ch06. 데이터 처리와 가공을 위한 오라클 함수

06-1. 오라클 함수

함수란?

함수 *function* : 수학에서 정의한 개념으로 x와 y변수가 존재하고 x값이 변하면 그 변화에 따라 어떤 연산 또는 가공을 거쳐 y값도 함께 변할 때 이 y를 함수라고 한다.
→ x값의 변화에 따라 y값이 종속적으로 변하기 때문에 '따름수'라고도 한다.

내장 함수의 종류

내장함수 :

입력 방식에 따라 데이터 처리에 사용하는 행이 나뉜다.

단일행 함수 *single-row function* :

데이터가 한 행씩 입력되고 입력된 한 행당 결과가 하나씩 나오는 함수

다중행 함수 *multiple-row function* :

여러 행이 입력되어 하나의 행으로 결과가 반환되는 함수

06-2. 문자 데이터를 가공하는 문자 함수

대소문자를 바꿔 주는 UPPER, LOWER, INITCAP 함수

함수	설명
UPPER (문자열)	괄호 안 문자 데이터를 모두 대문자로 변환하여 반환
LOWER (문자열)	괄호 안 문자 데이터를 모두 소문자로 변환하여 반환
INITCAP (문자열)	괄호 안 문자 데이터 중 첫 글자는 대문자로, 나머지 문자를 소문자로 변환 후 반환

```
SELECT ENAME, UPPER(ENAME), LOWER(ENAME), INITCAP(ENAME)
  FROM EMP;
```

문자열 길이를 구하느 LENGTH 함수

특정 문자열의 길이를 구할 때 LENGTH함수를 사용

```
SELECT ENAME, LENGTH(ENAME)
  FROM EMP;
```

```
SELECT ENAME, LENGTH(ENAME)
  FROM EMP
WHERE LENGTH(ENAME) >= 5;
```

LENGTH 와 LENGTHB 함수 비교

```
SELECT LENGTH('한글'), LENGTHB('한글')
  FROM DUAL;
```

한글은 한 문자당 2바이트로 처리 되는데

LENGTHB는 바이트 수를 출력하기 때문에 한글이 2글자면 4바이트가 된다.

문자열 일부를 추출하는 SUBSTR 함수

주민등록번호 중 생년월일 앞자리만 필요하거나 전화번호의 마지막 네 자리 숫자만 추출하는 경우와 같이 문자열 중 일부를 추출할 때 사용한다 .

SUBSTR (문자열DATA,시작위치,추출길이)	문자열 데이터의 시작 위치부터 추출 길이만큼 추출한다. 시작 위치가 음수일 경우에는 마지막 위치부터 거슬러 올라간 위치에서 시작한다.
SUBSTR (문자열DATA, 시작위치)	문자열 데이터의 시작 위치부터 문자열 데이터 끝까지 추출한다. 시작 위치가 음수일 경우에는 마지막 위치부터 거슬러 올라간 위치에서 끝까지 추출한다.

```
SELECT JOB, SUBSTR(JOB, 1, 2), SUBSTR(JOB, 3, 2), SUBSTR(JOB, 5)
  FROM EMP;
```

문자열 데이터 안에서 특정 문자 위치를 찾는 INSTR 함수

문자열 데이터 안에 특정 문자나 문자열이 어디에 포함되어 있는지를 알고자 할 때 사용.

- 기본형식

```
INSTR([대상 문자열 데이터(필수)],
      [위치를 찾으려는 부분 문자(필수)],
      [위치 찾기를 시작할 대상 문자열 데이터 위치(선택, 기본값은 1)],
      [시작 위치에서 찾으려는 문자가 몇 번째인지 지정(선택, 기본값은 1)])
```

```
SELECT INSTR('HELLO, ORACLE!', 'L') AS INSTR_1,
      INSTR('HELLO, ORACLE!', 'L', 5) AS INSTR_2,
      INSTR('HELLO, ORACLE!', 'L', 2, 3) AS INSTR_3,
FROM DUAL;
```

특정 문자를 다른 문자로 바꾸는 REPLACE 함수

- 기본 형식

```
REPLACE([문자열 데이터 또는 열 이름(필수)], [찾는 문자(필수)], [대체할 문자(선택)])
```

데이터의 빈 공간을 특정 문자로 채우는 LPAD, RPAD 함수

데이터와 자릿수를 지정한 후 데이터 길이가 지정한 자릿수보다 작을 경우에 나머지 공간을
특정 문자로 채우는 함수.

