6장 데이터 검색

1. 리터럴(Literal)
2. 수식
3. 스칼라 함수
4. 날짜 및 시간 처리
5. 데이터 형 변환 함수
6. 사용자 정의 변수
7. 시스템 변수

● SQL 명령문의 중요한 공통 요소

- 리터럴(Literal)
- 시스템 변수(System variable)
- 사용자 변수(User variable)
- 수치 수식(Numeric expression)
- 영숫자 수식(Alphanumeric expression)
- 데이터의 수식(Data expression)
- 스칼라 함수(Scalar function)
- 통계 함수(Statistical function)

6.1 리터럴(Literal)

- 정수 리터럴
- 십진 리터럴
- 부동 소수점 리터럴
- 영수치 리터럴
- 날짜 리터럴

6.1.1 정수 리터럴

-정수이거나 소수점이 없는 정수로써 양 의 부호(+), 음의 부호(-)를 가짐
-정수형 리터럴 : 38, +12, -3404, -16
-잘못된 정수형 리터럴 예 : 342.16, -14E5, jim, -123.45

6.1.2 십진 리터럴

-소수점을 가지고 있거나 가지지 않는 수, 양 또는 음의 부호를 사용
-십진 리터럴 예 : 49, 18.47, -3400, -16, 0.83459, -349
-소수점 앞에 있는 숫자의 수를 정밀도(precision)
-소수점 뒤에 있는 숫자의 수를 크기(scale)

SUCHI INT(5, 2) NOT NULL → 123.45, 12.3 값 입력 가능

Size Scale

6.1.3 부동 소수점 리터럴(floating point literal)

-지수를 가지고 있는 십진 리터럴
-부동 소수점 리터럴 예 : 49, 18.47, -34E2(-3400), 0.16E4(1600), 4E-3(0.004)

6.1.4 영수치 리터럴(alphanumeric literal)

-인용부호(')로 감싼 0 또는 그 이상의 영수치 문자로 구성된 문자열
-인용부호는 리터럴에 포함되지 않고 문자열의 시작과 끝을 나타냄
- 모든 영문자의 소문자 (a-z)
- 모든 영문자의 대문자 (A-Z)
- 모든 숫자 (0-9)
- 특수 기호 (+, =, ?, =, _)

-영수치 리터럴 예 : 'collins' (collins), '''tis' ('tis), '!?-' (!?-), '' (공백), '''' (‘), '1234' (1234)

6.1.5 날짜 리터럴(date literal)

-연도(year), 달(month), 일(day)로 구성하여 날짜를 표현
-날짜를 구성하는 3개의 요소는 슬래쉬('/')로 구분
-년도가 2자리 수치를 사용한다면 19가 두 자리 수치 앞에 생략

날짜 리터럴	한글 버전 입력 값	영문 버전 입력 값
1980/12/08	80/12/08	8 December 1980
1995/06/19	95/06/19	19 June 1995
99/1/1	99/01/01	1 January 1999
996/1/1	96/01/01	1 January 1996
2000/01/01	00/01/01	1 January 2000
1/1/1	01/01/01	1 January 0001

6.2 수식

- 하나 이상의 연산자와 필요할 경우 괄호를 사용하여 하나의 값을 표현
- 값은 영수치, 수치 또는 날짜 자료형 가짐
- SELECT 명령문의 SELECT와 WHERE 절에서 수식이 사용

6.2.1 수치 수식과 숫자 처리 함수

수치 수식은 정수, 십진수 또는 부동소수점수로 구성된 산술 수식
산술연산자(*, /, +, -)가 수식에 괄호와 함께 사용

- 수치 수식에서 열 명세가 NULL 값을 가진다면 전체 수식의 값은 NULL
- 수식의 계산 값은 다음과 같은 우선순위로 수행된다.
  (1) 왼쪽에서 오른쪽으로
  (2) 괄호
  (3) 곱셈과 나눗셈
  (4) 덧셈과 뺄셈
- 수치 수식에서 사용된 열 명세와 함수, 시스템 변수는 수치 자료형을 가짐
- 영수치 수식을 수치 수식으로 변환(변환은 자동적으로 수행된다).

