

[7일 완성] 생각하는 데이터베이스 모델링

(생각하는DB1탄)

Data Modeling #1 - 데이터베이스 요구사항분석

Data Modeling #2 - Conceptual Data Modeling(ERD)

Data Modeling #3 - Logical Data Modeling(RM)

Data Modeling #4 - Physical Data Modeling

Data Modeling #5 - SQL(Structured Query Language)

Data Modeling #6 - 데이터모델링 과제 수행

Data Modeling #7 - 생각하는 데이터베이스 모델링 종합시험

박 매일 강사

생각하는 데이터베이스 모델링

단계 번호	단계 이름	설명
1	데이터베이스 요구 사항 분석	데이터베이스 시스템의 요구 사항과 요구 사항을 이해합니다.
2	개념적 데이터 모델링(ERD)	높은 수준의 데이터 구조를 시각화하기 위해 엔터티-관계 다이어그램(ERD) 생성.
3	논리적 데이터 모델링(RM)	데이터 관계 및 제약 조건을 나타내는 관계형 모델 개발.
4	물리적 데이터 모델링	특정 데이터베이스 관리 시스템(DBMS)에 대한 실제 데이터베이스 스키마 설계
5	구조적 쿼리 언어(SQL)	데이터베이스와 상호 작용하고, 데이터를 쿼리하고, CRUD 작업을 수행하기 위해 SQL을 학습합니다.
6	데이터 모델링 작업 수행	실제 시나리오를 사용하여 데이터 모델링 개념을 구현하기 위한 실용적인 실습.
7	생각하는 데이터베이스 모델링 종합시험	데이터베이스 모델링 개념의 이해와 적용을 테스트하기 위한 종합 시험입니다.

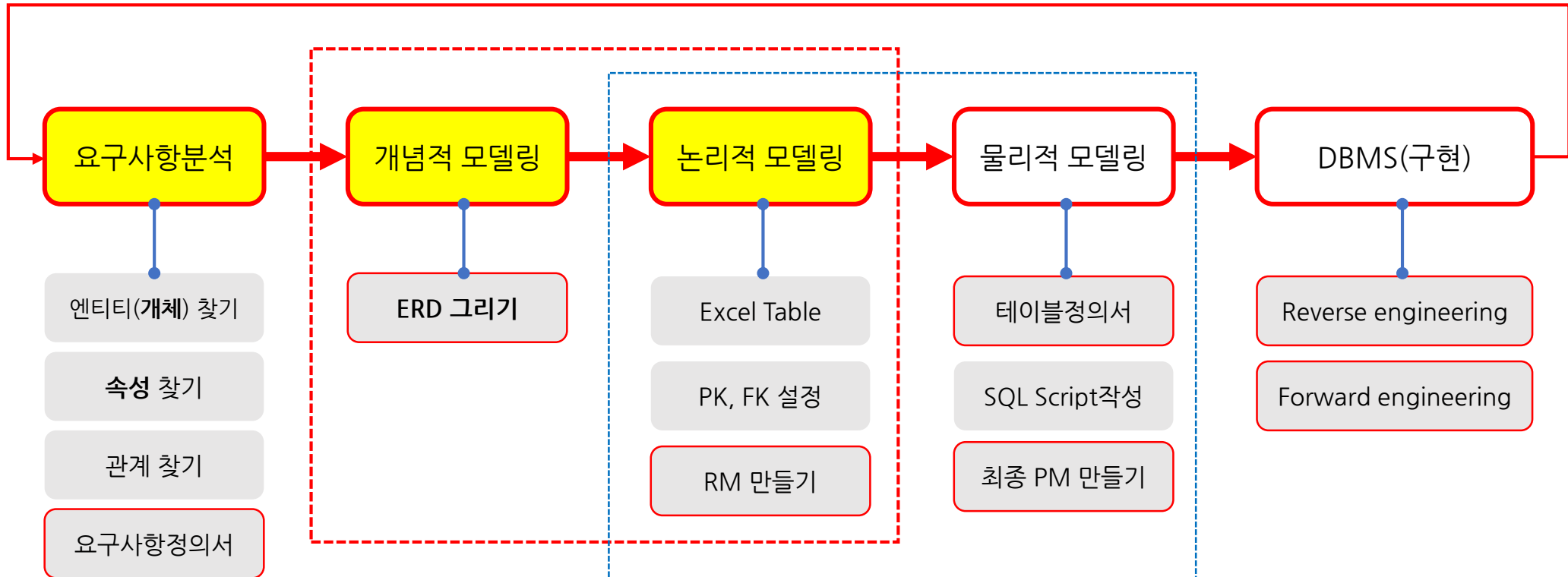
[7일 완성]생각하는 데이터베이스 모델링

Data Modeling #1 - 데이터베이스 요구사항분석

온라인 전자상거래 플랫폼 개발 요구사항

● 데이터 모델링(Data Modeling)

