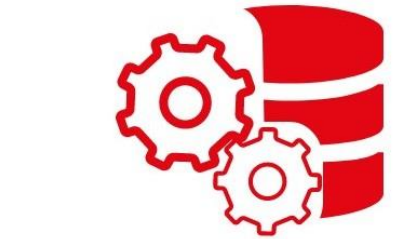


ORACLE®

DB Management

연산자 및 함수

SELECT문과 DUAL 테이블

- SELECT문
  - ✓ 데이터 질의문(DQL)
  - ✓ 연산자나 함수의 결과 확인을 위해서 사용하는 쿼리문
  - ✓ 필수 형식
    - ✓ SELECT 조회할_내용 FROM 조회할_테이블

- DUAL 테이블
  - ✓ 오라클의 SELECT문은 반드시 테이블이 필요함
  - ✓ 단순 연산이나 함수 결과를 확인할 때는 테이블이 필요 없는 경우가 있으나, 오라클에서는 SELECT문의 문법을 만족시키기 위하여 항상 테이블을 명시해야 함
    ▷ 이런 경우 SYS 계정이 가지고 있는 DUAL 테이블을 사용하면 됨
  - ✓ DUAL 테이블의 구조

COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
DUMMY	VARCHAR2(1 BYTE)	Yes	(null)	1	(null)

  - ✓ DUAL 테이블의 데이터

	DUMMY
1	X

연산자

- 산술 연산자

연산자	의미
+	더하기
-	빼기
*	곱하기
/	나누기

- NULL 비교 연산자

연산자	의미
IS NULL	NULL이면 TRUE, NULL이 아니면 FALSE
IS NOT NULL	NULL이 아니면 TRUE, NULL이면 FALSE

- 대입 연산자

연산자	의미
=	등호(=)의 오른쪽 값을 왼쪽으로 대입함

NULL

- NULL
  - ✓ 미확인 값
  - ✓ 아직 적용되지 않은 값
  - ✓ 값이 없는 상태
  - ✓ 0이나 공백(SPACE)을 의미하는 것이 아님
  - ✓ 모든 데이터 타입에서 나타날 수 있음

- NULL 의미
  - ① 회원의 나이가 NULL이다.
    - ➢ 회원의 나이를 모른다.
  - ② 제품 구매 이력이 NULL이다.
    - ➢ 제품을 한 번도 구매한 적이 없다.
  - ③ 담당 지도교수가 NULL이다.
    - ➢ 담당 지도교수가 배정되지 않았다.

NO	NAME	GENDER
1	AMANDA	F
2	JAMES	M
3	SMITH	
4	ALEX	M
5	JESSICA	F

값이 비어 있으면 NULL을 의미함

NULL 연산

- NULL 연산
  - ✓ 연산에 NULL이 포함되는 결과도 NULL임
    - ➢ NULL + 10 = NULL
    - ➢ NULL - 10 = NULL
    - ➢ NULL * 10 = NULL
    - ➢ NULL / 10 = NULL
    - ➢ 10 / NULL = NULL

- NULL 처리
  - ✓ NULL 값이 포함된 연산 결과는 NULL
  - ✓ 많은 함수들은 NULL 값을 제외하고 함수 처리
  - ✓ 따라서 NULL 값을 특정 값으로 변환한 뒤 연산하거나 함수에 포함시키기 위한 방법이 필요함
  - ✓ NVL 함수나 NVL2 함수를 이용해서 NULL 값을 다른 값으로 바꿔서 사용할 수 있음
  - ✓ NULL 값은 인덱스(INDEX)를 사용할 수 없음
  - ✓ NVL 함수나 NVL2 함수를 이용해서 NULL 값을 다른 값으로 바꾸면 인덱스(INDEX)를 사용할 수 있음

NULL 처리 함수

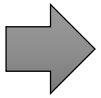
1. NVL 함수

- ✓ NULL값을 다른 값으로 바꿔서 사용
- ✓ NVL(STRING, NULL인 경우 사용할 값)

```
SELECT NVL(NAME, '아무개'), NVL(KOR, 0), NVL(ENG, 0), NVL(MATH, 0) FROM SAMPLE;
```

