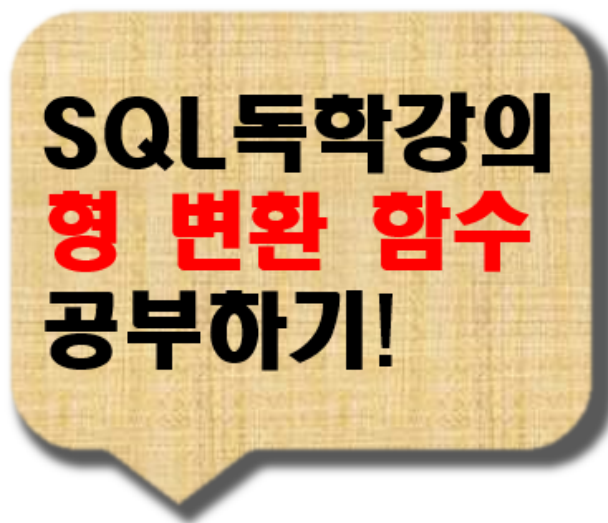


sql 독학 강의 # 단일행 함수 잘 사용 하기(형 변환 함수) 7편 -sTricky

sTricky 2020. 4. 7. 14:47

sql 독학 강의 # 단일행 함수 잘 사용 하기(형 변환 함수) 7편 -sTricky



sql 독학 강의 # 단일행 함수 잘 사용 하기(형 변환 함수)

[컨텐츠 index](#)

1. mysql의 데이터 타입 알아보기
2. 묵시적 형 변환 이란?
3. CAST, CONVERT 함수 사용 하기

안녕하세요.

코로나 여파로 여러 가지 신경 쓸 일들이 겹치다 보니 7편 업로드가 좀 늦어졌습니다.

이번 SQL 독학 강의의 주제는 단일행 함수 중에서도 **형 변환 함수**에 관련된 내용입니다.

데이터베이스에 데이터를 저장할때는 그냥 text 형태로 넣을 수도 있지만 여러 가지 데이터 형을 칼럼에 정의하고 형태에 맞는 데이터를 insert 하고 관리하게 됩니다.

그래야지 데이터의 정합성을 지키는데도 유리하며, 관리적인 측면에서도 수월해지기 때문 입니다.

하지만 데이터를 핸들링하다 보면 그 형태를 다른 형태로 변환해야 하는 경우가 생기기도 합니다.

데이터베이스는 데이터의 형태를 쉽게 변형할 수 있도록 함수를 제공하고 있습니다.

거기에 관해서 알아보도록 하겠습니다.

#지난 강의 보러 가기#

[2020/03/31 - \[Database/sql 강의\] - sql 독학 강의 # 단일행 함수 잘 사용 하기\(날짜 함수\) 6편 -sTricky.](#)

sql 독학 강의 # 단일행 함수 잘 사용 하기(날짜 함수) 6편 -sTric...

stricky.tistory.com

1. mysql의 데이터 타입 알아보기

mysql에도 여러 데이터 형태가 존재합니다. 우리는 그것을 데이터 타입이라고 부릅니다. mysql의 데이터 타입에 관해서 알아보겠습니다.

1) 문자형 데이터 타입

데이터 유형	정의
CHAR[(M)]	고정 길이를 갖는 문자열을 저장. M은 1 ~ 255($2^8 - 1$). CHAR(20)인 칼럼에10자만 저장을 하더라도, 20자만큼의 기억 장소를 차지.
VARCHAR[(M)]	가변 길이를 갖는 문자열을 저장. M은 1 ~ 65535($2^{16} - 1$). VARCHAR(20)인 칼럼에10자만 저장을 하면, 실제로도 10자만큼의 기억 장소를 차지.
TINYTEXT[(M)]	최대 255($2^8 - 1$) bytebyte
TEXT[(M)]	최대 65535($2^{16} - 1$) bytebyte
MEDIUMTEXT[(M)]	최대 16777215($2^{24} - 1$) bytebyte
LONGTEXT[(M)]	최대 4294967295($2^{32} - 1$) bytebyte
ENUM('value1', 'value2',...)	열거형. 정해진 몇 가지의 값들 중 하나만 저장. 최대 65535개의 개별 값을 가질 수 있고, 내부적으로 정수 값으로 표현된다.
SET('value1', 'value2',...)	집합형. 정해진 몇 가지의 값들 중 여러 개를 저장. 최대 64개의 요소로 구성될 수 있고, 내부적으로는 정수 값이다.