- 데이터 모델링이란?
 - 현실 세계에 존재하는 데이터를 컴퓨터 세계의 데이터 베이스로 옮기는 변환 과정
- 데이터 모델링 프로세스



- 온라인 전자상거래 플랫폼 개발 요구사항 분석
 - 목적
 - 사용자의 요구 사항을 수집하고 분석하여 개발할 데이터베이스의 용도를 파악
 - 업무에 필요한 데이터가 무엇인지, 그 데이터에 어떤 처리가 필요한지 등을 고려
 - 결과물
 - 요구 사항 명세서 (요구사항 정의서)
 - 주요 작업
 - 데이터베이스를 실제로 사용할 주요 **사용자의 범위를 결정**
 - 사용자가 조직에서 수행하는 **업무를 분석**
 - 면담, 설문 조사, 업무 관련 문서 분석 등의 방법을 이용해 요구 사항 수집
 - 수집된 요구 사항에 대한 분석 결과를 요구 사항 명세서로 작성

- 온라인 전자상거래 플랫폼 개발 요구사항

- 온라인 전자상거래 플랫폼 개발 요구사항

고객은 고객코드, 고객명, 전화번호, 이메일, 주소(기본주소, 상세주소), 지역, 가입일로 되어있다

고객은 지역별로 관리되도록 한다.

지역은 지역코드와 지역명으로 되어 있고 지역명은 대한민국의 지역코드(02:서울특별시)를 이용한다.

한 지역에는 여러 고객이 있을 수 있다.

제품은 제품코드, 제품명, 제품색상, 가격으로 되어있다.

하나의 제품은 여러 색상을 가질 수가 있다.

고객은 등록된 제품을 구매 할 수 있다.

한 명의 고객은 여러 제품을 구매 할 수 있고, 하나의 제품은 여러 고객이 구매 할 수 있다.

고객이 제품을 구매 시 구매수량과 구매일자를 기록한다.

- 온라인 전자상거래 플랫폼 개발 요구사항

- 요구사항 분석을 통한 데이터 찾기

- 엔티티 찾기(E) -> 속성 찾기(A) -> 관계 찾기(R) : 처음에는 이렇게

- 엔티티 찾기(E) -> 관계 찾기(R) -> 속성 찾기(A) : 숙달되면 이렇게

①고객은 고객코드, 고객명, 전화번호, 이메일, 주소(기본주소, 상세주소), 지역, 가입일로 되어있다

②고객은 지역별로 관리되도록 한다.

③지역은 지역코드와 지역명으로 되어 있고 지역명은 대한민국의 지역코드(02:서울특별시)를 이용한다.

④한 지역에는 여러 고객이 있을 수 있다.

⑤제품은 제품코드, 제품명, 제품구분, 제품색상, 가격으로 되어있다.

⑥하나의 제품은 여러 색상을 가질 수가 있다.

⑦고객은 등록된 제품을 구매 할 수 있다.

⑧한 명의 고객은 여러 제품을 구매 할 수 있고, 하나의 제품은 여러 고객이 구매 할 수 있다.

⑨고객이 제품을 구매 시 구매수량과 구매일자를 기록한다.

요구사항정의서

객체관계정의서

고객 : 고객코드, 고객명, 전화번호, 이메일,
주소(기본주소, 상세주소), 지역, 가입일
지역 : 지역코드, 지역명
제품 : 제품코드, 제품명, 제품구분, 제품색상, 가격
구매 : 고객은 제품을 구매(구매수량, 구매일자)
관리 : 고객은 지역별로 관리

● 온라인 전자상거래 플랫폼 개발 요구사항(산출물 예시)

■ 1. 요구사항 정의서

- ①고객은 고객코드, 고객명, 전화번호, 이메일, 주소(기본주소, 상세주소), 지역, 가입일로 되어있다
- ②고객은 지역별로 관리되도록 한다.
- ③지역은 지역코드와 지역명으로 되어 있고 지역명은 대한민국의 지역코드(02:서울특별시)를 이용한다.
- ④한 지역에는 여러 고객이 있을 수 있다.
- ⑤제품은 제품코드, 제품명, 제품구분, 제품색상, 가격으로 되어있다.
- ⑥하나의 제품은 여러 색상을 가질 수가 있다.
- ⑦고객은 등록된 제품을 구매 할 수 있다.
- ⑧한 명의 고객은 여러 제품을 구매 할 수 있고, 하 나의 제품은 여러 고객이 구매 할 수 있다.
- ⑨고객이 제품을 구매 시 구매수량과 구매일자를 기록한다.

