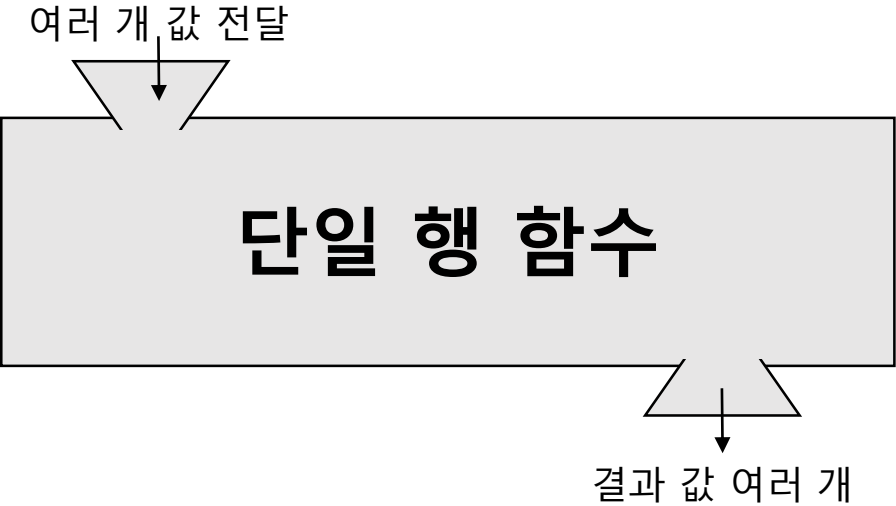
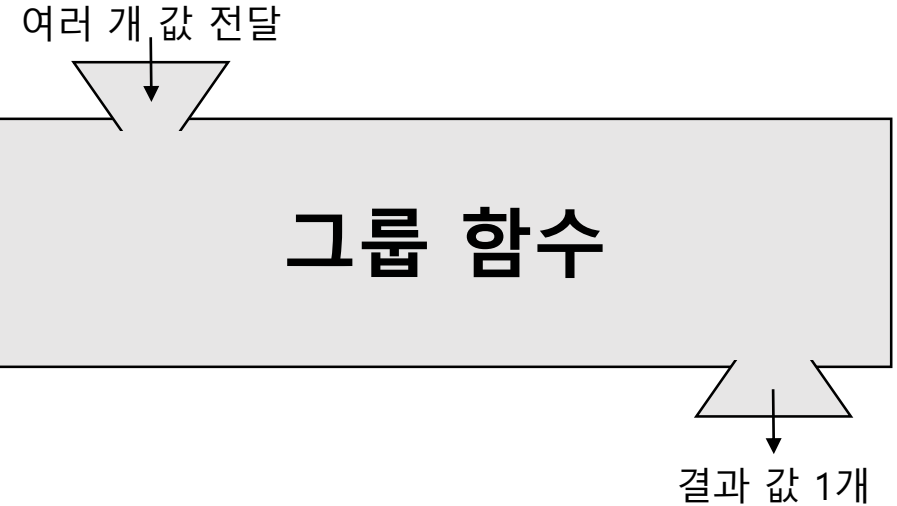


# 함수

# ▶ 함수(Function)

하나의 큰 프로그램에서 반복적으로 사용되는 부분들을 분리하여 작성해 놓은 작은 서브 프로그램 호출하며 값을 전달하면 결과를 리턴하는 방식으로 사용

## ✓ 유형

|   |  |
|---|--|
|  <p>여러 개 값 전달</p> <p><b>단일 행 함수</b></p> <p>결과 값 여러 개</p> |  <p>여러 개 값 전달</p> <p><b>그룹 함수</b></p> <p>결과 값 1개</p> |
| 단일 행 함수   | 그룹 함수  |
| 각 행마다 반복적으로 적용되어<br>입력 받은 행의 개수만큼 결과 반환   | 특정 행들의 집합으로 그룹이 형성되어 적용됨<br>그룹 당 1개의 결과 반환   |

# ▶ 문자 처리 함수

| 구분          | 입력 값 타입   | 리턴 값 타입   | 설명  |
|-------------|-----------|-----------|---|
| LENGTH      | CHARACTER | NUMBER    | 문자열 길이 반환                                       |
| LENGTHB     |           |           | 문자열의 바이트 크기 반환                                  |
| INSTR       |           |           | 특정 문자의 위치 반환                                    |
| INSTRB      |           |           | 특정 문자의 위치 바이트 크기 반환                             |
| LPAD/RPAD   | CHARACTER | CHARACTER | 지정 문자열을 입력한 크기만큼 본 문자열의 왼쪽/오른쪽부터 채워서 생성된 문자열 리턴 |
| LTRIM/RTRIM |           |           | 왼쪽/오른쪽부터 지정한 문자를 잘라내고 남은 문자 리턴                  |
| TRIM        |           |           | 왼쪽/오른쪽/양쪽부터 지정한 문자를 잘라내고 남은 문자 리턴               |
| SUBSTR      |           |           | 지정한 위치에서 지정한 길이만큼 문자 잘라내어 리턴                    |
| SUBSTRB     |           |           | 지정한 위치에서 지정한 바이트만큼 문자를 잘라내어 리턴                  |
| LOWER       |           |           | 전달받은 문자/문자열을 소문자로 변환하여 리턴                       |
| UPPER       |           |           | 전달받은 문자/문자열을 대문자로 변환하여 리턴                       |
| INITCAP     |           |           | 전달받은 문자/문자열의 첫 글자만 대문자로, 나머지는 소문자로 변환하여 리턴      |
| CONCAT      |           |           | 인자로 전달받은 두 개의 문자/문자열을 합쳐서 리턴                    |
| REPLACE     |           |           | 전달받은 문자열 중에서 지정한 문자를 인자로 전달받은 문자로 변환하여 리턴       |

# ▶ 문자 처리 함수

## ✓ LENGTH

주어진 컬럼 값/문자열의 길이(문자 개수) 반환

| 작성법                           | 리턴 값 타입   |
|-------------------------------|-----------|
| <b>LENGTH</b> (CHAR   STRING) | CHARACTER |

