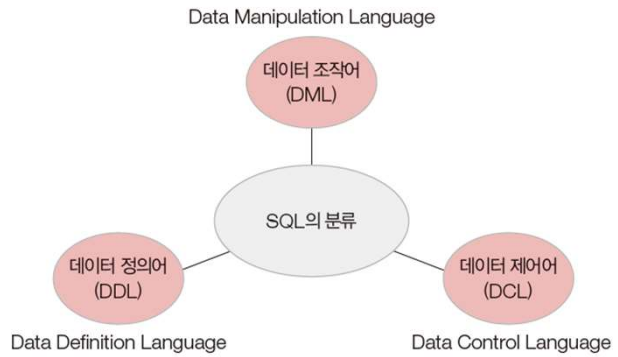
1-1 SQL 분류



1/32

1-1 SQL 분류

고객 테이블

고객아이디	고객이름	나이	등급	직업	적립금
apple	정소화	20	gold	학생	1000
banana	김선우	25	vip	간호사	2500
carrot	고명석	28	gold	교사	4500
orange	김용욱	22	silver	학생	0
melon	성원용	35	gold	회사원	5000
peach	오형준	NULL	silver	의사	300
pear	채광주	31	silver	회사원	500

2/32

1-1 SQL 분류

제품 테이블

제품번호	제품명	재고량	단가	제조업체
p01	그냥만두	5000	4500	대한식품
p02	매운쫄면	2500	5500	민국푸드
p03	쿵떡파이	3600	2600	한빛제과
p04	맛난초콜릿	1250	2500	한빛제과
p05	얼큰라면	2200	1200	대한식품
p06	통통우동	1000	1550	민국푸드
p07	달콤비스킷	1650	1500	한빛제과

3/32

1-1 SQL 분류

주문 테이블

주문번호	주문고객	주문제품	수량	배송지	주문일자
o01	apple	p03	10	서울시 마포구	2019-01-01
o02	melon	p01	5	인천시 계양구	2019-01-10
o03	banana	p06	45	경기도 부천시	2019-01-11
o04	carrot	p02	8	부산시 금정구	2019-02-01
o05	melon	p06	36	경기도 용인시	2019-02-20
o06	banana	p01	19	충청북도 보은군	2019-03-02
o07	apple	p03	22	서울시 영등포구	2019-03-15
o08	pear	p02	50	강원도 춘천시	2019-04-10
o09	banana	p04	15	전라남도 목포시	2019-04-11
o10	carrot	p03	20	경기도 안양시	2019-05-22

4/32

1-1 SQL 분류

❖ **SQL의 데이터 정의 기능**

– 테이블 생성, 변경, 삭제

1-1 SQL 분류

❖ **테이블 생성 : CREATE TABLE 문**

```
CREATE TABLE 테이블_이름 (
        ❶ 속성_이름  데이터_타입 [NOT NULL] [DEFAULT 기본_값]
        ❷ [PRIMARY KEY (속성_리스트)]
        ❸ [UNIQUE (속성_리스트)]
        ❹ [FOREIGN KEY (속성_리스트) REFERENCES 테이블_이름(속성_리스트)]
          [ON DELETE 옵션] [ON UPDATE 옵션]
        ❺ [CONSTRAINT 이름] [CHECK(조건)]
);
```

❖ [ ]의 내용은 생략이 가능

❖ SQL 질의문은 세미콜론(;)으로 문장의 끝을 표시

❖ SQL 질의문은 대소문자를 구분하지 않음

1-1 SQL 분류

❖ **테이블 변경 : ALTER TABLE 문**

– 새로운 속성 추가

```
ALTER  TABLE  테이블_이름
   ADD  속성_이름  데이터_타입  [NOT NULL]  [DEFAULT 기본_값];
```


