

12주차 3차시 내장 함수 3

【학습목표】

1. MySQL에서 제공하는 문자열 내장 함수에 대하여 살펴보고, 실습을 통하여 설명할 수 있다.
2. MySQL에서 제공하는 타입 변환 내장 함수에 대하여 살펴보고, 실습을 통하여 설명할 수 있다.

학습내용1 : 문자열 함수 (String Functions)

1. 문자열 비교 함수

- LIKE, NOT LIKE
- STRCMP() : 두 문자열 비교
 - * STRCMP(expr1, expr2)
 - . expr1과 expr2가 같으면, 0을 반환
 - . expr1이 expr2보다 작으면 -1을 반환
 - . expr1이 expr2보다 크면 1을 반환

2. 문자열 함수

| 함수명 | 설명 | 기본 형식 |
|--------------------|--------------------------------------|---------------------------------------|
| ASCII() | 가장 왼쪽 문자의 ASCII 코드 반환 | ASCII(str) |
| BIN() | 숫자의 이진 표현 문자열 반환 | BIN(N) |
| BIT_LENGTH() | 비트 길이를 반환 | BIT_LENGTH(str) |
| CHAR_LENGTH() | 문자의 수를 반환 | CHAR_LENGTH(str) |
| CHAR() | 수에 대응하는 문자를 반환 | CHAR(N, ... [USING charset_name]) |
| CHARACTER_LENGTH() | CHAR_LENGTH()와 동일 | CHARACTER_LENGTH(str) |
| CONCAT_WS() | 분리자를 사용하여 문자열 연결 | CONCAT_WS(separator, str1, str2, ...) |
| CONCAT() | 문자열 연결 | CONCAT(str1, str2, ...) |
| ELT() | 인덱스에 있는 문자열 반환 | ELT(N, str1, str2, str3, ...) |
| FIELD() | 첫 번째 인자를 두 번째 인자부터 끝 인자까지 찾아 인덱스를 반환 | FIELD(str, str1, str2, str3, ...) |
| FIND_IN_SET() | 첫 번째 인자를 두 번째 인자 내에서 찾아 인덱스를 반환 | FIND_IN_SET(str, strlist) |
| HEX() | 10진수 또는 문자열에 대한 16진수를 반환 | HEX(str), HEX(N) |
| UNHEX() | 16진수에 대응하는 문자열 반환 | UNHEX(str) |
| INSERT() | 지정된 위치부터 지정된 수 만큼의 문자를 다른 문자로 대체 | INSERT(str, pos, len, newstr) |
| INSTR() | 문자열의 첫번째 검색 위치를 반환 | INSTR(str, substr) |

| 함수명 | 설명 | 기본 형식 |
|-----------------------------|--------------------------|---|
| LEFT() | 지정된 갯수만큼 왼쪽부터 문자를 반환 | LEFT(str, len) |
| LENGTH(), OCTET_LENGTH() | 문자열의 길이를 바이트로 반환 | LENGTH(str), OCTET_LENGTH(str) |
| LIKE | 단순 패턴 매칭 | LIKE |
| LOAD_FILE() | 지정된 파일을 로드 | LOAD_FILE(file_name) |
| LOCATE(), POSITION() | 문자열의 첫 번째 검색 위치를 반환 | LOCATE(substr, str), LOCATE(substr, str, pos) POSITION(substr IN str) |
| LOWER(), LCASE() | 소문자로 변환 | LOWER(str), LCASE(str) |
| LPAD() | 지정된 문자열로 왼쪽에 첨가된 문자열을 반환 | LPAD(str, len, padstr) |
| LTRIM() | 왼쪽 스페이스를 제거 | LTRIM(str) |
| MATCH | 전체 문장 검색 수행 | MATCH |
| MID() | 지정된 위치부터 시작하는 문자열 반환 | MID(str, pos, len) |
| NOT LIKE | 단순 패턴 매칭의 부정 | NOT LIKE |
| NOT REGEXP | REGEXP의 부정 | NOT REGEXP |
| OCT() | 수에 대한 8진수 반환 | OCT(N) |
| ORD() | 가장 왼쪽 문자에 대한 코드 반환 | ORD(str) |