\* CHAR | STRING : 문자 타입 컬럼 또는 문자열

## ✓ 예시

```
SELECT EMP_NAME, LENGTH(EMP_NAME), EMAIL, LENGTH(EMAIL)
FROM EMPLOYEE;
```

|    | EMP_NAME | LENGTH(EMP_NAME) | EMAIL            | LENGTH(EMAIL) |
|----|----------|------------------|------------------|---------------|
| 1  | 선동일      | 3                | sun_di@kh.or.kr  | 15            |
| 2  | 송종기      | 3                | song_jk@kh.or.kr | 16            |
| 3  | 노웅철      | 3                | no_hc@kh.or.kr   | 14            |
| 4  | 송은희      | 3                | song_eh@kh.or.kr | 16            |
| 5  | 유재식      | 3                | yoo_js@kh.or.kr  | 15            |
| 6  | 정중하      | 3                | jung_jh@kh.or.kr | 16            |
| 7  | 박나라      | 3                | pack_nr@kh.or.kr | 16            |
| 8  | 하미유      | 3                | ha_iy@kh.or.kr   | 14            |
| 9  | 김해솔      | 3                | kim_hs@kh.or.kr  | 15            |
| 10 | 시보서      | 3                | si_bo@kh.or.kr   | 15            |

# ▶ 문자 처리 함수

## ✓ INSTR

지정한 위치부터 지정한 숫자 번째로 나타나는 문자의 시작 위치 반환

| 작성법  | 리턴 값 타입 |
|--|---------|
| <b>INSTR</b> (STRING, STR, [POSITION],[OCCURRENCE]]) | NUMBER  |

\* STRING : 문자 타입 컬럼 또는 문자열

\* STR : 찾으려는 문자열

\* POSITION : 찾을 위치 시작 값(기본 값 1)

POSITION > 0이면 STRING의 시작부터 끝 방향으로 찾고

POSITION < 0이면 STRING의 끝부터 시작 방향으로 찾음

\* OCCURRENCE : SUBSTRING이 반복될 때 지정하는 빈도(기본 값 1), 음수 사용 불가

## ✓ 예시

- EMAIL 컬럼의 문자열 중 '@'의 위치를 구하시오.

**SELECT** EMAIL, **INSTR**(EMAIL, '@', -1, 1) 위치

**FROM** EMPLOYEE;

|    | EMAIL            | 위치 |
|----|------------------|----|
| 1  | sun_di@kh.or.kr  | 7  |
| 2  | song_jk@kh.or.kr | 8  |
| 3  | no_hc@kh.or.kr   | 6  |
| 4  | song_eh@kh.or.kr | 8  |
| 5  | yoo_js@kh.or.kr  | 7  |
| 6  | jung_jh@kh.or.kr | 8  |
| 7  | pack_nr@kh.or.kr | 8  |
| 8  | ha_iy@kh.or.kr   | 6  |
| 9  | kim_hs@kh.or.kr  | 7  |
| 10 | sim_bs@kh.or.kr  | 7  |
| 11 | youn_eh@kh.or.kr | 8  |
| 12 | jun_hd@kh.or.kr  | 7  |

# ▶ 문자 처리 함수

## ✓ LTRIM/RTRIM

주어진 컬럼, 문자열의 왼쪽/오른쪽에서 지정한 STR에 포함된 모든 문자를 제거한 나머지 반환

| 작성법   | 리턴 값 타입   |
|---|-----------|
| <b>LTRIM</b> (STRING, STR) / <b>RTRIM</b> (STRING, STR) | CHARACTER |

\* STRING : 문자 타입 컬럼 또는 문자열

\* STR : 제거하려는 문자(열), 생략 시 공백문자

## ✓ 예시

```
SELECT EMP_NAME, LTRIM(PHONE, '010'), RTRIM(EMAIL, '@kh.or.kr')
FROM EMPLOYEE;
```

|    | EMP_NAME | LTRIM(P... | RTRIM.. |
|----|----------|------------|---------|
| 1  | 선동일      | 99546325   | sun_di  |
| 2  | 송중기      | 45686656   | song_j  |
| 3  | 노웅철      | 66656263   | no_hc   |
| 4  | 송은희      | 77607879   | song_e  |
| 5  | 유재식      | 99999129   | yoo_js  |
| 6  | 정중하      | 36654875   | jung_j  |
| 7  | 박나라      | 96935222   | pack_n  |
| 8  | 하이유      | 36654488   | ha_iy   |
| 9  | 김해솔      | 78634444   | kim_hs  |
| 10 | 심봉선      | 3654485    | sim_bs  |
| 11 | 으으해      | 78864233   | youn_e  |

# ▶ 문자 처리 함수

## ✓ LTRIM

| 수행 문장   | 결과     |
|---|--------|
| SELECT LTRIM(' KH') FROM DUAL;                  | KH     |
| SELECT LTRIM(' KH' , ' ') FROM DUAL;            | KH     |
| SELECT LTRIM('000123456' , '0') FROM DUAL;      | 123456 |
| SELECT LTRIM('123123KH' , '123') FROM DUAL;     | KH     |
| SELECT LTRIM('123123KH123' , '123') FROM DUAL;  | KH123  |
| SELECT LTRIM('ACABACCKH', 'ABC') FROM DUAL;     | KH     |
| SELECT LTRIM('5782KH', '0123456789') FROM DUAL; | KH     |

## ✓ RTRIM

| 수행 문장   | 결과     |
|---|--------|
| SELECT RTRIM('KH ') FROM DUAL;                  | KH     |
| SELECT RTRIM('KH ' , ' ') FROM DUAL;            | KH     |
| SELECT RTRIM('123456000' , '0') FROM DUAL;      | 123456 |
| SELECT RTRIM('KH123123' , '123') FROM DUAL;     | KH     |
| SELECT RTRIM('123KH123123' , '123') FROM DUAL;  | 123KH  |
| SELECT RTRIM('KHACABACC', 'ABC') FROM DUAL;     | KH     |
| SELECT RTRIM('KH5782', '0123456789') FROM DUAL; | KH     |

# ▶ 문자 처리 함수

## ✓ TRIM

주어진 컬럼, 문자열의 앞/뒤/양쪽에 있는 지정한 문자를 제거한 나머지 반환

| 작성법   | 리턴 값 타입   |
|---|-----------|
| <b>TRIM</b> ( STRING )<br><b>TRIM</b> ( CHAR FROM STRING)<br><b>TRIM</b> ( <b>LEADING</b>   <b>TRAILING</b>   <b>BOTH</b> [CHAR] FROM STRING) | CHARACTER |