1-1 SQL 분류

❖ **테이블 변경 : ALTER TABLE 문**

– 기존 속성 삭제

```
ALTER TABLE  테이블_이름  DROP COLUMN 속성_이름;
```

- *만약, 삭제할 속성과 관련된 제약조건이 존재한다면?*
  - 속성 삭제가 수행되지 않음
  - 관련된 제약조건을 먼저 삭제해야 함

1-1 SQL 분류

❖ **테이블 변경 : ALTER TABLE 문**

– 새로운 제약조건의 추가

```
ALTER TABLE  테이블_이름  ADD  CONSTRAINT  제약조건_이름  제약조건_내용;
```

9/32

1-1 SQL 분류

❖ 테이블 변경 : ALTER TABLE 문

– 기존 제약조건의 삭제

```
ALTER TABLE  테이블_이름  DROP  CONSTRAINT  제약조건_이름;
```

10/32

1-1 SQL 분류

❖ **테이블 삭제 : DROP TABLE 문**

```
DROP TABLE  테이블_이름;
```

- *만약, 삭제할 테이블을 참조하는 테이블이 있다면?*
  - 테이블 삭제가 수행되지 않음
  - 관련된 외래키 제약조건을 먼저 삭제해야 함

11/32

1-2 데이터베이스 형식의 종류

- **숫자 데이터 형식**
  - DECIMAL 형식은 정확한 수치를 저장하고 FLOAT, REAL 형식은 근사치를 저장
  - 소수점이 있는 실수는 되도록 DECIMAL 형식을 사용하여 저장하는 것이 바람직
  - 예를 들어 -999999.99~999999.99 범위의 숫자를 저장할 때는 DECIMAL(9,2)로 설정
  - 어떤 숫자를 부호 없는 정수로 지정하면
    - TINYINT는 0~255, SMALLINT는 0~65535,
    - MEDIUMINT는 0~16777215,
    - INT는 0~약 42억,
    - BIGINT는 0~약 1800경으로 표현할 수 있음
  - 부호 없는 정수를 지정할 때는 뒤에 UNSIGNED 예약어를 붙임

12/32

1-2 데이터베이스 형식의 종류

데이터 형식	바이트 수	숫자 범위	설명
BIT(N)	N/8		• 1~64bit 표현 • b'0000' 형식으로 저장
TINYINT	1	-128~127	• 정수 저장
BOOL BOOLEAN	1	-128~127	• 정수 저장 • TINYINT(1)과 동일 • 0은 false로, 그 외는 true로 취급
SMALLINT	2	-32768~32767	• 정수 저장
MEDIUMINT	3	-8388608~8388607	• 정수 저장
INT INTEGER	4	약 -21억~21억	• 정수 저장
BIGINT	8	약 -900경~900경	• 정수 저장
FLOAT	4	-3.40E+38~-1.17E-38	• 소수점 이하 7자리까지 저장
DOUBLE REAL	8	-1.22E-308~1.79E+308	• 소수점 이하 15자리까지 저장
DECIMAL(m,[d]) DEC(m,[d]) FIXED(m,[d]) NUMERIC(m,[d])	5~17	-1038+1~1038-1	• 전체 자릿수(m)와 소수점 이하 자릿수(d)를 가진 숫자 저장 예: DECIMAL(5,2)는 전체 자릿수를 5자리로 하되, 그중 소수점 이하를 2자리로 하겠다는 뜻

13/32

1-2 데이터베이스 형식의 종류

■ 문자 데이터 형식

- CHAR 형식은 고정 길이 문자형을 저장하고 자릿수가 고정되어 있음
- VARCHAR 형식은 가변 길이 문자형을 저장
- BINARY와 VARBINARY 형식은 바이트 단위의 이진 데이터 값을 저장
- TEXT 형식은 대용량 글자를 저장하기 위한 형식으로, 필요한 크기에 따라서 TINYTEXT, TEXT, MEDIUMTEXT, LONGTEXT 등의 형식을 사용할 수 있음
- BLOB(Binary Large OBject) 형식은 사진, 동영상, 문서 파일 등의 대용량 이진 데이터를 저장
- ENUM 형식은 열거형 데이터를 저장하는 데 사용
- SET 형식은 최대 64개의 데이터를 2개씩 세트로 묶어서 저장할 때 사용

14/32

1-2 데이터베이스 형식의 종류

데이터 형식		바이트 수	설명
CHAR(n)		1~255	• 고정 길이 문자형 저장(character의 약자) • n을 1~255까지 지정 • CHAR만 쓰면 CHAR(1)과 동일
VARCHAR(n)		1~65535	• 가변 길이 문자형 저장(variable character의 약자) • n을 1~65535까지 지정
BINARY(n)		1~255	• 고정 길이의 이진 데이터 값 저장
VARBINARY(n)		1~255	• 가변 길이의 이진 데이터 값 저장
TEXT 형식	TINYTEXT	1~255	• 255 크기의 TEXT 데이터 값 저장
	TEXT	1~65535	• N 크기의 TEXT 데이터 값 저장
	MEDIUMTEXT	1~16777215	• 16777215 크기의 TEXT 데이터 값 저장
	LONGTEXT	1~4294967295	• 최대 4GB 크기의 TEXT 데이터 값 저장
BLOB 형식	TINYBLOB	1~255	• 255 크기의 BLOB 데이터 값 저장
	BLOB	1~65535	• N 크기의 BLOB 데이터 값 저장
	MEDIUMBLOB	1~16777215	• 16777215 크기의 BLOB 데이터 값 저장
	LONGBLOB	1~4294967295	• 최대 4GB 크기의 BLOB 데이터 값 저장
ENUM(값들 …)		1 또는 2	• 최대 65535개의 열거형 데이터 값 저장
SET(값들 …)		1, 2, 3, 4, 8	• 최대 64개의 서로 다른 데이터 값 저장