| 함수명 | 설명 | 기본 형식 |
|-------------------|---------------------------------|--|
| REGEXP, RLIKE | 정규 표현식을 사용한 패턴 매칭 | REGEXP, RLIKE |
| REPEAT() | 문자열을 지정된 수만큼 반복 | REPEAT(str, count) |
| REPLACE() | 문자열 대치 | REPLACE(str, from_str, to_str) |
| REVERSE() | 문자열을 역순으로 반환 | REVERSE(str) |
| RIGHT() | 오른쪽부터 문자열을 지정된 수 만큼 반환 | RIGHT(str, len) |
| RPAD() | 문자열을 지정된 횟수 만큼 첨가 | RPAD(str, len, padstr) |
| RTRIM() | 오른쪽 스페이스를 제거 | RTRIM(str) |
| SPACE() | 지정된 갯 수 만큼 스페이스를 가진 문자열 반환 | SPACE(N) |
| STRCMP() | 두 문자열 비교 | STRCMP(expr1, expr2) |
| SUBSTR() | 지정된 문자열 반환 | SUBSTR(str, pos), SUBSTR(str FROM pos), SUBSTR(str, pos, len), SUBSTR(str FROM pos FOR len) |
| SUBSTRING_INDEX() | 분리자의 지정된 숫자만큼 발생된 위치 이전의 문자열 반환 | SUBSTRING_INDEX(str, delim, count) |
| SUBSTRING() | 지정된 문자열 반환 | SUBSTRING(str, pos), SUBSTRING(str FROM pos), SUBSTRING(str, pos, len), SUBSTRING(str FROM pos FOR len) |
| TRIM() | 문자열의 왼쪽과 오른쪽의 스페이스 제거 | TRIM(str) |
| UPPER(), UCASE() | 대문자로 변환 | UPPER(str), UCASE(str) |

- 예제 : ASCII(str)

```

MySQL
mysql> SELECT ASCII( '2' ), ASCII( 2 ), ASCII( 'dx' );
+-----+-----+-----+
| ASCII( '2' ) | ASCII( 2 ) | ASCII( 'dx' ) |
+-----+-----+-----+
|          50 |          50 |          100 |
+-----+-----+-----+
1 row in set (0.00 sec)

mysql>

```

- 예제 : BIN(N), CONV(N, 10, 2)

```

MySQL
mysql> SELECT BIN( 12 ), CONV( 12, 10, 2 );
+-----+-----+
| BIN( 12 ) | CONV( 12, 10, 2 ) |
+-----+-----+
| 1100      | 1100              |
+-----+-----+
1 row in set (0.00 sec)

mysql>

```

- 예제 : BIT_LENGTH(str), CHARACTER_LENGTH(str)

```

MySQL
mysql> SELECT BIT_LENGTH( 'text' ), CHARACTER_LENGTH( 'text' );
+-----+-----+
| BIT_LENGTH( 'text' ) | CHARACTER_LENGTH( 'text' ) |
+-----+-----+
| 32                    | 4                            |
+-----+-----+
1 row in set (0.00 sec)

mysql>

```

- 예제 : CONCAT(str1, str2, ...), CONCAT_WS(separator, str1, str2, ...)

```

MySQL
mysql> SELECT CONCAT( 'abc', '123', 'xyz' ), CONCAT_WS( ' ', 'abc', '123', 'xyz' );
+-----+-----+
| CONCAT( 'abc', '123', 'xyz' ) | CONCAT_WS( ' ', 'abc', '123', 'xyz' ) |
+-----+-----+
| abc123xyz                      | abc 123 xyz                          |
+-----+-----+
1 row in set (0.00 sec)

mysql>

```

- 예제 : ELT(N, str1, str2, str3, ...)

```

MySQL
mysql> SELECT ELT( 1, 'ej', 'Heja', 'hej', 'foo' ), ELT( 4, 'ej', 'Heja', 'hej', 'foo' );
+-----+-----+
| ELT( 1, 'ej', 'Heja', 'hej', 'foo' ) | ELT( 4, 'ej', 'Heja', 'hej', 'foo' ) |
+-----+-----+
| ej                                     | foo                                    |
+-----+-----+
1 row in set (0.00 sec)

mysql>