\* STRING : 문자 타입 컬럼 또는 문자열

\* CHAR : 제거하려는 문자(열), 생략 시 공백문자

\* LEADING : TRIM할 CHAR의 위치 지정, 앞(LEADING)/뒤(TRAILING)/양쪽(BOTH) 지정 가능(기본 값 양쪽)

## ✓ 예시

| 수행 문장  | 결과       |
|--|----------|
| SELECT TRIM(' KH ') FROM DUAL;                         | KH       |
| SELECT TRIM('Z' FROM 'ZZZKHZZZ') FROM DUAL;            | KH       |
| SELECT TRIM(LEADING 'Z' FROM 'ZZZ123456') FROM DUAL;   | 123456   |
| SELECT TRIM(TRAILING '1' FROM 'KH111111') FROM DUAL;   | KH       |
| SELECT TRIM(BOTH '3' FROM '333KH333333') FROM DUAL;    | KH       |
| SELECT TRIM(LEADING '2' FROM '222KH222222') FROM DUAL; | KH222222 |



## ▶ 문자 처리 함수

### ✓ SUBSTR

컬럼이나 문자열에서 지정한 위치부터 지정한 개수의 문자열을 잘라내어 반환

| 작성법  | 리턴 값 타입   |
|--|-----------|
| <b>SUBSTR</b> ( STRING, POSITION, [LENGTH] ) | CHARACTER |

\* STRING : 문자 타입 컬럼 또는 문자열

\* POSITION : 문자열을 잘라낼 위치로 양수면 시작방향에서 지정한 수만큼, 음수면 끝 방향에서 지정한 수만큼의 위치 의미

\* LENGTH : 반환할 문자 개수(생략 시 문자열의 끝까지 의미, 음수면 NULL 리턴)

### ✓ 예시

| 수행 문장  | 결과       |
|--|----------|
| SELECT SUBSTR('SHOWMETHEMONEY', 5, 2) FROM DUAL;   | ME       |
| SELECT SUBSTR('SHOWMETHEMONEY', 7) FROM DUAL;      | THEMONEY |
| SELECT SUBSTR('SHOWMETHEMONEY', 1, 6) FROM DUAL;   | SHOWME   |
| SELECT SUBSTR('SHOWMETHEMONEY', -8, 3) FROM DUAL;  | THE      |
| SELECT SUBSTR('SHOWMETHEMONEY', -10, 2) FROM DUAL; | ME       |
| SELECT SUBSTR('쇼우 미 더 머니', 2, 5) FROM DUAL;        | 우 미 더    |

# ▶ 문자 처리 함수

## ✓ LPAD/RPAD

주어진 컬럼, 문자열에 임의의 문자열을 왼쪽/오른쪽에 덧붙여 길이 N의 문자열 반환

| 작성법   | 리턴 값 타입   |
|---|-----------|
| <b>LPAD</b> (STRING, N, [STR]) / <b>RPAD</b> (STRING, N, [STR]) | CHARACTER |

\* STRING : 문자 타입 컬럼 또는 문자열

\* N : 반환할 문자(열)의 길이(바이트), 원래 STRING의 길이보다 작다면 N만큼 잘라서 표시

\* STR : 덧붙이려는 문자(열), 생략 시 공백문자

## ✓ 예시

```
SELECT LPAD(EMAIL, 20, '#')
```

```
FROM EMPLOYEE;
```

```
SELECT RPAD(EMAIL, 20, '#')
```

```
FROM EMPLOYEE;
```

| LPAD(EMAIL,20,'#')       | RPAD(EMAIL,20,...)       |
|--------------------------|--------------------------|
| 1 #####sun_di@kh.or.kr   | 1 sun_di@kh.or.kr#####   |
| 2 #####song_jk@kh.or.kr  | 2 song_jk@kh.or.kr#####  |
| 3 #####no_hc@kh.or.kr    | 3 no_hc@kh.or.kr#####    |
| 4 #####song_eh@kh.or.kr  | 4 song_eh@kh.or.kr#####  |
| 5 #####yoo_js@kh.or.kr   | 5 yoo_js@kh.or.kr#####   |
| 6 #####jung_jh@kh.or.kr  | 6 jung_jh@kh.or.kr#####  |
| 7 #####pack_nr@kh.or.kr  | 7 pack_nr@kh.or.kr#####  |
| 8 #####ha_iy@kh.or.kr    | 8 ha_iy@kh.or.kr#####    |
| 9 #####kim_hs@kh.or.kr   | 9 kim_hs@kh.or.kr#####   |
| 10 #####sim_bs@kh.or.kr  | 10 sim_bs@kh.or.kr#####  |
| 11 #####youn_eh@kh.or.kr | 11 youn_eh@kh.or.kr##### |
| 12 #####jun_hd@kh.or.kr  | 12 jun_hd@kh.or.kr#####  |
| 13 #####jang_zw@kh.or.kr | 13 jang_zw@kh.or.kr##### |
| 14 #####ha_dh@kh.or.kr   | 14 ha_dh@kh.or.kr#####   |
| 15 #####bang_ms@kh.or.kr | 15 bang_ms@kh.or.kr##### |

## ▶ 문자 처리 함수

### ✓ LOWER/UPPER/INITCAP

컬럼의 문자 혹은 문자열을 소문자/대문자/첫 글자만 대문자로 변환하여 반환

| 작성법   | 리턴 값 타입   |
|---|-----------|
| <b>LOWER</b> (STRING) / <b>UPPER</b> (STRING) / <b>INITCAP</b> (STRING) | CHARACTER |

\* STRING : 문자 타입 컬럼 또는 문자열

### ✓ 예시

| 수행 문장   | 결과                  |
|---|---------------------|
| SELECT LOWER('Welcome To My World') from dual;  | welcome to my world |
| SELECT UPPER('Welcome To My World') from dual;  | WELCOME TO MY WORLD |
| SELECT INITCAP('welcome to my world')from dual; | Welcome To My World |

