

```

1 REM Author :
2 REM Date :
3 REM Objective : Chapter 4. Single-Row Function
4 REM Environment : Ubuntu Server 20.04 LTS, HeidiSQL 10.2.0, MySQL Community Server 5.7.34.0
5
6 REM SQL function
7 -A function is a stored program that you can pass parameters into and then return a value.
8 1. Built Function(내장함수)
9 2. Stored Function(사용자 정의 함수)
10
11 REM 단일행 함수(Single Row function)
12 1. Syntax
13 function_name(column | expression [ arg1, arg2...])
14
15 2. 종류
16 1) 제어흐름 함수
17 2) 숫자 함수
18 3) 날짜시간 함수
19 4) 문자열 함수
20 5) 집합 함수
21 6) 변환 함수
22 7) 기타 함수
23
24
25 REM 제어 흐름 함수(Flow Control Functions)
26 1. IF()
27 1) Definition
28 -Returns a value if a condition is TRUE, or another value if a condition is FALSE.
29
30 2) Syntax
31 IF(expr1, expr2, expr3)
32
33 3) 만일 expr1이 참이면, expr2를 리턴한다.
34 4) 그렇지 않으면 expr3을 리턴한다.
35
36 SELECT IF(1 > 2, 2, 3); --> 3
37 SELECT IF(1 < 2, 'yes', 'no') --> 'yes'
38
39
40 2. CASE
41 1) Definition
42 -Goes through conditions and return a value when the first condition is met.
43 -like an IF-THEN-ELSE statement.
44 -So, once a condition is true, it will stop reading and return the result.
45 -If no conditions are true, it will return the value in the ELSE clause.
46 -If there is no ELSE part and no conditions are true, it returns NULL.
47
48 2) Syntax
49 CASE
50 WHEN compare_value1 THEN result1
51 WHEN compare_value2 THEN result2
52 WHEN compare_value3 THEN result3
53 ...
54 ELSE resultN
55 END
56
57 SELECT job, sal,
58 CASE WHEN job = 'ANALYST' THEN sal * 1.1
59 WHEN job = 'CLERK' THEN sal * 1.15
60 WHEN job = 'MANAGER' THEN sal * 1.2
61 ELSE sal
62 END AS "SALARY"
63 FROM emp;
64
65
66 3. IFNULL()
67 1) Definition
68 -Returns a specified value if the expression is NULL.
69 -If the expression is NOT NULL, this function returns the expression.
70
71 2) Syntax
72 IFNULL(expr1, expr2)
73 -If expr1 is not NULL, IFNULL() returns expr1; otherwise it returns expr2.
74 -expr1 : NULL

```

```

75     -expr2 : 치환값
76     -expr1값이 NULL 아니면 expr1 값을 그대로 사용
77     -만약 expr1 값이 NULL이면, expr2 값으로 대체
78
79
80 4. NULLIF()
81 1)Definition
82     -Compares two expressions and returns NULL if they are equal. Otherwise, the first expression is returned.
83
84 2)Syntax
85     NULLIF(expr1, expr2)
86
87     SELECT NULLIF(1,1); --> NULL
88     SELECT NULLIF(1,2); --> 1
89     SELECT NULLIF("Hello", "world"); --> 'Hello'
90
91
92
93 REM 숫자 함수(Numeric Functions)
94
95 1. ABS
96 1) 숫자 값을 절대값으로 바꾼다.
97 2)Syntax
98     ABS(expression)
99
100    SELECT ABS(-15)
101
102
103 2. CEIL(CEILING)
104 1>Returns the smallest integer value that is bigger than or equal to a number.
105 2)Syntax
106     CEIL(number)
107
108    SELECT CEIL(15.7)
109
110
111 3. DEGREES
112 1)Convert radians to degrees
113 2)Syntax
114     DEGREES(number)
115
116    SELECT DEGREES(PI()*2); --> 360
117    SELECT DEGREES(PI()); --> 180
118    SELECT DEGREES(PI() / 2); --> 90
119
120
121 4. FLOOR
122 1>Returns the largest integer value that is smaller than or equal to a number.
123 2)Syntax
124     FLOOR(number)
125
126    SELECT FLOOR(15.7)
127
128
129 5. MOD
130 1>Returns the remainder of a number divided by another number.
131 2)Syntax
132     MOD(m, n)
133     -m MOD n
134     -m % n
135
136    SELECT ename, sal, comm, MOD(sal, comm)
137    FROM emp
138    WHERE job = 'SALESMAN';
139
140    SELECT 10 / 3, MOD(10, 3);
141    SELECT sal, MOD(sal, 30);
142
143
144 6. PI
145    SELECT PI();
146
147
148 7. POW(POWER)

