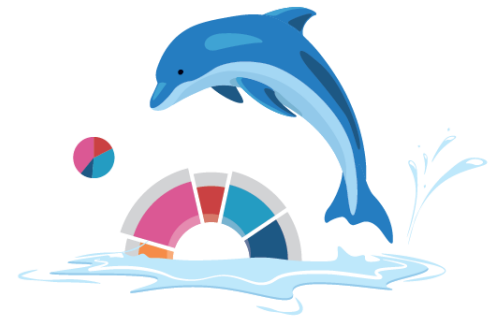


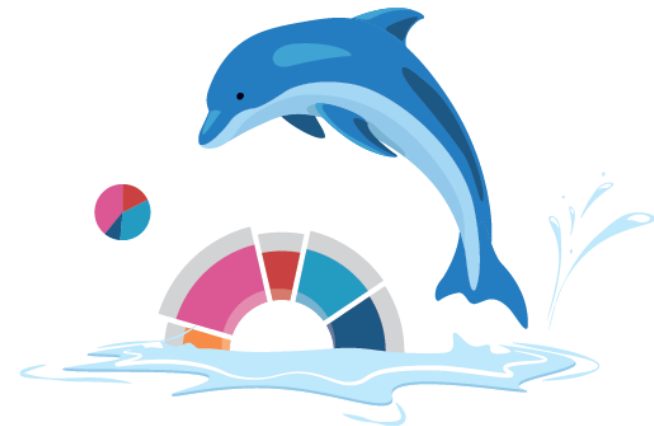
난생처음

MySQL



Chapter 03

함수1: 단일 행 함수



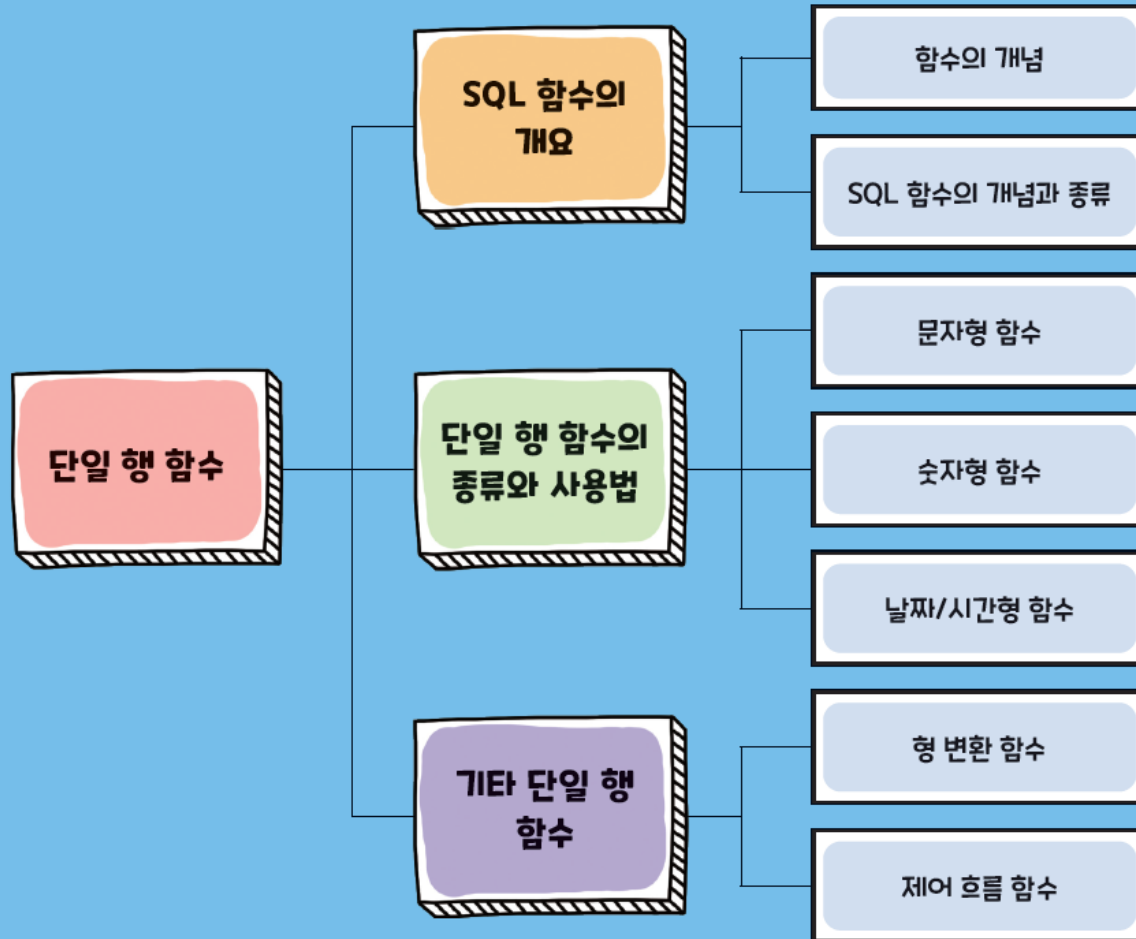
목차

1. SQL 함수의 개요
2. 단일 행 함수의 종류와 사용법
3. 기타 단일 행 함수

학습목표

- 함수의 개념과 종류를 이해할 수 있습니다.
- 단일 행 함수인 문자형 함수, 숫자형 함수, 날짜/시간형 함수를 활용할 수 있습니다.
- 기타 유용한 함수에 대해 이해할 수 있습니다.