# ▶ 숫자 처리 함수

| 구분    | 입력 값 타입 | 리턴 값 타입 | 설명                   |
|-------|---------|---------|----------------------|
| ABS   | NUMBER  | NUMBER  | 절대 값 리턴              |
| MOD   |         |         | 입력 받은 수를 나눈 나머지 값 반환 |
| ROUND |         |         | 특정 자릿수에서 반올림         |
| FLOOR |         |         | 버림(소수점 아래를 잘라냄)      |
| TRUNC |         |         | 특정 자릿수에서 잘라냄         |
| CEIL  |         |         | 올림(소수점 아래에서 올림)      |

# ▶ 숫자 처리 함수

## ✓ ABS

인자로 전달 받은 숫자의 절대값 반환

| 작성법                 | 리턴 값 타입 |
|---------------------|---------|
| <b>ABS</b> (NUMBER) | NUMBER  |

\* NUMBER : 숫자 혹은 숫자 데이터 컬럼

## ✓ 예시

| 수행 문장                        | 결과   |
|------------------------------|------|
| SELECT ABS(10.9) FROM DUAL;  | 10.9 |
| SELECT ABS(-10.9) FROM DUAL; | 10.9 |
| SELECT ABS(10) FROM DUAL;    | 10   |
| SELECT ABS(-10) FROM DUAL;   | 10   |

# ▶ 숫자 처리 함수

## ✓ MOD

인자로 전달 받은 숫자를 나누어 나머지 반환

| 작성법                           | 리턴 값 타입 |
|-------------------------------|---------|
| <b>MOD</b> (NUMBER, DIVISION) | NUMBER  |

\* NUMBER : 숫자 혹은 숫자 데이터 컬럼

\* DIVISION : 나눌 수 혹은 나눌 숫자 데이터 컬럼

## ✓ 예시

| 수행 문장                           | 결과  |
|---------------------------------|-----|
| SELECT MOD(10, 3) FROM DUAL;    | 1   |
| SELECT MOD(-10, 3) FROM DUAL;   | -1  |
| SELECT MOD(10.9, 3) FROM DUAL;  | 1.9 |
| SELECT ABS(10.9, -3) FROM DUAL; | 1.9 |

# ▶ 숫자 처리 함수

## ✓ ROUND

인자로 전달 받은 숫자 혹은 컬럼에서 지정한 위치부터 반올림하여 값 반환

| 작성법  | 리턴 값 타입 |
|--|---------|
| <b>ROUND(NUMBER)</b><br><b>ROUND(NUMBER, POSITION)</b> | NUMBER  |

\* NUMBER : 숫자 혹은 숫자 데이터 컬럼

\* POSITION : 반올림할 위치(생략 시 기본 값 0)

## ✓ 예시

| 수행 문장                                 | 결과       |
|---------------------------------------|----------|
| SELECT ROUND(10.11) FROM DUAL;        | 10       |
| SELECT ROUND(10.18) FROM DUAL;        | 10       |
| SELECT ROUND(10.51) FROM DUAL;        | 11       |
| SELECT ROUND(-10.61) FROM DUAL;       | -11      |
| SELECT ROUND(10.123456, 5) FROM DUAL; | 10.12346 |

# ▶ 숫자 처리 함수

## ✓ FLOOR

인자로 전달 받은 숫자 혹은 컬럼에서 소수점 자리의 수를 버림 후 반환

| 작성법                   | 리턴 값 타입 |
|-----------------------|---------|
| <b>FLOOR</b> (NUMBER) | NUMBER  |

\* NUMBER : 숫자 혹은 숫자 데이터 컬럼

## ✓ 예시

| 수행 문장                           | 결과  |
|---------------------------------|-----|
| SELECT FLOOR(10.11) FROM DUAL;  | 10  |
| SELECT FLOOR(10.18) FROM DUAL;  | 10  |
| SELECT FLOOR(10.51) FROM DUAL;  | 10  |
| SELECT FLOOR(-10.61) FROM DUAL; | -11 |



# ▶ 숫자 처리 함수

## ✓ TRUNC

인자로 전달 받은 숫자 혹은 컬럼에서 지정한 위치부터의 자리의 수를 버리고(절삭) 반환

| 작성법                             | 리턴 값 타입 |
|---------------------------------|---------|
| <b>TRUNC</b> (NUMBER, POSITION) | NUMBER  |

\* NUMBER : 숫자 혹은 숫자 데이터 컬럼

\* POSITION : 버릴 위치(생략 시 기본 값 0)

## ✓ 예시

| 수행 문장                             | 결과   |
|-----------------------------------|------|
| SELECT TRUNC(10.51) FROM DUAL;    | 10   |
| SELECT TRUNC(10.91, 0) FROM DUAL; | 10   |
| SELECT TRUNC(10.91, 1) FROM DUAL; | 10.9 |
| SELECT TRUNC(-10.61) FROM DUAL;   | -10  |

# ▶ 숫자 처리 함수

## ✓ CEIL

인자로 전달 받은 숫자 혹은 컬럼을 올림 후 반환

| 작성법                  | 리턴 값 타입 |
|----------------------|---------|
| <b>CEIL</b> (NUMBER) | NUMBER  |

\* NUMBER : 숫자 혹은 숫자 데이터 컬럼

## ✓ 예시

| 수행 문장                          | 결과  |
|--------------------------------|-----|
| SELECT CEIL(10.11) FROM DUAL;  | 11  |
| SELECT CEIL(10.19) FROM DUAL;  | 11  |
| SELECT CEIL(10.51) FROM DUAL;  | 11  |
| SELECT CEIL(-10.11) FROM DUAL; | -10 |

## ▶ 날짜 처리 함수

| 구분             | 입력 값 타입 | 리턴 값 타입 | 설명                                |
|----------------|---------|---------|-----------------------------------|
| SYSDATE        |         | DATE    | 시스템에 저장된 현재 날짜 반환                 |
| MONTHS_BETWEEN | DATE    | NUMBER  | 두 날짜를 전달받아 몇 개월 차이인지 계산하여 반환      |
| ADD_MONTHS     | DATE    | DATE    | 특정 날짜에 개월 수를 더하여 반환               |
| NEXT_DAY       |         |         | 특정 날짜에서 인자로 받은 요일이 최초로 다가오는 날짜 반환 |
| LAST_DAY       |         |         | 헤딩 달의 마지막 날짜 반환                   |
| EXTRACT        |         |         | 년, 월, 일 정보를 추출하여 반환               |