[예제 6-1] 등록금 총액을 변경하라.

```
mysql> update fee
    -> set fee_total = ifnull(fee_enter, 0) + fee_price;
Query OK, 24 rows affected (0.08 sec)
Rows matched: 24  Changed: 24  Warnings: 0
```

등록금총액(fee_total)은 “입학금(fee_enter) + 수업료(fee_price)”가 된다.
입학금이 NULL인 경우에는 가산을 할 수 없으므로 ifnull(fee_enter, 0)와 같이 ifnull( ) 함수를 사용하여 NULL 값을 0으로 변환하여 가산

[예제 6-2] 납입금 총액은 "등록금 총액 - 장학금 총액"이다.
납입금 총액을 변경하라.

```
mysql> update fee
    -> set fee_pay = fee_total - ifnull(jang_total, 0);
Query OK, 24 rows affected (0.01 sec)
Rows matched: 24  Changed: 24  Warnings: 0
```

ifnull(jang_total, 0) 함수를 이용하여 장학금액이 NULL인 경우는 0으로 변환하여 사용

6.2.2 영수치 수식

영수치 수식은 영수치 값을 갖는다. CHAR(ACTER)은 영수치 자료형

영수치 수식	값
'Jim'	Jim
'Pete andJim'	Pete andJim
‘2019'	2019
TOWN	stratford
1234	1234

수치 수식은 자동적으로 영수치 수식으로 변환된다.

[예제 6-3] 우편번호가 “06034”이고 도로명이 “압구정로2길” 인 우편번호, 시도이름, 시군이름, 도로명, 건물번호본번, 건물명, 벙정동명을 출력하라.

```
mysql> select post_no, sido_name, sigun_name, road_name, building_bon, town_building,
row_dongname
    -> from post
    -> where post_no = "06034" and road_name = "압구정로2길";
+---------+---------------+-----------+---------------+--------------+---------------------+-------------+
| post_no | sido_name     | sigun_name | road_name     | building_bon | town_building       | row_dongname |
+---------+---------------+-----------+---------------+--------------+---------------------+-------------+
| 06034   | 서울특별시    | 강남구    | 압구정로2길   |           36 |                     | 신사동      |
| 06034   | 서울특별시    | 강남구    | 압구정로2길   |           46 | 강남상가아파트      | 신사동      |
| 06034   | 서울특별시    | 강남구    | 압구정로2길   |           45 |                     | 신사동      |
| 06034   | 서울특별시    | 강남구    | 압구정로2길   |           49 |                     | 신사동      |
| 06034   | 서울특별시    | 강남구    | 압구정로2길   |           35 |                     | 신사동      |
| 06034   | 서울특별시    | 강남구    | 압구정로2길   |           37 |                     | 신사동      |
+---------+---------------+-----------+---------------+--------------+---------------------+-------------+
6 rows in set (5.46 sec)
```

6.2.3 숫자 처리 함수

1) ROUND, TRUNCATE 함수

[형식] ROUND(column_name or value, n)
TRUNCATE(column_name or value, n)

- ROUND 함수 : 소수점 이하 자릿수에서 반올림(자릿수는 양수, 0, 음수 가능)
- 자릿수 n을 생략하면 소숫점이 5 이상일 때 반올림
- 자릿수 n을 지정하면 지정한 자리수에서 반올림
- 자릿수 n이 음수일 경우는 소수점 이하는 버리고 해당 자릿수에서 반올림하고 자릿수 만큼 0값으로 대치
- TRUNCATE함수는 숫자를 소수점 이하 자릿수에서 버림
- 소숫점이하 숫자가 자릿수 n보다 모자랄 경우 0값으로 대치
- 자릿수가 음수일 경우는 소수점 이하는 버리고 자릿수 만큼 0값으로 대치

[실습 따라하기]

```
mysql> select round(123456.789,2), truncate(123456.789,2)
+--------------------+-----------------------+
| round(123456.789,2) | truncate(123456.789,2) |
+--------------------+-----------------------+
|          123456.79 |             123456.78 |
+--------------------+-----------------------+

mysql> select round(123456.789,5), truncate(123456.789,5)
+--------------------+-----------------------+
| round(123456.789,5) | truncate(123456.789,5) |
+--------------------+-----------------------+
|       123456.78900 |          123456.78900 |
+--------------------+-----------------------+

mysql> select round(12345678.901,-3), truncate(12345678.901, -3);
+----------------------+--------------------------+
| round(12345678.901,-3) | truncate(12345678.901, -3) |
+----------------------+--------------------------+
|              12346000 |                 12345000 |
+----------------------+--------------------------+
```