Preview



Section 01

SQL 함수의 개요

1. 함수의 개념

■ 함수

- 특별한 목적의 작업을 수행하기 위해 독립적으로 설계된 프로그램 코드의 집합
- 사용자는 어떤 로직에 의해서 함수가 동작하는지에 대해 알 필요 없이 필요한 값을 주고, 원하는 값을 얻으면 됨
- 함수를 실행할 때 필요한 값은 매개변수를 통해 전달됨
- [예] 문자열의 길이를 반환하는 LENGTH 함수
 - ✓ 함수 사용에 필요한 매개변수 값으로 문자열을 입력함

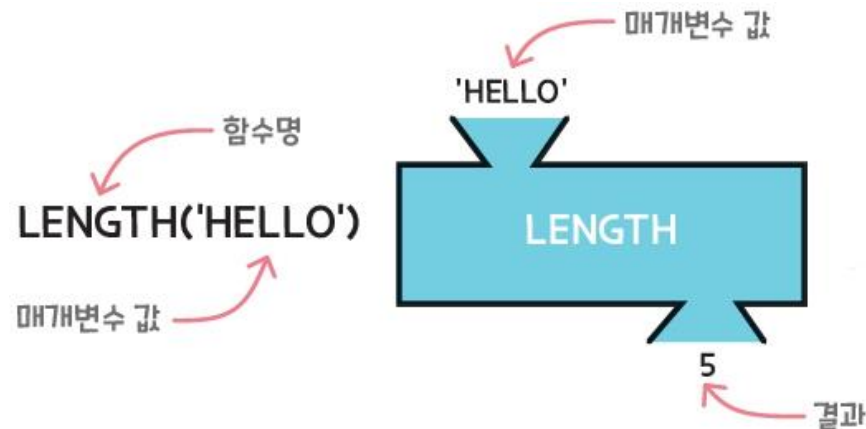


그림 3-1 함수의 사용 방법

2. SQL 함수의 개념과 종류

■ SQL 함수

- 데이터를 조회하거나 집계, 저장, 수정하는 과정에서 값을 가공하기 위해 제공되는 모듈화된 기능
- SQL 함수는 DBMS에 미리 만들어져 저장되어 있는지, 사용자가 필요에 따라서 직접 만드는지에 따라 내장 함수와 사용자 정의 함수로 구분함

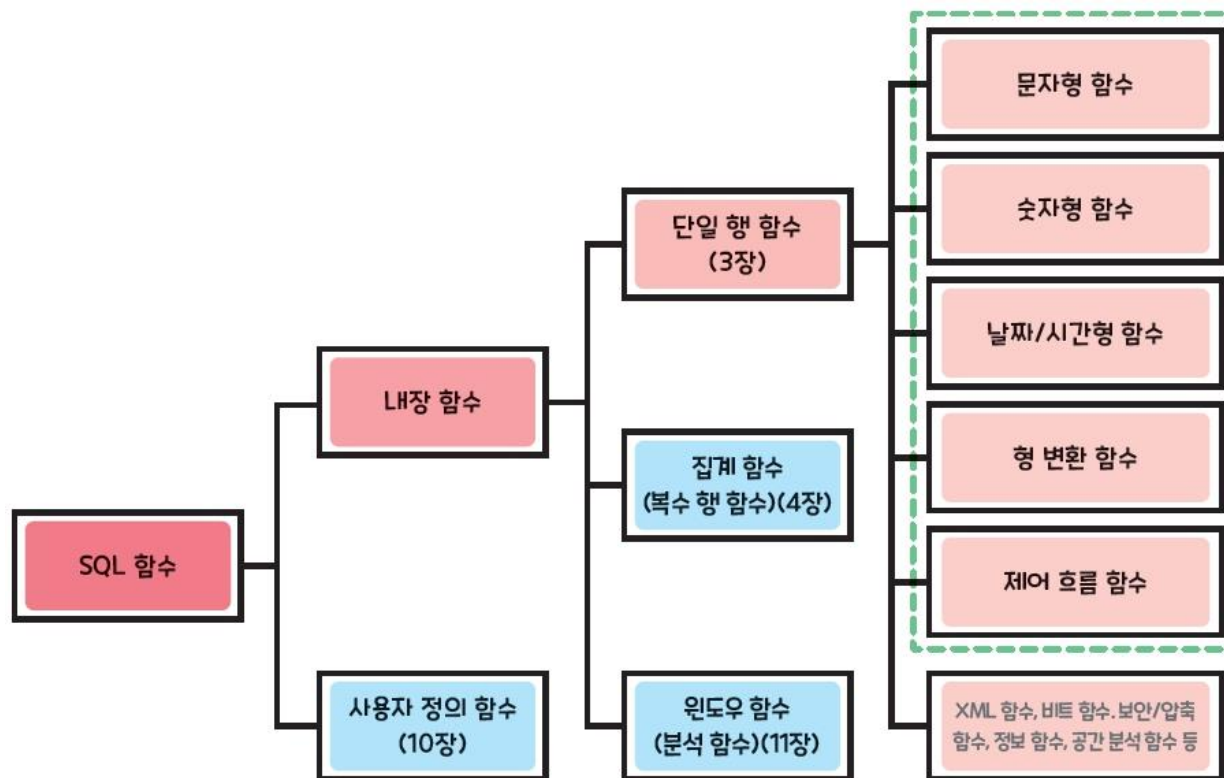


그림 3-2 SQL 함수의 종류

2. SQL 함수의 개념과 종류

확인문제

다음 빈칸에 들어갈 단어를 채우시오.