## ▶ 날짜 처리 함수

### ✓ SYSDATE

시스템에 저장되어 있는 현재 날짜 반환

| 작성법     | 리턴 값 타입 |
|---------|---------|
| SYSDATE | DATE    |

### ✓ 예시

```
SELECT SYSDATE  
FROM DUAL;
```

|   |          |
|---|----------|
|   | SYSDATE  |
| 1 | 18/12/20 |

✓ MONTHS\_BETWEEN

| 작성법                                  | 리턴 값 타입 |
|--------------------------------------|---------|
| <b>MONTHS_BETWEEN</b> (DATE1, DATE2) | NUMBER  |

\* DATE2 : 개월 수를 구하려는 날짜

## ✓ 예시

[illegible]

## ▶ 날짜 처리 함수

### ✓ ADD\_MONTHS

인자로 전달받은 날짜에 인자로 받은 숫자만큼 개월 수를 더하여 특정 날짜 반환

| 작성법                              | 리턴 값 타입 |
|----------------------------------|---------|
| <b>ADD_MONTHS</b> (DATE, NUMBER) | DATE    |

\* DATE1 : 기준이 되는 날짜

\* DATE2 : 더하려는 개월 수

### ✓ 예시

- EMPLOYEE테이블에서 사원의 이름, 입사일, 입사 후 6개월이 된 날짜 조회

```
SELECT EMP_NAME,
       HIRE_DATE,
       ADD_MONTHS(HIRE_DATE, 6)
FROM EMPLOYEE;
```

|    | EMP_NAME | HIRE_DATE | ADD_MONTHS(HIRE_DATE,6) |
|----|----------|-----------|-------------------------|
| 1  | 선동일      | 90/02/06  | 90/08/06                |
| 2  | 송종기      | 01/09/01  | 02/03/01                |
| 3  | 노용철      | 01/01/01  | 01/07/01                |
| 4  | 송은희      | 96/05/03  | 96/11/03                |
| 5  | 유재식      | 00/12/29  | 01/06/29                |
| 6  | 정중하      | 99/09/09  | 00/03/09                |
| 7  | 박나라      | 08/04/02  | 08/10/02                |
| 8  | 하미유      | 94/07/07  | 95/01/07                |
| 9  | 김해솔      | 04/04/30  | 04/10/31                |
| 10 | 신부서      | 11/11/11  | 12/05/11                |

## ▶ 날짜 처리 함수

### ✓ LAST\_DAY

인자로 전달받은 날짜가 속한 달의 마지막 날짜 반환

| 작성법                    | 리턴 값 타입 |
|------------------------|---------|
| <b>LAST_DAY</b> (DATE) | DATE    |

\* DATE : 기준이 되는 날짜

### ✓ 예시

- EMPLOYEE테이블에서 사원의 이름, 입사일, 입사한 달의 마지막 날 조회

```
SELECT EMP_NAME,
       HIRE_DATE,
       LAST_DAY(HIRE_DATE)
FROM EMPLOYEE;
```

|   | EMP_NAME | HIRE_DATE | LAST_DAY(HIRE_DATE) |
|---|----------|-----------|---------------------|
| 1 | 선동일      | 90/02/06  | 90/02/28            |
| 2 | 송종기      | 01/09/01  | 01/09/30            |
| 3 | 노용철      | 01/01/01  | 01/01/31            |
| 4 | 송은희      | 96/05/03  | 96/05/31            |
| 5 | 유재식      | 00/12/29  | 00/12/31            |
| 6 | 정중하      | 99/09/09  | 99/09/30            |
| 7 | 박나라      | 08/04/02  | 08/04/30            |
| 8 | 허미연      | 94/07/07  | 94/07/31            |

# ▶ 날짜 처리 함수

## ✓ EXTRACT

년, 월, 일 정보 추출하여 반환

| 작성법   | 리턴 값 타입 |
|---|---------|
| <b>EXTRACT</b> (YEAR FROM <u>DATE</u> )<br><b>EXTRACT</b> (MONTH FROM <u>DATE</u> )<br><b>EXTRACT</b> (DAY FROM <u>DATE</u> ) | DATE    |

\* DATE : 기준이 되는 날짜

## ✓ 예시

- EMPLOYEE테이블에서 사원의 이름, 입사 년, 입사 월, 입사 일 조회

```
SELECT EMP_NAME,
       EXTRACT(YEAR FROM HIRE_DATE) YEAR,
       EXTRACT(MONTH FROM HIRE_DATE) MONTH,
       EXTRACT(DAY FROM HIRE_DATE) DAY
FROM EMPLOYEE;
```

|    | EMP_NAME | YEAR | MONTH | DAY |
|----|----------|------|-------|-----|
| 1  | 선동일      | 1990 | 2     | 6   |
| 2  | 송종기      | 2001 | 9     | 1   |
| 3  | 노용철      | 2001 | 1     | 1   |
| 4  | 송은희      | 1996 | 5     | 3   |
| 5  | 유재식      | 2000 | 12    | 29  |
| 6  | 정중하      | 1999 | 9     | 9   |
| 7  | 박나라      | 2008 | 4     | 2   |
| 8  | 하미유      | 1994 | 7     | 7   |
| 9  | 김해솔      | 2004 | 4     | 30  |
| 10 | 심봉선      | 2011 | 11    | 11  |
| 11 | 윤은해      | 2001 | 2     | 3   |
| 12 | 전형돈      | 2012 | 12    | 12  |
| 13 | 장프위      | 2015 | 6     | 17  |
| 14 | 하동우      | 1999 | 12    | 31  |



## ▶ 형 변환 함수

| 구분        | 입력 값 타입             | 리턴 값 타입   | 설명                   |
|-----------|---------------------|-----------|----------------------|
| TO_CHAR   | DATE<br>NUMBER      | CHARACTER | 날짜형 혹은 숫자형을 문자형으로 변환 |
| TO_DATE   | CHARACTER<br>NUMBER | DATE      | 문자형 혹은 숫자형을 날짜형으로 변환 |
| TO_NUMBER | CHARACTER           | NUMBER    | 문자형을 숫자형으로 변환        |