	NAME	KOR	ENG	MATH
1	(null)	100	100	100
2	영숙	(null)	100	100
3	정수	100	(null)	100
4	지영	100	100	(null)

SAMPLE



	NVL(NAME,'아무개')	NVL(KOR,0)	NVL(ENG,0)	NVL(MATH,0)
1	아무개	100	100	100
2	영숙	0	100	100
3	정수	100	0	100
4	지영	100	100	0

SAMPLE

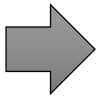
2. NVL2 함수

- ✓ NULL값을 다른 값으로 바꿔서 사용
- ✓ NVL2(STRING, NOT NULL인 경우 사용할 값, NULL인 경우 사용할 값)

```
SELECT NVL2(NAME, NAME || '님', '아무개') FROM SAMPLE;
```

	NAME	KOR	ENG	MATH
1	(null)	100	100	100
2	영숙	(null)	100	100
3	정수	100	(null)	100
4	지영	100	100	(null)

SAMPLE



	NVL2(NAME,NAME||'님','아무개')
1	아무개
2	영숙님
3	정수님
4	지영님

SAMPLE

연산자

- 비교 연산자

연산자	의미
>	크다
>=	크거나 같다
<	작다
<=	작거나 같다
=	같다 (등호 2개가 아님을 주의)
!=, <>	같지 않다

- 논리 연산자

연산자	의미
AND	모두 만족하면 TRUE, 아니면 FALSE
OR	하나라도 만족하면 TRUE, 아니면 FALSE
NOT	TRUE이면 FALSE, FALSE이면 TRUE

연산자

- 기타 연산자

연산자	의미
BETWEEN A AND B	A와 B 사이에 속하는 모든 값(A와 B를 포함함)
NOT BETWEEN A AND B	A와 B 사이에 속하지 않는 모든 값
IN(A, B, C)	A, B, C 중 하나임
NOT IN(A, B, C)	A, B, C 모두 아님
LIKE 패턴	패턴(PATTERN)과 일부 일치함
NOT LIKE 패턴	패턴(PATTERN)과 일치하는 부분이 없음
A || B	A와 B를 연결

와일드 카드와 패턴

- 와일드 카드

연산자	의미
%	모든 문자를 의미, 글자 수에 제한이 없음
_ (밑줄)	모든 문자를 의미, 글자 수가 1글자로 제한됨

- 패턴 예시

패턴	의미
김%	'김'으로 시작하는 모든 패턴 김, 김치, 김민서, 김이 모락모락 등
%김	'김'으로 끝나는 모든 패턴 김, 돌김, 매우 잘생김 등
%김%	'김'을 포함하는 모든 패턴 김, 김치, 돌김, 힘주면 김빠짐 등

집계 함수

- 주요 집계 함수

함수	의미
SUM(AGGREGATE_EXPRESSION)	AGGREGATE_EXPRESSION(집계 표현식) 합계
AVG(AGGREGATE_EXPRESSION)	AGGREGATE_EXPRESSION(집계 표현식) 평균
MAX(AGGREGATE_EXPRESSION)	AGGREGATE_EXPRESSION(집계 표현식) 최대값
MIN(AGGREGATE_EXPRESSION)	AGGREGATE_EXPRESSION(집계 표현식) 최소값
COUNT(AGGREGATE_EXPRESSION)	AGGREGATE_EXPRESSION(집계 표현식) 데이터 개수
COUNT(*)	전체 행(ROW)의 개수
STDDEV(AGGREGATE_EXPRESSION)	AGGREGATE_EXPRESSION(집계 표현식) 표준편차
VARIANCE(AGGREGATE_EXPRESSION)	AGGREGATE_EXPRESSION(집계 표현식) 분산

