select salary from emp;

--다중행

select round(salary/comm\_pct,1) from emp

where comm\_pct is not null;

--단일행(집계)

select sum(salary) from emp;

select round(avg(salary),2) from emp;

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

함수 - 문자열관련 함수

UPPER()/ LOWER() : 대문자/소문자 로 변환

INITCAP(): 단어 첫글자만 대문자 나머진 소문자로 변환

LENGTH() : 글자수 조회

LPAD(값, 크기, 채울값) : "값"을 지정한 "크기"의 고정길이 문자열로 만들고 모자라는 것은 왼쪽부터 "채울값"으로 채운다.

RPAD(값, 크기, 채울값) : "값"을 지정한 "크기"의 고정길이 문자열로 만들고 모자라는 것은 오른쪽부터 "채울값"으로 채운다.

SUBSTR(값, 시작index, 글자수) - "값"에서 "시작index"번째 글자부터 지정한 "글자수" 만큼의 문자열을 추출. 글자수 생략시 끝까지.

REPLACE(값, 찾을문자열, 변경할문자열) - "값"에서 "찾을문자열"을 "변경할문자열"로 바꾼다.

LTRIM(값): 왼공백 제거

RTRIM(값): 오른공백 제거

TRIM(값): 양쪽 공백 제거

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

select upper('abc'),lower('ABC'),initcap('abc def\_ghi') from dual;

select length('alala'), length('가나다') from dual;

select \* from emp where length(emp\_name)>7;

select lpad('test',10,'-') from dual; --'-'생략시 공백으로 채움

select rpad('test',10,'-') from dual;

select rpad('testtest',5,'-') from dual; --남는부분 오른쪽을 자름

select lpad('testtest',5,'-') from dual;

select substr('123456789',5,2) from dual; --5번째 글자부터 두글자만 자름

select replace('A--B--C','-','#') from dual; -- '-'를 '#'으로 바꿈

select ' hello ' o,trim(' hello ') b,ltrim(' hello ') l, rtrim(' hello ') r from dual; --공백을 없앰

select length(trim(' aaa ')) a from dual; --함수안에서 함수를 호출하는 경우: 안쪽 함수를 먼저 실행하고 그 결과를 넣어 바깥쪽 함수 실행.

--EMP 테이블에서 직원의 이름(emp\_name)을 모두 대문자, 소문자, 첫글자 대문자, 이름 글자수를 조회

select upper(emp\_name) 대문자이름, lower(emp\_name)소문자이름, initcap(emp\_name)첫글자대문자, length(emp\_name)글자수 from emp;

-- TODO: EMP 테이블에서 직원의 ID(emp\_id), 이름(emp\_name), 급여(salary), 부서(dept\_name)를 조회. 단 직원이름(emp\_name)은 모두 대문자, 부서(dept\_name)는 모두 소문자로 출력.

-- UPPER/LOWER

select emp\_id, upper(emp\_name), salary, lower(dept\_name) from emp;

--(아래 2개는 비교값의 대소문자를 확실히 모르는 가정)

--TODO: EMP 테이블에서 직원의 이름(emp\_name)이 PETER인 직원의 모든 정보를 조회하시오.

select \* from emp

where upper(emp\_name)='PETER';

--TODO: EMP 테이블에서 업무(job)가 'Sh\_Clerk' 인 직원의의 ID(emp\_id), 이름(emp\_name), 업무(job), 급여(salary)를 조회

select emp\_id, emp\_name, job, salary from emp

where initcap(job)='Sh\_Clerk';

--TODO: 직원 이름(emp\_name) 의 자릿수를 15자리로 맞추고 글자수가 모자랄 경우 공백을 앞에 붙여 조회. 끝이 맞도록 조회

select lpad(emp\_name,15, ' ') a from emp;-- lpad(emp\_name,15) 공백생략 가능

--TODO: EMP 테이블에서 모든 직원의 이름(emp\_name)과 급여(salary)를 조회.

