MySQL & Maria DB

SQL 주요 함수

목차

숫자함수

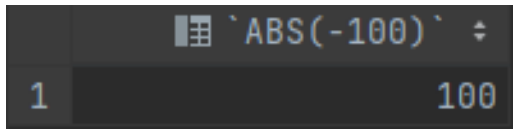
- 숫자 함수

구 분	설 명
ABS	절대값을 구한다.
ACOS(숫자),ASIN(숫자),ATAN(숫자) ATAN2(숫자1,숫자2), SIN(숫자), COS(숫자), TAN(숫자)	삼각 함수와 관련된 함수를 제공한다
CEILING(숫자),FLOOR(숫자), ROUND(숫자)	올림,내림,반올림을 계산한다.
CONV(숫자, 원래 진수, 변환할 진수)	숫자를 원래 진수에서 변환할 진수로 계산한다.
DEGREES(숫자),RADIANS(숫자),PI()	라디안 값을 각도값으로, 각도값을 라디안 값으로 변환한다.PI()는 파이값인 3.141592를 반환한다.
EXP(X),LN(숫자),LOG(숫자),LOG(밑수,숫자),LOG2(숫자),LOG10(숫자)	지수, 로그와 관련된 함수를 제공한다.
MOD(숫자1,숫자2) 숫자1 % 숫자2 / 숫자1 MOD 숫자2	숫자1을 숫자2로 나눈 나머지 값을 구한다.
POW(숫자1,숫자2), SQRT(숫자)	거듭제곱값 및 제곱근을 구한다.
RAND()	0 이상 1 미만의 실수를 구한다.
SIGN(숫자)	숫자가 양수,0,음수인지 구한다. 결과는 1, 0, -1 셋 중에 하나를 반환한다.
TRUNCATE(숫자, 정수)	숫자를 소수점을 기준으로 정수 위치까지 구하고 나머지는 버린다.

숫자함수

- ABS
  - ABS 함수는 절대값을 구합니다. 절대값은 방향은 없고 크기만 있는 것으로서 주어진 데이터가 음수일 경우 양수로 표현합니다.

예	**SELECT ABS(-100);**



	`ABS(-100)`
1	100

숫자함수

- CEILING, FLOOR, ROUND
  - CEILING은 올림, FLOOR는 내림, ROUND는 반올림한 숫자를 반환합니다. 아래에서는 4, 3, 4를 반환합니다.

예	**SELECT CEILING(3.7), FLOOR(3.7), ROUND(3.7);**

	`CEILING(3.7)`	`FLOOR(3.7)`	`ROUND(3.7)`
1	4	3	4

숫자함수

- MOD
  - MOD는 숫자1을 숫자2로 나눈 나머지 값을 구합니다. 아래에서는 모두 2를 반환합니다.

예	**SELECT MOD(14, 3), 14 % 3, 14 MOD 3**

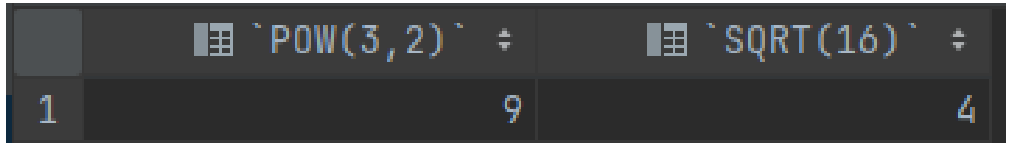
	`MOD(14, 3)`	`14 % 3`	`14 MOD 3`
1	2	2	2

  - 돌발 문제 : 사번이 홀수인 사람들을 검색해 보십시오.

숫자함수

- POW, SQRT
  - POW는 숫자1의 숫자2제곱 값을 구합니다. 아래에서는 9가 반환됩니다.
  - SQRT는 숫자의 제곱근 값을 구합니다. 아래에서는 4가 반환됩니다.

예	**SELECT POW(3,2), SQRT(16)**

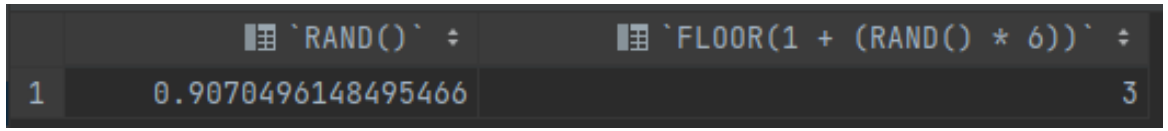


	`POW(3,2)`	`SQRT(16)`
1	9	4

숫자함수

- RAND
  - RAND는 0 이상 1 미만의 무작위 실수를 구합니다. 아래에서는 (0~1사이의 실수)와 (1,2,3,4,5,6 중 랜덤으로 하나)를 반환합니다.