모든 함수에서
NULL값은 제외됨

수학 함수

- 주요 수학 함수

함수	의미
POWER(A, B)	A의 B제곱
SQRT(A)	A의 제곱근(루트 A)
ABS(A)	A의 절대값
MOD(A, B)	A를 B로 나눈 나머지
SIGN(A)	A가 양수이면 1, 음수이면 -1, 0이면 0을 반환
CEIL(A)	실수 A를 정수로 올림
FLOOR(A)	실수 A를 정수로 내림
TRUNC(A, [DIGIT])	실수 A를 DIGIT 자릿수로 절사, DIGIT 생략하면 정수로 절사
ROUND(A, [DIGIT])	실수 A를 DIGIT 자릿수로 반올림, DIGIT 생략하면 정수로 반올림

ROUND 함수

1. ROUND
   - ✓ 지정한 자릿수로 숫자를 반올림 처리함

2. 사용법

```
ROUND(EXPR¦COLUMN, [DIGIT])
```

① 지정된 DIGIT 자릿수로 반올림 처리
② DIGIT이 양수이면 소수점 아래 반올림, DIGIT이 음수이면 정수부 반올림
③ DIGIT을 생략하면 정수로 반올림(DIGIT을 0으로 지정한 것과 동일)

3. 예시

함수	의미	결과
ROUND(5555.55, 2)	소수 2자리 반올림	5555.56
ROUND(5555.55, 1)	소수 1자리 반올림	5555.6
ROUND(5555.55)	정수 반올림	5556
ROUND(5555.55, -1)	일의 자리 반올림	5560
ROUND(5555.55, -2)	십의 자리 반올림	5600

TRUNC 함수

1. TRUNC
   - ✓ 지정한 자릿수로 숫자를 절사 처리함

2. 사용법

```
TRUNC(EXPR¦COLUMN, [DIGIT])
```

① 지정된 DIGIT 자릿수로 절사 처리
② DIGIT이 양수이면 소수점 아래 절사, DIGIT이 음수이면 정수부 절사
③ DIGIT을 생략하면 정수로 절사(DIGIT을 0으로 지정한 것과 동일)

3. 예시

함수	의미	결과
TRUNC(5555.55, 2)	소수 2자리 절사	5555.55
TRUNC(5555.55, 1)	소수 1자리 절사	5555.5
TRUNC(5555.55)	정수 절사	5555
TRUNC(5555.55, -1)	일의 자리 절사	5550
TRUNC(5555.55, -2)	십의 자리 절사	5500

날짜 함수

- 주요 날짜 함수

함수	의미
SYSDATE	오라클이 설치된 서버의 현재 날짜와 시간 (DATE 타입)
SYSTIMESTAMP	오라클이 설치된 서버의 현재 날짜와 시간 (TIMESTAMP 타입)
CURRENT_DATE	SESSIONTIMEZONE의 현재 날짜와 시간 (DATE 타입)
CURRENT_TIMESTAMP	SESSIONTIMEZONE의 현재 날짜와 시간 (TIMESTAMP 타입)
EXTRACT ({ YEAR | MONTH | DAY | HOUR | MINUTE | SECOND } FROM DATE)	지정된 DATE에서 필요한 정보 추출
NEXT_DAY(DATE, { 일 | 월 | 화 | 수 | 목 | 금 | 토 })	지정된 DATE의 다음 WEEKDAY(일~월) 반환
LAST_DAY(DATE)	지정된 DATE의 해당 월 말일 반환
ADD_MONTHS(DATE, N)	지정된 DATE의 N개월 후 날짜
MONTHS_BETWEEN(DATE1, DATE2)	두 날짜(최근 DATE1, 이전 DATE2) 사이에 경과한 개월 수