2) 숫자형 데이터 타입

데이터 유형	바이트	정의
BIT[(M)]	1	비트 값 유형. M은 값 당 비트 수를 나타내며 1에서 64 사이의 값을 나타냄.
BOOL, BOOLEAN		이 유형은 TINYINT (1)의 동의어. 0은 false, 0이 아닌 값은 true로 간주

TINYINT[(M)]	1	(signed) -128 ~ 127 (unsigned) 0 ~ 255(2^8)
SMALLINT[(M)]	2	(signed) -32768 ~ 32767 (unsigned) 0 ~ 65535(2^16)
MEDIUMINT[(M)]	3	(signed) -8388608 ~ 8388607 (unsigned) 0 ~ 16777215(2^24)
INT[(M)]	4	(signed) -2147483648 ~ 2147483647 (unsigned) 0 ~ 4294967295(2^32)
BIGINT[(M)]	8	(signed) -9223372036854775808 ~ 9223372036854775807 (unsigned) 0 ~ 18446744073709551615(2^64)
FLOAT[(M, D)]	4	(signed) -3.402823466E+38 ~ 1.175494351 E-38 (unsigned) 1.175494351 E-38 ~ 3.402823466E+38
DOUBLE[(M, D)] DOUBLE PRECISION[(M, D)] REAL[(M, D)]	8	(signed) -1.7976931348623157E+908 ~ -2.2250738585072014 E-308 (unsigned) 2.2250738585072014 E-308 ~ 1.7976931348623157E+308
FLOAT(p)		부동 소수점 숫자. p는 비트 정밀도를 가리키지만, MySQL은 결과 데이터 타입으로 FLOAT 또는 DOUBLE을 사용할지를 결정할 때에만 이 값을 사용한다.
DECIMAL[(M [, D])]	길이+1	묶음 고정 소수점 숫자 M은 전체 자릿수(Precision : 정밀도), D는 소수점 뒷자리수(Scale : 배율) - DECIMAL(5)의 경우 : -99999 ~ 99999 - DECIMAL(5, 1)의 경우 : -9999.9 ~ 9999.9 - DECIMAL(5, 2)의 경우 : -999.99 ~ 999.99 최대 65자리까지 지원
DEC[(M [, D])] NUMERIC[(M [, D])] FIXED[(M [, D])]		DECIMAL과 동의어다. FIXED 동의어는 다른 데이터베이스 시스템과의 호환을 위해서 사용하는 것이다.

3) 날짜형 데이터 타입

데이터 유형	바이트	정의
DATE	3	YYYY-MM-DD('1001-01-01' ~ '9999-12-31')

TIME	3	HH:MM:SS('838:59:59' ~ '838:59:59')
DATETIME	8	YYYY-MM-DD HH:MM:SS('1001-01-01 00:00:00' ~ '9999-12-31 23:59:59')
TIMESTAMP[(M)]	4	1970-01-01 ~ 2037년 임의 시간(1970-01-01 00:00:00을 0으로 해서 1초 단위로 표기)
YEAR[(2 4)]	1	2와 4를 지정할 수 있으며, 2인 경우에 값의 범위는 70 ~ 69, 4인 경우에는 1970 ~ 2069이다.

