

SQL – DML_FUNCTION

문혜영

```
DROP TABLE STUDENT CASCADE CONSTRAINT;
DROP TABLE DEPT CASCADE CONSTRAINT;
```

```
CREATE TABLE DEPT (
D_CODE CHAR(5) NOT NULL,
D_NAME VARCHAR2(40) NOT NULL,
CONSTRAINT DEPT_PK PRIMARY KEY (D_CODE)
);
```

```
CREATE TABLE STUDENT (
S_NAME VARCHAR2(20),
S_NO CHAR(10) NOT NULL,
S_CODE CHAR(5),
S_BIRTH DATE,
S_SCORE NUMBER,
S_ENGNAME VARCHAR2(20),
S_SCORE2 NUMBER,
CONSTRAINT S_PK PRIMARY KEY (S_NO),
CONSTRAINT S_FK FOREIGN KEY (S_CODE) REFERENCES DEPT (D_CODE) ON DELETE SET NULL,
CONSTRAINT S_BIRTH CHECK (S_BIRTH > TO_DATE('1980-01-01', 'YYYY-MM-DD'))
);
```

```
INSERT ALL
INTO DEPT VALUES ('A1000','경영정보')
INTO DEPT VALUES ('B1000','데이터사이언스')
INTO DEPT VALUES ('C1000','AI')
INTO DEPT VALUES ('D1000','성악')
SELECT * FROM DUAL;
```

```
INSERT ALL
INTO STUDENT VALUES ('대한민국','901','A1000', TO_DATE('2000-10-09', 'YYYY-MM-DD'),90,"",200)
INTO STUDENT VALUES ('홍장미','902','B1000', TO_DATE('2000-06-01', 'YYYY-MM-DD'),80,' LISA ',250)
INTO STUDENT VALUES ('이리아','903','C1000', TO_DATE('2001-11-02', 'YYYY-MM-DD'),95,'RUBY',220)
INTO STUDENT VALUES ('물보라','904','A1000', TO_DATE('2001-01-03', 'YYYY-MM-DD'),85,'Alice',230 )
INTO STUDENT VALUES ('다스리','905','B1000', TO_DATE('2000-02-25', 'YYYY-MM-DD'),66,'cody',"")
INTO STUDENT VALUES ('이루리','906','C1000', TO_DATE('2002-03-04', 'YYYY-MM-DD'),55,'denise',"")
INTO STUDENT VALUES ('은송이','907','D1000', TO_DATE('2000-07-06', 'YYYY-MM-DD'),77,"",300)
```

DEPT

	D_CODE	D_NAME
1	A1000	경영정보
2	B1000	데이터사이언스
3	C1000	AI
4	D1000	성악

STUDENT

	S_NAME	S_NO	S_CODE	S_BIRTH	S_SCORE	S_ENGNAME	S_SCORE2
1	대한민국	901	A1000	00/10/09	90	(null)	200
2	홍장미	902	B1000	00/06/01	80	LISA	250
3	이리아	903	C1000	01/11/02	95	RUBY	220
4	물보라	904	A1000	01/01/03	85	Alice	230
5	다스리	905	B1000	00/02/25	66	cody	(null)
6	이루리	906	C1000	02/03/04	55	denise	(null)
7	은송이	907	D1000	00/07/06	77	(null)	300

- 함수의 유형

- 생성 주체
 - 내장 함수 (Built-in Functions)
 - SUM, COUNT, AVG, MIN, MAX 등
 - 사용자 정의 함수 (User-Defined Functions, UDFs)
- 수행 범위
 - 단일행 함수 (Single-Row Functions)
 - 문자형 함수, 숫자형 함수, 날짜형 함수
 - 제어 함수, 변환 함수, NULL 관련 함수
 - UPPER, LOWER, CONCAT, SUBSTRING 등
 - 다중 행 함수 (Multi-Row Functions)
 - 보통 GROUP BY 절과 함께 사용
 - SUM, AVG, COUNT, MIN, MAX, ROW_NUMBER, RANK 등