LPAD *Left Padding*

→ 남은 빈 공간을 왼쪽에 채운다.

RPAD *Right Padding*

→ 남은 빈 공간을 오른쪽에 채운다.

- 기본 형식

```
LPAD([문자열 데이터 또는 열 이름(필수)], [데이터 자릿수(필수)], [채울 문자(선택)])
```

두 문자열 데이터를 합치는 CONCAT 함수

CONCAT함수는 두 개의 문자열 데이터를 하나의 데이터로 연결해 주는 역할을 한다.

두개의 입력 데이터 지정을 하고 열이나 문자열 데이터 모두 지정 가능.

```
SELECT CONCAT(EMPNO, ENAME),
       CONCAT(EMPNO, CONCAT(' : ', ENAME))
  FROM EMP
WHERE ENAME = 'SCOTT';
```

이와 비슷하게 문자열 데이터를 연결하는 || 문자가 있다

```
SELCT EMPNO || ENAME
  FROM EMP;
```

특정 문자를 지우는 TRIM, LTRIM, RTRIM 함수

문자열 데이터 내에서 특정 문자를 지우기 위해 사용한다.

- TRIM 함수 사용하기 (삭제할 문자가 없을 때)

```
SELECT '[' || TRIM(' _ _Oracle_ _ ') || ']' AS TRIM,
       '[' || TRIM(LEADING FROM ' _ _Oracle_ _ ') || ']' AS TRIM_LEADING,
       '[' || TRIM(TRAILING FROM ' _ _Oracle_ _ ') || ']' AS TRIM_TRAILING,
       '[' || TRIM(BOTH FROM ' _ _Oracle_ _ ') || ']' AS TRIM_BOTH
    FROM DUAL;
```

결과 x

SQL | 인출된 모든 행: 1(0.035초)

	TRIM	TRIM_LEADING	TRIM_TRAILING	TRIM_BOTH
1	[_ _Oracle_ _]	[_ _Oracle_ _]	[_ _Oracle_ _]	[_ _Oracle_ _]

- TRIM 함수 사용하기 (삭제할 문자가 있을 때)

```
SELECT '[' || TRIM('_' FROM '_ _Oracle_ _') || ']' AS TRIM,
       '[' || TRIM(LEADING '_' FROM '_ _Oracle_ _') || ']' AS TRIM_LEADING,
       '[' || TRIM(TRAILING '_' FROM '_ _Oracle_ _') || ']' AS TRIM_TRAILING,
       '[' || TRIM(BOTH '_' FROM '_ _Oracle_ _') || ']' AS TRIM_BOTH
    FROM DUAL;
```

의 결과 x

SQL | 인출된 모든 행: 1(0.004초)

	TRIM	TRIM_LEADING	TRIM_TRAILING	TRIM_BOTH
1	[_Oracle_]	[_Oracle_ _]	[_ _Oracle_]	[_Oracle_]

📎 (default) : 삭제할 문자가 생략될 경우에 기본적으로 공백을 제거한다.
LEADING : 왼쪽의 글자를 지움.
TRAILING : 오른쪽의 글자를 지움.
BOTH : 양쪽의 글자를 모두 지움.

LTRIM :

왼쪽의 지정문자를 삭제하는 데 사용

RTRIM :

오른쪽의 지정문자를 삭제하는 데 사용

- 기본 형식

```
LTRIM([원본 문자열 데이터(필수)], [삭제할 문자 집합(선택)])
```

```
SELECT '[' || TRIM(' _Oracle_ ') || ']' AS TRIM,
       '[' || LTRIM(' _Oracle_ ') || ']' AS LTRIM,
       '[' || LTRIM('<_Oracle_>', '_<') || ']' AS LTRIM_2,
       '[' || RTRIM(' _Oracle_ ') || ']' AS RTRIM,
       '[' || RTRIM('<_Oracle_>', '>_') || ']' AS RTRIM_2
    FROM DUAL;
```

의 결과 ×

SQL | 인출된 모든 행: 1(0.002초)

	TRIM	LTRIM	LTRIM_2	RTRIM	RTRIM_2
1	[_Oracle_]	[_Oracle_]	[Oracle_>]	[_Oracle_]	[<_Oracle]