예	**SELECT RAND(), FLOOR(1 + (RAND() * 6));**



	`RAND()`	`FLOOR(1 + (RAND() * 6))`
1	0.9070496148495466	3

숫자함수

- TRUNCATE
  - TRUNCATE는 소숫점을 기준으로 정수 위치까지 구하고 나머지를 버립니다. 아래에서는 1234.12, 1200을 반환합니다.

예	**SELECT TRUNCATE(1234.1234, 2), TRUNCATE(1234.1234, -2);**

	`TRUNCATE(1234.1234, 2)`	`TRUNCATE(1234.1234, -2)`
1	1234.12	1200

숫자함수

- CONV
  - CONV는 기존 진수의 숫자를 변환할 지수로 계산 후 반환합니다. 아래에서는 10진수 100을 2진수로 변환한 144가 반환됩니다.

예	**SELECT CONV(100, 10, 2);**

	`CONV(100, 10, 2)`
1	1100100

숫자함수

- DEGREES, PI, RADIANS
  - DEGREES는 라디안 값을 각도 값으로, RADIANS는 각도 값을 라디안 값으로 변환합니다.
    PI()는 3.14152를 반환합니다.
  - 아래에서는 180, 원주율값이 반환됩니다.

예	**SELECT DEGREES(PI()), RADIANS(180);**

	`DEGREES(PI())`	`RADIANS(180)`
1	180	3.141592653589793

문자함수

- 문자 함수

구 분	설 명
ASCII(아스키코드)/CHAR(숫자)	아스키 코드를 숫자로, 숫자를 아스키 코드로 돌려준다.
BIT_LENGTH(문자열), CHAR_LENGTH(문자열), LENGTH(문자열)	해당 문자열의 비트 크기를 반환한다. 해당 문자열의 문자 크기를 반환한다. 해당 문자열의 바이트 크기를 반환한다.
CONCAT(문자열1, 문자열2), CONCAT(문자열1,……)	문자열을 이어준다.
INSERT(기준 문자열, 위치, 길이, 삽입할 문자열)	기준 문자열의 위치부터 길이만큼을 지우고 삽입할 문자열을 끼워 넣는다.
LEFT(문자열, 길이), RIGHT(문자열, 길이)	왼쪽 또는 오른쪽에서 문자열의 길이만큼 반환한다.
UCASE(문자열)/UPPER(문자열) **LCASE(문자열)/LOWER(문자열)**	소문자를 대문자로, 대문자를 소문자로 변경한다.
LPAD(문자열, 길이, 채울 문자열) **RPAD(문자열, 길이, 채울 문자열)**	문자열을 길이만큼 늘린 후에 빈속을 채울 문자열로 채운다.
LTRIM(문자열),RTRIM(문자열)	문자열의 왼쪽/오른쪽 공백을 제거한다
TRIM(문자열)/ TRIM(방향 자를문자열 FROM 문자열)	TRIM은 문자열의 앞뒤 공백을 제거한다. 혹은 해당 문자를 제거하는데 제거할 문자는 LEADING(앞)/BOTH(양쪽)/TRAILING(뒤) 로 나뉠 수 있다.
REPEAT(문자열, 횟수)	문자열을 횟수만큼 반복한다.
REPLACE(문자열, 원래 문자열, 바꿀 문자열)	문자열에서 원래 문자열을 찾아서 바꿀 문자열로 바꿔준다.
REVERCE(문자열)	문자열의 순서를 거꾸로 만든다.
SPACE(길이)	길이 만큼의 공백을 반환한다.

문자함수

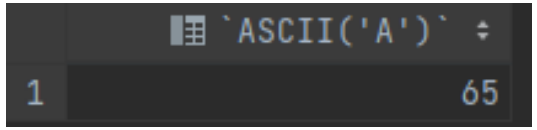
- 문자 함수

구 분	설 명
SUBSTRING(문자열, 시작위치, 길이) **SUBSTRING(문자열 FROM 시작위치 FOR 길이)**	시작 위치부터 길이만큼 문자를 반환한다. 길이가 생략되면 문자열의 끝까지 반환한다.
SUBSTRING_INDEX(문자열, 구분자, 횟수)	문자열에서 구분자가 왼쪽부터 횟수 번째까지 나오면 그 이후의 오른쪽은 버린다. 횟수가 음수면 오른쪽부터 세고 왼쪽을 버린다.

문자함수

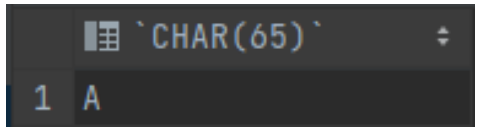
- ASCII
  - ASCII는 문자의 아스키 코드값을 반환합니다. 아래에서는 65가 반환됩니다.