Function

- 단일행 함수의 특징

- 예)

```
SELECT S_NAME, LENGTH(S_NAME) AS 길이  
FROM STUDENT;
```

```
SELECT S_NAME, CONCAT(S_SCORE, '점') AS 점수  
FROM STUDENT;
```

- 각 행(Row)에 대해 개별적으로 작용하고 그 결과를 반환함
 - 단일 행 내에 있는 하나 또는 복수의 값을 인수로 사용
 - 여러 행에 걸친 값을 사용할 수 없음
 - 함수 중첩(함수의 인자로 함수를 사용)이 가능함
 - SELECT, WHERE, ORDER BY 절에 사용 가능함

	S_NAME	길이
1	대한민국	4
2	홍장미	3
3	이리아	3
4	눌보라	3
5	다스리	3
6	이루리	3
7	은송이	3

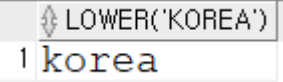
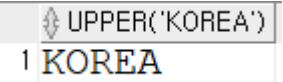
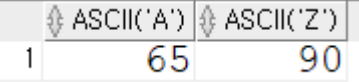
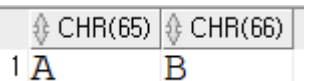
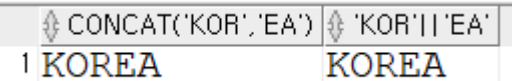
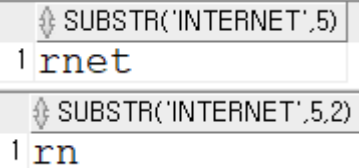
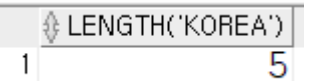
SQL 인출된 모든 행: 7(0,002초)		
	S_NAME	점수
1	대한민국	90점
2	홍장미	80점
3	이리아	95점
4	눌보라	85점
5	다스리	66점
6	이루리	55점
7	은송이	77점

- Oracle의 단일행 내장 함수

종류	개요	주요 함수
문자형 함수	문자형 변수 처리	LOWER, UPPER, ASCII, CHR, CONCAT, SUBSTR, LENGTH, LTRIM, RTRIM, TRIM
숫자형 함수	숫자형 변수 처리	ABS, SIGN, MOD, CEIL, FLOOR, ROUND, TRUNC, SIN, COS, TAN, EXP, POWER, SQRT, LOG, LN
변환 함수	문자, 숫자, DATE형 값의 타입 변환	TO_CHAR, TO_NUMBER, TO_DATE
날짜형 함수	DATE 타입의 변수 처리	SYSDATE, EXTRACT, TO_NUMBER(TO_CHAR(d,'DD' 'MM' 'YY'))
제어 함수	논리값에 따른 값의 처리	CASE, DECODE
NULL 관련 함수	NULL 처리	NVL, NULLIF, COALESCE

Function

- 문자형 함수

문자형 함수	함수 사용 예	SQL	결과
LOWER(문자열)	LOWER('SQL Expert')	SELECT LOWER('KorEa') FROM DUAL;	
UPPER(문자열)	UPPER('SQL Expert')	SELECT UPPER('KorEa') FROM DUAL;	
ASCII(문자)	ASCII('A'), ASCII('5')	SELECT ASCII('A'), ASCII('Z') FROM DUAL;	
CHR(ASCII 코드)	CHR(65), CHR(66)	SELECT CHR(65), CHR(66) FROM DUAL;	
CONCAT(문자열1, 문자열2)	CONCAT('KOR','EA'), cf) 'KOR' 'EA'	SELECT CONCAT('KOR','EA'), 'KOR' 'EA' FROM DUAL;	
SUBSTR(문자열, m), SUBSTR(문자열, m, n)	SUBSTR('internet', 5) SUBSTR('internet', 5,2)	SELECT SUBSTR('internet', 5) FROM DUAL; SELECT SUBSTR('internet', 5,2) FROM DUAL;	
LENGTH(문자열)	LENGTH('korea')	SELECT LENGTH('korea') FROM DUAL;	