06-3. 숫자 데이터를 연산하고 수치를 조정하는 숫자 함수

함수	설명
ROUND	지정된 숫자의 특정 위치에서 반올림한 값을 반환
TRUNC	숫자의 특정 위치에서 버림한 값을 반환
CEIL	지정된 숫자보다 큰 정수 중 가장 작은 정수를 반환
FLOOR	지정된 숫자보다 작은 정수 중 가장 큰 정수를 반환
MOD	지정된 숫자를 나눈 나머지 값을 반환

특정 위치에서 반올림하는 ROUND 함수

- 특정 숫자를 반올림하되 반올림할 위치를 지정할 수 있다.
- 반올림할 위치를 지정하지 않으면 소수점 첫째 자리에서 반올림한 결과가 반환.

```
ROUND([숫자(필수)], [반올림할 위치(선택])
```

특정 위치에서 버리는 TRUNC 함수

- 지정된 자리에서 숫자를 버림 처리하는 함수
- 반올림할 위치를 지정하지 않으면 소수점 첫째 자리에서 버림 처리하여 반환.

```
TRUNC([숫자(필수)], [버림 위치(선택)])
```

지정한 숫자와 가까운 정수를 찾는 CEIL, FLOOR 함수

- 각각 입력된 숫자와 가까운 큰 정수, 작은 정수를 반환하는 함수

```
CEIL([숫자(필수)])
```

숫자를 나눈 나머지 값을 구하는 MOD 함수

```
MOD([나눗셈 될 숫자(필수)], [나눌 숫자(필수)])
```

06-4. 날짜 데이터를 다루는 날짜 함수

연산	설명
날짜 데이터 + 숫자	날짜 데이터보다 숫자만큼 일수 이후의 날짜
날짜 데이터 - 숫자	날짜 데이터보다 숫자만큼 일수 이전의 날짜
날짜 데이터 - 날짜 데이터	두 날짜 데이터 간의 일수 차이
날짜 데이터 + 날짜 데이터	연산 불가, 지원하지 않음

SYSDATE 함수 이용하여 날짜 출력하기

```
SELECT SYSDATE AS NOW,
       SYSDATE-1 AS YESTERDAY,
       SYSDATE+1 AS TOMORROW
  FROM DUAL;
```

몇 개월 이후 날짜를 구하는 ADD_MONTHS 함수

```
ADD_MONTHS([날짜 데이터(필수)], [더할 개월 수(정수)(필수)]
```

```
SELECT SYSDATE,
       ADD_MONTHS(SYSDATE,3)
  FROM DUAL;
```

두 날짜 간의 개월 수 차이를 구하는 MONTHS_BETWEEN 함수

```
MONTHS_BETWEEN([날짜 데이터1(필수)],[날짜 데이터2(필수)]
```

돌아오는 요일, 달의 마지막 날짜를 구하는 NEXT_DAY, LAST_DAY 함수

```
NEXT_DAY([날짜 데이터(필수)], [요일 문자(필수)])
```

```
LAST_DAY([날짜 데이터(필수)]
```

- 날짜 데이터에도 반올림, 버림 처리에 사용하는 ROUND, TRUNC 함수를 사용할 수 있다.

오라클 날짜 데이터 기준 포맷 값

```
CC: 세기(Century)
YEAR: 년도
YYYY, YYY, YY, Y : 년도 자릿수 표기
BC, AD : 서기 등으로 표시
Q: 분기 표시
MM : 두자리로 월표시
MONTH: 영어로 표시
MON: 영어로 3자리로 월표시
RM: 로마자로 표시 i, ii. xi
WW, WI : 1년기준 몇째주 표시
W :한달기준 몇째주 표시
DDD: 365(1년기준 ) 의 몇째 일
DD: 날짜를 두자리로 표시
D: 요일을 숫자로 표시
DY: 요일 한자리로 표시
DAY: 요일 표시
AM, PM, A.M. , P.M. : 오전오후 표시
HH, HH12 : 12시 기준으로 표시
HH24 : 24시 기준으로 표시
/, "of" : 날짜의 중간에 문자 표시 ->
SPTH: 날짜를 영문 서수로 표시
SP : 날짜를 영문 숫자로 표시
```

06-5. 자료형을 변환하는 형 변환 함수

숫자처럼 생긴 문자 데이터는 숫자로 바꿔 주지만 그 외의 경우는 잘 동작하지 않는다.

사용자가 직접 지정하는 형변환을 명시적 형 변환이라고 한다.