```

149 1)Returns the value of a number raised to the power of another number.

150

151 SELECT POWER(3,2)

152

153

154 8. RADIANS

155 1)Converts a degree value into radians.

156 2)Syntax

157 RADIANS(number)

158

159 SELECT RADIANS(-45); --> -0.7853981633974483

160 SELECT RADIANS(90); --> 1.5707963267949

161

162

163 9. RAND

164 1)Returns a random number between 0 (inclusive) and 1 (exclusive).

165 2)Syntax

166 RAND(seed)

167

168 SELECT RAND(); --> 0.26097273012713784

169

170

171 10. ROUND

172 1)Rounds a number to a specified number of decimal places.

173 2)Syntax

174 ROUND(column | expression, n)

175 3) 열, 표현식, 값을 소수점 n째 자리로 반올림

176 4) n을 지정하지 않은 경우 소수점 이하 값이 없어짐

177 5) n이 음수이면 소수점 왼쪽 수가 반올림

178

179 SELECT ROUND(45.925, 2), ROUND(45.925, 0), ROUND(45.925, -1);

180 SELECT ROUND(-1.23);

181 SELECT ROUND(-1.58);

182 SELECT ROUND(1.298, 1);

183 SELECT ROUND(1.298, 0);

184

185

186 11. SIGN

187 1) 주어진 수가 양수이면 1, 0이면 0, 음수이면 -1

188

189 SELECT SIGN(-12);

190

191

192 12. SQRT

193 1)Returns the square root of a number.

194

195 SELECT SQRT(13);

196

197

198 13. TRUNCATE

199 1)Truncates a number to the specified number of decimal places.

200 2)열, 표현식, 값을 소수점 n째 자리까지 남기고 버린다.

201 3)Syntax

202 TRUNC (column | expression, n)

203

204 SELECT TRUNCATE(345.156, 0); --> 345

205 SELECT TRUNCATE(1.223,1);

206 SELECT TRUNCATE(1.999,1);

207 SELECT TRUNCATE(122, -2);

208

209

210

211 REM 날짜 함수

212

213 1. 날짜데이터

214 1)MySQL은 표준 출력 형식으로 주어진 날짜 또는 시간 유형에 대한 값을 검색하지만 사용자가 제공하는 입력 값에 대한 다양한 형식을 해석하려고 시도한다.

215 2)다른 형식의 값을 사용하면 예측할 수 없는 결과가 발생할 수 있다.

216 3)MySQL은 여러 형식으로 값을 해석하려고 시도하지만 날짜 부분은 항상 월-일-년 또는 일-월-보다는 년-월-일 순서(예: '98-09-04')로 지정해야 한다.

217 4)다른 곳에서 일반적으로 사용되는 연도 순서(예: '09-04-98', '04-09-98'), 다른 순서의 문자열을 년-월-일 순서로 변환하려면 STR_TO_DATE() 함수가 유용할 수 있다.

218 5)2자리 연도 값을 포함하는 날짜는 세기를 알 수 없기 때문에 모호하다.

219 6)MySQL은 다음 규칙을 사용하여 2자리 연도 값을 해석한다.

-Year values in the range 70-99 become 1970-1999.
-Year values in the range 00-69 become 2000-2069.

2. ADDDATE

1) Adds a time/date interval to a date and then returns the date.