예	SELECT ASCII('A');



- CHAR
  - CHAR은 아스키 코드값에 해당하는 문자를 반환합니다. 아래에서는 'A'가 반환됩니다.

예	SELECT CHAR(65);



문자함수

- BIT_LENGTH, CHAR_LENGTH, LENGTH()
  - BIT_LENGTH는 할당된 Bit 크기, CHAR_LENGTH는 문자의 개수, LENGTH()는 할당된 Byte 수를 반환합니다. 아래에서는 첫 번줄에서 24, 3, 3과 둘째 줄에서 72, 3, 9가 반환됩니다. (UTP-8 코드에서는 한글은 문자당 3바이트이기 때문.)

예	**SELECT BIT_LENGTH('abc'), CHAR_LENGTH('abc'), LENGTH('abc');** **SELECT BIT_LENGTH('가나다'), CHAR_LENGTH('가나다'), LENGTH('가나다');**

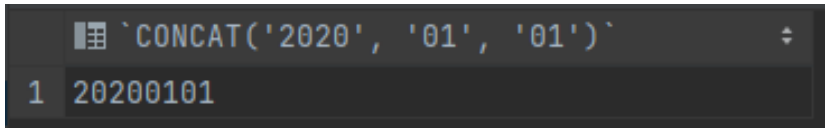
	`BIT_LENGTH('abc')`	`CHAR_LENGTH('abc')`	`LENGTH('abc')`
1	24	3	3

	`BIT_LENGTH('가나다')`	`CHAR_LENGTH('가나다')`	`LENGTH('가나다')`
1	72	3	9

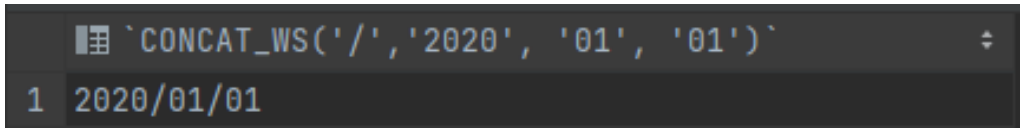
문자함수

- CONCAT, CONCAT_WS
  - CONCAT은 문자열을 이을 때 사용합니다. 아래에서는 ' 20200101 ' 이 반환됩니다.
  - CONCAT_WS는 구분자와 함께 문자열을 이을 때 사용합니다. 아래에서는 '2020/01/01'이 반환됩니다.

예	**SELECT CONCAT('2020', '01', '01');** **SELECT CONCAT_WS('/','2020', '01', '01');**



	`CONCAT('2020', '01', '01')`
1	20200101



	`CONCAT_WS('/','2020', '01', '01')`
1	2020/01/01

문자함수

- ELT, FIELD, FIND_IN_SET, INSTR, LOCATE
  - ELT는 위치 번째의 문자를 반환합니다. 아래에서는 'b'을 반환합니다.
  - FIELD는 찾을 문자열의 위치를 찾아서 있으면 위치를, 없으면 0을 반환합니다. 아래에서는 '2'를 반환합니다.
  - FIND_IN_SET은 찾을 문자열을 문자열 리스트에서 찾아서 위치를 반환합니다. 문자열 리스트는 콤마(,)로 구분되어 있어야 하며 공백이 없어야 합니다. 아래에서는 '2'를 반환합니다.
  - INSTR은 기준 문자열에서 부분 문자열을 찾아서 그 시작 위치를 반환합니다. 아래에서는 2를 반환합니다.
  - LOCATE는 INSTR와 동일하지만 파라미터의 순서가 반대로 되어있습니다. POSITION()과 동일한 함수입니다.

문자함수

- ELT, FIELD, FIND_IN_SET, INSTR, LOCATE

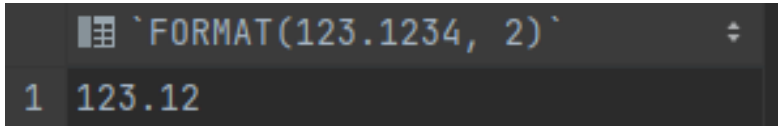
예	SELECT ELT(2, 'a', 'b', 'c'), -- ELT(위치, 문자열1, 문자열2, ...) FIELD('b', 'a', 'b', 'c'), -- FIELD(찾을 문자열, 문자열1, 문자열2, ...) FIND_IN_SET('b', 'a,b,c'), -- FIND_IN_SET(찾을 문자열, 문자열 리스트) INSTR('abcd', 'b'), -- INSTR(기준 문자열, 부분 문자열) LOCATE('b', 'abcd'); -- LOCATE(부분 문자열, 기준 문자열);

	ELT	FIELD	FIND_IN_SET	INSTR	LOCATE
1	b	2	2	2	2

문자함수

- FORMAT
  - 숫자를 소숫점 아래 자릿수까지만 표현합니다. 그리고 1000 단위마다 콤마를 표시합니다. 위에서는 123.12를 반환합니다..