2) FLOOR, CEIL 함수

[형식] FLOOR(column_name or value)
CEIL(column_name or value)

- FLOOR 함수는 소수점 아래의 수를 무조건 절삭하여 정수 값을 반환
- CEIL 함수는 소수점 아래의 수는 무시하고 무조건 올림을 하여 정수를 반환
- CEIL은 (0.5미만)이라도 인수보다 큰 다음 정수를 반환
- 소숫점이하 숫자가 자릿수 n보다 모자랄 경우 0값으로 대치
- 자릿수가 음수일 경우는 소수점 이하는 버리고 자릿수 만큼 0값으로 대치

[실습 따라하기]

```
mysql> select floor(123456.789), ceil(123456.123);
+--------------------+-------------------+
| floor(123456.789) | ceil(123456.123) |
+--------------------+-------------------+
|             123456 |            123457 |
+--------------------+-------------------+
```

3) MOD 함수

[형식] MOD(column_name1 or value1(분자), column_name2 or value2(분모))

- MOD 함수는 첫 번째 인수를 두 번째 인수로 나누어 나머지를 반환
- MOD 함수에서 두 번째 인수가 첫 번째 인수보다 크거나 0(zero)일 경우 결과 값은 첫 번째 인수를 반환
- 보통 수식계산은 0(zero)으로 나눌 경우 에러가 출력되지만,
- MOD 함수에서는 두 번째 인수가 0(zero)이라도 에러를 출력하지 않고 NULL로 출력된다.

[실습 따라하기]

```
mysql> select mod(10,3), mod(5,8), mod(7,0);
+-----------+----------+----------+
| mod(10,3) | mod(5,8) | mod(7,0) |
+-----------+----------+----------+
|         1 |        5 |     NULL |
+-----------+----------+----------+
1 row in set (0.00 sec)
```

4) ABS 함수

[형식] ABS(column_name or value)

- 절대값을 출력하는 함수

[실습 따라하기]

```
mysql> select abs(124), abs(-124);
+----------+-----------+
| abs(124) | abs(-124) |
+----------+-----------+
|      124 |       124 |
+----------+-----------+
```

5) POW함수 또는 POWER함수

[형식] POW(column_name or value, n)

- 제곱의 값을 구하는 함수이며 소숫점이 있는 경우에도 실행
- 단, 음수는 양수로 승처리된다.

[실습 따라하기]

```
mysql>  select pow(2,4), pow(-2.5,2), pow(1.5,6);
+---------+------------+-----------+
| pow(2,4) | pow(-2.5,2) | pow(1.5,6) |
+---------+------------+-----------+
|       16 |        6.25 |  11.390625 |
+---------+------------+-----------+
```

첫 번째는 2의 4승값을 구함
두 번째는 소수점을 포함한 음수지만 결과 값은 양수로 처리됨
세 번째는 소수점을 포함한 1.5의 6 승값을 구함

6) GREATEST, LEAST함수

[형식] GREAST(coulumn_name or values,coulumn_name or values,.....)
LEAST(coulumn_name or values,coulumn_name or values,.....)

- GREATEST함수는 주어진 숫자중 가장 큰 수를 반환
- LEAST는 반대로 가장 작은 수를 반환

[실습 따라하기]

```
mysql> select greatest(15,45,32,65), least(15,45,32,65);
+----------------------+-------------------+
| greatest(15,45,32,65) | least(15,45,32,65) |
+----------------------+-------------------+
|                   65 |                15 |
+----------------------+-------------------+
```

GREATEST는 4개의 숫자중 가장 큰 65를 반환
LEAST는 4개의 숫자중 가장 작은 15를 반환

[예제 6-4] 재학생들의 등록금액에 비례한 장학금의 비율은 몇 %인가?

```
mysql> select fee_total "등록금액", jang_total "장학금액",
    -> round(ifnull(jang_total,0)/fee_total*100,2) "비율"
    -> from fee;
```