날짜 연산 및 함수

- 날짜 연산

함수	의미	결과
TO_DATE('20/01/01') + 1	2020/01/01의 1일 후 날짜	'20/01/02'
TO_DATE('20/01/01') - 1	2020/01/01의 1일 전 날짜	'19/12/31'
TO_DATE('20/01/01') + (1 / 24)	2020/01/01 0:00:00의 1시간 후	'20/01/01 01:00:00'
TO_DATE('20/01/01') + (1 / 24 / 60)	2020/01/01 0:00:00의 1분 후	'20/01/01 00:01:00'
TO_DATE('20/01/01') + (1 / 24 / 60 / 60)	2020/01/01 0:00:00의 1초 후	'20/01/01 00:00:01'
TO_DATE('20/01/01') - TO_DATE('19/01/01')	2019/01/01 ~ 2020/01/01 사이 경과 일	365

- 날짜 함수

함수	의미	결과
ADD_MONTHS('20/01/01', 3)	2020/01/01의 3개월 후 날짜	'20/04/01'
MONTHS_BETWEEN('20/04/01', '20/01/01')	2020/01/01 ~ 2020/04/01 사이 경과 개월	3

타입 변환 함수

- 타입 변환 함수

함수	의미
TO_NUMBER(STRING)	문자열 STRING을 숫자로 변경
TO_CHAR(NUMBER|DATE, [FORMAT])	숫자 NUMBER나 날짜 DATE를 지정된 FORMAT의 문자열로 변경
TO_DATE(STRING, [FORMAT])	문자열 STRING을 FORMAT 형식으로 해석한 DATE 날짜로 변경
TO_TIMESTAMP(STRING, [FORMAT])	문자열 STRING을 FORMAT 형식으로 해석한 TIMESTAMP 날짜로 변경

FORMAT

- 숫자 FORMAT

종류	의미	예시	결과
9	숫자 한 자리	(1234, '999999')	1234
0	숫자 한 자리	(1234, '000000')	001234
.	소수점 표시	(1234, '9999.99') (1234, '0000.00')	1234.00 1234.00
,	천 단위 구분 기호	(1234, '9,999') (12345, '99,999')	1,234 12,345

- 날짜 FORMAT (2020-01-01 기준)

종류	의미	결과
YY	년도 2자리	20
YYYY	년도 4자리	2020
Q	분기	1 (1-3월)
MM	월 2자리	01
MON	월 영어 3글자	JAN
MONTH	월 영어	JANUARY
DD	일 2자리	01

종류	의미	결과
DY	요일 1글자	수
DAY	요일 전체	수요일
AM	오전/오후	오전
HH	12시각	1~12
HH24	24시각	0~23
MI	분	0~59
SS	초	0~59

타입 변환 함수

- TO_NUMBER

함수	의미	결과
TO_NUMBER('1234')	문자열 '1234'를 숫자 1234로 변환	1234
TO_NUMBER('3.14')	문자열 '3.14'를 숫자 3.14로 변환	3.14

- TO_CHAR

함수	의미	결과
TO_CHAR(1234, '999999')	공백 포함 6자리로 표현	' 1234'
TO_CHAR(1234, '000000')	0 포함 6자리로 표현	'001234'
TO_CHAR('20/01/01', 'YYYY-MM-DD')	YYYY-MM-DD 형식으로 표현	'2020-01-01'

- TO_DATE

함수	의미	결과
TO_DATE('05/01/2020', 'MM/DD/YYYY')	MM/DD/YYYY 형식으로 해석한 날짜	'2020/05/01'
TO_DATE('05/01/2020', 'DD/MM/YYYY')	DD/MM/YYYY 형식으로 해석한 날짜	'2020/01/05'

묵시적 타입 변환

- 묵시적 타입 변환
  - ✓ 정확한 연산을 위해 오라클에서 데이터 타입을 내부적으로 변환하는 것을 의미함
  - ✓ 주로 숫자와 문자를 연산할 때 발생

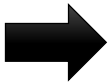
- 묵시적 타입변환 방식
  - ✓ 기본적인 변환은 문자(VARCHAR2) → 숫자(NUMBER) 방식임
  - ① STRING 타입 : NUMBER, 연산할 상수 : VARCHAR2 또는 CHAR
    - ➢ 연산할 상수를 NUMBER 타입으로 수정하여 연산
  - ② STRING 타입 : VARCHAR2 또는 CHAR, 연산할 상수 : NUMBER
    - ➢ STRING의 데이터를 NUMBER 타입으로 수정하여 연산