```

- 예제 : FIELD(str, str1, str2, str3, ...)

```

MySQL
mysql> SELECT FIELD( 'ej', 'Heja', 'ej', 'hej', 'foo' );
+-----+
| FIELD( 'ej', 'Heja', 'ej', 'hej', 'foo' ) |
+-----+
| 2 |
+-----+
1 row in set (0.00 sec)

mysql> SELECT FIELD( 'fo', 'Heja', 'ej', 'hej', 'foo' );
+-----+
| FIELD( 'fo', 'Heja', 'ej', 'hej', 'foo' ) |
+-----+
| 0 |
+-----+
1 row in set (0.00 sec)

mysql>

```

- 예제 : FIND_IN_SET(str, strlist)

```

MySQL
mysql> SELECT FIND_IN_SET( 'b', 'a,b,c,d' );
+-----+
| FIND_IN_SET( 'b', 'a,b,c,d' ) |
+-----+
| 2 |
+-----+
1 row in set (0.00 sec)

mysql>

```

- 예제 : HEX(str), HEX(N), CONV(N, 16, 10), UNHEX(str)

```

MySQL
mysql> SELECT 0x616263, HEX( 'abd' ), UNHEX( HEX( 'abc' ) );
+-----+-----+-----+
| 0x616263 | HEX( 'abd' ) | UNHEX( HEX( 'abc' ) ) |
+-----+-----+-----+
| abc      | 616264      | abc                    |
+-----+-----+-----+
1 row in set (0.00 sec)

mysql> SELECT HEX(255), CONV( HEX(255), 16, 10 );
+-----+-----+
| HEX(255) | CONV( HEX(255), 16, 10 ) |
+-----+-----+
| FF       | 255                       |
+-----+-----+
1 row in set (0.00 sec)

mysql>

```

- 예제 : UNHEX(str)

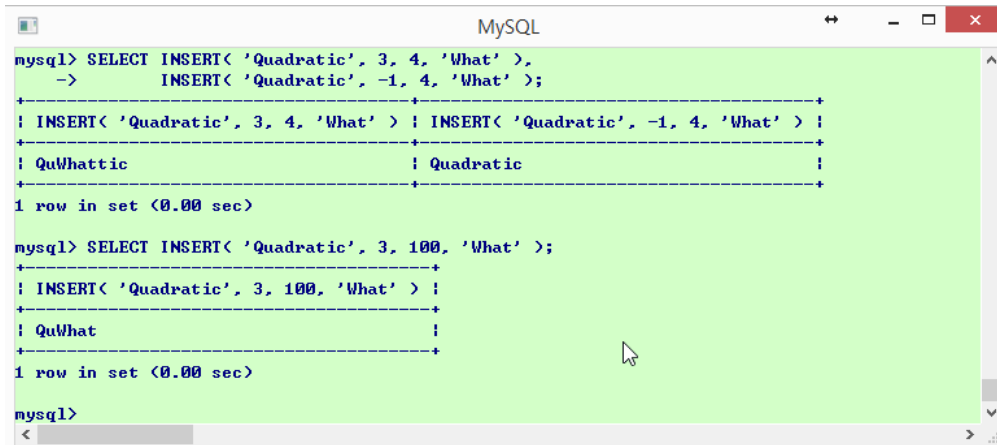
```

MySQL
mysql> SELECT UNHEX( '4D7953514C' ), 0x4D7953514C;
+-----+-----+
| UNHEX( '4D7953514C' ) | 0x4D7953514C |
+-----+-----+
| MySQL                  | MySQL         |
+-----+-----+
1 row in set (0.00 sec)

mysql>