4) 이진형 데이터 타입

데이터 유형	정의
BINARY[(M)]	CHAR 유형과 유사하지만 이진 바이트 문자열을 이진이 아닌 문자열로 저장. M은 바이트 단위의 열 길이를 나타냄.
VARBINARY[(M)]	VARCHAR 유형과 유사하지만 이진 바이트 문자열을 이진이 아닌 문자열로 저장. M은 바이트 단위의 열 길이를 나타냄.
TINYBLOB[(M)]	이진 데이터 타입. 최대 255(2 ⁸ - 1) byte
BLOB[(M)]	이진 데이터 타입. 최대 65535(2 ¹⁶ - 1) byte
MEDIUMBLOB[(M)]	이진 데이터 타입. 최대 16777215(2 ²⁴ - 1) byte
LOBLOB[(M)]	이진 데이터 타입. 최대 4294967295(2 ³² - 1) byte

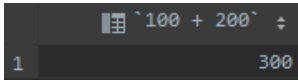
2. 묵시적 형 변환이란?

묵시적 형 변환이란, 데이터의 형태를 사용자의 의도에 맞춰서 데이터 베이스가 알아서 형 변환하여 결과를 출력하는 행위를 말합니다.

아래의 쿼리를 보겠습니다.

```
select 100 + 200 from dual;
```

100과 200을 더하는 SQL 명령입니다. 100, 200 모두 숫자로 입력을 했습니다. 당연히 결과값 300이 출력됩니다.

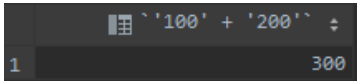


```
1 100 + 200
300
```

이번에는 100, 200 모두 문자로 바뀌서 입력 후 연산을 해보겠습니다.

```
select '100' + '200' from dual;
```

100, 200에 모두 따옴표를 붙여서 문자로 인식하도록 하고 연산을 실행했습니다.



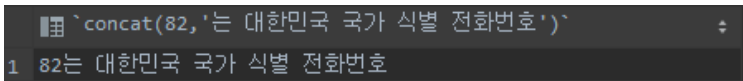
```
1 '100' + '200'
300
```

이렇게 해도 계산이 잘 되는군요.

그럼 이번에는 숫자 + 문자열을 합쳐서 하나의 문자열로 만드는 시도를 concat 함수를 통해서 해보겠습니다.

```
select concat(82, '는 대한민국 국가 식별 전화번호') from dual;
```

위 SQL을 보면 82는 숫자형으로 뒤에 있는 글씨는 문자열로 입력 후 concat을 통해 합치라고 했습니다. 그 결과는?



```
1 concat(82, '는 대한민국 국가 식별 전화번호')
82는 대한민국 국가 식별 전화번호
```

이렇게 잘 합쳐서 출력이 됩니다.

이와 같이 문자에서 숫자로, 숫자에서 문자로 사용자의 의도에 맞게 데이터 형태가 자동으로 변환되는 것을 묵시적 형 변환이라고 표현합니다.

3. CAST, CONVERT 함수 사용 하기

CAST 함수란 mysql에서 데이터 타입을 서로 변환시켜주는 형 변환 함수입니다. 사용법은 아주 간단합니다.

CAST (표현할 값 AS 데이터 형식[(길이)]);

CONVERT (표현할 값 , 데이터 형식[(길이)]);

위와 같이 사용을 하시면 됩니다. 매우 쉽죠? 그럼 예제를 통해서 한번 확인해보겠습니다. 아래 내용에 있는 설명은 **CAST** 함수를 기준으로 설명을 드리겠습니다. **CONVERT** 함수의 사용법도 거의 같기 때문에 직접 실습을 통해서 해보시길 바랍니다.

아래와 같은 데이터가 있습니다.

	number	text	date
1	1	korea	2020-04-07 14:00:11
2	2	USA	2020-04-07 14:00:13
3	3	PHP	2020-04-07 14:00:15
4	4	JAVA	2020-04-07 14:00:15
5	5	python	2020-04-07 14:00:16
6	6	law	2020-04-07 14:00:16
7	7	cup	2020-04-07 14:00:17

첫 번째 number 칼럼의 데이터를 보면 1,2,3.... 7 이렇게 들어있고, text라는 칼럼에는 문자, 글자가 입력되어 있습니다.

숫자 데이터의 경우 셀 안에서 우측에 붙어서 표현이 되고, 문자의 경우 좌측에 붙어서 출력이 되고 있습니다. 다시 말하면 결과 나오는 것을 보면 같은 숫자라도 우측에 붙어 있으면 숫자, 좌측에 붙어있으면 문자로 데이터베이스가 인식하고 있다는 말이 됩니다.