Function

- ASCII 코드표

10진수	16진수	문자	10진수	16진수	문자	10진수	16진수	문자	10진수	16진수	문자
0	0X00	NULL	16	0X10	DLE	32	0x20	SP	48	0x30	0
1	0X01	SOH	17	0X11	DC1	33	0x21	!	49	0x31	1
2	0X02	STX	18	0X12	SC2	34	0x22	"	50	0x32	2
3	0X03	ETX	19	0X13	SC3	35	0x23	#	51	0x33	3
4	0X04	EOT	20	0X14	SC4	36	0x24	\$	52	0x34	4
5	0X05	ENQ	21	0X15	NAK	37	0x25	%	53	0x35	5
6	0X06	ACK	22	0X16	SYN	38	0x26	&	54	0x36	6
7	0X07	BEL	23	0X17	ETB	39	0x27	'	55	0x37	7
8	0X08	BS	24	0X18	CAN	40	0x28	(56	0x38	8
9	0X09	HT	25	0x19	EM	41	0x29)	57	0x39	9
10	0X0A	LF	26	0x1A	SUB	42	0x2A	*	58	0x3A	:
11	0X0B	VT	27	0x1B	ESC	43	0x2B	+	59	0x3B	;
12	0X0C	FF	28	0x1C	FS	44	0x2C	,	60	0x3C	<
13	0X0D	CR	29	0x1D	GS	45	0x2D	-	61	0x3D	=
14	0X0E	SO	30	0x1E	RS	46	0x2E	.	62	0x3E	>
15	0X0F	SI	31	0x1F	US	47	0x2F	/	63	0x3F	?

Function

- ASCII 코드표

10진수	16진수	문자	10진수	16진수	문자	10진수	16진수	문자	10진수	16진수	문자
64	0x40	@	80	0x50	P	96	0x60	`	112	0x70	p
65	0x41	A	81	0x51	Q	97	0x61	a	113	0x71	q
66	0x42	B	82	0x52	R	98	0x62	b	114	0x72	r
67	0x43	C	83	0x53	S	99	0x63	c	115	0x73	s
68	0x44	D	84	0x54	T	100	0x64	d	116	0x74	t
69	0x45	E	85	0x55	U	101	0x65	e	117	0x75	u
70	0x46	F	86	0x56	V	102	0x66	f	118	0x76	v
71	0x47	G	87	0x57	W	103	0x67	g	119	0x77	w
72	0x48	H	88	0x58	X	104	0x68	h	120	0x78	x
73	0x49	I	89	0x59	Y	105	0x69	i	121	0x79	y
74	0x4A	J	90	0x5A	Z	106	0x6A	j	122	0x7A	z
75	0x4B	K	91	0x5B	[107	0x6B	k	123	0x7B	{
76	0x4C	L	92	0x5C	\	108	0x6C	l	124	0x7C	
77	0x4D	M	93	0x5D]	109	0x6D	m	125	0x7D	}
78	0x4E	N	94	0x5E	^	110	0x6E	n	126	0x7E	~
79	0x4F	O	95	0x5F	_	111	0x6F	o	127	0x7F	DEL

- 문자형 함수

문자형 함수	함수 사용 예
LTRIM (문자열) / LTRIM (문자열, 지정문자)	문자열의 왼쪽부터 시작해서 다른 문자를 만나기 전까지 지정 문자를 제거. (지정 문자가 생략되면 공백 값이 디폴트) (예) LTRIM('xxxYYZZxYZ') → 'xxxYYZZxYZ' (예) LTRIM('xxxYYZZxYZx','x') → 'YYZZxYZx'
RTRIM (문자열) / RTRIM (문자열, 지정문자)	문자열의 오른쪽부터 시작해서 다른 문자를 만나기 전까지 지정 문자를 제거. (지정 문자가 생략되면 공백 값이 디폴트) (예) RTRIM('zXXYYzzXYzz','z') → 'zXXYYzzXY'
TRIM (문자열) / TRIM (지정문자 FROM 문자열)	문자열의 양쪽에서 시작해서 다른 문자를 만나기 전까지 지정 문자를 제거. (지정 문자와 FROM이 생략되면 공백 값이 디폴트) (예) TRIM('x' FROM 'xxYYZZxYZxx') → 'YYZZxYZ' (예) TRIM(' xxxYYzz ') → 'xxxYYzz'

```
SELECT LENGTH(' Hello World ') FROM dual;
```