SQL 함수는 DBMS에 미리 저장되어 있는지, 사용자가 필요에 따라서 직접 만드는지에 따라 ① 와 ② 로 구분됩니다. ①에는 단일 행 함수, 집계 함수, 윈도우 함수 등이 있습니다.

① _____ ② _____

정답

① 내장 함수(Built-in Function) ② 사용자 정의 함수(User-defined Function)

Section 02

단일 행 함수의 종류와 사용법

1. 문자형 함수

■ 문자형 함수

- 문자열을 입력하면 문자열이나 숫자 값을 반환하는 함수

■ CHAR_LENGTH(), LENGTH()

- CHAR_LENGTH() : 문자의 개수를 반환하는 함수
- LENGTH() : 문자열에 할당된 바이트(Byte) 수를 반환하는 함수
- 형식

CHAR_LENGTH(문자열)

LENGTH(문자열)

1. 문자형 함수

- [예제 3-1] 영문 'HELLO'와 한글 '안녕'의 문자 개수를 세고 결과를 확인하시오.

```
SELECT CHAR_LENGTH('HELLO') ●———— 영문자 개수 반환  
      ,LENGTH('HELLO') ●———— 영문자의 바이트 수 반환  
      ,CHAR_LENGTH('안녕') ●———— 문자 개수 반환  
      ,LENGTH('안녕'); ●———— UTF-8의 바이트 수 반환
```

▶ 실행결과

CHAR_LENGTH('HELLO')	LENGTH('HELLO')	CHAR_LENGTH('안녕')	LENGTH('안녕')
5	5	2	6

1. 문자형 함수

■ CONCAT(), CONCAT_WS()

- CONCAT() : 문자열을 연결할 때 사용하는 함수
- CONCAT_WS() : 구분자와 함께 문자열을 연결할 때 사용하는 함수
- 형식

CONCAT(문자열1, 문자열2, ...)

CONCAT_WS(구분자, 문자열1, 문자열2, ...)

- ## ■ [예제 3-2] CONCAT() 함수와 CONCAT_WS() 함수를 사용하여 문자열을 연결했을 때의 결과값을 비교하시오.

```
SELECT CONCAT('DREAMS', 'COME', 'TRUE')  
       ,CONCAT_WS('-', '2023', '01', '29');
```

문자열 연결
구분자 '-'로 연결

▶ 실행결과

CONCAT('DREAMS', 'COME', 'TRUE')	CONCAT_WS('-', '2023', '01', '29')
DREAMSCOMETRUE	2023-01-29

1. 문자형 함수

■ LEFT(), RIGHT(), SUBSTR()

- LEFT() : 문자열의 왼쪽부터 길이만큼 문자열을 반환하는 함수
- RIGHT() : 문자열의 오른쪽부터 길이만큼 문자열을 반환하는 함수
- SUBSTR() : 지정한 위치로부터 길이만큼의 문자열을 반환하는 함수
- 형식

LEFT(문자열, 길이)

RIGHT(문자열, 길이)

SUBSTR(문자열, 시작_위치, 길이) 또는 SUBSTRING(문자열, 시작_위치, 길이)

1. 문자형 함수

- [예제 3-3] 'SQL 완전정복' 문자열에서 지정한 길이만큼 문자열을 반환하시오.

```
SELECT LEFT('SQL 완전정복', 3) ●————— 왼쪽부터 3문자  
      ,RIGHT('SQL 완전정복', 4) ●————— 오른쪽부터 4문자  
      ,SUBSTR('SQL 완전정복', 2, 5) ●————— 2번째 문자부터 5문자  
      ,SUBSTR('SQL 완전정복', 2); ●————— 2번째 문자부터 끝까지
```

▶ 실행결과

LEFT('SQL 완전정복', 3)	RIGHT('SQL 완전정복', 4)	SUBSTR('SQL 완전정복', 2, 5)	SUBSTR('SQL 완전정복', 2)
SQL	완전정복	QL 완전	QL 완전정복

1. 문자형 함수

■ SUBSTRING_INDEX()

- SUBSTRING_INDEX() : 지정한 구분자를 기준으로 문자열을 분리해서 가져올 때 사용하는 함수
- 형식

```
SUBSTRING_INDEX(문자열, 구분자, 인덱스)
```

- [예제 3-4] '서울시 동작구 흑석로'에서 앞/뒤 2번째 위치에 있는 공백(' ')을 기준으로 문자열을 분리한 결과를 확인하시오.

```
SELECT SUBSTRING_INDEX('서울시 동작구 흑석로', ' ', 2) ● 왼쪽부터 두 번째 공백 이후 제거  
       ,SUBSTRING_INDEX('서울시 동작구 흑석로', ' ', -2); ● 오른쪽부터 두 번째 공백 이전 제거
```

▶ 실행결과

SUBSTRING_INDEX('서울시 동작구 흑석로', ' ', 2)	SUBSTRING_INDEX('서울시 동작구 흑석로', ' ', -2)
서울시 동작구	동작구 흑석로

1. 문자형 함수

■ LPAD(), RPAD()

- LPAD()와 RPAD() : 지정한 길이에서 문자열을 제외한 빈칸을 특정 문자로 채울 때 사용하는 함수
- LPAD()는 왼쪽에, RPAD()는 오른쪽에 특정 문자를 채움
- 형식

LPAD(문자열, 길이, 채울_문자열)

RPAD(문자열, 길이, 채울_문자열)