아래 쿼리를 확인해보겠습니다.

```
select cast(100 as char) as num_to_char, cast('100' as signed) as char_to_num from dual;
```

num_to_char 칼럼을 보면 100 이라는 숫자 데이터를 char 즉, 문자 데이터 형태로 변경해라, 그리고 **char_to_num** 칼럼을 보시면 '100'이라는 문자 데이터를 숫자 데이터 형식으로 변경하라는 명령입니다. 결과를 보실까요?

	num_to_char	char_to_num
1	100	100

똑같이 100이라고 결과가 나와있지만, **num_to_char**를

각각 원하는 형태로 변경이 잘 된 것 이죠.

이번엔 문자열을 날짜로 변경하는 것을 해보도록 하겠습니다.

아래와 같이 SQL을 실행해보도록 할게요.

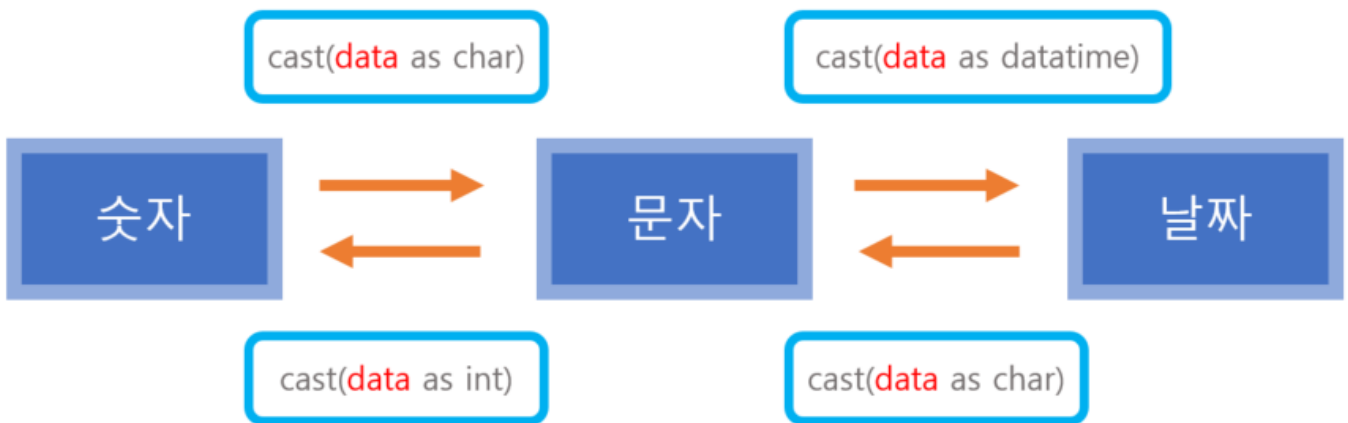
```
select '2016-08-25 03:30:00', cast('2016-08-25 03:30:00' as datetime) as char_to_datetime from dual;
```

앞에는 그냥 문자열로 인식이 되는 데이터이고요, 뒤에 있는 **char_to_datetime**은 날짜 형식의 데이터로 인식이 되는 형태로 출력이 될 것입니다.

	2016-08-25 03:30:00	char_to_datetime
1	2016-08-25 03:30:00	2016-08-25 03:30:00

이렇게 결과가 나왔습니다. 눈으로 확인 하긴 어렵지만.. 이렇게 형 변환이 일어난 것으로 볼 수 있습니다.

여기까지 예를 들어드린 것을 그림으로 설명을 해보면 아래와 같습니다.



이렇게 그림으로 보닌 칸 이해가 좀 쉽지 않습니까?

포스팅 내용에 관한 질문은 댓글로 남겨두시면 답변 해드리겠습니다.

오늘 내용은 이렇게 간단하면서도 중요하게 쓰이는 데이터 형 변환에 관해서 알아보았습니다.

다음 시간에는 **일반 함수**에 관해서 알아볼 예정입니다. 많은 성원 부탁드립니다.

감사합니다.