```

- 예제 : INSERT(str, pos, len, newstr)



```

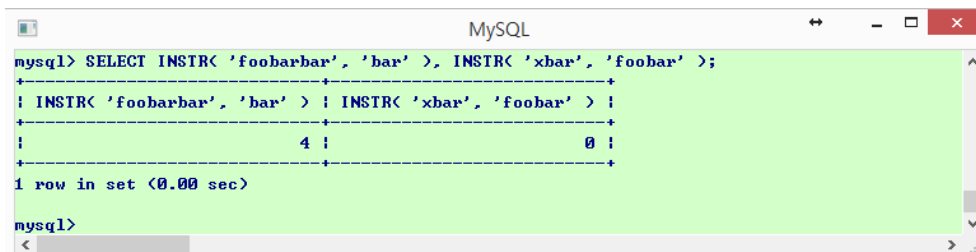
mysql> SELECT INSERT< 'Quadratic', 3, 4, 'What' >,
->      INSERT< 'Quadratic', -1, 4, 'What' >;
+-----+-----+
| INSERT< 'Quadratic', 3, 4, 'What' > | INSERT< 'Quadratic', -1, 4, 'What' > |
+-----+-----+
| QuWhattic                           | Quadratic                           |
+-----+-----+
1 row in set (0.00 sec)

mysql> SELECT INSERT< 'Quadratic', 3, 100, 'What' >;
+-----+
| INSERT< 'Quadratic', 3, 100, 'What' > |
+-----+
| QuWhat                                |
+-----+
1 row in set (0.00 sec)

mysql>

```

- 예제 : INSTR(str, substr)



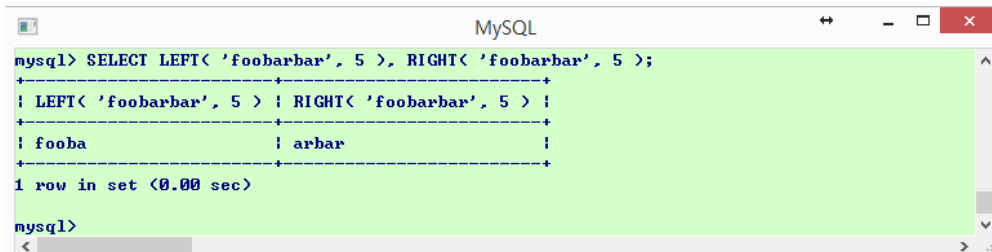
```

mysql> SELECT INSTR< 'foobarbar', 'bar' >, INSTR< 'xbar', 'foobar' >;
+-----+-----+
| INSTR< 'foobarbar', 'bar' > | INSTR< 'xbar', 'foobar' > |
+-----+-----+
| 4                            | 0                            |
+-----+-----+
1 row in set (0.00 sec)

mysql>

```

- 예제 : LEFT(str, len), RIGHT(str, len)



```

mysql> SELECT LEFT< 'foobarbar', 5 >, RIGHT< 'foobarbar', 5 >;
+-----+-----+
| LEFT< 'foobarbar', 5 > | RIGHT< 'foobarbar', 5 > |
+-----+-----+
| fooba                  | arbar                    |
+-----+-----+
1 row in set (0.00 sec)

mysql>

```

- 예제 : LOCATE(substr, str), LOCATE(substr, str, pos),
POSITION(substr IN str)

```

MySQL
mysql> SELECT LOCATE( 'bar', 'foobarbar' ), POSITION( 'bar' IN 'foobarbar' );
+-----+-----+
| LOCATE( 'bar', 'foobarbar' ) | POSITION( 'bar' IN 'foobarbar' ) |
+-----+-----+
| 4 | 4 |
+-----+-----+
1 row in set (0.00 sec)

mysql> SELECT LOCATE( 'bar', 'foobarbar', 5 );
+-----+
| LOCATE( 'bar', 'foobarbar', 5 ) |
+-----+
| 7 |
+-----+
1 row in set (0.00 sec)

mysql>

```

- 예제 : LOWER(str), LCASE(str), UPPER(str), UCASE(str)

```

MySQL
mysql> SELECT LOWER( 'QUADratic' ), LCASE( 'QUADratic' );
+-----+-----+
| LOWER( 'QUADratic' ) | LCASE( 'QUADratic' ) |
+-----+-----+
| quadratic | quadratic |
+-----+-----+
1 row in set (0.00 sec)

mysql> SELECT UPPER( 'QUADratic' ), UCASE( 'QUADratic' );
+-----+-----+
| UPPER( 'QUADratic' ) | UCASE( 'QUADratic' ) |
+-----+-----+
| QUADRATIC | QUADRATIC |
+-----+-----+
1 row in set (0.00 sec)

mysql>