예	**SELECT FORMAT(123.1234, 2); -- FORMAT(숫자, 소숫점 자릿수)**



문자함수

- BIN, HEX, OCT
  - BIN은 2진수, HEX는 16진수, OCT는 8진수 값을 반환합니다. 위에서는 11111, 1F, 37을 반환됩니다.

예	**SELECT BIN(31), -- BIN(숫자) HEX(31), -- HEX(숫자) OCT(31); -- OCT(숫자)**

	`BIN(31)`	`HEX(31)`	`OCT(31)`
1	11111	1F	37

문자함수

- INSERT
  - 기준 문자열의 위치부터 길이만큼을 삽입할 문자열로 변경합니다.. 위에서는 '가@@@마'가 반환됩니다.

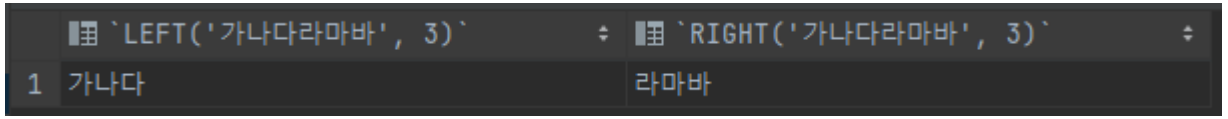
예	**SELECT INSERT('가나다라마', 2, 3, '@@@');** **-- INSERT(기준 문자열, 위치, 길이, 삽입할 문자열)**

	`INSERT('가나다라마', 2, 3, '@@@')`
1	가@@@마

문자함수

- LEFT, RIGHT
  - LEFT는 문자열의 왼쪽부터 길이만큼 반환합니다. 위에서는 '가나다'가 반환됩니다.
  - RIGHT는 문자열의 오른쪽부터 길이만큼 반환합니다. 위에서는 '라마바'가 반환됩니다.

예	**SELECT LEFT('가나다라마바', 3), -- LEFT(문자열,길이) RIGHT('가나다라마바', 3); -- RIGHT(문자열,길이)**



	`LEFT('가나다라마바', 3)`	`RIGHT('가나다라마바', 3)`
1	가나다	라마바

문자함수

- LCASE, UCASE
  - LCASE는 대문자를 소문자로 변경한 후 반환합니다. 위에서는 'abcde'가 반환됩니다. LOWER함수와 동일합니다.
  - UCASE는 소문자를 대문자로 변경한 후 반환합니다. 위에서는 'ABCDE'가 반환됩니다. UPPER함수와 동일합니다.

예	**SELECT LCASE(‘aBcDe’), -- LCASE(문자열) UCASE(‘aBcDe’); -- UCASE(문자열)**

	`LCASE('aBcDe')`	`UCASE('aBcDe')`
1	abcde	ABCDE

문자함수

- LPAD, RPAD
  - LPAD/RPAD는 문자열을 길이만큼 왼쪽/오른쪽에서 늘린 후에 빈 곳을 채울 문자열로 채웁니다. 위에서는 '@@가나다', '가나다@@'을 반환합니다.

예	**SELECT LPAD('가나다', 5, '@@'), -- LPAD(문자열, 길이, 채울 문자열) RPAD('가나다', 5, '@@'); -- RPAD(문자열, 길이, 채울 문자열)**

	`LPAD('가나다', 5, '@@')`	`RPAD('가나다', 5, '@@')`
1	@@가나다	가나다@@

문자함수

- LTRIM, RTRIM, TRIM
  - LTRIM/RTRIM은 문자열의 왼쪽/오른쪽 공백을 제거합니다.
  - TRIM은 양쪽의 공백을 모두 제거 합니다.
  - 중간의 공백은 제거하지 않습니다. 위에서는 모두 'abc'를 반환합니다.

예	**SELECT LTRIM(' abc'), -- LTRIM(문자열) RTRIM('abc '), -- RTRIM(문자열) TRIM(' abc '); -- TRIM(문자열)**

	`LTRIM('  abc')`	`RTRIM('abc  ')`	`TRIM('  abc  ')`
1	abc	abc	abc

문자함수

- TRIM FROM
  - TRIM FROM은 앞(LEADING)또는 뒤(TRAILING)또는 양쪽(BOTH)에서 원하는 문자열을 자를 수 있습니다. 위는 'bab'를 반환합니다..

예	**SELECT TRIM(BOTH 'a' FROM 'aababaa'); -- TRIM(방향 자를문자열 FROM 문자열)**

	`TRIM(BOTH 'a' FROM 'aababaa')`
1	bab

문자함수

- REPEAT
  - REPEAT은 문자열을 횟수만큼 반복합니다. 위에서는 'abcabcabc'를 반환합니다.