2) Syntax

ADDDATE(date, INTERVAL value addunit)

OR

ADDDATE(date, days)

SELECT ADDDATE("2017-06-15 09:34:21", INTERVAL 15 MINUTE); --> 2017-06-15 09:49:21

SELECT ADDDATE("2017-06-15 09:34:21", INTERVAL -3 HOUR); --> 2017-06-15 06:34:21

SELECT ADDDATE("2017-06-15", INTERVAL -2 MONTH); --> 2017-04-15

SELECT DATE_ADD('2008-01-02', INTERVAL 31 DAY); --> '2008-02-02'

SELECT ADDDATE('2008-01-02', INTERVAL 31 DAY); --> '2008-02-02'

SELECT ADDDATE('2008-01-02', 31); --> '2008-02-02'

3. ADDTIME

1) Adds a time interval to a time/datetime and then returns the time/datetime.

2) Syntax

ADDTIME(datetime, addtime)

--Add 5 seconds and 3 microseconds to a time and return the datetime:

SELECT ADDTIME("2017-06-15 09:34:21.000001", "5.000003"); --> 2017-06-15 09:34:26.000004

--Add 2 hours, 10 minutes, 5 seconds, and 3 microseconds to a time and return the datetime:

SELECT ADDTIME("2017-06-15 09:34:21.000001", "2:10:5.000003"); --> 2017-06-15 11:44:26.000004

--Add 5 days, 2 hours, 10 minutes, 5 seconds, and 3 microseconds to a time and return the datetime:

SELECT ADDTIME("2017-06-15 09:34:21.000001", "5 2:10:5.000003"); --> 2017-06-20 11:44:26.000004

--Add 2 hours, 10 minutes, 5 seconds, and 3 microseconds to a time and return the time:

SELECT ADDTIME("09:34:21.000001", "2:10:5.000003"); --> 11:44:26.000004

4. CURDATE

1) Returns the current date.

2) The date is returned as "YYYY-MM-DD" (string) or as YYYYMMDD (numeric).

3) This function equals the **CURRENT_DATE()** function.

4) Syntax

CURDATE()

SELECT CURDATE() + 1; --> 20210831

SELECT CURDATE(); --> '2021-08-30'

SELECT CURDATE() + 0; --> 20210830

5. CURRENT_DATE

1) Returns the current date.

2) Syntax

CURRENT_DATE()

SELECT CURRENT_DATE() + 1; --> 20210831

6. CURRENT_TIME

1) Returns the current time.

2) The time is returned as "HH-MM-SS" (string) or as HHMMSS.uuuuuu (numeric).

3) This function equals the **CURTIME()** function.

4) Syntax

CURRENT_TIME()

SELECT CURRENT_TIME() + 1; --> 224909

SELECT CURTIME(); --> --> '22:49:58'

SELECT CURTIME() + 0; --> 224958.000000

7. CURRENT_TIMESTAMP

1) Returns the current date and time.

2) The date and time is returned as "YYYY-MM-DD HH-MM-SS" (string) or as YYYYMMDDHHMMSS.uuuuuu (numeric).

```

293 SELECT CURRENT_TIMESTAMP(); --> '2021-08-30 22:52:13'
294 SELECT CURRENT_TIMESTAMP() + 1 --> 20210830225329
295
296
297 8. DATE
298 1)Extracts the date part from a datetime expression.
299 2)Syntax
300     DATE(expression)
301
302     SELECT DATE("2017-06-15 09:34:21"); --> '2017-06-15'
303
304
305 9. DATEDIFF
306 1>Returns the number of days between two date values.
307 2)Syntax
308     DATEDIFF(date1, date2)
309
310     SELECT DATEDIFF("2017-06-25 09:34:21", "2017-06-15 15:25:35"); --> 10
311     SELECT DATEDIFF("2017-01-01", "2016-12-24"); --> 8
312
313
314 10. DATE_FORMAT
315 1)Formats a date as specified.
316 2)Syntax
317     DATE_FORMAT(date, format)
318
319     SELECT DATE_FORMAT("2017-06-15", "%M %d %Y"); --> June 15 2017
320     SELECT DATE_FORMAT("2017-06-15", "%W %M %e %Y"); --> Thursday June 15 2017
321
322
323 11. DAY
324 1>Returns the day of the month for a given date (a number from 1 to 31).
325 2)This function equals the DAYOFMONTH() function.
326 3)Syntax
327     DAY(date)
328
329     SELECT DAY("2017-06-15 09:34:21"); --> 15
330     SELECT DAY(CURDATE()); --> 30
331
332
333 12. DAYNAME
334 1>Returns the weekday name for a given date.
335 2)Syntax
336     DAYNAME(date)
337
338     SELECT DAYNAME("2017-06-15 09:34:21"); --> Thursday
339     SELECT DAYNAME(CURDATE()); --> Monday
340
341
342 13. LAST_DAY
343 1)Extracts the last day of the month for a given date.
344 2)Syntax
345     LAST_DAY(date)
346
347     SELECT LAST_DAY("2017-02-10 09:34:00"); --> 2017-02-28
348
349
350 14. MAKEDATE
351 1)Creates and returns a date based on a year and a number of days value.
352 2)Syntax
353     MAKEDATE(year, day)
354
355     SELECT MAKEDATE(2017, 175); --> 2017-06-24
356
357
358 15. MAKETIME
359 1)Creates and returns a time based on an hour, minute, and second value.
360 2)Syntax
361     MAKETIME(hour, minute, second)
362
363     SELECT MAKETIME(16, 1, 0); --> 16:01:00
364
365
366 16. NOW