```

- 예제 : LPAD(str, len, padstr), RPAD(str, len, padstr)

```

MySQL
mysql> SELECT LPAD( 'hi', 4, '??' ), LPAD( 'hi', 1, '??' ), LPAD( 'hi', 5, '??' );
+-----+-----+-----+
| LPAD( 'hi', 4, '??' ) | LPAD( 'hi', 1, '??' ) | LPAD( 'hi', 5, '??' ) |
+-----+-----+-----+
| ??hi | h | ???hi |
+-----+-----+-----+
1 row in set (0.00 sec)

mysql> SELECT RPAD( 'hi', 5, '?' ), RPAD( 'hi', 1, '?' );
+-----+-----+
| RPAD( 'hi', 5, '?' ) | RPAD( 'hi', 1, '?' ) |
+-----+-----+
| hi??? | h |
+-----+-----+
1 row in set (0.00 sec)

mysql>

```

- 예제 : LTRIM(str), RTRIM(str), TRIM(str)

```

MySQL
mysql> SELECT LTRIM( ' abc ' ), RTRIM( ' abc ' ), TRIM( ' abc ' );
+-----+-----+-----+
| LTRIM( ' abc ' ) | RTRIM( ' abc ' ) | TRIM( ' abc ' ) |
+-----+-----+-----+
| abc              | abc              | abc              |
+-----+-----+-----+
1 row in set (0.00 sec)

mysql>

```

- 예제 : SUBSTRING(str, pos), SUBSTRING(str FROM pos),
SUBSTRING(str, pos, len), SUBSTRING(str FROM pos FOR len)

```

MySQL
mysql> SELECT SUBSTRING( 'Quafratically', 5 ), SUBSTRING( 'foobarbar' FROM 4 );
+-----+-----+
| SUBSTRING( 'Quafratically', 5 ) | SUBSTRING( 'foobarbar' FROM 4 ) |
+-----+-----+
| ratically                        | barbar                          |
+-----+-----+
1 row in set (0.00 sec)

mysql> SELECT SUBSTRING( 'ChangSeungKim', -3 ), SUBSTRING( 'ChangSeungKim', -8, 5 );
+-----+-----+
| SUBSTRING( 'ChangSeungKim', -3 ) | SUBSTRING( 'ChangSeungKim', -8, 5 ) |
+-----+-----+
| Kim                              | Seung                             |
+-----+-----+
1 row in set (0.00 sec)

mysql> SELECT SUBSTRING( 'ChangSeungKim' FROM -8 FOR 5 );
+-----+
| SUBSTRING( 'ChangSeungKim' FROM -8 FOR 5 ) |
+-----+
| Seung                                       |
+-----+
1 row in set (0.00 sec)

mysql>

```

- 예제 : OCT(N)

```

MySQL
mysql> SELECT OCT( 12 );
+-----+
| OCT( 12 ) |
+-----+
| 14        |
+-----+
1 row in set (0.00 sec)

mysql>

```

- 예제 : REPEAT(str, count)

```

MySQL
mysql> SELECT REPEAT( 'MySQL', 3 );
+-----+
| REPEAT( 'MySQL', 3 ) |
+-----+
| MySQLMySQLMySQL      |
+-----+
1 row in set (0.00 sec)

mysql>

```

- 예제 : REPLACE(str, from_str, to_str)

```

MySQL
mysql> SELECT REPLACE( 'www.mysql.com', 'w', 'Ww' );
+-----+
| REPLACE( 'www.mysql.com', 'w', 'Ww' ) |
+-----+
| WwWwWw.mysql.com                       |
+-----+
1 row in set (0.00 sec)

mysql>