STRING의 데이터를 NUMBER 타입으로
변환하기 위해서 내부적으로
TO_NUMBER 함수가 사용됨

묵시적 타입 변환

1. STRING 타입 ➢ NUMBER
   연산 데이터 ➢ VARCHAR2

```
SELECT *
FROM DEPARTMENT
WHERE DEPT_NO = '10';
```

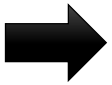


```
SELECT *
FROM DEPARTMENT
WHERE DEPT_NO = TO_NUMBER('10');
```

DEPT_NO STRING을 그대로 사용할 수 있으므로 아무 문제 없음

2. STRING 타입 ➢ VARCHAR2
   연산 데이터 ➢ NUMBER

```
SELECT *
FROM STUDENT
WHERE STU_NO = 10101;
```



```
SELECT *
FROM STUDENT
WHERE TO_NUMBER(STU_NO) = 10101;
```

STU_NO STRING에 빠른 검색을 위한 인덱스가 설정되어 있더라도,
TO_NUMBER 함수로 인해서
인덱스 사용이 불가능하기 때문에
처리 속도가 느려짐

문자열 함수

- 주요 문자열 함수

함수	의미
UPPER(STRING)	STRING을 모두 대문자로 변환
LOWER(STRING)	STRING을 모두 소문자로 변환
INITCAP(STRING)	STRING의 첫 글자는 대문자 나머지 글자는 소문자로 변환
LENGTH(STRING)	STRING의 글자 수
CONCAT(STRING1, STRING2)	STRING1, STRING2를 연결
SUBSTR(STRING, BEGIN, LENGTH)	STRING의 BEGIN 위치부터 LENGTH만큼 가져옴
INSTR(STRING, FIND)	STRING에서 FIND의 위치를 가져옴
LPAD(STRING, WIDTH, CHAR)	WIDTH에 맞춰 STRING의 왼쪽에 CHAR를 채움
RPAD(STRING, WIDTH, CHAR)	WIDTH에 맞춰 STRING의 오른쪽에 CHAR를 채움
LTRIM(STRING, [CHAR])	STRING의 왼쪽 CHAR 제거, CHAR 생략 시 공백 제거
RTRIM(STRING, [CHAR])	STRING의 오른쪽 CHAR 제거, CHAR 생략 시 공백 제거
TRIM(STRING)	STRING의 양쪽 공백 제거

CONCAT 함수

1. CONCAT
   - ✓ 두 문자열을 하나로 연결

2. 사용법

```
CONCAT(EXPR¦COLUMN, EXPR¦COLUMN)
```

① 오직 2개의 문자열만 연결 가능
② 3개 이상의 문자열을 연결하려면 CONCAT 함수를 여러 개 사용해야 함

3. 예시

함수	의미	결과
CONCAT('APPLE', 'JUICE')	'APPLE'과 'JUICE'를 연결	'APPLEJUICE'
CONCAT('APPLE', CONCAT(' ', 'JUICE'))	'APPLE'과 ' '과 'JUICE'를 연결	'APPLE JUICE'

SUBSTR 함수

1. SUBSTR
   - ✓ 문자열의 일부만 추출

2. 사용법

```
SUBSTR(EXPR¦COLUMN, BEGIN, [LENGTH])
```

① 문자열의 BEGIN번째 문자부터 LENGTH개의 문자열을 추출
② 첫 번째 문자의 BEGIN 값은 1
③ BEGIN이 음수이면 뒤에서부터 BEGIN번째 문자라는 의미임
④ LENGTH를 생략하면 마지막 문자까지 추출