LENGTH('HELLOWORLD')	
1	13

```
SELECT LENGTH(LTRIM(' Hello World ')) FROM dual;
```

LENGTH(LTRIM('HELLOWORLD'))	
1	12

```
SELECT LENGTH(RTRIM(' Hello World ')) FROM dual;
```

LENGTH(RTRIM('HELLOWORLD'))	
1	12

```
SELECT LENGTH(TRIM(' Hello World ')) FROM dual;
```

LENGTH(TRIM('HELLOWORLD'))	
1	11

```
SELECT ' Hello World ' || 'TEST' FROM dual;
```

'HELLOWORLD' 'TEST'	
1	Hello World TEST

```
SELECT LTRIM('xxxYYZZxYZ') FROM DUAL;
```

LTRIM('XXXYYZZXYZ')	
1	xxxYYZZxYZ

```
SELECT LTRIM('xxxYYZZxYZ', 'x') FROM DUAL;
```

LTRIM('XXXYYZZXYZ','X')	
1	YYZZxYZ

```
SELECT RTRIM('zXXYYzzXYzz','z') FROM DUAL;
```

RTRIM('ZXXYYZZXYZ','Z')	
1	zXXYYzzXY

Function

- 문자형 함수

- Q) S_NAME의 맨 마지막 문자를 '**'로 대체한 RESULT 칼럼을 출력하시오.(STUDENT 테이블 이용)

- Hint) CONCAT, SUBSTR, LENGTH 함수 사용

	S_NAME	RESULT
1	대한민국	대한**
2	홍장미	홍**
3	이리아	이**
4	둘보라	둘**
5	다스리	다**
6	이루리	이**
7	은송이	은**

- 숫자형 함수

– Q) 아래 숫자 함수의 실행 결과를 확인하고, 그 기능을 유추하시오.

숫자형 함수	함수 사용 예
ABS(숫자)	(예) ABS(-7)
SIGN(숫자)	(예) SIGN(-20), SIGN(0), SIGN(10)
MOD(숫자1, 숫자2)	(예) MOD(17, 3)
CEIL(숫자)	(예) CEIL(18.123), CEIL(-18.123)
FLOOR(숫자)	(예) FLOOR(18.123), FLOOR(-18.123)
ROUND(숫자), ROUND(숫자, m)	(예) ROUND(18.5235, 3), ROUND(18.5235) ROUND(18.5235, -1)
TRUNC(숫자), TRUNC(숫자, m)	(예) TRUNC(18.5235, 3), TRUNC(18.5235)
그 외: SIN(숫자), EXP(숫자), POWER(숫자1, 숫자2), SQRT(숫자), LOG(숫자1, 숫자2), LN(숫자)	

- 숫자형 함수

- Q) STUDENT 테이블에서 S_CODE를 활용하여 전체 학생을 4개의 그룹(0 ~ 3)에 배정하시오.

- Hint) MOD 사용

```
SELECT S_NO, S_NAME, MOD(S_NO,4) AS 그룹
FROM STUDENT;
```

	S_NO	S_NAME	그룹
1	901	대한민국	1
2	902	홍장미	2
3	903	이리아	3
4	904	눌보라	0
5	905	다스리	1
6	906	이루리	2
7	907	은송이	3

- Q) STUDENT 테이블에서 행번호를 활용하여 전체 학생을 4개의 그룹(0 ~ 3)에 배정하시오.

- Hint) ROWNUM, MOD 사용

```
SELECT S_NO, S_NAME, ROWNUM, MOD(ROWNUM,4) AS 그룹
FROM STUDENT;
```

	S_NO	S_NAME	ROWNUM	그룹
1	901	대한민국	1	1
2	902	홍장미	2	2
3	903	이리아	3	3
4	904	눌보라	4	0
5	905	다스리	5	1
6	906	이루리	6	2
7	907	은송이	7	3