```

```

367 1)Returns the current date and time.
368
369 SELECT NOW();
370
371
372 17. PERIOD_ADD
373 1)Adds a specified number of months to a period.
374 2)Return the result formatted as YYYYMM.
375 3)Syntax
376     PERIOD_ADD(period, number)
377
378 SELECT PERIOD_ADD(201703, 15); --> 201806
379
380
381 18. PERIOD_DIFF
382 1)Returns the difference between two periods. The result will be in months.
383 2)Syntax
384     PERIOD_DIFF(period1, period2)
385
386 SELECT PERIOD_DIFF(201703, 201803); --> -12
387 SELECT PERIOD_DIFF(1703, 1612); --> 3
388
389
390 19. QUARTER
391 1)Returns the quarter of the year for a given date value (a number from 1 to 4).
392 2)Syntax
393     QUARTER(date)
394
395 SELECT QUARTER("2017-01-01 09:34:21"); --> 1
396
397
398 20. STR_TO_DATE
399 1)Returns a date based on a string and a format.
400
401 SELECT STR_TO_DATE('01,5,2013','%d,%m,%Y'); --> '2013-05-01'
402 SELECT STR_TO_DATE('May 1, 2013','%M %d,%Y'); --> '2013-05-01'
403
404
405
406 REM 문자 함수
407 1. ASCII, CHAR
408 1)Returns the ASCII value for the specific character.
409 2)Returns the String value for the specific ASCII code.
410 3)Syntax
411     ASCII(str)
412     CHAR(number)
413
414 SELECT ASCII('2'); --> 50
415 SELECT CHAR(77,121,83,81,'76'); --> 'MySQL'
416
417
418 2. BIT_LENGTH
419 1)Returns the length of the string str in bits.
420 2)Syntax
421     BIT_LENGTH(str)
422
423 SELECT BIT_LENGTH('hello'); --> 40
424 SELECT BIT_LENGTH('안녕'); --> 48
425
426
427 3. CHAR_LENGTH
428 1)Returns the length of the string str, measured in characters.
429 2)Syntax
430     CHAR_LENGTH(str)
431
432 SELECT CHAR_LENGTH("SQL Tutorial"); --> 12
433 SELECT CHAR_LENGTH("안녕"); --> 2
434
435
436 4. LENGTH
437 1)Returns the length of a string (in bytes).
438 2)Syntax
439     LENGTH(str)
440

```

```
441 SELECT LENGTH("SQL Tutorial"); --> 12
442 SELECT CHAR_LENGTH("안녕"); --> 6
443
444
445
```

5. FORMAT

1) The **FORMAT()** function formats a number to a format like "#,###,###.##", rounded to a specified number of decimal places, then it returns the result as a string.