## ▶ 형 변환 함수

### ✓ TO\_CHAR

날짜 혹은 숫자형 데이터를 문자형 데이터로 변환하여 반환

| 작성법  | 리턴 값 타입   |
|--|-----------|
| <b>TO_CHAR</b> (DATE[, FORMAT])<br><b>TO_CHAR</b> (NUMBER[, FORMAT]) | CHARACTER |

- \* DATE : 문자형으로 변환하려는 날짜형 데이터
- \* NUMBER : 문자형으로 변환하려는 숫자형 데이터
- \* FORMAT : 문자형으로 변환 시 지정할 출력 형식

### ✓ FORMAT 형식

| 형식   | 의미         | 형식  | 의미          |
|------|------------|-----|-------------|
| YYYY | 년도 표현(4자리) | YY  | 년도 표현(2자리)  |
| MM   | 월을 숫자로 표현  | MON | 월을 알파벳으로 표현 |
| DD   | 일 표현       | Q   | 분기 표현       |
| DAY  | 요일 표현      | DY  | 요일을 약어로 표현  |

# ▶ 형 변환 함수

## ✓ TO\_CHAR 예시1

```
SELECT EMP_NAME,
       TO_CHAR(HIRE_DATE, 'YYYY-MM-DD'),
       TO_CHAR(HIRE_DATE, 'YY/MON, DAY, DY')
FROM EMPLOYEE;
```

| EMP_NAME | TO_CHAR(HIRE_DATE, 'YYYY-MM-DD') | TO_CHAR(HIRE_DATE, 'YY/MON, DAY, DY') |
|----------|----------------------------------|---------------------------------------|
| 1 선동일    | 1990-02-06                       | 90/2월 , 화요일, 화                        |
| 2 송종기    | 2001-09-01                       | 01/9월 , 토요일, 토                        |
| 3 노웅철    | 2001-01-01                       | 01/1월 , 월요일, 월                        |
| 4 송은희    | 1996-05-03                       | 96/5월 , 금요일, 금                        |
| 5 유재식    | 2000-12-29                       | 00/12월, 금요일, 금                        |
| 6 정중하    | 1999-09-09                       | 99/9월 , 목요일, 목                        |
| 7 박나라    | 2008-04-02                       | 08/4월 , 수요일, 수                        |
| 8 하미유    | 1994-07-07                       | 94/7월 , 목요일, 목                        |
| 9 김해술    | 2004-04-30                       | 04/4월 , 금요일, 금                        |
| 10 심봉선   | 2011-11-11                       | 11/11월, 금요일, 금                        |
| 11 윤은혜   | 2001-02-03                       | 01/2월 , 토요일, 토                        |
| 12 전형돈   | 2012-12-12                       | 12/12월, 수요일, 수                        |
| 13 장쯔위   | 2015-06-17                       | 15/6월 . 수요일 . 수                       |

## ✓ TO\_CHAR 예시2

```
SELECT EMP_NAME,
       TO_CHAR(SALARY, 'L999,999,999'),
       TO_CHAR(SALARY, '000,000,000')
FROM EMPLOYEE;
```

| EMP_NAME | TO_CHAR(SALARY, 'L999,999,999') | TO_CHAR(SALARY, '000,000,000') |
|----------|---------------------------------|--------------------------------|
| 1 선동일    | ₩8,000,000                      | 008,000,000                    |
| 2 송종기    | ₩6,000,000                      | 006,000,000                    |
| 3 노웅철    | ₩3,700,000                      | 003,700,000                    |
| 4 송은희    | ₩2,800,000                      | 002,800,000                    |
| 5 유재식    | ₩3,400,000                      | 003,400,000                    |
| 6 정중하    | ₩3,900,000                      | 003,900,000                    |
| 7 박나라    | ₩1,800,000                      | 001,800,000                    |
| 8 하미유    | ₩2,200,000                      | 002,200,000                    |
| 9 김해술    | ₩2,500,000                      | 002,500,000                    |
| 10 심봉선   | ₩3,500,000                      | 003,500,000                    |
| 11 윤은혜   | ₩2,000,000                      | 002,000,000                    |
| 12 전형돈   | ₩3,000,000                      | 003,000,000                    |

# ▶ 형 변환 함수

## ✓ TO\_DATE

숫자 혹은 문자형 데이터를 날짜형 데이터로 변환하여 반환

| 작성법   | 리턴 값 타입 |
|---|---------|
| <b>TO_DATE</b> (CHARACTER[, FORMAT])<br><b>TO_DATE</b> (NUMBER[, FORMAT]) | DATE    |

- \* CHARACTER : 날짜형으로 변환하려는 문자형 데이터
- \* NUMBER : 날짜형으로 변환하려는 숫자형 데이터
- \* FORMAT : 날짜형으로 변환 시 지정할 출력 형식

## ✓ 예시

- EMPLOYEE테이블에서 2000년도 이후에 입사한 사원의 사번, 이름, 입사일 조회

```
SELECT EMP_NO,
       EMP_NAME,
       HIRE_DATE
FROM EMPLOYEE
WHERE HIRE_DATE > TO_DATE(20000101, 'YYYYMMDD');
```

| EMP_NO            | EMP_NAME | HIRE_DATE |
|-------------------|----------|-----------|
| 1 631156-1548654  | 송종기      | 01/09/01  |
| 2 861015-1356452  | 노용철      | 01/01/01  |
| 3 660508-1342154  | 유재식      | 00/12/29  |
| 4 630709-2054321  | 박나라      | 08/04/02  |
| 5 870927-1313564  | 김해솔      | 04/04/30  |
| 6 750206-1325546  | 심봉선      | 11/11/11  |
| 7 650505-2356985  | 윤은해      | 01/02/03  |
| 8 830807-1121321  | 전형돈      | 12/12/12  |
| 9 780923-2234542  | 장프위      | 15/06/17  |
| 10 856795-1313513 | 방명수      | 10/04/04  |
| 11 881130-1050911 | 대복훈      | 17/06/19  |
| 12 770808-1364897 | 차태연      | 13/03/01  |
| 13 770808-2665412 | 전지연      | 07/03/20  |
| 14 870427-2232123 | 미오리      | 16/11/28  |
| 15 770823-1113111 | 이중석      | 14/09/18  |