- 변환형 함수

- 데이터 타입 변환

- 명시적(Explicit) 데이터 타입 변환

- 함수를 사용하여 명시적으로 데이터 타입을 변환

- 암시적(Implicit) 데이터 타입 변환

- 시스템이 자동으로 데이터 타입 변환
 - » 예: MOD(S_NO, 4) 문자열을 숫자로 변환
 - 성능 저하 및 에러 발생의 가능성 존재

```
SELECT S_NO, S_NAME, MOD(S_NO, 4) AS 그룹
```

```
FROM STUDENT;
```

```
SELECT S_NO, S_NAME, MOD(TO_NUMBER(S_NO), 4) AS 그룹
```

```
FROM STUDENT;
```

● 변환형 함수

변환형 함수	함수 설명
TO_CHAR(숫자 날짜), TO_CHAR(숫자 날짜, FORMAT)	숫자나 날짜를 문자열로 변환 (예) SELECT TO_CHAR(SYSDATE, 'YYYY/MM/DD') AS 타입1, TO_CHAR(SYSDATE, 'YYYY.MM.DD.HH24.MI.SS') AS 타입2, TO_CHAR(SYSDATE) AS 타입3 FROM DUAL;
TO_NUMBER(문자열)	문자열을 숫자로 변환 (예) SELECT '1' + '1' AS 계산 FROM DUAL; 암시적변환 (예) SELECT TO_NUMBER('1') + TO_NUMBER('1') AS 계산 F ROM DUAL; 명시적변환
TO_DATE(문자열), TO_DATE(문자열, FORMAT)	문자열을 날짜로 변환 (예) SELECT EXTRACT (YEAR FROM '20220123') AS 연도 -> ERROR!!! FROM DUAL; (예) SELECT EXTRACT (YEAR FROM TO_DATE ('20220123', 'YYYY/MM/DD')) AS 연도 FROM DUAL;

Function

● 날짜형 함수

날짜형 함수	함수 설명
SYSDATE	현재 날짜와 시각 (예) SELECT SYSDATE FROM DUAL; (예) SELECT TO_CHAR(SYSDATE, 'YYYY.MM.DD.HH24.MI.SS') FROM DUAL;
EXTRACT('YEAR' 'MONTH' 'DAY' FROM 날짜)	날짜 데이터에서 년/월/일 정보 추출 (예) EXTRACT(YEAR FROM SYSDATE) (예) SELECT S_BIRTH, EXTRACT(YEAR FROM S_BIRTH) FROM STUDENT; SELECT S_BIRTH, EXTRACT(MONTH FROM S_BIRTH) FROM STUDENT; SELECT S_BIRTH, EXTRACT(DAY FROM S_BIRTH) FROM STUDENT;
TRUNC(날짜, 'DD')	날짜 데이터에서 시/분/초를 잘라냄 (예1) SELECT TO_CHAR(SYSDATE, 'YYYY.MM.DD.HH24.MI.SS') FROM DUAL; (예2) SELECT TRUNC(SYSDATE, 'DD') FROM DUAL; (예3) SELECT TO_CHAR(TRUNC(SYSDATE, 'DD'), 'YYYY.MM.DD.HH24.MI.SS') FROM DUAL;

S_BIRTH	EXTRACT(YEAR FROM S_BIRTH)
1 00/10/09	2000
2 00/06/01	2000
3 01/11/02	2001
4 01/01/03	2001
5 00/02/25	2000
6 02/03/04	2002
7 00/07/06	2000

TO_CHAR(TRUNC(SYSDATE, 'DD'), 'YYYY.MM.DD.HH24.MI.SS')
1 2023.11.09.00.00.00

- Q) STUDENT 테이블에서 S_NAME, S_BIRTH와 함께 나이를 출력하시오.

	S_NAME	S_BIRTH	AGE
1	대한민국	00/10/09	23
2	홍장미	00/06/01	23
3	이리아	01/11/02	22
4	불보라	01/01/03	22
5	다스리	00/02/25	23
6	이루리	02/03/04	21
7	은송이	00/07/06	23

- Q) STUDENT 테이블에서 S_NAME, S_BIRTH와 함께 오늘까지 지난 날 수를 출력하시오.