예	**SELECT TRIM(BOTH 'a' FROM 'aababaa'); -- TRIM(방향 자를문자열 FROM 문자열)**

	`REPEAT('abc', 3)`
1	abcabcabc

- REPLACE
  - REPLACE는 문자열에서 원래 문자열을 찾아서 바꿀 문자열로 바꿔줍니다. 위에서는 'It is apple'을 반환합니다.

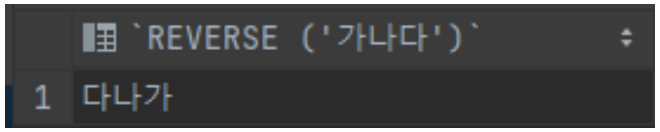
예	**SELECT REPLACE ('It is banana', 'banana', 'apple'); -- REPLACE(문자열, 원래 문자열, 바꿀 문자열))**

	`REPLACE ('It is banana', 'banana', 'apple')`
1	It is apple

문자함수

- REVERSE
  - REVERSE는 문자열을 거꾸로 만듭니다. 위에서는 '다나가'를 반환합니다.

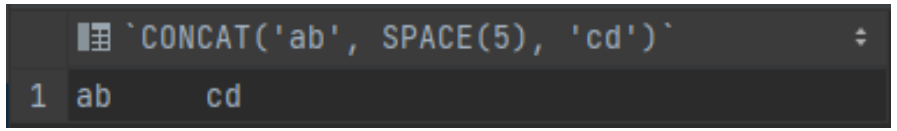
예	**SELECT REVERSE ('가나다'); -- REVERSE(문자열)**



	`REVERSE ('가나다')`
1	다나가

- SPACE
  - SPACE는 길이만큼의 공백을 반환합니다. 위에서는 'ab cd'를 반환합니다.

예	**SELECT CONCAT('ab', SPACE(5), 'cd'); -- SPACE(길이)**

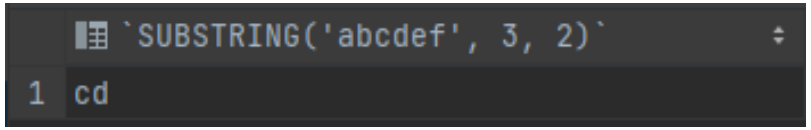


	`CONCAT('ab', SPACE(5), 'cd')`
1	ab cd

문자함수

- SUBSTRING
  - SUBSTRING은 시작 위치부터 길이만큼 문자를 반환합니다. 위에서는 'cd'를 반환합니다. SUBSTRING(문자열 FROM 시작위치 FOR 길이)문법으로 작성하기도 합니다. SUBSTR, MID 함수와 동일합니다.

예	**SELECT SUBSTRING('abcdef', 3, 2); -- SUBSTRING(문자열, 시작위치, 길이)**



문자함수

- SUBSTRING_INDEX
  - SUBSTRING_INDEX는 문자열에서 시작부터 구분자가 횟수번나오면 그 이후는 버립니다. 횟수가 음수면 오른쪽부터 시작합니다. 위에서는 'jaehoney.tistory', 'tistory.com'이 반환됩니다.

예	**SELECT SUBSTRING_INDEX('jaehoney.tistory.com', '.', 2);** **SELECT SUBSTRING_INDEX('jaehoney.tistory.com', '.', -2);** **-- SUBSTRING_INDEX (문자열, 구분자, 횟수)**