## ▶ 형 변환 함수

### ✓ TO\_NUMBER

날짜 혹은 문자형 데이터를 숫자형 데이터로 변환하여 반환

| 작성법                                    | 리턴 값 타입 |
|--|---------|
| <b>TO_NUMBER</b> (CHARACTER, [FORMAT]) | NUMBER  |

\* CHARACTER : 숫자형으로 변환하려는 문자형 데이터

\* FORMAT : 날짜형으로 변환 시 지정할 출력 형식

### ✓ 예시

```
SELECT TO_NUMBER('1,000,000', '99,999,999')
       - TO_NUMBER('550,000', '999,999')
FROM DUAL;
```

|   |  |
|---|--|
|   | TO_NUMBER('1,000,000', '99,999,999') - TO_NUMBER('550,000', '999,999') |
| 1 | 450000   |

# ▶ NULL 처리 함수

## ✓ NVL

NULL로 되어 있는 컬럼의 값을 인자로 지정한 숫자 혹은 문자로 변경하여 반환

| 작성법         | 리턴 값 타입             |
|-------------|---------------------|
| NVL(P1, P2) | NUMBER<br>CHARACTER |

\* P1 : NULL데이터를 처리할 컬럼명 혹은 값

\* P2 : NULL값을 대체하고자 하는 값

## ✓ 예시

```
SELECT EMP_NO,
       EMP_NAME,
       SALARY,
       NVL(BONUS, 0),
       (SALARY + (SALARY * NVL(BONUS, 0)))*12
FROM EMPLOYEE;
```

| EMP_NO            | EMP_NAME | SALARY  | NVL(BONUS,0) | ((SALARY*12)+(SALARY*12)*NVL(BONUS,0)) |
|-------------------|----------|---------|--------------|--|
| 1 621235-1985634  | 선동일      | 8000000 | 0.3          | 124800000                              |
| 2 631156-1548654  | 송종기      | 6000000 | 0            | 72000000                               |
| 3 861015-1356452  | 노용철      | 3700000 | 0            | 44400000                               |
| 4 631010-2653546  | 송은희      | 2800000 | 0            | 33600000                               |
| 5 660508-1342154  | 유재식      | 3400000 | 0.2          | 48960000                               |
| 6 770102-1357951  | 정중하      | 3900000 | 0            | 46800000                               |
| 7 630709-2054321  | 박나라      | 1800000 | 0            | 21600000                               |
| 8 690402-2040612  | 하미유      | 2200000 | 0.1          | 29040000                               |
| 9 870927-1313564  | 김해술      | 2500000 | 0            | 30000000                               |
| 10 750206-1325546 | 심봉선      | 3500000 | 0.15         | 48300000                               |
| 11 650505-2356985 | 윤은혜      | 2000000 | 0            | 24000000                               |
| 12 830807-1121321 | 전형돈      | 2000000 | 0            | 24000000                               |
| 13 780923-2234542 | 장프위      | 2550000 | 0.25         | 38250000                               |
| 14 621111-1785463 | 하동운      | 2320000 | 0.1          | 30624000                               |
| 15 856795-1313513 | 방명수      | 1380000 | 0            | 16560000                               |
| 16 881130-1050911 | 대북훈      | 3760000 | 0            | 45120000                               |

# ▶ 선택 함수

## ✓ DECODE

비교하고자 하는 값 또는 컬럼이 조건식과 같으면 결과 값 반환

| 작성법   | 리턴 값 타입 |
|---|---------|
| <b>DECODE</b> (표현식, 조건1, 결과1, 조건2, 결과2, ..., DEFAULT) | 결과      |

- \* 표현식 : 값에 따라 선택을 다르게 할 컬럼 혹은 값
- \* 조건 : 해당 값이 참인지 거짓인지 여부 판단
- \* 결과 : 해당 조건과 일치하는 경우 반환할 값
- \* DEFAULT : 모든 조건이 불일치 시 반환할 값

## ✓ 예시

```
SELECT EMP_ID,
       EMP_NAME,
       EMP_NO,
       DECODE(SUBSTR(EMP_NO, 8, 1), '1', '남', '2', '여') AS 성별
FROM EMPLOYEE;
```

|    | EMP_ID | EMP_NAME | EMP_NO         | 성별 |
|----|--------|----------|----------------|----|
| 1  | 200    | 선동일      | 621235-1985634 | 남  |
| 2  | 201    | 송종기      | 631156-1548654 | 남  |
| 3  | 202    | 노용철      | 861015-1356452 | 남  |
| 4  | 203    | 송은희      | 631010-2653546 | 여  |
| 5  | 204    | 유재식      | 660508-1342154 | 남  |
| 6  | 205    | 정중하      | 770102-1357951 | 남  |
| 7  | 206    | 박나라      | 630709-2054321 | 여  |
| 8  | 207    | 하미유      | 690402-2040612 | 여  |
| 9  | 208    | 김해솔      | 870927-1313564 | 남  |
| 10 | 209    | 심봉선      | 750206-1325546 | 남  |
| 11 | 210    | 윤은혜      | 650505-2356985 | 여  |
| 12 | 211    | 전형돈      | 830807-1121321 | 남  |
| 13 | 212    | 장쯔위      | 780923-2234542 | 여  |
| 14 | 213    | 하동운      | 621111-1785463 | 남  |
| 15 | 214    | 방명수      | 856795-1313513 | 남  |
| 16 | 215    | 대불호      | 881130-1050911 | 남  |

## ▶ 선택 함수

### ✓ CASE

비교하고자 하는 값 또는 컬럼이 조건식과 같으면 결과 값 반환(조건은 범위 값 가능)

| 작성법  | 리턴 값 타입 |
|--|---------|
| <b>CASE WHEN</b> 조건1 <b>THEN</b> 결과1<br><b>WHEN</b> 조건2 <b>THEN</b> 결과2<br><b>WHEN</b> 조건3 <b>THEN</b> 결과3<br>...<br><b>ELSE</b> 결과N<br><b>END</b> | 결과      |

- \* 조건 : 해당 값이 참인지 거짓인지 여부 판단
- \* 결과 : 해당 조건과 일치하는 경우 반환할 값
- \* DEFAULT : 모든 조건이 불일치 시 반환할 값