	S_NAME	S_BIRTH	DAY_PASSED
1	대한민국	00/10/09	8432
2	홍장미	00/06/01	8562
3	이리아	01/11/02	8043
4	불보라	01/01/03	8346
5	다스리	00/02/25	8659
6	이루리	02/03/04	7921
7	은송이	00/07/06	8527

참조

```
SELECT S_NAME, S_BIRTH, (SYSDATE-S_BIRTH) AS RESULT
FROM STUDENT ;
```

	S_NAME	S_BIRTH	DAY_PASSED
1	대한민국	00/10/09	8432.00099537037037037037037037037037
2	홍장미	00/06/01	8562.00099537037037037037037037037037
3	이리아	01/11/02	8043.00099537037037037037037037037037
4	불보라	01/01/03	8346.00099537037037037037037037037037
5	다스리	00/02/25	8659.00099537037037037037037037037037
6	이루리	02/03/04	7921.00099537037037037037037037037037
7	은송이	00/07/06	8527.00099537037037037037037037037037

Function

- 날짜형 함수

```
SELECT S_NAME, S_BIRTH,  
EXTRACT(YEAR FROM S_BIRTH) AS 생년,  
EXTRACT(MONTH FROM S_BIRTH) AS 생월,  
EXTRACT(DAY FROM S_BIRTH) AS 생일  
FROM STUDENT;
```

	S_NAME	S_BIRTH	생년	생월	생일
1	대한민국	00/10/09	2000	10	9
2	홍장미	00/06/01	2000	6	1
3	이리아	01/11/02	2001	11	2
4	불보라	01/01/03	2001	1	3
5	다스리	00/02/25	2000	2	25
6	이루리	02/03/04	2002	3	4
7	은송이	00/07/06	2000	7	6

```
SELECT S_NAME, S_BIRTH,  
TO_NUMBER (TO_CHAR(S_BIRTH, 'YYYY')) AS 생년,  
TO_NUMBER (TO_CHAR(S_BIRTH, 'MM')) AS 생월,  
TO_NUMBER (TO_CHAR(S_BIRTH, 'DD')) AS 생일  
FROM STUDENT;
```

Function

- CASE Expression

- 표현식이지만 함수의 성격을 갖고 있음

예) S_SCORE 가 90이상이면 "BEST" 표시
그외에는 '***' 표시

```
SELECT S_NAME,S_SCORE,  
CASE  
    WHEN S_SCORE >= 90 THEN 'BEST '  
    ELSE '*** '  
END AS NEW_SCORE  
FROM STUDENT;
```

	S_NAME	S_SCORE	NEW_SCORE
1	대한민국	90	BEST
2	홍장미	80	***
3	이리아	95	BEST
4	불보라	85	***
5	다스리	66	***
6	이루리	55	***
7	은송이	77	***

- CASE Expression

구분	사용 예	특징
Searched Case Expression	<pre>SELECT S_NAME,S_SCORE, CASE WHEN S_SCORE >= 90 THEN 'A' WHEN S_SCORE >= 80 THEN 'B' WHEN S_SCORE >= 70 THEN 'C' WHEN S_SCORE >= 60 THEN 'D' ELSE 'F' END AS NEW_SCORE FROM STUDENT;</pre>	<ul style="list-style-type: none">• 표준 SQL• 다양한 조건 사용 가능• 표현식이 복잡함
Simple Case Expression	<pre>SELECT S_NAME,S_CODE, CASE S_CODE WHEN 'A1000' THEN '1층' WHEN 'B1000' THEN '2층' WHEN 'C1000' THEN '3층' WHEN 'D1000' THEN '4층' ELSE '대기' END AS 강의실 FROM STUDENT;</pre>	<ul style="list-style-type: none">• 표준 SQL• 동등(=) 비교에만 사용• 표현식이 명료함

- CASE Expression

- Searched Case Expression만 가능한 경우의 예
 - 동등(=) 이외의 조건은 Simple Case Expression 또는 Decode 함수로 표현 불가

```
SELECT S_NAME, S_SCORE,  
CASE  
    WHEN S_SCORE >= 90 THEN '장학생 '  
    WHEN S_SCORE BETWEEN 75 AND 89 THEN '보통'  
    ELSE '노력대상'  
END AS RESULT  
FROM STUDENT;
```