- [예제 3-5] 'SQL' 문자열의 앞 7칸에 # 기호를 채우고, SQL 뒤 2칸에는 * 기호를 채우시오.

```
SELECT LPAD('SQL', 10, '#') ●———— 문자열 왼쪽에 # 채우기  
       ,RPAD('SQL', 5, '*'); ●———— 문자열 오른쪽에 * 채우기
```

▶ 실행결과

LPAD('SQL', 10, '#')	RPAD('SQL', 5, '*')
#####SQL	SQL**

1. 문자형 함수

■ LTRIM(), RTRIM()

- LTRIM() : 왼쪽의 공백을 제거할 때 사용하는 함수
- RTRIM() : 오른쪽 공백을 제거할 때 사용하는 함수
- 형식

LTRIM(문자열)

RTRIM(문자열)

- [예제 3-6] ' SQL ' 문자열의 앞/뒤에 있는 공백을 제거하고, 결과를 문자열의 길이로 확인하시오.

```
SELECT LENGTH(LTRIM(' SQL ')) ● ————— 왼쪽 공백을 지운 후 길이 반환  
      ,LENGTH(RTRIM(' SQL ')) ● ————— 오른쪽 공백을 지운 후 길이 반환  
      ,LENGTH(TRIM(' SQL ')); ● ————— 양쪽 공백을 지운 후 길이 반환
```

▶ 실행결과

LENGTH(LTRIM(' SQL '))	LENGTH(RTRIM(' SQL '))	LENGTH(TRIM(' SQL '))
4	4	3

1. 문자형 함수

■ TRIM(BOTH/LEADING/TRAILING)

- 양쪽에 있는 동일 문자열을 제거하고자 할 때에는 TRIM()의 첫 번째 매개변수에 BOTH를 넣음
- BOTH 대신 LEADING을 넣으면 왼쪽에 있는 문자열이, TRAILING을 넣으면 오른쪽에 있는 문자열이 제거됨
- 형식

TRIM(문자열)

TRIM(제거할_문자열의_방향(BOTH/LEADING/TRAILING) 제거할_문자열 FROM 문자열)

- [예제 3-7] 'abcSQLabcbabc'에서 BOTH/LEADING/TRAILING을 사용하여 앞뒤에 있는 'abc' 문자열을 제거하시오.

```
SELECT TRIM(BOTH 'abc' FROM 'abcSQLabcbabc') ● 양쪽의 모든 abc 제거  
      ,TRIM(LEADING 'abc' FROM 'abcSQLabcbabc') ● 왼쪽의 abc 제거  
      ,TRIM(TRAILING 'abc' FROM 'abcSQLabcbabc'); ● 오른쪽의 abc 제거
```

▶ 실행결과

TRIM(BOTH '*' FROM '**SQL**')	TRIM(LEADING 'abc' FROM 'abcSQLabc')	TRIM(TRAILING 'abc' FROM 'abcSQLabcbabc')
SQL	SQLabc	abcSQL

1. 문자형 함수

■ FIELD(), FIND_IN_SET(), INSTR(), LOCATE()

- FIELD() : 여러 문자열 중에서 찾는 문자열이 있으면 문자열의 위치 값을 반환하는 함수. 만약 찾는 문자열이 없으면 0을 반환함

- 형식

```
FIELD(찾을_문자열, 문자열1, 문자열2, ...)
```

- FIND_IN_SET() : 문자열 리스트에서 지정한 문자열을 찾아서 위치 값을 반환함

- 형식

```
FIND_IN_SET(찾을_문자열, 문자열_리스트)
```

- INSTR(), LOCATE() : 기준 문자열 중에서 부분 문자열을 찾아서 위치 값을 반환함

- 형식

```
INSTR(기준_문자열, 부분_문자열)
```

```
LOCATE(부분_문자열, 기준_문자열)
```

1. 문자형 함수

- [예제 3-8] 문자열의 위치를 찾는 4가지 함수 FIELD(), FIND_IN_SET(), INSTR(), LOCATE()를 사용하고 결과를 확인하시오.

```
SELECT FIELD('JAVA', 'SQL', 'JAVA', 'C')  
      ,FIND_IN_SET('JAVA', 'SQL,JAVA,C')  
      ,INSTR('네 인생을 살아라', '인생')  
      ,LOCATE('인생', '네 인생을 살아라');
```

첫 매개변수인 JAVA의 위치
'.'로 구분된 두 번째 매개변수 값 중 JAVA의 위치
인생의 위치

▶ 실행결과

FIELD('JAVA', 'SQL', 'JAVA', 'C')	FIND_IN_SET('JAVA', 'SQL,JAVA,C')	INSTR('네 인생을 살아라', '인생')	LOCATE('인생', '네 인생을 살아라')
2	2	3	3

1. 문자형 함수

■ ELT()

- ELT() : 지정한 위치에 있는 문자열을 반환하는 함수
- FIELD()는 첫 매개변수 값에 찾고자 하는 문자열을 넣어주지만, ELT()는 찾을 문자열의 위치 값을 넣음
- 형식

```
ELT(찾을_문자열_위치, 문자열1, 문자열2, ...)
```

- [예제 3-9] 문자열의 위치를 찾는 4가지 함수 FIELD(), FIND_IN_SET(), INSTR(), LOCATE()를 사용하고 결과를 확인하시오.

```
SELECT ELT(2, 'SQL', 'JAVA', 'C');
```