--(단, "급여(salary)" 열을 길이가 7인 문자열로 만들고, 길이가 7이 안될 경우 왼쪽부터 빈 칸을 '\_'로 채우시오. EX) \_\_\_\_\_\_5000) -LPAD() 이용

select emp\_name, lpad(salary,7,'\_') salary from emp;

-- TODO: EMP 테이블에서 이름(emp\_name)이 10글자 이상인 직원들의 이름(emp\_name)과 이름의 글자수 조회

select emp\_name, length(emp\_name) from emp

where length(emp\_name)>=10;

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

함수 - 숫자관련 함수

round(값, 자릿수) : 자릿수에서 반올림 (양수 - 실수부, 음수 - 정수부)

trunc(값, 자릿수) : 자릿수에서 절삭(양수 - 실수부, 음수 - 정수부)

ceil(값) : 올림, 정수

floor(값) : 내림, 정수

mod(나뉘는수, 나누는수) : 나눗셈의 나머지 연산

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

select 50.123, round(50.128,2)반올림, ceil(50.128) 올림, floor(50.128)내림, trunc(50.128,2)절삭 from dual;

select round(12345,-1), --10단위 이하에서 반올림

round(12378, -2) from dual; --100단위 이하에서 반올림

-- 자리수 : 0(단단위-기본값), 음수 > 실수부, 양수>정수부

select trunc(1234.56,-2),

trunc(1234.56,-1),

trunc(1234.56,0),

trunc(1234.56,1),

trunc(1234.56,2)

from dual;

select mod(10,3) from dual; --10/3의 나머지

--TODO: EMP 테이블에서 각 직원에 대해 직원ID(emp\_id), 이름(emp\_name), 급여(salary) 그리고 15% 인상된 급여(salary)를 조회하는 질의를 작성하시오.

--(단, 15% 인상된 급여는 올림해서 정수로 표시하고, 별칭을 "SAL\_RAISE"로 지정.)

select emp\_id, emp\_name, salary, ceil(salary\*1.15) SAL\_RAISE from emp;

--TODO: 위의 SQL문에서 인상 급여(sal\_raise)와 급여(salary) 간의 차액을 추가로 조회 (직원ID(emp\_id), 이름(emp\_name), 15% 인상급여, 인상된 급여와 기존 급여(salary)와 차액)

select emp\_id, emp\_name, salary, ceil(salary\*1.15) SAL\_RAISE, ceil(salary\*1.15)-salary from emp;

-- TODO: EMP 테이블에서 커미션이 있는 직원들의 직원\_ID(emp\_id), 이름(emp\_name), 커미션비율(comm\_pct), 커미션비율(comm\_pct)을 8% 인상한 결과를 조회.

--(단 커미션을 8% 인상한 결과는 소숫점 이하 2자리에서 반올림하고 별칭은 comm\_raise로 지정)

select emp\_id, emp\_name, comm\_pct, round(comm\_pct\*1.08,2) comm\_raise from emp where comm\_pct is not null;

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

함수 - 날짜관련 계산 및 함수

Date +- 정수 : 날짜 계산.

sysdate : 현재 시스템 시간을 불러옴(실행시점의 일시) date타입으로 저장

systimestamp: 실행시점의 일시. timestamp타입으로 저장(밀리초까지)

months\_between(d1, d2) -경과한 개월수(d1이 최근, d2가 과거)

add\_months(d1, 정수) - 정수개월 지난 날짜. 마지막 날짜의 1개월 후는 달의 마지막 날이 된다.

next\_day(d1, '금요일') - d1이후 첫번째 금요일의 날짜. 요일은 한글(locale)로 지정한다.

last\_day(d) - d 달의 마지막날.

extract(year|month|day from date) - date에서 year/month/day만 추출

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

select sysdate from dual;

select to\_char(sysdate, 'yyyy/mm/dd hh24:mi:ss') from dual;

select sysdate -10, sysdate+10 from dual;

select add\_months(sysdate, 10), add\_months(sysdate,-10), add\_months(sysdate, 12\*3) from dual;

select months\_between(sysdate, '2015-07-05') , months\_between(sysdate, '2015-07-01') from dual;

select next\_day('&year-&month-01', '금요일') from dual;

select last\_day(sysdate) from dual;

select extract(year from hire\_date), extract(month from hire\_date), extract(day from hire\_date) from emp;

select \*from emp where extract(year from hire\_date)=2004;

--TODO: EMP 테이블에서 부서이름(dept\_name)이 'IT'인 직원들의 '입사일(hire\_date)로 부터 10일전', 입사일과 '입사일로 부터 10일후', 의 날짜를 조회.

select hire\_date-10,hire\_date,hire\_date+10 from emp where dept\_name='IT';

--TODO: 부서가 'Purchasing' 인 직원의 이름(emp\_name), 입사 6개월전과 입사일(hire\_date), 6개월후 날짜를 조회.

select emp\_name, add\_months(hire\_date,-6), hire\_date, add\_months(hire\_date, 6) from emp where dept\_name='Purchasing';

--TODO: EMP 테이블에서 입사일과 입사일 2달 후, 입사일 2달 전 날짜를 조회.

select hire\_date, add\_months(hire\_date,2), add\_months(hire\_date,-2) from emp;

-- TODO: 각 직원의 이름(emp\_name), 근무 개월수 (입사일에서 현재까지의 달 수)를 계산하여 조회.