형변환 함수

TO_CHAR	숫자 또는 날짜 데이터를 문자 데이터로 변환
TO_NUMBER	문자 데이터를 숫자 데이터로 변환
TO_DATE	문자 데이터를 날짜 데이터로 변환

날짜, 숫자 데이터를 문자 데이터로 변환하는 TO_CHAR 함수

```
TO_CHAR([날짜 데이터(필수)], '[출력되길 원하는 문자 형태(필수)])
```

- 현재날짜와 시간을 '연/월/일/ 시:분:초' 로 출력하기 (기준 포맷 사용)

```
SELECT TO_CHAR(SYSDATE, 'YYYY/MM/DD HH24:MI:SS') AS 현재날짜시간
  FROM DUAL;
```

CC	세기
YYYY, RRRR	연 (4자리 숫자)
YY, RR	연 (2자리 숫자)
MM	월 (2자리 숫자)
MON	월 (언어별 월 이름 약자)
MONTH	월 (언어별 월 이름 전체)
DD	일 (2자리 숫자)
DDD	1년 중 며칠(1~366)
DY	요일 (언어별 요일 이름 약자)
DAY	요일 (언어별 요일 이름 전체)
W	1년 중 몇 번째 주 (1~53)

- 월과 요일을 다양한 형식으로 출력

```
SELECT SYSDATE,
      TO_CHAR(SYSDATE, 'MM') AS MM,
      TO_CHAR(SYSDATE, 'MON') AS MON,
      TO_CHAR(SYSDATE, 'MONTH') AS MONTH,
      TO_CHAR(SYSDATE, 'DD') AS DD,
      TO_CHAR(SYSDATE, 'DY') AS DY,
      TO_CHAR(SYSDATE, 'DAY') AS DAY
  FROM DUAL;
```

- 특정 언어에 맞춰서 날짜 출력하기

```
TO_CHAR([날짜 데이터(필수)], '[출력되길 원하는 문자 형태(필수)]',
'NLS_DATE_LANGUAGE = language'(선택))
```

시간 형식 지정하여 출력하기

```
SELECT SYSDATE,
      TO_CHAR(SYSDATE, 'HH24:MI:SS') AS HH24MISS,
      TO_CHAR(SYSDATE, 'HH12:MI:SS AM') AS HHMISS_AM,
      TO_CHAR(SYSDATE, 'HH:MI:SS P.M.') AS HHMISS_PM
  FROM DUAL;
```

형식	설명
HH24	24시간으로 표현한 시간
HH, HH12	12시간으로 표현한 시간
MI	분
SS	초
AM, PM, A.M., P.M.	오전, 오후 표시

숫자 데이터 형식을 지정하여 출력하기

```
SELECT SAL,
        TO_CHAR(SAL, '$999,999') AS SAL_$,
        TO_CHAR(SAL, 'L999,999') AS SAL_L,
        TO_CHAR(SAL, '999,999.00') AS SAL_1,
        TO_CHAR(SAL, '000,999,999.00') AS SAL_2,
        TO_CHAR(SAL, '000999999.99') AS SAL_3,
        TO_CHAR(SAL, '999,999,00') AS SAL_4
  FROM EMP;
```

문자 데이터를 숫자 데이터로 변환하는 TO_NUMBER 함수

- 숫자처럼 생긴 문자 데이터 간의 암시적 형변환이 일어나는데 중간에 쉼표(,)가 들어가 있으면 암시적 형 변환이 일어나지 않는다. 그럴 경우 TO_NUMBER 함수를 사용하여

강제적으로 형 변환을 시켜준다.

```
TO_NUMBER('[문자열 데이터(필수)]', '인식될 숫자형태(필수)]')
```

문자 데이터를 날짜 데이터로 변환하는 TO_DATE 함수

```
TO_DATE('[문자열 데이터(필수)]', '인식될 날짜형태(필수)]')
```

06-6. NULL 처리 함수

NVL 함수의 기본 사용법

```
NVL([NULL인지 여부를 검사할 데이터 또는 열(필수)],
    [앞의 데이터가 NULL일 경우 반환할 데이터(필수)])
```

```
SELECT EMPNO, ENAME, SAL, COMM, SAL+COMM,
        NVL(COMM,0),
        SAL+NVL(COMM, 0)
  FROM EMP;
```

NVL2 함수의 기본 사용법

```
NVL2([NULL인지 여부를 검사할 데이터 또는 열(필수)],
    [앞의 데이터가 NULL이 아닐 경우 반환할 데이터 또는 계산식(필수)]),
    [앞의 데이터가 NULL일 경우 반환할 데이터 또는 계산식(필수)])
```

```
SELECT EMPNO, ENAME, COMM,
        NVL2(COMM,'0', 'X'),
        SAL+NVL(COMM, SAL*12+COMM, SAL*12) AS ANNIMAL
  FROM EMP;
```

06-7. 상황에 따라 다른 데이터를 반환하는 DECODE 함수와 CASE 문

특정 열 값이나 데이터 값에 따라 어떤 데이터를 반환할지 정한다.

