SQL 기초와 데이터 분석

하홍석

programmers

5. 다양한 데이터 타입 다루기

5. 다양한 데이터 타입 다루기

1. 숫자
2. 문자
3. 이진
4. Array
5. Key-value

숫자

숫자

데이터 타입	의미	비고
BIT(M)	0과 1로만 구성 예> b'111' = 7	1<=M<=64 기본값 : M=1
TINYINT	매우 작은 정수	Signed 범위 : -127 ~ 127 Unsigned 범위 : 0 ~ 255
BOOL, BOOLEAN	True/False	TINYINT(1) 과 같음 0이면 False, 1이면 True
SMALLINT	작은 정수	-32768 ~ 32767 0 ~ 65535
MEDIUMINT	중간 크기 정수	-8388608 ~ 8388607 0 ~ 16777215
INT, INTEGER	정수	-2147483648 ~ 2147483647 0 ~ 4294967295
BIGINT, SERIAL	INT 타입보다 2배 많은 비트를 사용하는 정수	-9223372036854775808 ~ 9223372036854775807 0 ~ 18446744073709551615.
DECIMAL(M,D), DEC, FIXED	고정소수점 타입	M<=65 0<=D<=30
FLOAT	부동소수점 타입	4 bytes
DOUBLE	부동소수점 타입	8 bytes

Ref : https://dev.mysql.com/doc/refman/8.0/en/numeric-type-syntax.html

문자

문자

데이터 타입	의미	비고
CHAR	고정된 길이의 문자열	길이 0~255 선언된 값보다 짧은 문자열이 들어오면, 빈 문자열로 나머지 길이를 채움
VARCHAR	변동 가능한 길이의 문자열	길이 0~65535
TEXT	변동 가능한 길이의 문자열	길이 최대 65535 (길이 설정 불가) Non-binary strings (=character strings) 기본값 지정 불가
TINYTEXT	작은 TEXT	길이 최대 255
MEDIUMTEXT	중간 크기 TEXT	길이 최대 16777215
LONGTEXT	큰 크기 TEXT	길이 최대 4294967295
ENUM	최초에 지정해 둔 리스트에 포함되는 값만 저장	효율적인 데이터 저장 유연성, 확장성 낮음
SET	최초에 지정해 둔 리스트에 포함되는 값들을 중복으로 저장	64개까지

Ref : https://dev.mysql.com/doc/refman/8.0/en/string-types.html

이진

이진

데이터 타입	의미	비고
BLOB	Binary Large Object	길이 최대 65535 (길이 설정 불가) 기본값 지정 불가
TINYBLOB	작은 BLOB	길이 최대 255
MEDIUMBLOB	중간 크기 BLOB	길이 최대 16777215
LONGBLOB	큰 BLOB	길이 최대 4294967295
BINARY	Binary strings (고정된 길이)	길이 0~255 선언된 값보다 짧은 문자열이 들어오면, 빈 문자열로 나머지 길이를 채움
VARBINARY	Binary strings (변동 가능한 길이)	길이 0~255

Array

Array

1. Array (배열) : 데이터가 저장된 리스트
    1. 예> [‘a’, ‘b’, ‘c’] / [1, 2, 3]
2. Element (원소) : Array에 저장된 각 데이터
3. JSON 타입으로 배열을 저장
4. 기본값 설정 불가

1. JSON_ARRAY : 입력을 JSON 배열로 반환하는 함수

```
CREATE TABLE IF NOT EXISTS 'products' (
  'product_id' int(6) NOT NULL,
  'category' varchar(40) NOT NULL,
  'name' varchar(10) NOT NULL,
  'price' int unsigned NOT NULL,
  'options' JSON NULL,
  PRIMARY KEY ('product_id')
);

INSERT INTO 'products' ('product_id', 'category', 'name', 'price',
                        'options') VALUES
  (0, '키즈', '어린이칫솔', 1500, JSON_ARRAY('빨강','파랑')),
  (1, '스포츠', '손목보호대', 10000, JSON_ARRAY('S', 'M','L')),
  (2, '주방용품', '밥그릇', 2000, JSON_ARRAY('소', '중','대'))
  (3, '디지털', '마우스', 15000, NULL) ;
```

2. JSON_TYPE : JSON 데이터의 타입을 반환하는 함수

실습 : https://www.programiz.com/sql/online-compiler/

3. JSON_EXTRACT

```
SELECT JSON_EXTRACT(options, '$') as all_elements
FROM products
```

```
SELECT JSON_EXTRACT(options, '$[0]') as first_element
FROM products
```

Key-value

Key-value

1. Key-value : Key와 Value로 이루어진 데이터
    1. Key를 통해 Value에 접근할 수 있음
    2. 예
        1. {‘이름’ : ‘홍길동’, ‘부서’ : ‘개발팀’, ‘직책’ : ‘팀장’, ‘근무지’ : ‘판교’}
        2. {‘색상’ : [‘빨강’, ‘파랑’], ‘사이즈’ : [’S’, ‘M’, ‘L’]}
2. JSON 타입으로 key-value를 저장

Key-value

1. JSON_OBJECT

```
INSERT INTO 'managers_v2' ('id', 'name', 'managing', 'info') VALUES
  (0, '영희', '스포츠', JSON_OBJECT('off', JSON_ARRAY('일', '월'), 'substitute', '민수')),
  (1, '철수', '주방용품', JSON_OBJECT('off', JSON_ARRAY('화', '수'), 'substitute', '길순')),
  (2, '민수', '디지털', JSON_OBJECT('off', JSON_ARRAY('목', '금'), 'substitute', '철수')),
  (3, '길순', '키즈', JSON_OBJECT('off', JSON_ARRAY('금', '토'), 'substitute', '영희'))
```

2. JSON_EXTRACT

```
SELECT JSON_EXTRACT(info, '$.off') as off
FROM managers_v2
```

3. JSON_INSERT

```
UPDATE managers_v2 set info = JSON_INSERT(info, '$.new', JSON_ARRAY(1,2,3,4));
```

4. JSON_REPLACE

```
UPDATE managers_v2 set info = JSON_REPLACE(info, '$.new', 1);
```

End of Document