● ————— 두 번째 위치에 있는 문자열

▶ 실행결과

ELT(2, 'SQL', 'JAVA', 'C')
JAVA

1. 문자형 함수

■ REPEAT()

- REPEAT() : 문자열을 반복하고자 할 때 사용하는 함수
 - ✓ 매개변수에 반복할 문자열과 반복할 횟수를 넣음
- 형식

```
REPEAT(문자열, 횟수)
```

■ [예제 3-10] REPEAT()를 사용하여 '*' 5개를 반복하시오.

```
SELECT REPEAT('*', 5);
```

▶ 실행결과

```
REPEAT('*', 5)
```

```
*****
```

1. 문자형 함수

■ REPLACE()

- REPLACE() : 문자열의 일부를 다른 문자열로 대체하고자 할 때 사용하는 함수
- 형식

```
REPLACE(문자열, 원래_문자열, 바꿀_문자열)
```

■ [예제 3-11] '.'로 구분되어 있는 전화번호의 구분자를 '-'로 대체하시오.

```
SELECT REPLACE('010.1234.5678', '.', '-');
```

▶ 실행결과

REPLACE('010.1234.5678', '.', '-') 010-1234-5678

1. 문자형 함수

■ REVERSE()

- REVERSE() : 문자열을 거꾸로 뒤집을 때 사용하는 함수
- 형식

```
REVERSE(문자열)
```

■ [예제 3-12] 'OLLEH'를 맨 뒤쪽의 문자부터 나타내시오.

```
SELECT REVERSE('OLLEH');
```

▶ 실행결과

REVERSE('OLLEH')
HELLO

2. 숫자형 함수

■ 숫자형 함수

- 숫자를 다루는 단일 행 함수를 통해 올림, 버림, 반올림, 절댓값 반환, 나머지, 제곱승 등의 기능을 수행할 수 있음

■ CEILING(), FLOOR(), ROUND(), TRUNCATE()

- CEILING() : 올림, FLOOR() : 버림, ROUND() : 지정한 위치에서 반올림, TRUNCATE() : 지정한 위치에서 버림 작업을 수행하는 함수
- 형식

CEILING(숫자)

FLOOR(숫자)

ROUND(숫자)

ROUND(숫자, 반올림할_자릿수)

TRUNCATE(숫자, 버림할_자릿수)

2. 숫자형 함수

- [예제 3-13] 123.56에 대해 CEILING(), FLOOR(), ROUND(), TRUNCATE()를 사용한 결과를 확인하시오.

```
SELECT CEILING(123.56)
       ,FLOOR(123.56)
       ,ROUND(123.56)
       ,ROUND(123.56, 1)
       ,TRUNCATE(123.56, 1);
```

▶ 실행결과

CEILING(123.56)	FLOOR(123.56)	ROUND(123.56)	ROUND(123.56, 1)	TRUNCATE(123.56, 1)
124	123	124	123.6	123.5

2. 숫자형 함수

■ ABS(), SIGN()

- ABS() : 절댓값을 반환하는 함수
- SIGN() : 양수의 경우 1, 음수의 경우 -1을 반환하는 함수
- 형식

ABS(숫자)

SIGN(숫자)

■ [예제 3-14] -120과 120의 절댓값과 음수, 양수 여부를 판별하시오.

```
SELECT ABS(-120)  
      ,ABS(120)  
      ,SIGN(-120)  
      ,SIGN(120);
```

▶ 실행결과

ABS(-120)	ABS(120)	SIGN(-120)	SIGN(120)
120	120	-1	1

2. 숫자형 함수

■ MOD()

- MOD() : 나머지를 구하는 함수로 세 가지 방법을 사용할 수 있음
- 형식

MOD(숫자1, 숫자2)

숫자1 % 숫자2

숫자1 MOD 숫자2

■ [예제 3-15] 203을 4로 나눈 나머지를 확인하시오

```
SELECT MOD(203, 4)
      ,203 % 4
      ,203 MOD 4;
```

▶ 실행결과

MOD(203, 4)	203 % 4	203 MOD 4
3	3	3

2. 숫자형 함수

■ POWER(), SQRT(), RAND()

- POWER() : n제곱승 값을 반환하는 함수
- SQRT() : 제곱근 값을 반환하는 함수
- RAND() : 0과 1사이 임의의 실수 값을 반환하는 함수
 - ✓ 매개변수 값을 넣지 않으면 실행할 때마다 0과 1 사이에 있는 임의의 실수를 반환함
 - ✓ RAND() 안에 시드(Seed)를 설정하면 매번 동일한 임의의 값을 얻을 수 있음
 - ✓ ROUND(), TRUNCATE() 등의 함수를 같이 사용하면 임의의 정수를 만들 수 있음
- 형식

POWER(숫자1, 숫자2) ● ————— 숫자1의 숫자2 제곱승

SQRT(숫자)

RAND()

RAND(숫자)

2. 숫자형 함수

- [예제 3-16] 제곱과 제곱근, 랜덤값을 구하는 함수를 사용하고, 그 결과를 보여시오.

```
SELECT POWER(2, 3)
      ,SQRT(16)
      ,RAND()
      ,RAND(100)
      ,ROUND(RAND() * 100);
```

0~100 사이의 정수

▶ 실행결과

POWER(2, 3)	SQRT(16)	RAND()	RAND(100)	ROUND(RAND() * 100)
8	4	0.838829533046884	0.17353134804734155	76

3. 날짜/시간형 함수

■ 현재 날짜/시간 반환 함수

- NOW(), SYSDATE(), CURDATE(), CURTIME()
 - ✓ 현재 날짜와 시간을 반환하는 함수
- NOW(), SYSDATE() : 시스템의 현재 날짜와 시간을 반환함
- CURDATE() : 시스템의 현재 날짜를 반환함
- CURTIME() : 시스템의 현재 시간을 반환함