	`SUBSTRING_INDEX('jaehoney.tistory.com', '.', 2)`
1	jaehoney.tistory

	`SUBSTRING_INDEX('jaehoney.tistory.com', '.', -2)`
1	tistory.com

날짜 및 시간 함수

날짜 및 시간함수

구분	설명	구분	설명
SYSDATE	시스템 저장된 현재 날짜를 반환한다.	ADDDATE(날짜, 차이), SUBDATE(날짜, 차이)	날짜를 기준으로 차이를 더하거나 뺀 날짜를 구한다.
MONTHS_BETWEEN	두 날짜 사이가 몇 개월인지를 반환한다.	ADDTIME(날짜/시간, 시간), SUBTIME(날짜/시간, 시간)	날짜/시간을 기준으로 시간을 더하거나 뺀 결과를 구한다.
YEAR(날짜),MONTH(날짜),DAY(날짜), HOUR(시간),MINUTE(시간), SECOND(시간),MICROSECOND(시간)	날짜 또는 시간에서 연,월,일,시,분,초,밀리초를 구한다.	CURDATE(),CURTIME(),NOW(), SYSDATE()	CURDATE()는 현재 년-월-일, CURTIME()은 현재 시:분:초를 구한다 나머지 두 함수는 연월일시분초를 전부 구한다.
LAST_DAY	해당 달의 마지막 날짜를 반환한다.	ADD_MONTHS	특정 날짜에 개월 수를 더한다.
NEXT_DAY	특정 날짜에서 최초로 도래하는 인자로 받은 요일의 날짜를 반환한다.	DATE(),TIME()	DATETIME 형식에서 연월일 및 시분초만 추출한다.
PERIOD_ADD(연월, 개월수), PERIOD_DIFF(연월1, 연월2)	연월에서 개월만큼의 개월이 지난 연월을 구한다.	DATEDIFF(날짜1,날짜2),TIMEDIFF(날짜1 또는 시간1, 날짜1 또는 시간2)	DATEDIFF()는 날짜1-날짜2의 일수를 결과로 구한다. TIMEDIFF()는 시간1-시간2의 결과를 구한다.
TRUNC	인자로 받은 날짜를 특정 기준으로 버린다.	DAYOFWEEK(날짜), MONTHNAME(),DAYOFYEAR(날짜)	요일(1:일 ~ 7:토) 중 하나를 반환하거나 월 이름을 반환 혹은 1년 중 몇 번째 날짜인지 구한다.
ROUND	인자로 받은 날짜를 특정 기준으로 반올림한다.	LAST_DAY(날짜)	주어진 날짜의 마지막 날짜를 구한다.
QUARTER(날짜)	날짜가 4분기중에서 몇 분기인지 구한다	MAKEDATE(연도,정수)	연도에서 정수만큼 지난 날짜를 구한다
TIME_TO_SEC(시간)	시간을 초 단위로 구한다.	MAKETIME(시,분,초)	시,분,초를 이용해서 '시:분:초' 의 TIME형식을 만든다

날짜 및 시간 함수

- ADDDATE, SUBDATE
  - ADDDATE는 날짜를 기준으로 차이를 더한 날짜를 구합니다. 위에서는 '2020-02-01'이 반환됩니다.
  - SUBDATE는 날짜를 기준으로 차이를 뺀 날짜를 구합니다. 위에서는 '2019-12-01'이 반환됩니다.
  - DATE_ADD, DATE_SUB와 동일한 함수입니다.

예	**SELECT ADDDATE('2020-01-01', INTERVAL 31 DAY), -- ADDDATE(날짜, 차이) SUBDATE('2020-01-01', INTERVAL 31 DAY); -- SUBDATE(날짜, 차이)**

	`ADDDATE('2020-01-01', INTERVAL 31 DAY)`	`SUBDATE('2020-01-01', INTERVAL 31 DAY)`
1	2020-02-01	2019-12-01

날짜 및 시간 함수

- ADDTIME, SUBTIME
  - ADDTIME은 날짜/시간을 기준으로 시간을 더한 결과를 반환합니다. 위에서는 '2020-01-01 22:20:00)이 반환됩니다.
  - SUBTIME은 날짜/시간을 기준으로 시간을 뺀 결과를 반환합니다. 위에서는 '2020-01-01 20:20:00)이 반환됩니다.

예	**SELECT ADDTIME('2020-01-01 21:20:00', '1:0:0'), -- ADDTIME(날짜/시간, 시간) SUBTIME('2020-01-01 21:20:00', '1:0:0'); -- SUBTIME(날짜/시간, 시간)**

	`ADDTIME('2020-01-01 21:20:00', '1:0:0')`	`SUBTIME('2020-01-01 21:20:00', '1:0:0')`
1	2020-01-01 22:20:00	2020-01-01 20:20:00

날짜 및 시간 함수

- YEAR, MONTH, DAYOFMONTH, HOUR, MINUTE, SECOND, MICROSECOND
  - YEAR, MONTH, DAYOFMONTH, HOUR, MINUTE, SECOND, MICROSECOND는 특정 날짜나 시간에 대한 연, 월, 일, 시, 분, 초, 밀리초를 반환합니다.

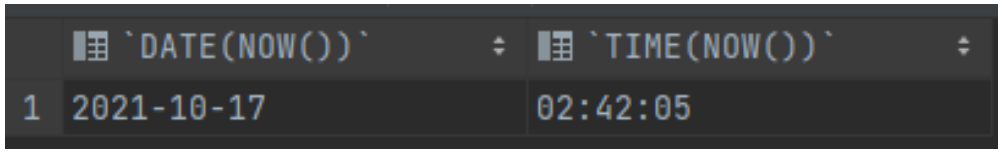
예	**SELECT YEAR(NOW()), MONTH(NOW()), DAYOFMONTH(NOW()), HOUR(NOW()), MINUTE(NOW()), SECOND(NOW()), MICROSECOND(NOW()));**

	`YEAR(NOW())`	`MONTH(NOW())`	`DAYOFMONTH(NOW())`	`HOUR(NOW())`	`MINUTE(NOW())`	`SECOND(NOW())`	`MICROSECOND(NOW())`
1	2021	10	17	2	40	49	0

날짜 및 시간 함수

- DATE, TIME
  - DATE는 '연-월-일', TIME은 '시:분:초'를 반환합니다.