2) Syntax

FORMAT(number, decimal_places)

```
450
451 SELECT FORMAT(250500.5634, 0); --> '250,501'
452 SELECT FORMAT(12332.123456, 4); --> '12,332.1235'
453 SELECT FORMAT(12332.1,4); --> '12.332.1000'
454 SELECT FORMAT(12332.2,0); --> '12,332'
455 SELECT FORMAT(12332.2,2,'de_DE'); --> '12.332,20'
456 -If no locale is specified, the default is 'en_US'
457
458
```

6. LOWER

1) 소문자로 변환

2) Syntax

LOWER(column | expression)

```
463
464 SELECT empno, ename
465 FROM emp
466 WHERE LOWER(ename) = 'scott';
467
468
```

7. UPPER

1) 대문자로 변환

2) Syntax

UPPER (column | expression)

```
473
474 SELECT empno, ename, deptno
475 FROM emp
476 WHERE ename = 'blake';
477
478 SELECT empno, ename, deptno
479 FROM emp
480 WHERE ename = UPPER('blake');
481
482
```

8. CONCAT

1) Adds two or more expressions together.

2) Syntax

CONCAT(expression1, expression2, expression3,...)

```
487
488 SELECT CONCAT("SQL ", "Tutorial ", "is ", "fun!")
489
490
```

9. SUBSTR[ING]

1) Extracts a substring from a string (starting at any position).

2) Syntax

SUBSTR(string, start, length)

```
495
496 SELECT SUBSTRING('Quadratically',5); --> 'ratically'
497 SELECT SUBSTRING('foobarbar' FROM 4); --> 'barbar'
498 SELECT SUBSTRING('Quadratically',5,6); --> 'ratica'
499 SELECT SUBSTRING('Sakila', -3); --> 'ila'
500 SELECT SUBSTRING('Sakila', -5, 3); --> 'aki'
501
502
```

10. INSTR

1) Returns the position of the first occurrence of substring substr in string str.

2) Syntax

INSTR(str,substr)

```
507
508 SELECT INSTR('foobarbar', 'bar'); --> 4
509 SELECT INSTR('xbar', 'foobar'); --> 0
510
511
```

11. LPAD | RPAD

1) Left-pads a string with another string, to a certain length.

```

514 2)Syntax
515 LPAD(string, length, lpad_string)
516
517 SELECT LPAD("SQL Tutorial", 20, "ABC"); --> ABCABCABSQ SQL Tutorial
518
519
520
521 12. LTRIM | RTRIM
522 1)Removes leading spaces from a string.
523 2)Syntax
524 LTRIM(string)
525
526 SELECT LTRIM(" SQL Tutorial"); --> SQL Tutorial
527
528
529
530 13. REPLACE
531 1)Replaces all occurrences of a substring within a string, with a new substring.
532 2)Syntax
533 REPLACE(string, substring, new_string)
534
535 SELECT REPLACE("SQL Tutorial", "SQL", "HTML"); --> HTML Tutorial
536
537
538 14. REPEAT
539 1)Repeats a string as many times as specified.
540 2)Syntax
541 REPEAT(string, number)
542
543
544 SELECT REPEAT("SQL Tutorial", 3); --> SQL TutorialSQL TutorialSQL Tutorial
545
546
547 15. REVERSE
548 1)Reverses a string and returns the result.
549 2)Syntax
550 REVERSE(string)
551
552 SELECT REVERSE("SQL Tutorial"); --> lairoTu LQS
553
554
555 16. SPACE
556 1>Returns a string of the specified number of space characters.
557 2)Syntax
558 SPACE(number)
559
560 SELECT SPACE(6); --> '      '
561
562
563
564 REM 변환함수
565 1. CAST
566 1)Converts a value (of any type) into the specified datatype.
567 2)Syntax
568 CAST(value AS datatype)
569
570 SELECT CAST(150 AS CHAR); --> '150'
571 SELECT CAST("14:06:10" AS TIME); --> 14:06:10
572
573
574 2. CONVERT
575 1)Converts a value into the specified datatype or character set.
576 2)Syntax
577 CONVERT(value, type)
578 OR
579 CONVERT(value USING charset)
580
581 SELECT CONVERT(150, CHAR); --> '150'

```