--(단 근무 개월수가 실수 일 경우 정수로 반올림. 근무개월수 내림차순으로 정렬.)

select emp\_name, round(months\_between(sysdate, hire\_date))||'개월' from emp order by 2 desc;

--TODO: 직원 ID(emp\_id)가 100 인 직원의 입사일 이후 첫번째 금요일의 날짜를 구하시오.

select next\_day(hire\_date, '금요일') from emp where emp\_id=100;

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

함수 - 변환 함수

to\_char() : 숫자형, 날짜형을 문자형으로 변환

to\_number() : 문자형을 숫자형으로 변환

to\_date() : 문자형을 날짜형으로 변환

-구문: 함수(변환대상값, 형태)

-to\_char()의 형태: 변환 처리 결과의 형태(출력형태)

-to\_number()/to\_date(): 변환할 대상의 형태

형식(format)문자

숫자 : 0(남는 자리 0처리), 9(남는 자리 공백처리) ex) 533> 0000> 0533, 546>9999>546

fm으로 시작하면 9일경우 공백을 제거,

'.'(정수/실수부 구분자) , ','(정수부 단위구분자),

'L'(로컬통화기호), '$'(통화표시)

일시 : yyyy(연도 4자리, yy:연도 2자리(2000년대)),

rr(연도 2자리(50이상:90년대, 50미만:2000년대)),

mm(월 2자리), ex) 04,11

dd(일 2자리),

hh24(시간 00~23 2자리), hh(01~12)

mi, (분 2자리)

ss, (초 2자리)

day(요일),

am 또는 pm (오전/오후)

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

select '10' + 10 from dual; --oracle에서 실행 할 때 알아서 문자열을 숫자로 바꿈 ('10'+10>>10+10)

select to\_date('10','yy') - 10 from dual; --위의 경우 때문에 10년도에서 10일을 뺀 값을 보고 싶을 때 10만 써서는 제대로 처리가 안되기 때문에 직접 알려줘야한다.

select to\_number('2,900','0,000') from dual;

select to\_char(1234567,'9,999,999')c,

to\_char(12345,'9,999,999')a,

to\_char(12345,'fm9,999,999')b ,

to\_char(1234567,'9,999') d ,--형식문제에서 자릿수는 값보다 크거나 같아야한다. 적으면 #으로 나온다.

to\_char(12345.6789,'99999.99') e ,--반올림

to\_char(5000,'$9,999')f, --달러표시

to\_char(5000,'L9,999')g, --로컬통화기호

to\_char(5000,'9,999')||'원'h --변환함수에서format문자만 사용 가능

--숫자는 형식문자열에서 형식문자를 제외한 것은 사용 못함.

from dual; --fm사용시 공백 제거

select to\_number('4,000','9,999') +10 a,

to\_number('$5,000','$9,999')+50 b

from dual;

select sysdate,

to\_char(sysdate) a,

to\_char(sysdate,'yyyy') b,

to\_char(sysdate,'yy') c,

to\_char(sysdate,'mm') d,

to\_char(sysdate, 'yyyy-mm-dd') e

from dual;

select to\_char(sysdate,'hh24') a,

to\_char(sysdate,'hh')b,

to\_char(sysdate,'hh24:mi:ss') c,

to\_char(sysdate,'yyyy-mm-dd hh24:mi:ss') d,

to\_char(sysdate, 'hh am pm') e, --am이나 pm이나 현재 오전인지 오후인지 확인한다.

to\_char(sysdate, 'mm-dd day') f

from dual;

--date 관련 형식문자를 제외한 문자열은 " " 로 감싸서 넣어줄 수 있다.

select to\_char(sysdate,'yyyy"년"')a,

to\_char(sysdate,'yyyy"년" mm"월" dd"일" hh24"시" mi"분" ss"초"')b

from dual;

select to\_char(sysdate, 'w') 몇번째주,

to\_char(sysdate,'ww') 일년중몇번째주,

to\_char(sysdate,'q') "4분기 중 몇번째분기", --1분기:1~3, 2분기:4~6, 3분기:7~9, 4분기:10~12

to\_char(sysdate,'d') "요일을 숫자로 변환" --일:1 월:2 화:3~ 토:7

from dual;

select to\_date('20191010','yyyymmdd')-10,

to\_date('201905','yyyymm') --default는 첫번째 값(일시분초01 00 00 00)

from dual;

-- EMP 테이블에서 업무(job)에 "CLERK"가 들어가는 직원들의 ID(emp\_id), 이름(name), 업무(job), 급여(salary)를 조회

--(급여는 단위 구분자 , 를 사용하고 앞에 $를 붙여서 출력.)

select emp\_id, emp\_name, job, to\_char(salary,'fm$99,999.99') salary, salary

from emp

where job like '%CLERK%';

-- 문자열 '20030503' 를 2003년 05월 03일 로 출력.