예	**SELECT DATE(NOW()), TIME(NOW());**



	`DATE(NOW())`	`TIME(NOW())`
1	2021-10-17	02:42:05

날짜 및 시간 함수

- DATEDIFF, TIMEDIFF
  - DATEDIFF는 날짜2에서 날짜1까지 몇 일 남았는지를 반환합니다. 위에서는 5가 반환됩니다.
  - TIMEDIFF는 시간이 얼마나 남았는지를 반환합니다. 위에서는 08:00:000이 반환됩니다.

예	**SELECT DATEDIFF('2020-1-5', '2020-1-1'), -- DATEDIFF(날짜1, 날짜2) TIMEDIFF('14:30:00', '06:30:00'); -- TIMEDIFF(날짜1 OR 시간1, 날짜2 OR 시간2)**

	`DATEDIFF('2020-1-5', '2020-1-1')`	`TIMEDIFF('14:30:00', '06:30:00')`
1	4	08:00:00

날짜 및 시간 함수

- DAYOFWEEK, MONTHNAME, DAYOFYEAR
  - DAYOFWEEK는 요일을(월:2 화:3), MONTHNAME은 해당 월의 영어이름, DAYOFYEAR는 1년 중 몇일이 지났는지를 반환합니다.

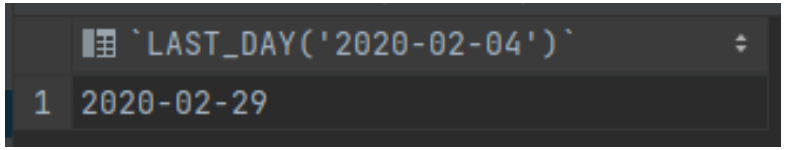
예	**SELECT DAYOFWEEK(NOW()), MONTHNAME(NOW()), DAYOFYEAR(NOW());**

	`DAYOFWEEK(NOW())`	`MONTHNAME(NOW())`	`DAYOFYEAR(NOW())`
1	1	October	290

날짜 및 시간 함수

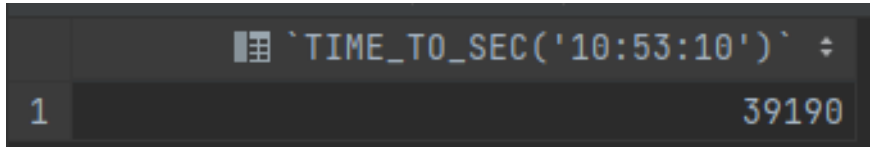
- LAST_DAY
  - LAST_DAY는 주어진 월의 마지막날을 반환합니다. 위에서는 '2020-02-29'가 반환됩니다.

예	**SELECT LAST_DAY('2020-02-04');**



- TIME_TO_SEC
  - TIME_TO_SEC은 시간을 초 단위로 구합니다. 위에서는 '39190'이 출력됩니다.

예	**SELECT TIME_TO_SEC('10:53:10');**



형 변환 함수

- 형 변환 함수

구 분	설 명
CAST(expr AS type) **CONVERT(expr, type)**	지정한 값을 다른 데이터 타입으로 변환하고 싶을 때 사용한다. type에 들어가는 타입의 종류는 아래와 같다. - BINARY[(N)] - CHAR[(N)] [charset_info] - DATE - DATETIME - DECIMAL[(M[,D])] - JSON - NCHAR[(N)] - SIGNED [INTEGER] - TIME - UNSIGNED [INTEGER]

형 변환 함수

- 형 변환 함수

예	SELECT CAST(NOW() AS SIGNED), CONVERT(NOW(), SIGNED), CAST(20200101 AS DATE), CONVERT(20200101, DATE);

	CAST(NOW() AS SIGNED)	CONVERT(NOW(), SIGNED)	CAST(20200101 AS DATE)	CONVERT(20200101, DATE)
1	20211017192202	20211017192202	2020-01-01	2020-01-01