등록금액	장학금액	비율
3500000	500000	14.29
3000000	2500000	83.33
3000000	2000000	66.67
3000000	800000	26.67
3000000	1000000	33.33
2500000	2500000	100.00
2800000	2500000	89.29
2800000	2500000	89.29
3000000	2500000	83.33
3000000	2500000	83.33
3000000	2000000	66.67
3000000	2500000	83.33
3500000	500000	14.29
3000000	2000000	66.67
3500000	500000	14.29
3000000	NULL	0.00
3500000	500000	14.29
3000000	NULL	0.00
3500000	500000	14.29
3000000	NULL	0.00
3500000	500000	14.29
3000000	NULL	0.00
3500000	500000	14.29
3000000	2500000	83.33

24 rows in set (0.01 sec)

6.3 스칼라 함수

스칼라 함수는 전달 인수를 사용하지 않거나 하나 이상의 전달 인수를 사용

[형식] CONCAT (column_name1 or string1, column_name2 or string2)
SUBSTRING (column_name or string, m, n)
LENGTH (column_name or string)
INSTR (column_name or string, character)
LPAD (column_name or string, m, character)
RPAD (column_name or string, m, character)
LOWER (column_name or string)
UPPER (column_name or string)
INITCAP (column_name or string)

-CONCAT 함수는 두 문자열을 연결시켜 합쳐준다.
-SUBSTRING 함수는 지정된 위치에서 지정된 길이만큼의 문자열을 추출한다.
-LENGTH 함수는 문자열의 길이를 정수값으로 반환한다.
-INSTR 함수는 문자열에서 특정 문자의 위치를 반환한다.
-LPAD 함수는 왼쪽에 지정된 문자를 지정된 길이만큼 채워준다.
-RPAD 함수는 오른쪽에 지정된 문자를 지정된 길이만큼 채워준다.
-LOWER 함수는 문자열을 모두 소문자로 바꾸어 준다.
-UPPER 함수는 문자열을 모두 대문자로 바꾸어준다.
-INITCAP 함수는 단어별로 첫 글자를 대문자로, 나머지 부분은 소문자로 변환

```
CONCAT('Data', 'Base') ---> DataBase
SUBSTRING('Korea', 1, 3) ---> Kor
LENGTH('lee_woo') ---> 7
INSTR('Korea', 'e') ---> 4
LPAD('Korea', 15, '*') ---> **********Korea
RPAD('Korea', 15, '#') ---> Korea##########
LOWER ('Korea') → korea
UPPER  ('Korea') → KOREA
INITCAP ('KOREA UNIVERSITY') → Korea University  //ORACLE명령어
```

[예제 6-5] 학번이 20191001인 학생의 학번, 이름, 영문이름을 출력하라. 단, 영문이름은 대문자로 출력하라.

```
mysql> select stu_no, stu_name, upper(stu_ename)
    -> from student
    -> where stu_no = '20191001';
+----------+-----------+-----------------+
| stu_no   | stu_name  | upper(stu_ename) |
+----------+-----------+-----------------+
| 20191001 | 김유신    | KIM YOO-SHIN     |
+----------+-----------+-----------------+
```

[예제 6-6] 2학년 학생의 번호와 이름, 영문이름 그리고 영문이름의 길이 출력

```
mysql> select stu_no, stu_name, stu_ename, length(rtrim(stu_ename))
    -> from student
    -> where grade = 2;
+----------+----------+-------------+-------------------------+
| stu_no   | stu_name | stu_ename   | length(rtrim(stu_ename)) |
+----------+----------+-------------+-------------------------+
| 20181001 | 장수인   | Jang Soo-In |                      11 |
| 20181002 | 정인정   | Jung In-Jung |                     12 |
| 20181003 | 이상진   | Lee Sang-Gin |                     12 |
+----------+----------+-------------+-------------------------+
```

[예제 6-7] 영문이름의 길이가 정확히 12자인 각 학생의 번호와 영문이름 출력

```
mysql> select stu_no, stu_ename
    -> from student
    -> where length(rtrim(stu_ename)) = 12;
+----------+--------------+
| stu_no   | stu_ename    |
+----------+--------------+
| 20141001 | Park Do-Sang |
| 20161001 | Park Jung-In |
| 20181002 | Jung In-Jung |
| 20181003 | Lee Sang-Gin |
| 20191001 | Kim Yoo-Shin |
| 20191004 | Lee Sun-Shin |
| 20201001 | Kim Young-Ho |
+----------+--------------+
```

