hello World INOUT parameter Jimin
501
502 PL/SQL procedure successfully completed.
503 -----
504 DECLARE
505     a NUMBER;
506 PROCEDURE squareNum(x IN OUT NUMBER) IS
507 BEGIN
508     x := x * x;
509 END;
510 BEGIN
511     a := 23;
512     squareNum(a);
513     DBMS_OUTPUT.PUT_LINE(' Square of (23) : ' || a);
514 END;
515 /
516 -----
517 --13. 복합 파라미터
518 CREATE OR REPLACE PROCEDURE test_proc
519 (
520     p_name IN VARCHAR2,
521     p_msg OUT VARCHAR2
522 )
523 IS
524 BEGIN
525     p_msg := 'My name is ' || p_name;
526 END;
527 /
528
529 DECLARE
530     v_msg VARCHAR2(100);
531 BEGIN
532     test_proc('조용필', v_msg);
533     DBMS_OUTPUT.PUT_LINE(v_msg);
534 END;
535 /
536
537 --14. 파라미터에 기본값 사용하기
538 CREATE OR REPLACE PROCEDURE test_proc
539 (
540     p_name VARCHAR2 DEFAULT '송대관'
541 )
542 IS
543 BEGIN
544     DBMS_OUTPUT.PUT_LINE('My name is ' || p_name);
545 END;
546 /
547
548 EXEC test_proc();
549
550 --15. 저장 프로시저의 매개변수 실행 순서
551 CREATE OR REPLACE PROCEDURE test_proc
552 (
553     p_dan NUMBER,
554     p_max_num NUMBER
555 )
556 IS
557 i NUMBER;
558 BEGIN
559     FOR i IN 1..p_max_num
560     LOOP
561         DBMS_OUTPUT.PUT_LINE(p_dan || ' * ' || i || ' = ' || p_dan * i);
562     END LOOP;
563 END;
564 /
565 EXEC test_proc(3, 9); --3단
566
567 CREATE OR REPLACE PROCEDURE test_proc
568 (

```

```

570         p dan NUMBER DEFAULT 2,
571         p max num NUMBER DEFAULT 9
572     )
573     IS
574         i NUMBER;
575     BEGIN
576         FOR i IN 1..p_max_num
577             LOOP
578                 DBMS_OUTPUT.PUT_LINE(p dan || ' * ' || i || ' = ' || p dan * i);
579             END LOOP;
580     END;
581 /
582 EXEC test proc(4);  --4단, 즉 값을 한 개만 넣으면 제일 앞의 매개변수로 인식
583
584 --만일 2번째 매개변수로 인식하려면
585 EXEC test proc(p max num => 5);
586
587
588 --16. 저장 프로시저 삭제
589 DROP PROCEDURE [schema].procedure name;
590
591
592 REM Stored FUNCTION
593 --1)오라클에서 기본적으로 제공하는 함수 이외에 사용자가 필요에 따라 만든 함수
594 --2)프로시저와 성격이 매우 비슷하지만, 반환값이 있느냐 없느냐가 가장 큰 차이
595 --3) 실행시 반드시 하나의 값을 RETURN하기 위해 사용
596 --4) 함수 선언에서 Data Type이 있는 RETURN 절을 추가하고 PL/SQL 블록에 적어도 한개 이상의 RETURN문을 포함
597 --5)Syntax
598 CREATE OR REPLACE FUNCTION [SCHEMA.]function_name
599 [(argument1 [mode] datatype[, argument2 [mode] datatype,...])]
600 RETURN data type
601 {IS | AS}
602 변수 선언부
603 BEGIN
604     statement1;
605     statement2;
606     RETURN variable_name;
607 END ;
608
609 --실행
610 EXECUTE :variable_name := function_name(argument_list);
611
612 -----
613 CREATE FUNCTION chk_sal
614 (v_sal NUMBER)
615     RETURN NUMBER
616 IS
617     v_chk NUMBER;
618 BEGIN
619     v_chk := v_sal * 0.01;
620     RETURN v_chk;
621 END;
622 /
623
624 -EXECUTE 명령어에 의해 실행(SQL*Plus 환경)
625 variable v_sal NUMBER
626 EXECUTE :v_sal := chk_sal(7900);
627 PL/SQL procedure successfully completed.
628 SQL> print v_sal
629
630      V SAL
631 -----
632      79
633
634 -표현식의 일부로서 호출될 수 있다.
635 SQL> SELECT empno, sal, chk_sal(sal)
636        2 FROM emp
637        3 WHERE deptno = 10;
638
639          EMPNO          SAL  CHK_SAL(SAL)
640 -----
641          7782          2450           24.5
642          7839          5000            50
643          7934          1300            13
644
645 SQL> SELECT SUM(sal), chk_sal(SUM(sal))
646        2 FROM emp
647        3 WHERE deptno = 10;
648
649          SUM(SAL)  CHK_SAL(SUM(SAL))
650 -----
651          8750          87.5
652
653 -----
654 CREATE OR REPLACE FUNCTION tax
655 (v_value IN NUMBER)
656 --함수는 IN argument 만 사용한다.
657     RETURN number
658 IS
659 BEGIN
660     RETURN v_value * 0.07;
661 END;
662 /
663
664 SQL> @demo.sql

```