	S_NAME	S_SCORE	RESULT
1	대한민국	90	장학생
2	홍장미	80	보통
3	이리아	95	장학생
4	눌보라	85	보통
5	다스리	66	노력대상
6	이루리	55	노력대상
7	은송이	77	보통

- **CASE Expression**

- CASE 표현식의 중첩

```
SELECT S_NAME, S_SCORE,  
CASE  
  WHEN S_SCORE >= 90 THEN '장학생 '  
  ELSE(  
    CASE  
      WHEN S_SCORE BETWEEN 75 AND 89 THEN '보통'  
      ELSE '노력대상'  
    END)  
  END AS RESULT  
FROM STUDENT;
```


- CASE Expression

- DECODE

- Oracle에서만 사용되는 함수
 - DECODE(표현식, 기준값1, 출력값1 [, 기준값2, 출력값2, ... , 디폴트값])
 - 표현식의 값이 기준값1이면 값1을 출력, 기준값2이면 값2를 출력
 - 기준값이 없으면 디폴트 값을 출력

```
SELECT S_NAME,S_CODE,  
       DECODE(S_CODE,  
              'A1000', '1층',  
              'B1000','2층',  
              'C1000', '3층',  
              'D1000','4층',  
              '대기') AS 강의실  
FROM STUDENT;
```

Function

- NULL이란?

- 비어있는 값
- 공백(space), 0과는 다른 의미
- NULL을 포함하는 모든 산술 연산의 결과는 NULL
 - $NULL+0$, $NULL-1$, $NULL*0$, $NULL/0 \rightarrow NULL$
- NULL과 공집합도 역시 다른 의미
 - Q) 다음 두 문장의 실행 결과를 비교하시오.

```
SELECT S_NAME, S_ENGNAME FROM STUDENT  
WHERE S_NAME='대한민국';
```

```
SELECT S_NAME, S_ENGNAME FROM STUDENT  
WHERE S_NAME='우리나라'
```

	S_NAME	S_NO	S_CODE	S_BIRTH	S_SCORE	S_ENGNAME
1	대한민국	901	A1000	00/10/09	90	(null)
2	홍장미	902	B1000	00/06/01	80	LISA
3	이리아	903	C1000	01/11/02	95	RUBY
4	늘보라	904	A1000	01/01/03	85	Alice
5	다스리	905	B1000	00/02/25	66	cody
6	이루리	906	C1000	02/03/04	55	denise
7	은송이	907	D1000	00/07/06	77	(null)

S_NAME	S_ENGNAME
1 대한민국	(null)

S_NAME	S_ENGNAME

Function

- NULL 관련 함수

NULL 관련 함수	함수 설명
NVL (표현식, 대체값)	<ul style="list-style-type: none">표현식의 값이 NULL이면 대체값을, NULL이 아니면 표현식의 값을 반환표현식의 값과 대체값의 데이터 타입이 같아야 함 예) <code>SELECT S_NAME, S_ENGNAME, NVL(S_ENGNAME, '없음') AS NAME FROM STUDENT;</code>
NULLIF (표현식1, 표현식2)	<ul style="list-style-type: none">두 식이 같으면 NULL을, 같지 않으면 표현식1의 값을 반환 예) <code>SELECT S_NAME, S_ENGNAME, NULLIF(S_ENGNAME, 'RUBY') FROM STUDENT;</code>
COALESCE (표현식1, 표현식2, ...)	<ul style="list-style-type: none">임의의 개수의 표현식에서 NULL이 아닌 최초의 표현식을 반환모든 표현식이 NULL이라면 NULL을 반환 예) <code>SELECT S_ENGNAME, S_NAME, COALESCE(S_ENGNAME, S_NAME) FROM STUDENT;</code>

	S_NAME	S_ENGNAME	NAME
1	대한민국	(null)	없음
2	홍장미	LISA	LISA
3	이리아	RUBY	RUBY
4	불보라	Alice	Alice
5	다스리	cody	cody
6	이루리	denise	denise
7	은송이	(null)	없음