[예제 6-8] 현주소의 우편번호가 “01”으로 시작하는 학생의 학번과 이름, 우편번호를 나타내어라.

```
mysql> select stu_no, stu_name, post_no
    -> from student
    -> where substring(post_no,1,2) = '01';
+----------+-----------+---------+
| stu_no   | stu_name  | post_no |
+----------+-----------+---------+
| 20141001 | 박도상    | 01066   |
| 20191004 | 이순신    | 01901   |
+----------+-----------+---------+
```

[예제 6-9] 학번이 20141001, 20191002인 학생의 학번, 이름, 우편번호, 주소를 출력하라.

```
mysql> select stu_no, stu_name, post_no, address
    -> from student
    -> where stu_no = '20141001' or  stu_no = '20191002';
+----------+-----------+---------+-----------------------------------------------------------+
| stu_no   | stu_name  | post_no | address                                                   |
+----------+-----------+---------+-----------------------------------------------------------+
| 20141001 | 박도상    | 01066   | 101동 203호                                               |
| 20191002 | 홍길동    | 59635   | 전라남도 여수시 시청로 금호아파트 104동 605호            |
+----------+-----------+---------+-----------------------------------------------------------+
2 rows in set (0.00 sec)
```

[예제 6-10] 학번이 '20141001'인 학생의 학번과 이름, 우편번호, 주소를 출력하라. 출력에는 rtrim( ) 함수를 이용하여 오른쪽 공백 부분을 삭제하여 출력하고 문자열을 연결시키는 CONCAT함수를 이용한다. (단, 도로명이 "덕릉로41길", 아파트 이름은 "다우빌라2차" 이다.)

```
mysql> select s.stu_no, s.stu_name, s.post_no,
    ->  concat(rtrim(p.sido_name), ' ', rtrim(p.sigun_name), ' ', p.road_name, ' ', p.town_building, ' ',
     rtrim(s.address)) "주소"
    -> from student s, post p
    -> where s.post_no = p.post_no
    -> and road_name = '덕릉로41길'
    -> and p.town_building = '다우빌라2차'
    -> and s.stu_no = '20141001';
+----------+----------+---------+-----------------------------------------------------------------------+
| stu_no   | stu_name | post_no | 주소                                                                  |
+----------+----------+---------+-----------------------------------------------------------------------+
| 20141001 | 박도상   | 01066   | 서울특별시 강북구 덕릉로41길 다우빌라2차 101동 203호                 |
+----------+----------+---------+-----------------------------------------------------------------------+
1 row in set (5.81 sec)
```

[예제 6-11] 학번이 '200141001' 인 학생의 주소를 "서울특별시 강북구 덕릉로41길 다우빌라2차 101동 203호 " 로 변경하라.

```
mysql> update student
    -> set address = '서울특별시 강북구 덕릉로41길 다우빌라2차 101동 203호'
    -> where stu_no = '20141001';
Query OK, 1 row affected (0.02 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

[예제 6-12] 학번 '200141001'인 학생의 학번, 이름, 우편번호, 주소를 출력하라.

```
mysql> select stu_no, stu_name, post_no, address
    -> from student
    -> where stu_no = '20141001';
+----------+-----------+---------+-----------------------------------------------------------------------+
| stu_no   | stu_name  | post_no | address                                                               |
+----------+-----------+---------+-----------------------------------------------------------------------+
| 20141001 | 박도상    | 01066   | 서울특별시 강북구 덕릉로41길 다우빌라2차 101동 203호                 |
+----------+-----------+---------+-----------------------------------------------------------------------+

1 row in set (0.00 sec)
```

6.4 날짜 및 시간 처리

-MySQL은 표준시간과 다양한 날짜 및 시간 관련 칼럼 타입과 함수를 지원

6.4.1 날짜 및 시간 관련 칼럼 타입

① DATE : 날짜 타입, '1000-01-01'에서 '9999-12-31'까지 표현 가능
② DATETIME : 날짜와 시간이 합쳐진 타입, '1000-01-01 00:00:00'에서 '9999-12-31 23:59:59'까지 표현 가능
③ TIMESTAMP[(M)] : 날짜 및 시간 타입, '1970-01-01 00:00:00'에서 2037년까지 나타낸다. [(M)]자리에는 출력될 길이를 나타내는 숫자를 쓸 수 있는데 14나 12나 8혹은 6을 사용 가능. 숫자를 쓰지 않으면 기본적으로 14자리로 나타낸다. TIMESTAMP의 특징은 자동 변경 칼럼 타입이라는 것이다. 이것은 INSERT나 UPDATE문을 사용할 때 매우 유용하다. 즉 데이터의 수정을 언제 했는지를 바로 확인할 수 있다.
④ TIME : 시간 타입, '-838:59:59'에서 '838:59:59'까지 사용 가능 기본적으로 지원하는 형태는 'HH:MM:SS'이다.
⑤ YEAR[(2/4)] : 연도를 나타내는 타입, 2자리 혹은 4자리로 나타낼 수 있으면 자리수를 지정하지 않으면 기본적으로 4자리로 나타낸다. 4자리로 사용할 때는 1901에서 2155년까지 지원하며 2자리로 사용할 때는 1970에서 2069년까지 지원한다.

6.4.2 날짜 및 시간 관련 함수

[실습 따라하기]

1) NOW() 또는 SYSDATE() : 현재 날짜와 시간을 반환한다.