■ 2. 객체 관계 정의서

객체 명	속성 명
고객	고객코드, 고객명, 전화번호, 이메일, 주소(기본주소, 상세주소), 지역, 가입일
지역	지역코드, 지역 명
제품	제품코드, 제품명, 제품구분, 제품색상, 가격

관계	참여객체	관계유형	속성
관리	고객,지역	1:N	
구매	고객,제품	N:M	구매수량, 구매일자

[7일 완성]생각하는 데이터베이스 모델링

Data Modeling #2 - Conceptual Data Modeling(ERD)

온라인 전자상거래 플랫폼 ERD 그리기

- Conceptual Data Modeling(개념적 모델링)

- 목적

- DBMS에 독립적인 개념적 스키마 설계
 - 요구 사항 분석 결과물을 개념적 데이터 모델을 이용해 개념적 구조로 표현

- 개념적 모델링

- 엔터티-관계 다이어그램(ERD) :

- ERD는 엔터티(개체), 해당 속성(속성) 및 엔터티 간의 관계를 시각적으로 나타냄

- 결과물

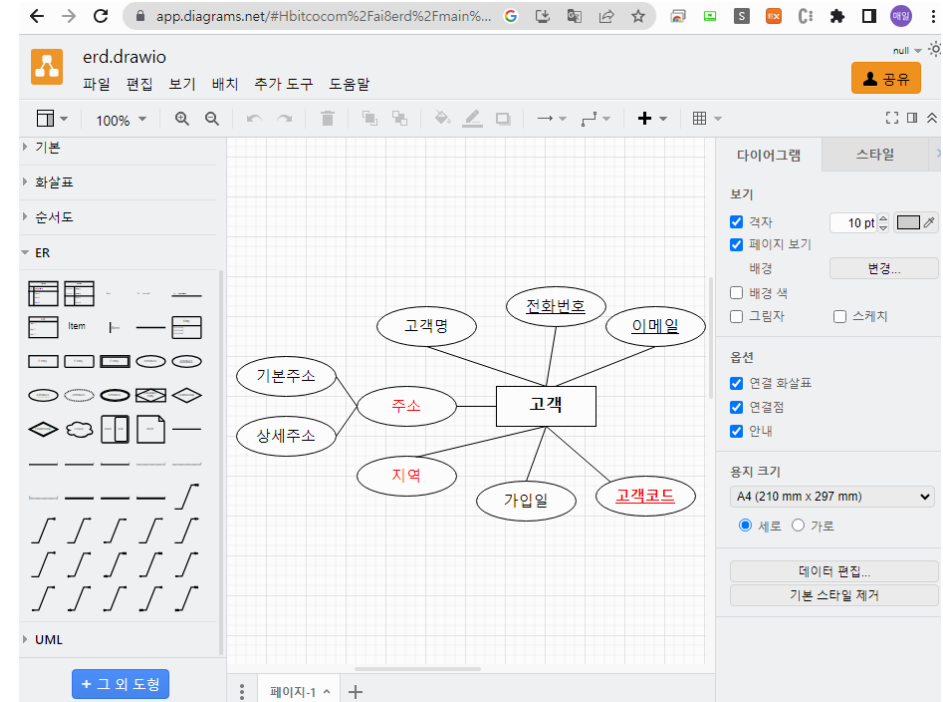
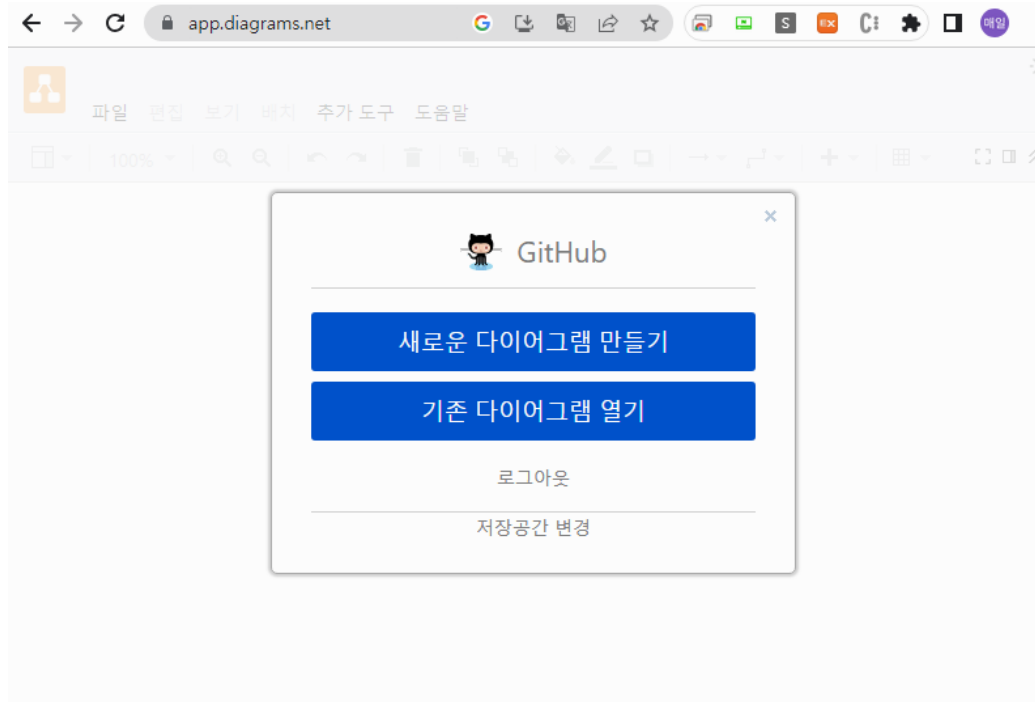
- 개념적 스키마 : E-R 다이어그램

