

# 함수 & 조인

## Functions

### Single-Row Functions

- Lower : 소문자로 만들기
- Upper: 대문자로 만들기

예시)

```
select lower(job) from emp;
```

의 결과 x

SQL | 인출된 모든 행: 14(0초)

	LOWER(JOB)
1	clerk
2	salesman
3	salesman
4	manager
5	salesman
6	manager
7	manager
8	analyst
9	president
10	salesman
11	clerk
12	clerk
13	analyst

- concat: 문자열 합치기(두개 단어만 쓸 수 있다.)

```
select concat(concat('A', '-'), ename) from emp;
```

의 결과 x	
SQL   인출된 모든 행: 14(0.002초)	
CONCAT(CONCAT('A','-'),ENAME)	
1 A-SMITH	
2 A-ALLEN	
3 A-WARD	
4 A-JONES	
5 A-MARTIN	
6 A-BLAKE	
7 A-CLARK	
8 A-SCOTT	
9 A-KING	
10 A-TURNER	
11 A-ADAMS	
12 A-JAMES	
13 A-FORD	

→ 함수는 무한 중첩이 가능하다.

→ concat을 중첩시키는 것보다 ||를 쓰는 것이 훨씬 더 효율적이라고 볼 수 있다.

- substr: 인덱스만큼 꺼내라(start자리에서 4개를 뽑아라)

예시) 휴대폰 번호에서 가운데 번호를 꺼내라

```
select substr('010-1234-5678',5, 4) from dual;
```

의 결과 x	
SQL   인출된 모든 행: 1(0초)	
SUBSTR('010-1234-5678',5,4)	
1 1234	

- length('String') : 길이 반환
- instr('string', 'r') : r이 몇번째에 있는지 인덱스 반환

- lpad(sal, 10,'\*'): pad는 채우는것. lpad는 왼쪽으로 채우고 rpad는 오른쪽을 채운다. 10글자가 안되면 나머지는 \*로 채워라.

```
select lpad(sal, 10, '*') from emp;
```

실행 결과 x

SQL | 인출된 모든 행: 14(0초)

	LPAD(SAL,10,'*')
4	*****2975
5	*****1250
6	*****2850
7	*****2450
8	*****3000
9	*****5000
10	*****1500
11	*****1100
12	*****950
13	*****3000

- trim('s' from 'ssmith') : s라는 글자를 모두 버려
- replace

```
--A라는 글자를 000으로 대체해라
select replace('AACK and AUE', 'A', '000') from dual;

select '-'||replace('A BCD', ' ', '')||'-' from dual;
select '-'||replace(' A BCD', ' ', '')||'-' from dual;
select '-'||replace(' A BCD ', ' ', '')||'-' from dual;
```

- round(45.926, 2): 두번째 자리까지 반올림해서 보여줘

- trunc: 두번째 자리까지 버림 해서 보여줘
- mod: 나눗셈 나머지 반환
- to\_char:

```
--9는 숫자를 의미. , 찍기 표현
select to_char(10000000000, '999,999,999,999') from dual;
```

질의 결과 x

SQL | 인출된 모든 행: 1(0초)

	TO_CHAR(10000000000,'999,999,999,999')
1	10,000,000,000

- null 처리

```
--null 처리
select avg(comm), avg(nvl(comm,0)) from emp;
```

질의 결과 x

SQL | 인출된 모든 행: 1(0초)

	AVG(COMM)	AVG(NVL(COMM,0))
1	550	157.142857142857142857142857142857142857

- decode

```
select deptno, decode(deptno, 10, '십',
                        20, '이십',
                        30, '삼십',
                        '대기') from emp;
```

DEPTNO	DECODE(DEPTNO,10,'십',20,'이십',30,'삼십','대기')
4	20 이십
5	30 삼십
6	30 삼십
7	10 십
8	20 이십
9	10 십
10	30 삼십
11	20 이십
12	30 삼십
13	20 이십
14	10 십

— 커미션이 null이면 0으로 그렇지 않으면 그대로 출력

```
select comm, decode(comm, null, 0,
                    comm) from emp;
```

결과 x

SQL | 인출된 모든 행: 14(0.001초)

COMM	DECODE(COMM, NULL, 0, COMM)
(null)	0
300	300
500	500
(null)	0
1400	1400
(null)	0
(null)	0
(null)	0

—커미션을 받는 사원이면 우수, 그렇지 않으면 평사원

```
--커미션을 받는 사원이면 우수, 그렇지 않으면 평사원
select comm, decode(nvl(comm, 0), 0, '평사원',
                    '우수') as label from emp;
```

의 결과 x

SQL | 인출된 모든 행: 14(0초)

COMM	LABEL
(null)	평사원
300	우수
500	우수
(null)	평사원
1400	우수
(null)	평사원
(null)	평사원
(null)	평사원

—커미션을 받는 사원이면 comm값, 그렇지 않으면 평사원

```
--커미션을 받는 사원이면 comm값, 그렇지 않으면 평사원
select comm, decode(nvl(comm,0), 0, '평사원',
                    comm) from emp;
```

의 결과 x

SQL | 인출된 모든 행: 14(0초)

	COMM	DECODE(NVL(COMM,0),0,'평사원',COMM)
1	(null)	평사원
2	300	300
3	500	500
4	(null)	평사원
5	1400	1400
6	(null)	평사원
7	(null)	평사원
8	(null)	평사원

- case





```
select job, avg(sal)
from emp
group by job;
```

[illegible]

## Group by

: where 안에는 group 함수 못온다. (예를 들어, avg(sal) 같은거.)

: group 함수는 having 뒤에 온다.

```
-- 사원정보 테이블에서 각 부서별 평균 급여가 2000이상인 부서와
-- 평균 급여를 출력하시오.
select deptno, avg(sal)
from emp
group by deptno
having avg(sal) >= 2000;
```

[illegible]

- ‘일반컬럼’과 ‘그룹 함수’를 같이 조회할 경우 반드시 group by ‘일반컬럼’을 사용해야.

```
select job, sum(sal)
from emp
where job!='SALESMAN'
group by job
having sum(sal)>5000
order by sum(sal);
```

	JOB	SUM(SAL)
1	ANALYST	6000
2	MANAGER	8275

—각 부서별 평균 급여 중 가장 많은 평균 급여를 받는 부서의 부서, 평균 급여를 출력하시오. (서브 쿼리 사용)

```
select max(asal)
from (select deptno, avg(sal) as asal from emp group by deptno)
;
```

**MAX(ASAL)**

1 2916.667

## 조인

- 1) Inner Join
- 2) Self Join
- 3) Outer Join
  - Full
  - Left
  - Right

## 기타 설명

: 조인은 틀마다 명령어가 다를 수 있다.