3. 예시

함수	의미	결과
SUBSTR('APPLE', 1, 3)	1번째 문자부터 3개 추출	'APP'
SUBSTR('APPLE', 3)	3번째 문자부터 끝까지 추출	'PPL'
SUBSTR('010-1111-2222', 10)	10번째 문자부터 끝까지 추출	'2222'
SUBSTR('010-1111-2222', -4)	-4번째(끝에서 4번째) 문자부터 끝까지 추출	'2222'

INSTR 함수

1. INSTR
   - ✓ 문자열에서 지정된 문자가 존재하는 위치를 반환

2. 사용법

```
INSTR(EXPR¦COLUMN, CHAR, [BEGIN, [N]])
```

① 문자열의 BEGIN번째 문자부터 N번째 CHAR를 찾아서 그 위치를 반환함
② BEGIN 음수이면 뒤에서 BEGIN번째 문자부터 검색하라는 의미임
③ BEGIN을 생략하면 처음부터 찾음
④ N을 생략하면 일치하는 1번째 문자를 찾음

3. 예시

함수	의미	결과
INSTR('APPLE APPLY', 'APP')	1번째 문자부터 검색, 1번째 APP의 위치	1
INSTR('APPLE APPLY', 'APP', 4)	4번째 문자부터 검색, 1번째 APP의 위치	7
INSTR('APPLE APPLY', 'APP', 1, 2)	1번째 문자부터 검색, 2번째 APP의 위치	7
INSTR('APPLE APPLY', 'APP', -1, 2)	-1번째 문자부터 검색, 2번째 APP의 위치	1

LPAD/RPAD 함수

1. LPAD(RPAD)
   - ✓ 문자열을 지정한 길이로 맞추기 위해서 왼쪽(오른쪽)에 지정한 문자를 채움

2. 사용법

```
LPAD(EXPR¦COLUMN, WIDTH, [CHAR])
RPAD(EXPR¦COLUMN, WIDTH, [CHAR])
```

   ① 문자열을 WIDTH 길이로 맞추기 위해서 왼쪽(오른쪽)에 CHAR를 채움
   ② CHAR를 생략하면 공백(스페이스)을 채움

3. 예시

함수	의미	결과
LPAD('APPLE', 10)	APPLE을 10자리로 표시, 왼쪽에 공백 채움	' APPLE'
LPAD('APPLE', 10, '*')	APPLE을 10자리로 표시, 왼쪽에 * 채움	'*****APPLE'
RPAD('APPLE', 10)	APPLE을 10자리로 표시, 오른쪽에 공백 채움	'APPLE '
RPAD('APPLE', 10, '*')	APPLE을 10자리로 표시, 오른쪽에 * 채움	'APPLE*****'

LTRIM/RTRIM 함수

1. LTRIM(RTRIM)
   - ✓ 문자열 왼쪽(오른쪽)의 지정한 문자를 제거

2. 사용법

```
LTRIM(EXPR¦COLUMN, [CHAR])
RTRIM(EXPR¦COLUMN, [CHAR])
```

   - ① 문자열의 왼쪽(오른쪽)에 있는 CHAR를 삭제함
   - ② CHAR를 생략하면 공백(스페이스)을 삭제함

3. 예시

함수	의미	결과
LTRIM(' APPLE')	왼쪽의 공백을 제거	'APPLE'
LTRIM('*****APPLE', '*')	왼쪽의 *를 제거	'APPLE'
RTRIM('APPLE ')	오른쪽의 공백을 제거	'APPLE'
RTRIM('APPLE*****', '*')	오른쪽의 *를 제거	'APPLE'

TRIM 함수

1. TRIM
   - ✓ 문자열 양쪽(왼쪽과 오른쪽)의 공백을 제거

2. 사용법

```
TRIM(EXPR¦COLUMN)
```

   ① 문자열 양쪽의 불필요한 공백을 모두 제거함
   ② 제거할 문자를 지정하는 것은 불가능함

3. 예시

함수	의미	결과
TRIM(' APPLE ')	양쪽(왼쪽, 오른쪽 모두)의 공백을 제거	'APPLE'