■ [예제 3-17] 현재 날짜와 시간을 다양한 형태로 반환하시오.

```
SELECT NOW()  
       ,SYSDATE()  
       ,CURDATE()  
       ,CURTIME();
```

▶ 실행결과

NOW()	SYSDATE()	CURDATE()	CURTIME()
2023-01-31 14:31:07	2023-01-31 14:31:07	2023-01-31	14:31:07

3. 날짜/시간형 함수

■ 연도, 분기, 월, 일, 시, 분, 초 반환 함수

- YEAR(), QUARTER(), MONTH(), DAY(), HOUR(), MINUTE(), SECOND()
 - ✓ 날짜 중 해당 값을 반환함
 - ✓ [예] YEAR()는 날짜 중 연도를, QUARTER()는 날짜 중 분기를 반환함

■ [예제 3-18] 날짜에 관한 다양한 함수의 결과를 확인하시오.

```
SELECT NOW()           ● ————— 현재 날짜
      ,YEAR(NOW())      ● ————— 연도
      ,QUARTER(NOW())   ● ————— 분기
      ,MONTH(NOW())     ● ————— 월
      ,DAY(NOW())       ● ————— 일
      ,HOUR(NOW())      ● ————— 시
      ,MINUTE(NOW())    ● ————— 분
      ,SECOND(NOW());   ● ————— 초
```

▶ 실행결과

NOW()	YEAR(NOW())	QUARTER(NOW())	MONTH(NOW())	DAY(NOW())	HOUR(NOW())	MINUTE(NOW())	SECOND(NOW())
2023-01-29 20:23:32	2023	1	1	29	20	23	32

3. 날짜/시간형 함수

■ 기간 반환 함수

- DATEDIFF(), TIMESTAMPDIFF()
 - ✓ 지정한 기간을 반환하는 함수
- DATEDIFF() : 기간을 일자 기준으로 반환함
- TIMESTAMPDIFF() : 기간을 지정한 단위 기준으로 보여줌
 - ✓ DATEDIFF()는 끝 일자를 앞에 넣어주고, TIMESTAMPDIFF()는 끝 일자를 뒤에 넣음
- 형식

DATEDIFF(끝_일자, 시작_일자)

TIMESTAMPDIFF(단위, 시작_일자, 끝_일자)

표 3-1 TIMESTAMPDIFF() 함수에 매개변수로 들어가는 단위

단위	의미	단위	의미
SECOND	초	WEEK	주
MINUTE	분	MONTH	월
HOURL	시	QUARTER	분기
DAY	일	YEAR	연도

3. 날짜/시간형 함수

- [예제 3-19] 현재 날짜를 기준으로 날짜 사이의 기간을 확인하시오.

```
SELECT NOW()  
      ,DATEDIFF('2025-12-20', NOW()) ●————— 1  
      ,DATEDIFF(NOW(), '2025-12-20')  
      ,TIMESTAMPDIFF(YEAR, NOW(), '2025-12-20')  
      ,TIMESTAMPDIFF(MONTH, NOW(), '2025-12-20')  
      ,TIMESTAMPDIFF(DAY, NOW(), '2025-12-20'); ●————— 2
```

▶ 실행결과

NOW()	DATEDIFF('2025-12-20', NOW())	DATEDIFF(NOW(), '2025-12-20')	TIMESTAMPDIFF(YEAR, NOW(), '2025-12-20')	TIMESTAMPDIFF(MONTH, NOW(), '2025-12-20')	TIMESTAMPDIFF(DAY, NOW(), '2025-12-20')
2023-01-31 14:35:11	1054	-1054	2	34	1053

3. 날짜/시간형 함수

■ 기간을 반영하는 날짜 함수

- ADDDATE() : 지정한 날짜를 기준으로 그 기간만큼 더한 날짜를 반환하는 함수
- SUBDATE() : 기간만큼 뺀 날짜를 반환함
- 형식

ADDDATE(날짜, 기간)
또는 ADDDATE(날짜, INTERVAL 기간 단위)
SUBDATE(날짜, 기간)
또는 SUBDATE(날짜, INTERVAL 기간 단위)

■ [예제 3-20] 오늘 날짜 및 오늘 날짜로부터 50일 후, 50개월 후, 50시간 전의 날짜를 확인하시오.

```
SELECT NOW()  
      ,ADDDATE(NOW(), 50) ●————— ① 50일 후  
      ,ADDDATE(NOW(), INTERVAL 50 DAY) ●————— ② 50일 후  
      ,ADDDATE(NOW(), INTERVAL 50 MONTH) ●————— 50개월 후  
      ,SUBDATE(NOW(), INTERVAL 50 HOUR); ●————— 50시간 전
```

▶ 실행결과

NOW()	ADDDATE(NOW(), 50)	ADDDATE(NOW(), INTERVAL 50 DAY)	ADDDATE(NOW(), INTERVAL 50 MONTH)	SUBDATE(NOW(), INTERVAL 50 HOUR)
2023-01-31 14:59:41	2023-03-22 14:59:41	2023-03-22 14:59:41	2027-03-31 14:59:41	2023-01-29 12:59:41