## ▶ 선택 함수

### ✓ CASE 예시1

```
SELECT EMP_ID, EMP_NAME, EMP_NO,
       CASE WHEN SUBSTR(EMP_NO, 8, 1) = 1 THEN '남'
       ELSE '여'
       END AS 성별
FROM EMPLOYEE;
```

| EMP_ID | EMP_NAME | EMP_NO         | 성별 |
|--------|----------|----------------|----|
| 1 200  | 선동일      | 621235-1985634 | 남  |
| 2 201  | 송종기      | 631156-1548654 | 남  |
| 3 202  | 노용철      | 861015-1356452 | 남  |
| 4 203  | 송은희      | 631010-2653546 | 여  |
| 5 204  | 유재식      | 660508-1342154 | 남  |
| 6 205  | 정중하      | 770102-1357951 | 남  |
| 7 206  | 박나라      | 630709-2054321 | 여  |
| 8 207  | 하미유      | 690402-2040612 | 여  |
| 9 208  | 김해솔      | 870927-1313564 | 남  |
| 10 209 | 심봉선      | 750206-1325546 | 남  |
| 11 210 | 윤은해      | 650505-2356985 | 여  |
| 12 211 | 정철우      | 830807-1121321 | 남  |

### ✓ CASE 예시2

```
SELECT EMP_NAME, SALARY,
       CASE WHEN SALARY > 5000000 THEN '1등급'
       WHEN SALARY > 3500000 THEN '2등급'
       WHEN SALARY > 2000000 THEN '3등급'
       ELSE '4등급'
       END 등급
FROM EMPLOYEE;
```

| EMP_NAME | SALARY  | 등급  |
|----------|---------|-----|
| 1 김성춘    | 3000000 | 3등급 |
| 2 선동일    | 8000000 | 1등급 |
| 3 송종기    | 6000000 | 1등급 |
| 4 노용철    | 3700000 | 2등급 |
| 5 송은희    | 2800000 | 3등급 |
| 6 유재식    | 3400000 | 3등급 |
| 7 정중하    | 3900000 | 2등급 |
| 8 박나라    | 1800000 | 4등급 |
| 9 하미유    | 2200000 | 3등급 |
| 10 김해솔   | 2500000 | 3등급 |
| 11 심봉선   | 3500000 | 3등급 |
| 12 윤은해   | 2000000 | 4등급 |
| 13 정철우   | 3000000 | 3등급 |

## ▶ 그룹 함수

하나 이상의 행을 그룹으로 묶어 연산하며 총합, 평균 등을 하나의 컬럼으로 반환하는 함수

| 구분    | 설명           |
|-------|--------------|
| SUM   | 그룹의 누적 합계 반환 |
| AVG   | 그룹의 평균 반환    |
| COUNT | 그룹의 총 개수 반환  |
| MAX   | 그룹의 최대 값 반환  |
| MIN   | 그룹의 최소 값 반환  |

## ▶ 그룹 함수

### ✓ SUM

해당 컬럼 값들의 총합 반환

### ✓ 예시

- EMPLOYEE테이블에서 남자 사원의 급여 총합 조회

```
SELECT SUM(SALARY),
FROM EMPLOYEE
WHERE SUBSTR(EMP_NO, 8, 1) = 1;
```

|   | SUM(SALARY) |
|---|-------------|
| 1 | 49760000    |

- EMPLOYEE테이블에서 부서코드가 D5인 직원의 보너스 포함 연봉 조회

```
SELECT SUM(SALARY + (SALARY*NVL(BONUS, 0))*12)
FROM EMPLOYEE
WHERE DEPT_CODE = 'D5';
```

|   | SUM((SALARY+(SALARY*NVL(BONUS,0))*12)) |
|---|--|
| 1 | 24700000                               |

## ▶ 그룹 함수

### ✓ AVG

해당 컬럼 값들의 평균 반환

### ✓ 예시

- EMPLOYEE테이블에서 전 사원의 보너스 평균을 소수 셋째 자리에서 반올림 한 것 조회

```
SELECT ROUND(AVG(NVL(BONUS, 0)), 2)
```

```
FROM EMPLOYEE;
```

\* NVL을 하지 않을 시 NULL 값을 가진 행은 평균 계산에서 제외되어 계산

| ROUND(AVG(NVL(BONUS,0)),2) |      |
|----------------------------|------|
| 1                          | 0.08 |

## ▶ 그룹 함수

### ✓ MAX/MIN

그룹의 최대값과 최소값 반환

### ✓ 예시

- EMPLOYEE테이블에서 가장 높은 급여와 가장 낮은 급여 조회

```
SELECT MAX(SALARY), MIN(SALARY)
FROM EMPLOYEE;
```

|   | MAX(SALARY) | MIN(SALARY) |
|---|-------------|-------------|
| 1 | 8000000     | 1380000     |

- EMPLOYEE테이블에서 가장 오래된 입사일과 가장 최근인 입사일 조회

```
SELECT MAX(HIRE_DATE), MIN(HIRE_DATE)
FROM EMPLOYEE;
```

|   | MAX(HIRE_DATE) | MIN(HIRE_DATE) |
|---|----------------|----------------|
| 1 | 17/06/19       | 90/02/06       |

## ▶ 그룹 함수

### ✓ COUNT

테이블 조건을 만족하는 행의 개수 반환

### ✓ 예시

- EMPLOYEE테이블에서 전체 사원 수 조회

```
SELECT COUNT(*)  
FROM EMPLOYEE;
```

|   | COUNT(*) |
|---|----------|
| 1 | 23       |

- EMPLOYEE테이블에서 부서코드가 D5인 직원의 수 조회

```
SELECT COUNT(DEPT_CODE)  
FROM EMPLOYEE  
WHERE DEPT_CODE = 'D5';
```

|   | COUNT(DEPT_CODE) |
|---|------------------|
| 1 | 6                |

- EMPLOYEE테이블에서 사원들이 속해있는 부서의 수 조회

```
SELECT COUNT(DISTINCT DEPT_CODE)  
FROM EMPLOYEE;
```

|   | COUNT(DISTINCT DEPT_CODE) |
|---|---------------------------|
| 1 | 6                         |