- 주요 작업

- 요구 사항 분석 결과를 기반으로 중요한 개체를 추출하고 개체 간의 관계를 결정하여 E-R 다이어그램으로 표현

Data Modeling #2 - Conceptual Data Modeling(ERD)

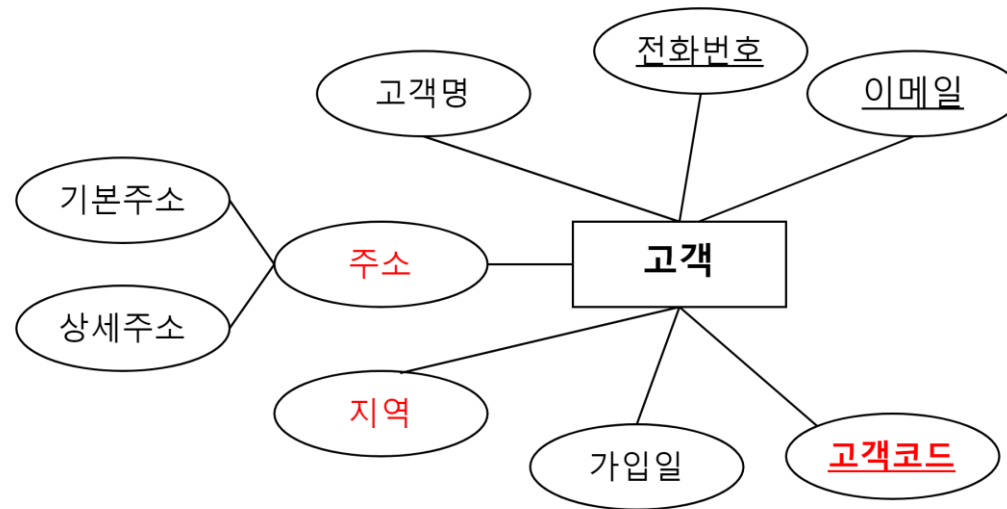
- 온라인 전자상거래 플랫폼 ERD 그리기
 - drawio.io에 접속하여 ERD를 그려볼 수 있다.
 - drawio.io에서 자신의 깃 허브(github)에 만들어진 저장소와 연결한다.



- 온라인 전자상거래 플랫폼 ERD 그리기

- 키 속성 (Key Attributes)

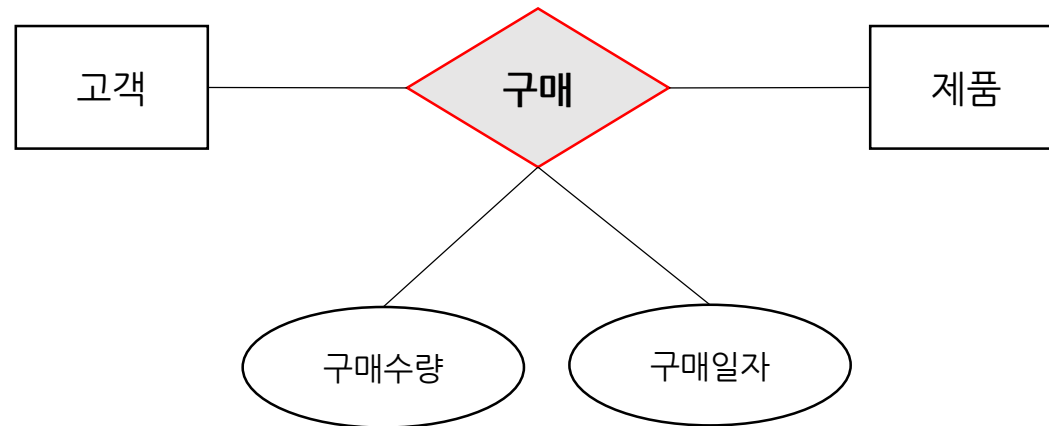
- 각 객체 인스턴스를 식별하는 데 사용되는 속성
 - 모든 객체 인스턴스의 키 속성 값이 다름
 - 둘 이상의 속성들로 구성되기도 함
 - 예) 고객 개체의 고객코드 속성
 - ERD에서 **밑줄**로 표현