-->char->char; char->date->char

select to\_char(to\_date('20030503','yyyymmdd'),'yyyy"년" mm"월" dd"일"') a

from dual;

-- TODO: 부서명(dept\_name)이 'Finance'인 직원들의 ID(emp\_id), 이름(emp\_name)과 입사년도(hire\_date) 4자리만 출력하시오. (ex: 2004);

--to\_char()

select emp\_id, emp\_name, to\_char(hire\_date, 'yyyy')입사년도 from emp where dept\_name='Finance';

--TODO: 직원들중 11월에 입사한 직원들의 직원ID(emp\_id), 이름(emp\_name), 입사일(hire\_date)을 조회

--to\_char()

select emp\_id, emp\_name, hire\_date

from emp

where to\_char(hire\_date,'mm')='11';

--where extract(month from hire\_date)='11';

--TODO: 2006년에 입사한 모든 직원의 이름(emp\_name)과 입사일(yyyy-mm-dd 형식)을 입사일(hire\_date)의 오름차순으로 조회

--to\_char()

select emp\_name, to\_char(hire\_date, 'yyyy-mm-dd') hire\_date from emp where to\_char(hire\_date,'yyyy')='2006' order by hire\_date;

--TODO: 2004년 01월 이후 입사한 직원 조회의 이름(emp\_name)과 입사일(hire\_date) 조회

select emp\_name, hire\_date from emp

where hire\_date >= to\_date('200401','yyyymm')

--where to\_char(hire\_date, 'yyyymm')>='200401')

order by 2;

--TODO: 문자열 '20100107232215' 를 2010년 01월 07일 23시 22분 15초 로 출력. (dual 테입블 사용)

select to\_char(to\_date('20100107232215' ,'yyyymmddhh24miss'),'yyyy"년" mm"월" dd"일" hh24"시" mi"분" ss"초"')

from dual;

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

함수 - null 관련 함수

NVL(expr1, expr2) : expr1-널이 아니면 expr1반환 널이면 expr2반환

NVL2(expr, nn, null) : expr이 null이 아니면 nn, 널이면 세번째

nullif(ex1, ex2) : 둘이 같으면 null, 다르면 ex1

coalesce(ex1,ex2,ex3,...) ex1~exn 중 null이 아닌 첫번째 값 변환.

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

select nvl(10,0),nvl(null,70) from dual; --앞에가 null일 경우 뒤에꺼

select nvl2(10,1,0), nvl2(null,1,0) from dual;

select nullif(10,10), nullif(10,20) from dual; --같으면 null 아니면 첫번째값

select coalesce(null, null, 10, null,23) from dual;

select emp\_id, comm\_pct, mgr\_id, dept\_name ,

coalesce(comm\_pct, mgr\_id) "notnull" --한 컬럼에 들어가는 값들의 타입을 같아야한다. coalesce(comm\_pct, mgr\_id, dept\_name)는 에러

from emp

where emp\_id in(150, 100, 101);

-- EMP 테이블에서 직원 ID(emp\_id), 이름(emp\_name), 급여(salary), 커미션비율(comm\_pct)을 조회. 단 커미션비율이 NULL인 직원은 0이 출력되도록 한다..

select emp\_id, emp\_name, salary, nvl(comm\_pct,0), nvl2(comm\_pct\*salary,'커미션있음','커미션없음')"커미션유무"

from emp;

--TODO: EMP 테이블에서 직원의 ID(emp\_id), 이름(emp\_name), 업무(job), 부서(dept\_name)을 조회. 부서가 없는 경우 '부서미배치'를 출력.

select emp\_id, emp\_name, job, nvl(dept\_name,'부서미배치') from emp;

--TODO: EMP 테이블에서 직원의 ID(emp\_id), 이름(emp\_name), 급여(salary), 커미션 (salary \* comm\_pct)을 조회. 커미션이 없는 직원은 0이 조회되록 한다.