```
mysql> select sysdate(), now();
+---------------------+---------------------+
| sysdate()           | now()               |
+---------------------+---------------------+
| 2019-02-11 10:50:46 | 2019-02-11 10:50:46 |
+---------------------+---------------------+
```

2) CURDATE() 또는 CURRENT_DATE() : 현재 날짜를 반환한다.

```
mysql> select curdate(), current_date();
+------------+--------------+
| curdate()  | current_date() |
+------------+--------------+
| 2019-02-11 | 2019-02-11     |
+------------+--------------+
1 row in set (0.01 sec)
```

3) CURTIME() 또는 CURRENT_TIME() : 현재 시간을 반환한다.

```
mysql> select curtime(), current_time();
+-----------+---------------+
| curtime() | current_time() |
+-----------+---------------+
| 10:24:54  | 10:24:54       |
+-----------+---------------+
```

4) DAYOFMONTH(date) : 몇일인지를 리턴 한다.

```
mysql> select now();
+---------------------+
| now()               |
+---------------------+
| 2019-02-11 10:53:31 |
+---------------------+

mysql> select dayofmonth(now());
+------------------+
| dayofmonth(now()) |
+------------------+
|                 2 |
+------------------+
```

5) DAYOFWEEK(date) / WEEKDAY(date) : 숫자로 요일을 리턴 한다.

```
mysql> select dayofmonth(now()), dayofweek(now());
+------------------+-----------------+
| dayofmonth(now()) | dayofweek(now()) |
+------------------+-----------------+
|                 2 |              11 |
+------------------+-----------------+
```

6) DAYOFYEAR(date) : 1년 중 며칠이 지났는가를 리턴 한다.

```
mysql> select dayofyear(now());
+-----------------+
| dayofyear(now()) |
+-----------------+
|              42 |
+-----------------+
```

7) DATE_ADD와 DATE_SUB

[형식] DATE_ADD(column_name or date, interval기본값)
DATE_SUB(column_name or date, interval기본값)

DATE_ADD함수는 날짜에서 기준값 만큼 더한 값, DATE_SUB함수는 날짜에서 기준값 만큼 뺀 값

오늘이 2019-02-11 이라면 3일후와 현재의 날짜에서 3일 전 을 나타냄

```
mysql> select date_add(now(), interval 3 day), date_sub(now(), interval 3
day);
+-----------------------------------+-----------------------------------+
| date_add(now(), interval 3 day) | date_sub(now(), interval 3 day) |
+-----------------------------------+-----------------------------------+
| 2019-02-14 10:55:29               | 2019-02-08 10:55:29               |
+-----------------------------------+-----------------------------------+
```

8) YEAR, MONTH

```
mysql> select year(now()), month(now());
+-------------+--------------+
| year(now()) | month(now()) |
+-------------+--------------+
|        2019 |            2 |
+-------------+--------------+
```

현재일자(오늘) 년도(4자리), 월(영문), 일(00이 포함된 날짜)을 출력하라.