```

- 예제 : REVERSE(str)

```

MySQL
mysql> SELECT REVERSE( 'abc' );
+-----+
| REVERSE( 'abc' ) |
+-----+
| cba              |
+-----+
1 row in set (0.00 sec)

mysql>

```

- 예제 : SPACE(N)

```

MySQL
mysql> SELECT CONCAT( 'abc', SPACE( 6 ), '123' );
+-----+
| CONCAT( 'abc', SPACE( 6 ), '123' ) |
+-----+
| abc      123                       |
+-----+
1 row in set (0.00 sec)

mysql>

```

- 예제 : SUBSTR(str, pos), SUBSTR(str FROM pos),
SUBSTR(str, pos, len), SUBSTR(str FROM pos FOR len)

```

MySQL
mysql> SELECT SUBSTR( 'Quafratically', 5 ), SUBSTR( 'foobarbar' FROM 4 );
+-----+-----+
| SUBSTR( 'Quafratically', 5 ) | SUBSTR( 'foobarbar' FROM 4 ) |
+-----+-----+
| ratically                    | barbar                      |
+-----+-----+
1 row in set (0.00 sec)

mysql> SELECT SUBSTR( 'ChangSeungKim', -3 ), SUBSTR( 'ChangSeungKim', -8, 5 );
+-----+-----+
| SUBSTR( 'ChangSeungKim', -3 ) | SUBSTR( 'ChangSeungKim', -8, 5 ) |
+-----+-----+
| Kim                           | Seung                         |
+-----+-----+
1 row in set (0.00 sec)

mysql> SELECT SUBSTR( 'ChangSeungKim' FROM -8 FOR 5 );
+-----+
| SUBSTR( 'ChangSeungKim' FROM -8 FOR 5 ) |
+-----+
| Seung                                    |
+-----+
1 row in set (0.00 sec)

mysql>

```


- 예제 : SUBSTRING_INDEX(str, delim, count)

```

MySQL
mysql> SELECT SUBSTRING_INDEX( 'www.mysql.com', '.', 2 ),
-> SUBSTRING_INDEX( 'www.mysql.com', '.', -2 );
+-----+-----+
| SUBSTRING_INDEX( 'www.mysql.com', '.', 2 ) | SUBSTRING_INDEX( 'www.mysql.com', '.', -2 ) |
+-----+-----+
| www.mysql                                | mysql.com                                |
+-----+-----+
1 row in set (0.00 sec)

mysql>

```

학습내용2 : 타입 변환 함수 (Type Conversion Functions)

| 함수명 | 설명 | 기본 형식 |
|-----------|-------------------|--|
| BINARY | 문자열을 바이너리 문자열로 변환 | |
| CAST() | 값을 다른 데이터 타입으로 변환 | CAST(str AS type) |
| CONVERT() | 값을 다른 데이터 타입으로 변환 | CONVERT(str, type) CONVERT(expr USING transcoding_name) |

- type : BINARY[(N)], CHAR[(N)], DATE, DATETIME, DECIMAL[(M[,D])], SIGNED [INTEGER], TIME, UNSIGNED [INTEGER]

1. BINARY

- 문자열을 바이너리 문자열로 변환
- 속성값 비교를 강화하기 위한 방법
 - . 문자끼리 비교가 아닌 바이트 간의 비교
- 대소문자 구분함
- 문자뒤의 스페이스도 중요한 문자로 인식

```

MySQL
mysql> SELECT 'a' = 'A';
+-----+
| 'a' = 'A' |
+-----+
|          1 |
+-----+
1 row in set (0.00 sec)

mysql> SELECT BINARY 'a' = 'A';
+-----+
| BINARY 'a' = 'A' |
+-----+
|                  0 |
+-----+
1 row in set (0.00 sec)

mysql>

```

```

mysql> SELECT 'a' = 'a ';
+-----+
| 'a' = 'a ' |
+-----+
|          1 |
+-----+
1 row in set (0.00 sec)

mysql>
mysql> SELECT BINARY 'a' = 'a ';
+-----+
| BINARY 'a' = 'a ' |
+-----+
|                0 |
+-----+
1 row in set (0.00 sec)