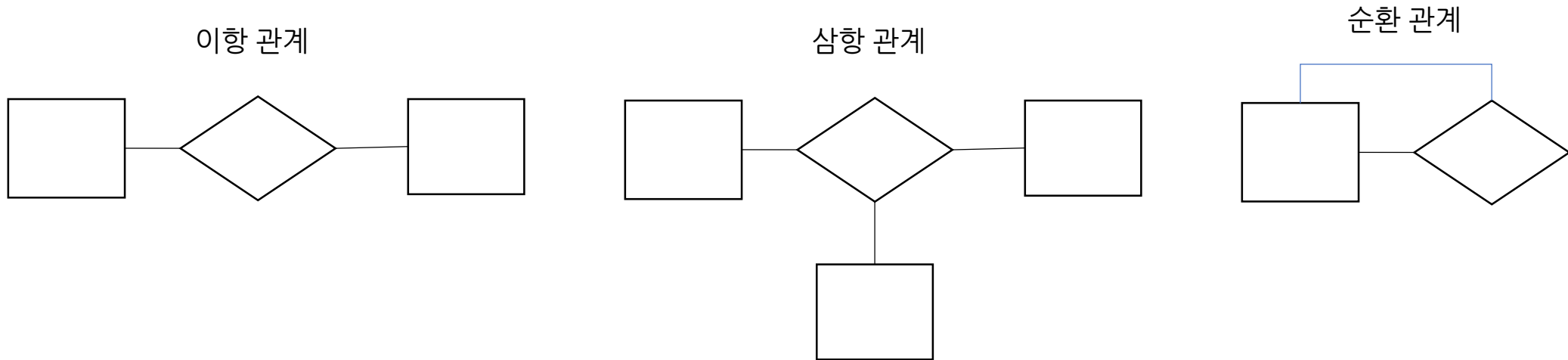
- 온라인 전자상거래 플랫폼 ERD 그리기

- **관계(Relationship)**

- 개체와 개체가 맺고 있는 의미 있는 연관성
 - 개체 집합들 사이의 대응 관계, 즉 매핑(mapping)을 의미
 - 예) 고객 개체와 제품 개체 간의 구매 관계
+ 고객은 제품을 구매한다.
 - ERD에서 **마름모**로 표현



- 온라인 전자상거래 플랫폼 ERD 그리기
 - **관계의 유형 : 관계에 참여하는 개체 타입의 수 기준**
 - 이항 관계 : 개체 타입 두 개가 맺는 관계
 - 삼항 관계 : 개체 타입이 세 개가 맺는 관계
 - 순환 관계 : 개체 타입 하나가 자기 자신과 맺는 관계



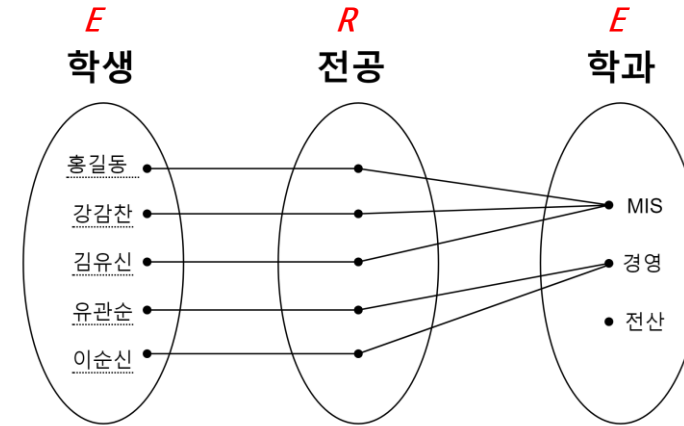
Data Modeling #2 - Conceptual Data Modeling(ERD)

● 온라인 전자상거래 플랫폼 ERD 그리기

▪ 대응 수에 따른 관계의 분류(관계유형)

- 일대일(1:1) 관계
- 일대다(1:N)관계
- 다대다(N:M)관계

학생은 한 학과에 연결된다.
학과는 여러 학생이 연결된다.