- NULL 관련 함수

- NVL

- NULL 값을 특정 값으로 변환할 때 사용
 - S_NAME과 S_ENGNAME을 , S_ENGNAME이 없는 경우는 '없음'으로 출력하는 예

```
SELECT S_NAME,S_ENGNAME,  
       NVL(S_ENGNAME, '없음') AS NAME  
FROM STUDENT;
```

- 위의 문장을 IS NULL과 CASE 구문으로 표현하시오.

```
SELECT S_NAME,S_ENGNAME,  
CASE  
  WHEN S_ENGNAME IS NULL THEN '없음'  
  ELSE S_ENGNAME  
END AS NAME  
FROM STUDENT;
```

	S_NAME	S_ENGNAME	NAME
1	대한민국	(null)	없음
2	홍장미	LISA	LISA
3	이리아	RUBY	RUBY
4	닐보라	Alice	Alice
5	다스리	cody	cody
6	이루리	denise	denise
7	은송이	(null)	없음

Function

- NULL 관련 함수

- NVL

- $SCORE = S_SCORE * 0.3 + S_SCORE2 * 0.7$ 이라고 할 때, 다음 식의 결과는?

```
SELECT S_NAME, S_SCORE, S_SCORE2, S_SCORE*0.3+S_SCORE2*0.7 AS SCORE  
FROM STUDENT;
```

	S_NAME	S_SCORE	S_SCORE2	SCORE
1	대한민국	90	200	167
2	홍장미	80	250	199
3	이리아	95	220	182.5
4	불보라	85	230	186.5
5	다스리	66	(null)	(null)
6	이루리	55	(null)	(null)
7	은송이	77	300	233.1

```
SELECT S_NAME, S_SCORE, S_SCORE2, NVL(S_SCORE, 0)*0.3+NVL(S_SCORE2,0)*0.7  
AS SCORE FROM STUDENT;
```

	S_NAME	S_SCORE	S_SCORE2	SCORE
1	대한민국	90	200	167
2	홍장미	80	250	199
3	이리아	95	220	182.5
4	불보라	85	230	186.5
5	다스리	66	(null)	19.8
6	이루리	55	(null)	16.5
7	은송이	77	300	233.1

- NULL 관련 함수

- NULLIF

- 특정 값을 NULL로 변환할 때 사용
 - S_NAME과 S_ENGNAME을 출력하되, S_ENGNAME이 'RUBY' 인 경우는 NULL로 출력하는 예
 - CASE 구문으로 표현 가능

```
SELECT S_NAME, S_ENGNAME,  
CASE  
    WHEN S_ENGNAME = 'RUBY' THEN NULL  
    ELSE S_ENGNAME  
END AS NAME  
FROM STUDENT;
```

	S_NAME	S_ENGNAME	S_ENGNAME_1
1	대한민국	(null)	(null)
2	홍장미	LISA	LISA
3	이리아	RUBY	(null)
4	물보라	Alice	Alice
5	다스리	cody	cody
6	이루리	denise	denise
7	은송이	(null)	(null)

- 위 문장을 NULLIF 구문으로 표현하시오.

```
SELECT S_NAME,S_ENGNAME, NULLIF(S_ENGNAME, 'RUBY') AS  
S_ENGNAME  
FROM STUDENT;
```

- NULL 관련 함수

- COALESCE

- 임의의 개수의 표현식에서 NULL이 아닌 최초의 표현식을 반환할 때 사용

```
SELECT S_ENGNAME,S_NAME, COALESCE(S_ENGNAME,S_NAME) AS  
NAME  
FROM STUDENT;
```

- CASE 구문으로 표현 가능

```
SELECT S_ENGNAME,S_NAME,  
CASE  
    WHEN S_ENGNAME IS NOT NULL THEN S_ENGNAME  
    ELSE(  
        CASE  
            WHEN S_NAME IS NOT NULL THEN S_NAME ELSE NULL  
        END)  
    END AS NAME  
FROM STUDENT;
```

	S_ENGNAME	S_NAME	NAME
1	(null)	대한민국	대한민국
2	LISA	홍장미	LISA
3	RUBY	이리아	RUBY
4	Alice	눌보라	Alice
5	cody	다스리	cody
6	denise	이루리	denise
7	(null)	은송이	은송이

수고하셨습니다 🙌