3. 날짜/시간형 함수

■ 기타 날짜 반환 함수

- LAST_DAY() : 해당 월의 마지막 일자를 반환함
- DAYOFYEAR() : 현재 연도에서 며칠이지났는지를 반환함
- MONTHNAME()은 월을 영문으로, WEEKDAY()는 요일을 정수로 보여줌
- 형식

LAST_DAY(날짜)

DAYOFYEAR(날짜)

MONTHNAME(날짜)

WEEKDAY(날짜)

3. 날짜/시간형 함수

- [예제 3-21] 오늘 날짜를 기준으로 이번 달 마지막 날짜와 일 년 중 몇 일째인지, 그리고 월 이름과 요일을 확인하시오.

```
SELECT NOW()  
      , LAST_DAY(NOW())  
      , DAYOFYEAR(NOW())  
      , MONTHNAME(NOW())  
      , WEEKDAY(NOW());
```

▶ 실행결과

NOW()	LAST_DAY(NOW())	DAYOFYEAR(NOW())	MONTHNAME(NOW())	WEEKDAY(NOW())
2023-01-31 15:09:37	2023-01-31	31	January	1

3. 날짜/시간형 함수

확인문제

다음 문제를 해결하려면 어떤 함수를 써야 할까요?

1. 오늘은 내가 태어난 지 몇 일째일까? ()
2. 내 생일부터 10,000일이 지난 날짜는 언제일까? ()
3. 나는 무슨 요일에 태어났을까? ()

정답

1. DATEDIFF() 또는 TIMESTAMPDIFF()
2. ADDDATE()
3. WEEKDAY()

Section 03

기타 단일 행 함수

1. 형 변환 함수

■ 형 변환 함수

- 데이터를 검색, 삽입할 때 컬럼에 맞는 형식으로 지정하지 않으면 오류가 나는 경우에 사용하는 함수

■ CAST(), CONVERT()

- 원하는 형태로 데이터타입을 변경하여 처리하거나 확인할 수 있음
- 형식

CAST(값 AS 데이터타입)

CONVERT(값, 데이터타입)

1. 형 변환 함수

- [예제 3-22] 문자 '1'을 부호 없는 숫자 형식으로, 숫자 2를 문자 형식으로 변환하시오.

```
SELECT CAST('1' AS UNSIGNED)  
      ,CAST(2 AS CHAR(1))  
      ,CONVERT('1', UNSIGNED)  
      ,CONVERT(2, CHAR(1));
```

▶ 실행결과

CAST('1' AS UNSIGNED)	CAST(2 AS CHAR(1))	CONVERT('1', UNSIGNED)	CONVERT(2, CHAR(1))
1	2	1	2

2. 제어 흐름 함수

■ 제어 흐름 함수

- 프로그램의 흐름을 제어할 때 사용함
- 조건에 따른 결과를 SQL 문장 하나로 얻을 수 있음

■ IF

- IF(조건, 수식1, 수식2) : 조건의 결과가 참이면 수식1을 반환하고, 그렇지 않으면 수식2의 결과를 반환함
- 형식

IF(조건, 수식1, 수식2)

- [예제 3-23] 12,500원짜리 제품을 450개 이상 주문한 금액이 5,000,000원 이상이면 '초과달성', 미만이면 '미달성'이라고 보이시오.

```
SELECT IF(12500 * 450 > 5000000, '초과달성', '미달성');
```

▶ 실행결과

IF(12500 * 450 > 5000000, '초과달성', '미달성')
초과달성

2. 제어 흐름 함수

■ IFNULL(), NULLIF()

- NULL과 관련된 제어 흐름 함수
- IFNULL() : 수식1이 NULL이 아니면 수식1의 값을 반환하고, NULL이면 수식2의 값을 반환함
- NULLIF() : 두 수식 값을 비교하여 값이 같으면 NULL을 반환하고, 값이 다르면 수식1의 값을 반환함
- 형식

IFNULL(수식1, 수식2)

NULLIF(수식1, 수식2)

2. 제어 흐름 함수

- [예제 3-24] IFNULL()의 첫 매개변수 값이 NULL인지 여부에 따라 어떻게 결과가 다르게 나오는지 확인하시오.

```
SELECT IFNULL(1, 0)
      ,IFNULL(NULL, 0)
      ,IFNULL(1/0, 'OK');
```

▶ 실행결과

IFNULL(1, 0)	IFNULL(NULL, 0)	IFNULL(1/0, 'OK')
1	0	OK

- [예제 3-25] NULLIF()을 사용하여 두 결과를 비교하시오.

```
SELECT NULLIF(12 * 10, 120)
      ,NULLIF(12 * 10, 1200);
```

▶ 실행결과

NULLIF(12 * 10, 120)	NULLIF(12 * 10, 1200)
NULL	120

3. CASE문

■ CASE문

- 함수는 아니지만 조건 비교가 여러 개일 때 유용하게 사용할 수 있음
- WHEN 조건1 THEN 값1 WHEN 조건2 THEN 값2...
 - ✓ 모든 조건을 만족하지 않으면 ELSE 다음에 값을 넣어줌
 - ✓ CASE문은 END로 마무리되어야 함

- [예제 3-26] 주문 금액이 5,000,000원 이상이면 '초과달성', 4,000,000원 이상이면 '달성' 그 나머지는 '미달성'이라고 할 때, 12,500원짜리 제품을 450개 이상 주문했다면 어디에 해당하는지 보이시오.

```
SELECT CASE WHEN 12500 * 450 > 5000000 THEN '초과달성'
          WHEN 2500 * 450 > 4000000 THEN '달성'
          ELSE '미달성'
END;
```

▶ 실행결과

CASE WHEN 12500 * 450 > 5000000 THEN '초과달성' WHEN 2500 * 450 > 4000000 THEN '달성' ELSE '미달성' END
초과달성

3. CASE문

확인문제

다음 빈칸에 들어갈 함수명을 넣어주세요.