15/32

1-2 데이터베이스 형식의 종류

■ 날짜와 시간 데이터 형식

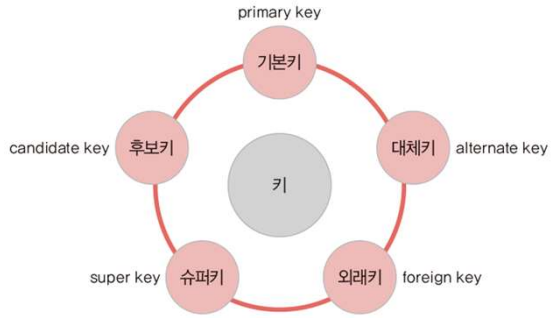
데이터 형식	바이트 수	설명
DATE	3	• 'YYYY-MM-DD' 형식으로 날짜 저장 • 저장 범위는 1001-01-01~9999-12-31
TIME	3	• 'HH:MM:SS' 형식으로 시간 저장 • 저장 범위는 -838:59:59.000000~838:59:59.000000
DATETIME	8	• 'YYYY-MM-DD HH:MM:SS' 형식으로 날짜와 시간 저장 • 저장 범위는 1001-01-01 00:00:00~9999-12-31 23:59:59
TIMESTAMP	4	• 'YYYY-MM-DD HH:MM:SS' 형식으로 날짜와 시간 저장 • 저장 범위는 1001-01-01 00:00:00~9999-12-31 23:59:59 • time_zone 시스템 변수와 관련이 있으며 UTC 시간대로 변환하여 저장
YEAR	1	• 'YYYY' 형식으로 연도 저장 • 저장 범위는 1901~2155

16/32

1-3 제약조건

❖ **키(key)**

▪ 릴레이션에서 투플들을 유일하게 구별하는 속성 또는 속성들의 집합



17/32

1-3 제약조건

- 키의 특성
- 데이터베이스에서 조건에 만족하는 레코드를 찾거나 순서대로 정렬할 때 각 레코드들을 서로 구분할 수 있는 기준이 되는 속성
  - 유일성(uniqueness)
    - 하나의 테이블에서 모든 레코드는 서로 다른 키 값을 가져야 함
  - 최소성(minimality)
    - 꼭 필요한 최소한의 속성들로만 키를 구성함

18/32

1-3 제약조건

▪ **primary key**

▪ 어떤 테이블에서 **특정 레코드를 유일하게 구별**할 수 있는 속성

기본키

고객아이디	고객이름	나이	등급	직업	적립금	주소
apple	김현준	20	gold	학생	1000	서울시 구로구 고척로 27
banana	정소화	25	vip	간호사	2500	부천시 원미구 석천로 58
carrot	원유선	28	gold	교사	4500	서울시 영등포구 대림로 73
orange	정지영	22	silver	학생	0	서울시 마포구 토정로 56

19/32

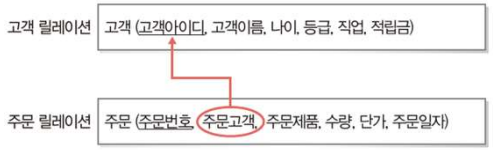
1-3 제약조건

- **unique key**
  - 대체키를 지정하는 키워드
  - 대체키로 지정되는 속성의 값은 유일성을 가지며 기본키와 달리 NULL 값이 허용됨

20/32

1-3 제약조건

- **foreign key**
  - 다른 테이블의 기본키를 참조하는 속성 또는 속성들의 집합
  - 테이블들 간의 관계를 표현
    - 참조하는 테이블 : 외래키를 가진 테이블
    - 참조되는 테이블 : 외래키가 참조하는 기본키를 가진 테이블



1-3 제약조건

고객 릴레이션

고객 릴레이션의 기본키

고객아이디	고객이름	나이	등급	직업	적립금
apple	김현준	20	gold	학생	1000
banana	정소화	25	vip	간호사	2500
carrot	원유선	28	gold	교사	4500
orange	정지영	22	silver	학생	0

주문 릴레이션

주문 릴레이션의 기본키

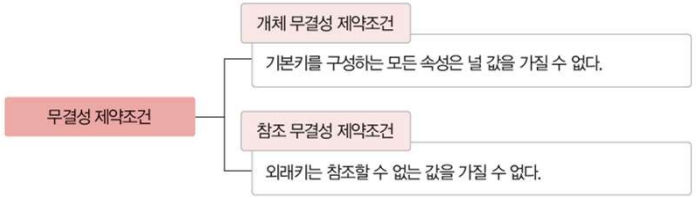
주문번호	주문고객	주문제품	수량	단가	주문일자
1001	apple	진짜우동	10	2000	2019-01-01
1002	carrot	맛있는파이	5	500	2019-01-10
1003	banana	그대로만두	11	4500	2019-01-11

주문 릴레이션의 외래키

외래키 속성과 그것이 참조하는 기본키 속성의 이름은 달라도 되지만 도메인은 같아야 한다.