select emp\_id, emp\_name, salary, to\_char(nvl(salary\*comm\_pct,0),'$9,999') commition from emp;

/\*

nvl(10,0)

if 10 이 null이 아니면

then 10

else 0

nvl2(10,1,0)

if 10 이 null이 아니면

then 1

else 0

\*/

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

DECODE함수와 CASE 문

decode(컬럼, [비교값, 출력값, ...] , else출력)

ex) decode(emp\_id,100,'a',200,'b',300,'c','d') => emp\_id가 100이면 a, 200이면 b 300이면 c 아무것도 아니면 d (d를 주지 않으면 null반환)

case문 동등비교

case 컬럼 when 비교값 then 출력값

[when 비교값 then 출력값]

[else 출력값]

end

case문 조건문

case when 조건 then 출력값

[when 조건 then 출력값]

[else 출력값]

end

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

select decode(dept\_name, null, '부서없음') from emp where dept\_name is null;

select case dept\_name when null then '부서없음' end a,

case when dept\_name is null then '부서없음' end b from emp where dept\_name is null;

--null의 경우 반드시 동등비교가 아닌 조건으로 is null을 사용해야한다.

--EMP테이블에서 급여와 급여의 등급을 조회하는데 급여 등급은 10000이상이면 '1등급', 10000미만이면 '2등급' 으로 나오도록 조회

select salary, case when salary>=10000 then '1등급'

else '2등급'

end 등급

from emp

order by 1;

--TODO: EMP 테이블에서 업무(job)이 'AD\_PRES'거나 'FI\_ACCOUNT'거나 'PU\_CLERK'인 직원들의 ID(emp\_id), 이름(emp\_name), 업무(job)을 조회.

-- 업무(job)가 'AD\_PRES'는 '대표', 'FI\_ACCOUNT'는 '회계', 'PU\_CLERK'의 경우 '구매'가 출력되도록 조회

--decode문

select emp\_id, emp\_name, decode(job,'AD\_PRES','대표',

'FI\_ACCOUNT','회계',

'PU\_CLERK','구매','기타') job, job

from emp

where job in ('AD\_PRES','FI\_ACCOUNT','PU\_CLERK','IT\_PROG');

--case문

select emp\_id, emp\_name, case job when 'AD\_PRES' then '대표'

when 'FI\_ACCOUNT' then '회계'

when 'PU\_CLERK' then '구매'

else '기타'

end job2, job

from emp

where job in ('AD\_PRES','FI\_ACCOUNT','PU\_CLERK','IT\_PROG');

--TODO: EMP 테이블에서 부서이름(dept\_name)과 급여 인상분을 조회. 급여 인상분은 부서이름이 'IT' 이면 급여(salary)에 10%를 'Shipping' 이면 급여(salary)의 20%를 'Finance'이면 30%를 나머지는 0을 출력

-- decode 와 case문을 이용해 조회

--decode문

select dept\_name, decode(dept\_name, 'IT', salary\*0.1, 'Shipping', salary\*0.2, 'Finance', salary\*0.3,0) "급여 인상분"

from emp;

--case문

select dept\_name, case dept\_name when 'IT' then salary\*0.1

when 'Shipping' then salary\*0.2

when 'Finance' then salary\*0.3

else 0 end "급여 인상분"

from emp;

--TODO: EMP 테이블에서 직원의 ID(emp\_id), 이름(emp\_name), 급여(salary), 인상된 급여를 조회한다.

--단 급여 인상율은 급여가 5000 미만은 30%, 5000이상 10000 미만는 20% 10000 이상은 10% 로 한다.

select emp\_id, emp\_name, salary,

case when salary<5000 then salary\*0.3

when salary<10000 then salary\*0.2

else salary\*0.1 end "인상된 급여"

from emp;

--decode()/case 를 이용한 정렬

-- 직원들의 모든 정보를 조회한다. 단 정렬은 업무(job)가 'ST\_CLERK', 'IT\_PROG', 'PU\_CLERK', 'SA\_MAN' 순서대로 먼저나오도록 한다. (나머지 JOB은 상관없음)

select \* from emp

order by decode(job, 'ST\_CLERK',1,

'IT\_PROG',2,

'PU\_CLERK', 3,

'SA\_MAN',4,9999),salary desc;

select \* from emp

order by case job when 'ST\_CLERK' then 1

when 'IT\_PROG' then 2

when 'PU\_CLERK' then 3

when 'SA\_MAN' then 4

else 9999 end;