IFNULL()과 NULLIF()는 NULL과 관련된 제어 흐름 함수입니다. 첫 번째 매개변수가 NULL이 아니면 첫 번째 값을 반환하고, NULL이면 두 번째 값을 반환하는 함수는 ① 입니다. 두 매개변수를 비교하여 값이 같으면 NULL을 반환하고, 값이 다르면 첫 번째 수식 값을 반환하는 함수는 ② 입니다.

① _____ ② _____

정답

① IFNULL() ② NULLIF()

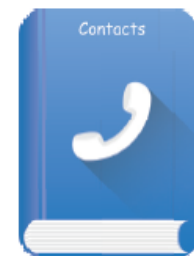
점검문제

문제 1

다음 조건에 따라 고객 테이블에서 고객회사명과 전화번호를 다른 형태로 보이도록 함수를 사용해봅시다. 고객회사명2와 전화번호2를 만드는 조건은 다음과 같습니다.

조건

- 고객회사명2 : 기존 고객회사명 중 앞의 두 자리를 *로 변환한다.
- 전화번호2 : 기존 전화번호의 (xxx)xxx-xxxx 형식을 xxx-xxx-xxxx 형식으로 변환한다.



▶ 실행결과

고객회사명	고객회사명2	전화번호	전화번호2
굿모닝서울	**닝서울	(02)978-1984	02-978-1984
얼케이 상사	**이 상사	(02)345-1945	02-345-1945
씨엔그룹	**그룹	(02)31-0345	02-31-0345

문제 2

다음 조건에 따라 주문세부 테이블의 모든 컬럼과 주문금액, 할인금액, 실제 주문금액을 보이시오. 이때 모든 금액은 1의 단위에서 버림을 하고 10원 단위까지 보이도록 합니다.



조건

- 주문금액 : 주문수량 * 단가
- 할인금액 : 주문수량 * 단가 * 할인율
- 실주문금액 : 주문금액 - 할인금액

▶ 실행결과

주문번호	제품번호	단가	주문수량	할인율	주문금액	할인금액	실주문금액
H0250	51	4200	35	0.15	147000	22050	124950
H0250	65	1700	15	0.15	25500	3820	21680
H0251	22	1700	6	0.05	10200	510	9690
H0251	57	1600	15	0.05	24000	1200	22800
	65						

문제 3

사원 테이블에서 전체 사원의 이름, 생일, 만나이, 입사일, 입사일수, 입사한 지 500일 후의 날짜를 보이시오.

▶ 실행결과

이름	생일	만나이	입사일	입사일수	500일후
이소미	1985-12-05	38	2019-04-13	1706	2020-08-25
배재용	1973-02-17	50	2019-01-01	1808	2020-05-15
유대현	1988-08-27	35	2019-03-14	1736	2020-07-26
최소민	1987-09-17	36	2019-04-15	1704	2020-08-27
	1980-03-01		2018-12-29	1811	2020-05-12



문제 4

고객 테이블에서 도시 컬럼의 데이터를 다음 조건에 따라 '대도시'와 '도시'로 구분하고, 마일리지 점수에 따라서 'VVIP', 'VIP', '일반 고객'으로 구분하시오.



조건

- 도시 구분 : '특별시'나 '광역시'는 '대도시'로, 그 나머지 도시는 '도시'로 구분한다.
- 마일리지 구분 : 마일리지 100,000점 이상이면 'VVIP고객', 10,000점 이상이면 'VIP고객', 그 나머지는 '일반고객'으로 구분한다.

▶ 실행결과

담당자명	고객회사명	도시	도시구분	마일리지	마일리지구분
이은광	굿모닝서울	서울특별시	대도시	15911	VIP고객
김병현	엘케이 상사	서울특별시	대도시	406	일반고객
김성민	씨엔그룹	광명시	도시	8788	일반고객
유리안	유리안	대도시	대도시	15911	VIP고객

문제 5

주문 테이블에서 주문번호, 고객번호, 주문일 및 주문년도, 분기, 월, 일, 요일, 한글요일을 보이시오.

▶ 실행결과

주문번호	고객번호	주문일	주문년도	주문분기	주문월	주문일	주문요일	한글요일
H0248	NETVI	2020-03-12	2020	1	3	12	3	목요일
H0249	MSPTO	2020-03-13	2020	1	3	13	4	금요일
H0250	NARHA	2020-03-16	2020	1	3	16	0	월요일
H0251	CTEVI	2020-03-16	2020	1	3	16	0	월요일
H0252	PRDSU	2020-03-17	2020	1	3	17	1	화요일

문제 6

주문 테이블에서 요청일보다 발송일이 7일 이상 늦은 주문내역을 보이시오.

▶ 실행결과

주문번호	고객번호	사원번호	주문일	요청일	발송일	지연일수
H0380	NGOHU	E07	2020-08-20	2020-09-17	2020-09-24	7
H0423	URLGO	E06	2020-10-01	2020-10-15	2020-11-02	18
H0427	CCOPI	E04	2020-10-05	2020-11-02	2020-11-09	7
H0451	ICKQU	E04	2020-10-28	2020-11-11	2020-11-18	7

Thank you!