```
mysql> select date_format(now(), '%Y %M %d');
+--------------------------------+
| date_format(now(), '%Y %M %d') |
+--------------------------------+
| 2019 February 11               |
+--------------------------------+
```

9) DATE_FORMAT(날짜, '형식') : 날짜를 형식에 맞게 출력

DATE 타입	구분 기호	설 명	구분 기호	설 명
년도	%Y	4자리 년도	%y	2자리 년도
월	%M %b	긴 월 이름(January,...) 짧은 월 이름(Jan,...)	%m %c	숫자의 월(01...12) 숫자의 월(1...12)
요일	%W	긴 요일 이름(Sunday,...)	%a	짧은 요일 이름(Sun,...)
일	%D %w	월 내에서 서수 형식의 일(1th,...) 숫자의 요일(1=Sunday,...)	%d %e %i	월 내의 일자 (01...31) 월 내의 일자 (1...31) 일년 중의 날수 (001...366)
시	%l %h	12시간제의 시(1...12) 12시간제의 시(01...12)	%k %H	12시간제의 시 (0...23) 12시간제의 시 (00...23)
분	%i	숫자의 분(00...59)		
초	%S	숫자의 초(00...59)	%s	숫자의 초 (00...59)
시간	%r	12시간제의 시간(hh:mm:ss AM 또는 PM)	%T	24시간제의 시간 (hh:mm:ss)
주	%U	일요일을 기준으로 한 주(0...53)	%u	일요일을 기준으로 한 주 (0...53)
기타	%%	문자 '%'	%p	AM 또는 PM

[예제 6-13] 교수테이블에서 교수코드, 교수명, 임용일자를 년도(4자리), 월(영문), 일(0이 포함된 날짜) 형식으로 출력하라.

```
mysql> select prof_code, prof_name, date_format(create_date, '%Y %M %d')
    -> from professor;
+-----------+-----------+-------------------------------------+
| prof_code | prof_name | date_format(create_date, '%Y %M %d') |
+-----------+-----------+-------------------------------------+
| 4001      | 정진용    | 1995 September 01                    |
| 4002      | 나인섭    | 2006 February 02                     |
| 4003      | 오승재    | 1993 September 01                    |
| 4004      | 고진광    | 1988 March 01                        |
| 4005      | 정병열    | 1998 March 01                        |
| 4006      | 박심심    | 2000 January 15                      |
| 4007      | 김영식    | 2013 March 01                        |
| 4008      | 최우철    | 1997 March 01                        |
| 4009      | 문창우    | 1995 March 01                        |
| 5010      | 정종선    | 1997 March 01                        |
| 5011      | 최종주    | 1992 March 05                        |
+-----------+-----------+-------------------------------------+
11 rows in set (0.03 sec)
```

6.5 데이터형 변환 함수

1) 형변환 함수(Cast Functions)

[형식] CAST(expression AS type)
CONVERT(expression,type)
CONVERT(expr USING transcoding_name)

[type의 종류] ·BINARY , ·CHAR , ·DATE , ·DATETIME , ·SIGNED ,
·TIME , ·UNSIGNED

- cast 함수는 CREATE ... SELECT 구문에서 특정 타입으로 칼럼을 생성하고자 하는데 유용

```
mysql> CREATE TABLE new_table SELECT CAST('2020-01-01' AS DATE);
Query OK, 1 row affected (0.03 sec)
Records: 1  Duplicates: 0  Warnings: 0

mysql> select * from new_table;
+--------------------------+
| CAST('2020-01-01' AS DATE) |
+--------------------------+
| 2020-01-01               |
+--------------------------+
```

-cast 함수는 ENUM 칼럼을 사전순으로 정렬하는데도 유용하게 사용

```
SELECT enum_col FROM tbl_name ORDER BY CAST(enum_col AS CHAR);
```

ENUM 칼럼의 sorting은 내부 수치값을 사용하여 발생한다.
그 값을 CHAR 결과값으로 형변환하면 사전순으로 정렬된다.

-CAST(string AS BINARY)는 BINARY string과 동일하다. CAST(expr AS CHAR)는 구문을 디폴트 캐릭터 셋을 가진 문자열로 취급한다.
-DATE, DATETIME, 또는 TIME으로 CAST()를 하면 그 칼럼은 특정 타입으로만 표시되고 칼럼의 값은 바뀌지 않는다.

```
mysql> SELECT CAST(NOW() AS DATE);
+--------------------+
| CAST(NOW() AS DATE) |
+--------------------+
| 2019-02-28          |
+--------------------+
1 row in set (0.00 sec)
```

NOW()의 년월일 시분초
데이터 값을 DATE(날짜)로 변환

-문자열(string)을 숫자 값으로 형변환 하려면, CAST()를 사용 할 필요가 없다.