```

665      Function created.
666      SQL> variable x number
667      SQL> execute :x := tax(100)
668      PL/SQL procedure successfully completed.
669
670      SQL> print x
671
672              X
673      -----
674              7
675
676      SQL> SELECT sal, tax(sal) FROM emp
677      2  WHERE empno = 7900;
678
679              SAL      TAX(SAL)
680      -----
681              950        66.5
682
683      SQL> UPDATE emp SET sal = tax(sal)
684      2  WHERE empno = 7900;
685
686      1 row updated.
687      -----
688
689      CREATE OR REPLACE FUNCTION test_func(p_name IN VARCHAR2)
690      RETURN VARCHAR2
691      IS
692          v_name    VARCHAR2(100) := '';
693      BEGIN
694          v_name := 'My Name is ' || p_name;
695          RETURN v_name;
696      END ;
697
698      FUNCTION created.
699
700      SQL> SELECT test_func('조성모') AS "이름" FROM dual;
701
702      이름
703      -----
704      My Name is 조성모
705
706      OR
707
708      SQL> VAR v_name VARCHAR2(100);
709      SQL> EXEC  :v_name := test_func('설운도');
710
711      PL/SQL procedure successfully completed.
712
713      SQL> PRINT v_name;
714      V NAME
715      -----
716      My Name is 설운도
717
718      --4) 주의 사항
719      --a. SELECT 문에서 사용되는 함수에는 DML 을 포함해서는 안된다.
720      --b. UPDATE 나 DELETE 문에서 사용되는 함수에는 같은 테이블에 대한 DML 과 SELECT 문이 포함되서는 안된다.
721
722      --보너스는 급여의 200%를 지급한다.
723
724      CREATE OR REPLACE FUNCTION cal_bonus
725      ( v empno IN emp.empno%TYPE )
726      RETURN NUMBER
727      IS
728          v_sal    NUMBER(7,2);
729      BEGIN
730          SELECT sal
731          INTO v_sal
732          FROM emp
733          WHERE empno = v_empno;
734
735          RETURN v_sal * 200;
736      END ;
737      /
738
739      SQL> VAR v_rebonus NUMBER;
740      SQL> EXECUTE  :v_rebonus := cal_bonus(7788);
741
742      PL/SQL procedure successfully completed.
743
744      SQL> PRINT v_rebonus;
745      SQL> SELECT empno, sal, cal_bonus(7369)
746      FROM emp WHERE empno = 7369;
747
748      --사원명으로 검색하여 해당 사원의 직급을 얻어 오는 함수를 fun_sel_empname라는 이름으로 작성하시오.
749      CREATE OR REPLACE FUNCTION fun_sel_empname
750      ( v ename IN emp.ename%TYPE)
751      RETURN VARCHAR2
752      IS
753          v_job emp.job%TYPE;
754      BEGIN
755          SELECT job INTO v_job
756          FROM emp
757          WHERE ename = UPPER(v_ename);
758
759          RETURN v_job;

```



```

760 END ;
761 /
762
763 SQL>VAR v_job VARCHAR2(10);
764 SQL>EXEC :v_job := fun_sel_empname('scott');
765 SQL>PRINT v_job;
766 SQL>SELECT ename, job, fun_sel_empname('scott')
767        FROM emp WHERE ename = UPPER('scott');
768
769 --5) 삭제
770 DROP FUNCTION function_name;
771
772 -emp table에서 이름으로 부서 번호를 검색하는 함수를 작성하시오
773 --CREATE OR REPLACE FUNCTION ename_deptno
774 --(
775 --  v ename      IN  emp.ename%TYPE
776 --)
777 --RETURN      NUMBER
778 --IS
779 --  v deptno      emp.deptno%TYPE;
780 --BEGIN
781 --  SELECT deptno
782 --    INTO v deptno
783 --    FROM emp
784 --    WHERE ename = UPPER(v_ename);
785 --RETURN v_deptno;
786 --END;
787 --VAR g_deptno NUMBER
788 --EXECUTE : g_deptno := ename_deptno('SCOTT')
789
790 --emp table에서 이름을 입력받아 부서번호, 부서명, 급여를 검색하는 function을 작성하시오. 단 부서번호를
791 RETURN에 사용하시오.
792 --CREATE OR REPLACE FUNCTION emp_disp
793 --(
794 --  v ename      IN  emp.ename%TYPE,
795 --  v_dname      OUT dept.dname%TYPE,
796 --  v_sal        OUT emp.sal%TYPE
797 --)
798 --RETURN      NUMBER
799 --IS
800 --  v_deptno      emp.deptno%TYPE;
801 --  v_dname temp    dept.dname%TYPE;
802 --  v_sal_temp    emp.sal%TYPE;
803 --BEGIN
804 --  SELECT sal, deptno
805 --    INTO v sal temp, v deptno
806 --    FROM emp
807 --    WHERE ename = UPPER(v_ename);
808 --  SELECT dname
809 --    INTO v dname temp
810 --    FROM dept
811 --    WHERE deptno = v_deptno;
812 --  v_dname := v_dname temp;
813 --  v_sal := v_sal_temp;
814 --END;
815 --VAR g_deptno NUMBER
816 --VAR g_dname VARCHAR(20)
817 --VAR g_sal NUMBER
818 --EXECUTE :g_deptno := emp_disp('SCOTT', :g_dname, :g_sal)
819
820 REM Procedure vs Function
821 --PROCEDURE                      FUNCTION
822 --PL/SQL 문으로서 실행            식의 일부로서 실행
823 --RETURN datatype이 없습        RETURN datatype이 필수
824 --값을 RETURN 할 수 있습        값을 RETURN 하는 것이 필수

```