```

2. CAST()

- 입력 : 모든 데이터 타입의 표현식
- 출력 : 지정된 데이터 타입으로 결과 값을 변환
- CONVERT()와 유사
- 기본 형식

CAST(expr AS type)

3. CONVERT()

- 입력 : 모든 데이터 타입의 표현식
- 출력 : 지정된 데이터 타입으로 결과 값을 변환
- 기본 형식

CONVERT(expr, type)

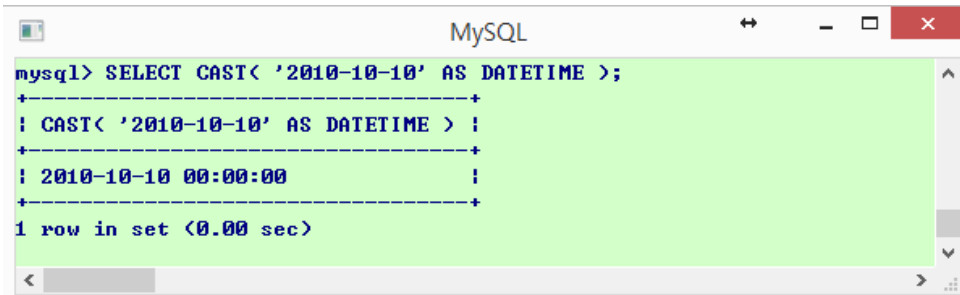
CONVERT(expr USING 인코딩문자세트)

- 변환가능한 데이터 type

. BINARY [(N)]
 . CHAR [(N)]
 . DATE
 . DATETIME
 . DECIMAL [(M [, D])]
 . SIGNED [INTEGER]
 . TIME
 . UNSIGNED [INTEGER]

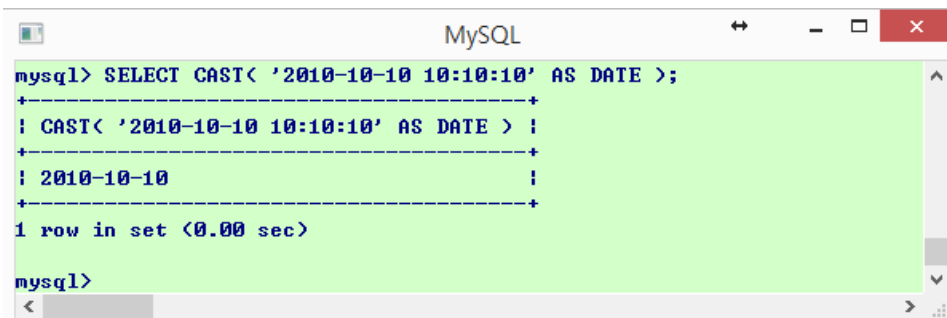
- CONVERT(... USING ...) : 다른 인코딩 문자세트로 변환하기 위하여 사용
 . SELECT CONVERT('한글' USING utf8) ;

- 예제
 . '2010-10-10' 문자열을 DATETIME 데이터 타입으로 변환
 ==> SELECT CAST('2010-10-10' AS DATETIME);



```
mysql> SELECT CAST( '2010-10-10' AS DATETIME );
+-----+
| CAST( '2010-10-10' AS DATETIME ) |
+-----+
| 2010-10-10 00:00:00                |
+-----+
1 row in set (0.00 sec)
```

. '2010-10-10 10:10:10'를 DATE 데이터 타입으로 변환
 ==> SELECT CAST('2010-10-10 10:10:10' AS DATE);



```
mysql> SELECT CAST( '2010-10-10 10:10:10' AS DATE );
+-----+
| CAST( '2010-10-10 10:10:10' AS DATE ) |
+-----+
| 2010-10-10                             |
+-----+
1 row in set (0.00 sec)

mysql>
```

【학습정리】

1. 문자열 내장 함수는 문자열을 비교하여 검색하기 위한 내장 함수와 검색된 데이터내의 문자열을 처리하는 내장함수로 구분할 수 있다.
2. DBMS는 자동적으로 데이터 타입을 변환하는 기능을 가지고 있지만, 명시적으로 변환하기 위한 내장함수를 제공한다.