DECODE 함수

기준이 되는 데이터를 먼저 지정한 후 해당 데이터 값에 따라 다른 결과 값을 내보내는 함수

```
DECODE([검사 대상이 될 열 또는 데이터, 연산이나 함수의 결과],
       [조건1], [데이터가 조건 1과 일치할 때 반환할 결과],
       [조건2], [데이터가 조건 2과 일치할 때 반환할 결과],
          .
          .
          .
       [조건n], [데이터가 조건 n과 일치할 때 반환할 결과],
       [위 조건1~조건n과 일치한 경우가 없을 때 반환할 결과])
```

- 만약 EMP 테이블에서 직책이 MANAGER인 사람은 급여의 10% 인상한 급여, SALESMAN인 사람은 급여의 5%, ANALYST인 사람은 그대로, 나머지는 3%만큼 인상된 급여를 출력하고싶다면,

```
SELECT EMPNO, ENAME, JOB, SAL,
      DECODE(JOB
            'MANAGER', SAL*1.1,
            'SALESMAN', SAL*1.05,
            'ANALYST', SAL,
            SAL*1.03) AS UPSAL
  FROM EMP;
```

CASE문

특정 조건에 따라 반환할 데이터를 설정할 때 사용

DECODE와 달리 각조건에 사용하는 데이터가 서로 상관이 없어도 됨.

- 기준 데이터 없이 조건식만으로 CASE문 사용하기

```
SELECT EMPNO, ENAME, COMM,
    CASE
    WHEN COMM IS NULL THEN '해당사항 없음'
    WHEN COMM = 0 THEN '수당없음'
    WHEN COMM > 0 THEN '수당 : ' || COMM
    END AS COMM_TEXT
FROM EMP;
```

질의 결과 x

SQL | 인출된 모든 행: 14(0.051초)

	EMPNO	ENAME	COMM	COMM_TEXT
1	7369	SMITH	(null)	해당사항 없음
2	7499	ALLEN	300	수당 : 300
3	7521	WARD	500	수당 : 500
4	7566	JONES	(null)	해당사항 없음
5	7654	MARTIN	1400	수당 : 1400
6	7698	BLAKE	(null)	해당사항 없음
7	7782	CLARK	(null)	해당사항 없음
8	7788	SCOTT	(null)	해당사항 없음
9	7839	KING	(null)	해당사항 없음
10	7844	TURNER	0	수당없음
11	7876	ADAMS	(null)	해당사항 없음
12	7900	JAMES	(null)	해당사항 없음
13	7902	FORD	(null)	해당사항 없음
14	7934	MILLER	(null)	해당사항 없음

Q

1.

```
SELECT EMPNO,
       RPAD(SUBSTR(EMPNO, 1, 2), 4, '*') AS MASKING_EMPNO,
       ENAME,
       RPAD(SUBSTR(ENAME, 1, 1), LENGTH(ENAME), '*') AS MASKING_ENAME
```

```
    FROM  EMP
 WHERE LENGTH(ENAME) = 5;
```

2.

```
SELECT EMPNO, ENAME, SAL,
       TRUNC(SAL / 21.5, 2) AS DAY_PAY,
       ROUND(SAL / 21.5 / 8, 1) AS TIME_PAY
  FROM EMP;
```

3.

```
SELECT EMPNO, ENAME, HIREDATE,
       TO_CHAR(NEXT_DAY(ADD_MONTHS(HIREDATE, 3), '월요일'), 'YYYY-MM-DD'),
       NVL(TO_CHAR(COMM), 'N/A')
  FROM EMP;
```

4.

```
SELECT EMPNO, ENAME, MGR,
       CASE
          WHEN MGR IS NULL THEN '0000'
          WHEN SUBSTR(MGR, 1, 2) = '78' THEN '8888'
          WHEN SUBSTR(MGR, 1, 2) = '77' THEN '7777'
          WHEN SUBSTR(MGR, 1, 2) = '76' THEN '6666'
          WHEN SUBSTR(MGR, 1, 2) = '75' THEN '5555'
          ELSE MGR
    FROM EMP;
```