```
mysql> SELECT 1+'1';
결과=> 2
```

문자열값을 숫자로 사용

-문자열에서 숫자를 사용하게 되면, 숫자는 자동적으로 binary string으로 변환

```
mysql> SELECT CONCAT("hello you ",2);
결과=> "hello you 2"
```

-signed, unsigned 64bit 값을 가진 수학적인 연산을 지원
-연산자 중의 하나가 unsigned integer라면, 결과는 unsigned로 나타남
-signed나 unsigned 64bit integer로 각각 형변환하는데 오버라이드 가능

```
mysql> SELECT CAST(1-2 AS UNSIGNED);
결과=> 18446744073709551615


mysql> SELECT CAST(CAST(1-2 AS UNSIGNED) AS SIGNED);
결과=> -1
```

-두 연산자 모두 실수 포인트값을 가지고 있다면 결과도 실수 포인트 값

```
mysql> SELECT CAST(1 AS UNSIGNED) - 2.0;
결과=> -1.0
```

-unsigned integer 칼럼에서 뺄셈을 할 때 부호가 있는 정수값은 뺄셈이 일어나기 전에 그 칼럼이 실수 포인트 값으로 형변환된다.

```
SELECT (unsigned_column_1+0.0)-(unsigned_column_2+0.0);
```

6.6 사용자 정의 변수

-사용자 정의 변수이름은 alphanumeric 문자와 '_', '$', '.'로 구성
-변수에 초기값이 지정되지 않으면, NULL이 기본 값이 됨
-초기 값으로 integer, real, string 값을 저장 가능
-변수이름은 버전 5.x부터는 대·소문자 구분이 없다.

● 방법1 : SET 문을 사용하여 변수를 설정

```
mysql> set @t3=5;
mysql> select @t3;
+------+
| @t3  |
+------+
| 5    |
+------+
```

● 방법2 : @variable:=expr 문을 사용하여 설정

```
mysql> SELECT @t1:=(@t2:=1)+@t3:=4,@t1,@t2,@t3;
+----------------------+------+------+------+
| @t1:=(@t2:=1)+@t3:=4 | @t1  | @t2  | @t3  |
+----------------------+------+------+------+
|                    5 | 5    | 1    | 4    |
+----------------------+------+------+------+
```

6.7 시스템 변수

-시스템 변수는 thread-specific 변수와 global 변수가 있다.
-global 변수는 SET GLOBAL 명령으로 변경
-session 변수는 SET SESSION 명령으로 변경

● global 변수 변경

```
SET GLOBAL sort_buffer_size=value; 또는 SET @@global.sort_buffer_size=value;
```

-변수 확인 명령

```
SELECT @@global.sort_buffer_size;
SHOW GLOBAL VARIABLES LIKE 'sort_buffer_size';
```

● session 변수 변경 (여기서 session=LOCAL을 의미함)

```
SET SESSION sort_buffer_size=value; 또는 SET @@session.sort_buffer_size=value;
    또는 SET session.sort_buffer_size=value;
```

-변수 확인 명령

```
SELECT @@session.sort_buffer_size;
SHOW SESSION VARIABLES LIKE 'sort_buffer_size';
```

[실습 따라하기]

```
mysql> set global sort_buffer_size=530000;
Query OK, 0 rows affected (0.01 sec)

mysql> set session sort_buffer_size=524280;
Query OK, 0 rows affected (0.00 sec)

mysql> select @@global.sort_buffer_size;
+---------------------------+
| @@global.sort_buffer_size |
+---------------------------+
|                    530000 |
+---------------------------+
1 row in set (0.01 sec)

mysql> select @@session.sort_buffer_size;
+----------------------------+
| @@session.sort_buffer_size |
+----------------------------+
|                     524280 |
+----------------------------+
1 row in set (0.00 sec)
```

[실습 따라하기]

현재의 autocommit 상태를 확인하는 방법은 다음 예제와 같다.

```
mysql> select @@session.autocommit;
+----------------------+
| @@session.autocommit |
+----------------------+
|                    1 |
+----------------------+
1 row in set (0.03 sec)

mysql> set autocommit=0;
Query OK, 0 rows affected (0.00 sec)

mysql> select @@session.autocommit;
+----------------------+
| @@session.autocommit |
+----------------------+
|                    0 |
+----------------------+
1 row in set (0.01 sec)
```