제어 흐름 함수

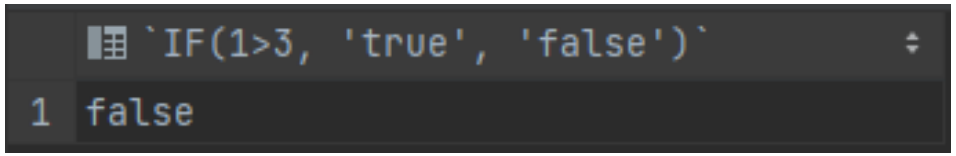
- 제어 흐름 함수

구 분	설 명
IF(수식, 참, 거짓)	진리값에 따른 값이 반환
IFNULL(수식1,수식2)	수식1이 NULL이면 수식2를 반환하고, 그렇지 않으면 수식1을 리턴
NULLIF(수식1, 수식2)	수식1과 수식2가 같다면 NULL을, 아니면 수식1을 반환
CASE ~ WHEN ~ ELSE~ END	다중 분기에 사용

제어 흐름 함수

- IF
  - IF는 수식과 참일 때 반환할 값과 거짓일 때 반환할 값을 입력하면, 수식의 진리 값에 따른 값이 반환됩니다. 위에서는 false가 반환됩니다.

예	**SELECT IF(1>3, 'true', 'false');**

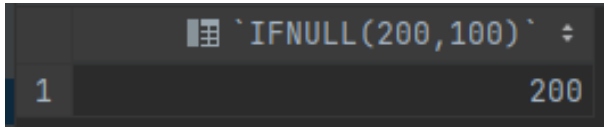


	`IF(1>3, 'true', 'false')`
1	false

제어 흐름 함수

- IFNULL
  - IFNULL은 수식1이 NULL이면 수식2를 반환하고, 그렇지 않으면 수식1을 리턴합니다. 위에서는 200이 반환됩니다.

예	**SELECT IFNULL(200,100)**

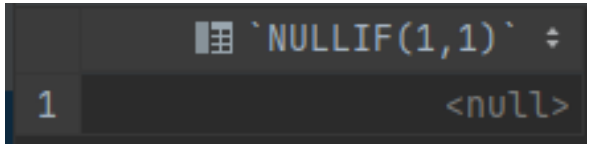


	`IFNULL(200,100)`
1	200

제어 흐름 함수

- NULLIF
  - NULLIF는 수식1과 수식2가 같다면 NULL을, 아니면 수식1을 반환합니다. 위에서는 NULL이 반환됩니다.

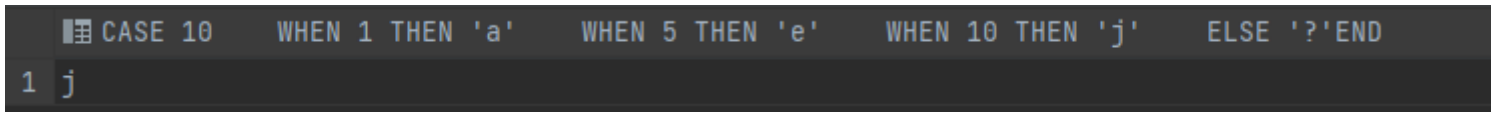
예	**SELECT NULLIF(1,1);**



제어 흐름 함수

- CASE
  - CASE는 내장함수는 아니고 다중 분기에 사용하는 Operator(연산자)입니다. switch문과 유사하며 위에서는 'j'가 반환됩니다. WHEN에서 찾을 수 없다면 ELSE의 값이 반환됩니다.

예	**SELECT CASE 10** **WHEN 1 THEN 'a'** **WHEN 5 THEN 'e'** **WHEN 10 THEN 'j'** **ELSE '?'** **END;**



제어 흐름 함수

- CASE
  - CASE 문의 또 다른 예

예

```
SELECT ENAME, DEPTNO,
            CASE WHEN DEPTNO=10 THEN 'ACCOUNTING'
                 WHEN DEPTNO=20 THEN 'RESEARCH'
                 WHEN DEPTNO=30 THEN 'SALES'
                 WHEN DEPTNO=40 THEN 'OPERATIONS'
             END AS DNAME
FROM EMP;
```

	ENAME	DEPTNO	DNAME
1	SMITH	20	RESEARCH
2	ALLEN	30	SALES
3	WARD	30	SALES
4	JONES	20	RESEARCH
5	MARTIN	30	SALES
6	BLAKE	30	SALES
7	CLARK	10	ACCOUNTING
8	SCOTT	20	RESEARCH
9	KING	10	ACCOUNTING
10	TURNER	30	SALES
11	ADAMS	20	RESEARCH
12	JAMES	30	SALES
13	FORD	20	RESEARCH
14	MILLER	10	ACCOUNTING

시스템 함수

- 시스템 함수

구 분	설 명
USER(), DATABASE()	현재 사용자 및 현재 선택된 데이터 베이스를 구한다.
FOUND_ROWS()	바로 앞의 SELECT 문에서 조회된 행의 개수를 구한다.
ROW_COUNT()	바로 앞의 INSERT,UPDATE,DELETE 문에서 입력, 수정, 삭제된 행의 개수를 구한다.
VERSION()	현재 MYSQL의 버전을 구한다