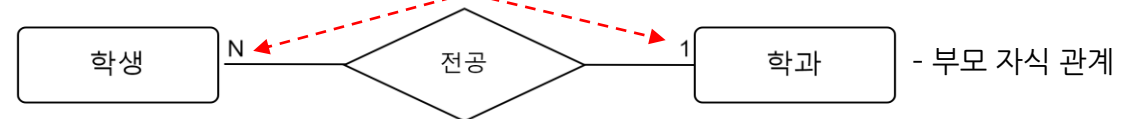
▪ 매핑 카디널리티(Cardinality, 대응수)

- 관계를 맺는 구 개체 집합에서,
각 객체 인스턴스가 연관성을 맺고 있는
상대 객체 집합의 인스턴스 개수

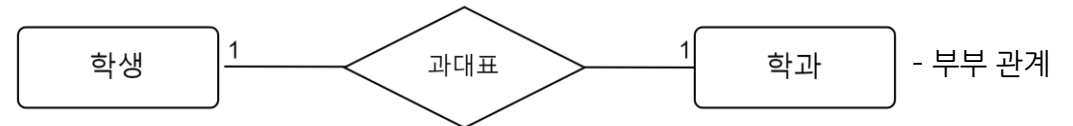
*하나의 학생은 몇 개의 학과에 연결되는가?

*하나의 학과는 몇 명의 학생과 연결되는가?

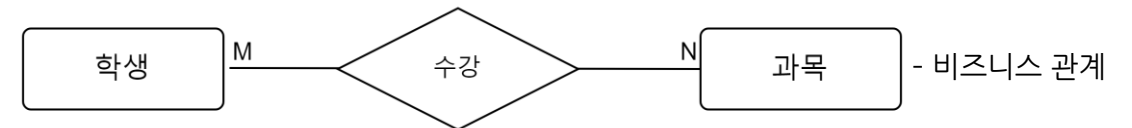
- 일대다 (1:N) 관계



- 일대일 (1:1) 관계



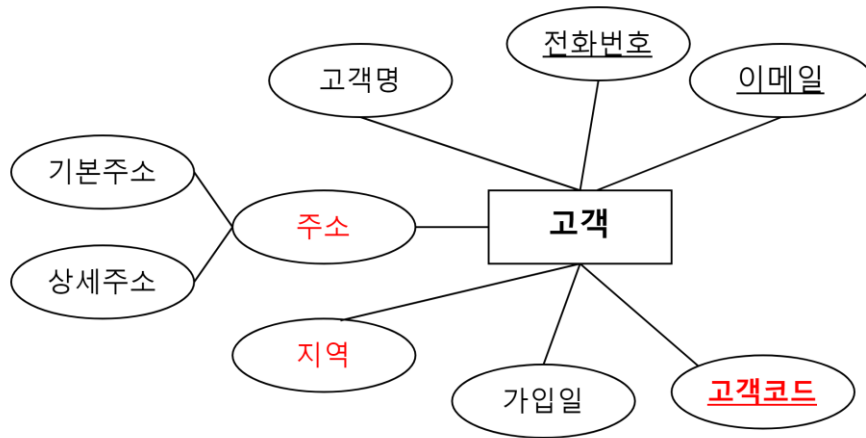
- 다대다 (M:N) 관계



Data Modeling #2 - Conceptual Data Modeling(ERD)

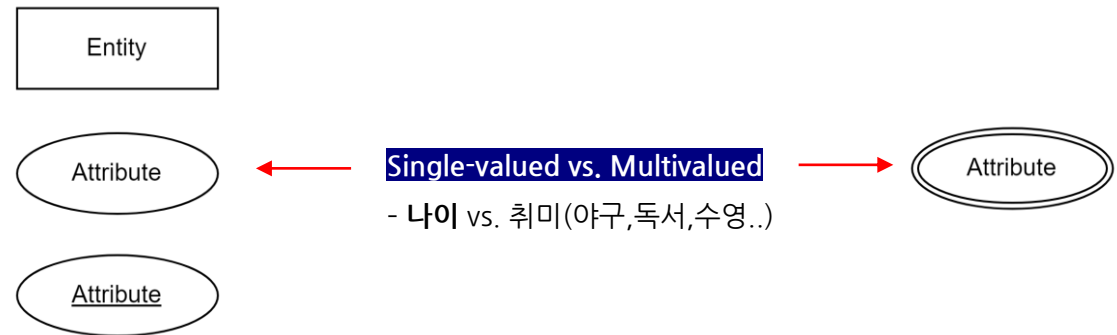
- 온라인 전자상거래 플랫폼 ERD 그리기
 - 엔티티 속성 중 **후 보키, 기본 키, 복합속성, 다중속성**을 유심히 찾아본다.
 - 고객 엔티티에서 찾아볼 수 있는 속성들