1-3 제약조건

- **무결성 제약조건(integrity constraint)**
  - 데이터의 무결성을 보장하고 일관된 상태로 유지하기 위한 규칙
  - 무결성 : 데이터를 결함이 없는 상태, 즉 정확하고 유효하게 유지하는 것



1-3 제약조건

- **기본키는 NULL을 가질 수 없다!**

고객아이디	고객이름	나이	등급	직업	적립금
apple	김현준	20	gold	학생	1000
NULL	정소화	25	vip	간호사	2500
carrot	원유선	28	gold	교사	4500
NULL	정지영	22	silver	학생	0

1-3 제약조건

■ **참조할 수 없는 외래키는 가질수 없다!**

고객 릴레이션

고객아이디	고객이름	나이	등급	직업	적립금
apple	김현준	20	gold	학생	1000
banana	정소화	25	vip	간호사	2500
carrot	원유선	28	gold	교사	4500
orange	정지영	22	silver	학생	0

고객 릴레이션의 기본키

주문 릴레이션

주문번호	주문고객	주문제품	수량	단가	주문일자
1001	cherry	진짜우동	10	2000	2019-01-01
1002	carrot	맛있는파이	5	500	2019-01-10
1003	banana	그대로만두	11	4500	2019-01-11

주문 릴레이션의 기본키

주문 릴레이션의 외래키

1-3 제약조건

■ NOT NULL

- 속성이 널 값을 허용하지 않음을 의미하는 키워드
- 예) 고객아이디 VARCHAR(20) NOT NULL

■ DEFAULT

- 속성의 기본 값을 지정하는 키워드
- 예) 적립금 INT DEFAULT 0
- 예) 담당자 VARCHAR(10) DEFAULT '홍길동'

1-3 제약조건

■ **CHECK**

- 테이블에 정확하고 유효한 데이터를 유지하기 위해 특정 속성에 대한 제약조건을 지정
- CONSTRAINT 키워드와 함께 고유의 이름을 부여할 수도 있음
- 예) CHECK(재고량 >= 0 AND 재고량 <= 10000)
- 예) CONSTRAINT CHK_CPY CHECK(제조업체 = '한빛제과')

1-4 데이터 조작

- **데이터 삽입 : INSERT 문**

```
INSERT
INTO    테이블_이름[(속성_리스트)]
VALUES (속성값_리스트);
```

– INTO 절에는 데이터를 삽입할 테이블의 이름과 속성의 이름을 나열
  - 속성 리스트를 생략하면 테이블을 정의할 때 지정한 속성의 순서대로 값이 삽입됨

– VALUES 키워드와 함께 삽입할 속성 값들을 나열

– INTO 절의 속성 이름과 VALUES 절의 속성 값은 순서대로 일대일 대응되어야 함

1-4 데이터 조작

판매 데이터베이스의 고객 테이블에 고객아이디가 strawberry, 고객이름이 최유경, 나이가 30세, 등급이 vip, 직업이 공무원, 적립금이 100원인 새로운 고객의 정보를 삽입해보자. 그런 다음 고객 테이블에 있는 모든 내용을 검색하여 삽입된 새로운 투플을 확인해보자.

```
INSERT
INTO      고객(고객아이디, 고객이름, 나이, 등급, 직업, 적립금)
VALUES    ('strawberry', '최유경', 30, 'vip', '공무원', 100);
```


1-4 데이터 조작

```
INSERT
INTO    고객(고객아이디, 고객이름, 나이, 등급, 직업, 적립금)
VALUES  ('strawberry', '최유경', 30, 'vip', '공무원', 100);
```

=

```
INSERT
INTO    고객
VALUES  ('strawberry', '최유경', 30, 'vip', '공무원', 100);
```


1-4 데이터 조작

```
INSERT
INTO    고객(고객아이디, 고객이름, 나이, 등급, 적립금)
VALUES  ('tomato', '정은심', 36, 'gold', 4000);
```

```
INSERT
INTO    고객
VALUES  ('tomato', '정은심', 36, 'gold', NULL, 4000);
```


1-4 데이터 조작

- **데이터 삽입 : INSERT 문**
  - 부속 질의문을 이용한 데이터 삽입
    - SELECT 문을 이용해 다른 테이블에서 검색한 데이터를 삽입

```
INSERT
INTO  테이블_이름[(속성_리스트)]
SELECT 문;
```

```
예) INSERT
    INTO      한빛제품(제품명, 재고량, 단가)
    SELECT    제품명, 재고량, 단가
    FROM      제품
    WHERE     제조업체 = '한빛제과';
```