ERD란 Entity Relationship Diagram의 약어로, 데이터베이스 구조를 한눈에 알아보기 위해서 쓰인다. DB를 개발하기 전에 보다 많은 아이디어를 도출하고, 데이터베이스 설계의 이해를 높이기 위해 사용된다



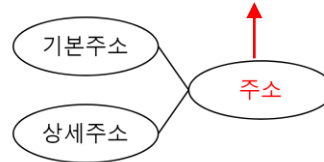
다른 Entity에 연결될 수 있다.

지역코드	지역명
02	서울특별시
062	광주광역시
064	제주특별자치도



Single-valued vs. Multivalued
- 나이 vs. 취미(야구, 독서, 수영..)

Composite Attributes(복합속성)

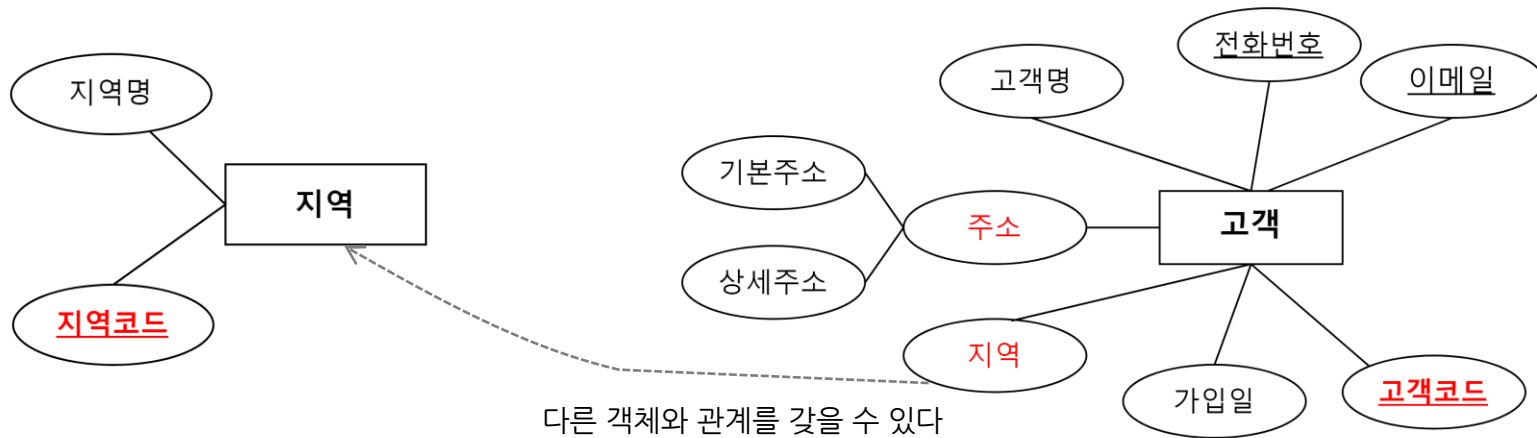


Simple vs. Composite

- Simple Attribute - 더 이상 쪼개지지 않는 원자 값을 갖는 속성
 - 나이, 학번, ...
- Composite Attribute - 몇 개의 요소로 분해될 수 있는 속성
 - 주소 -> 시, 군, 구, 번지, ...

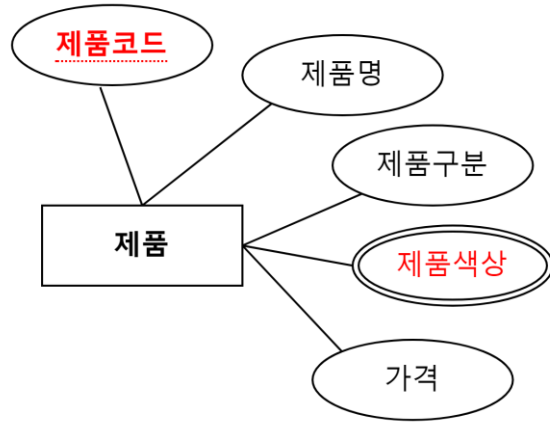
Data Modeling #2 - Conceptual Data Modeling(ERD)

- 온라인 전자상거래 플랫폼 ERD 그리기
 - 엔티티 속성 중 후 보기, 기본 키, 복합속성, 다중속성을 유심히 찾아본다.
 - 지역 엔티티에서 찾아볼 수 있는 속성들



Data Modeling #2 - Conceptual Data Modeling(ERD)

- 온라인 전자상거래 플랫폼 ERD 그리기
 - 엔티티 속성 중 후 보키, 기본 키, 복합속성, 다중속성을 유심히 찾아본다.
 - 제품 엔티티에서 찾아볼 수 있는 속성들

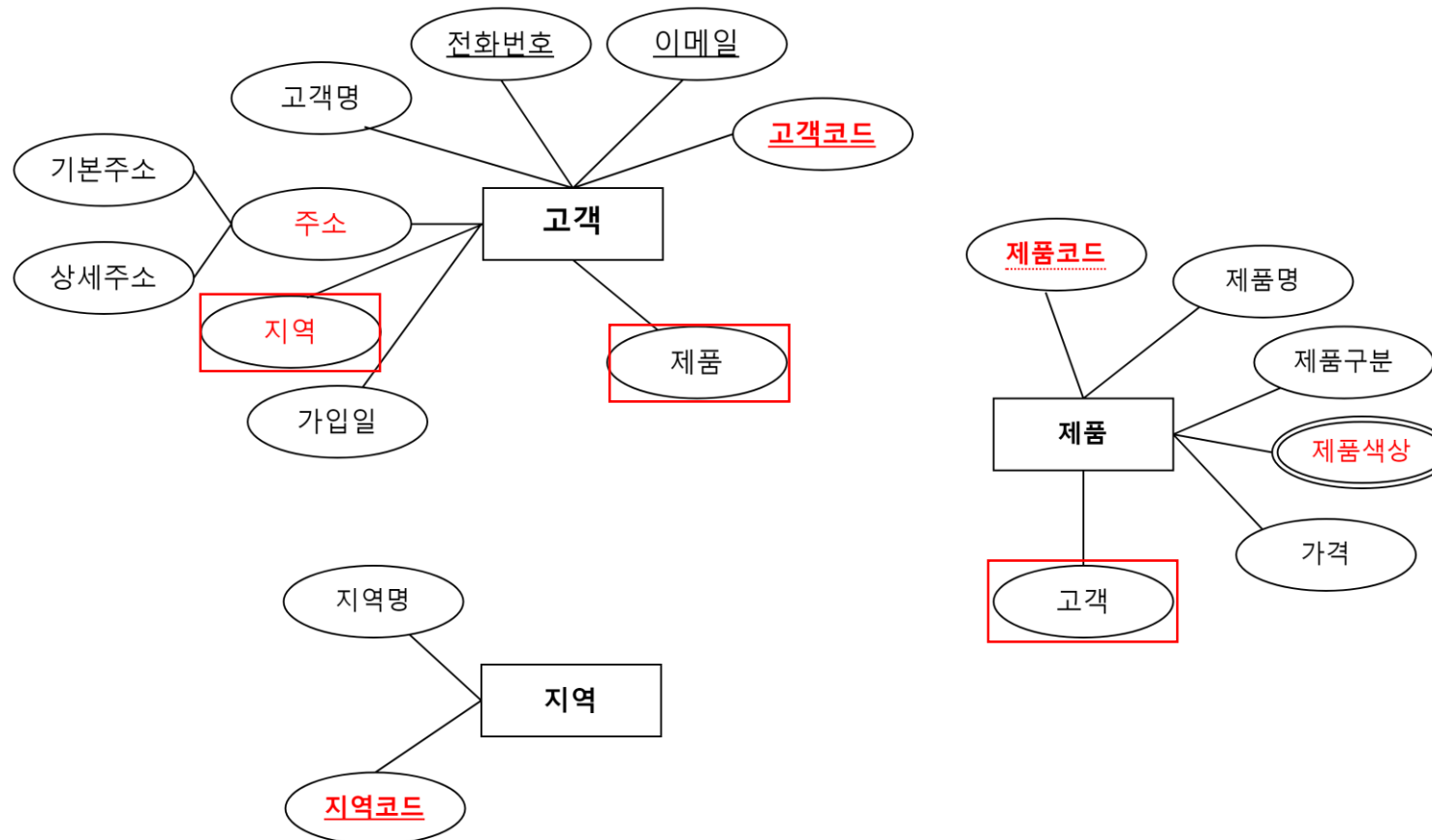


Composite Attribute - 몇 개의 요소로 분해될 수 있는 속성

제품색상
흰색, 빨강
녹색, 검정
파랑, 노랑

Data Modeling #2 - Conceptual Data Modeling(ERD)

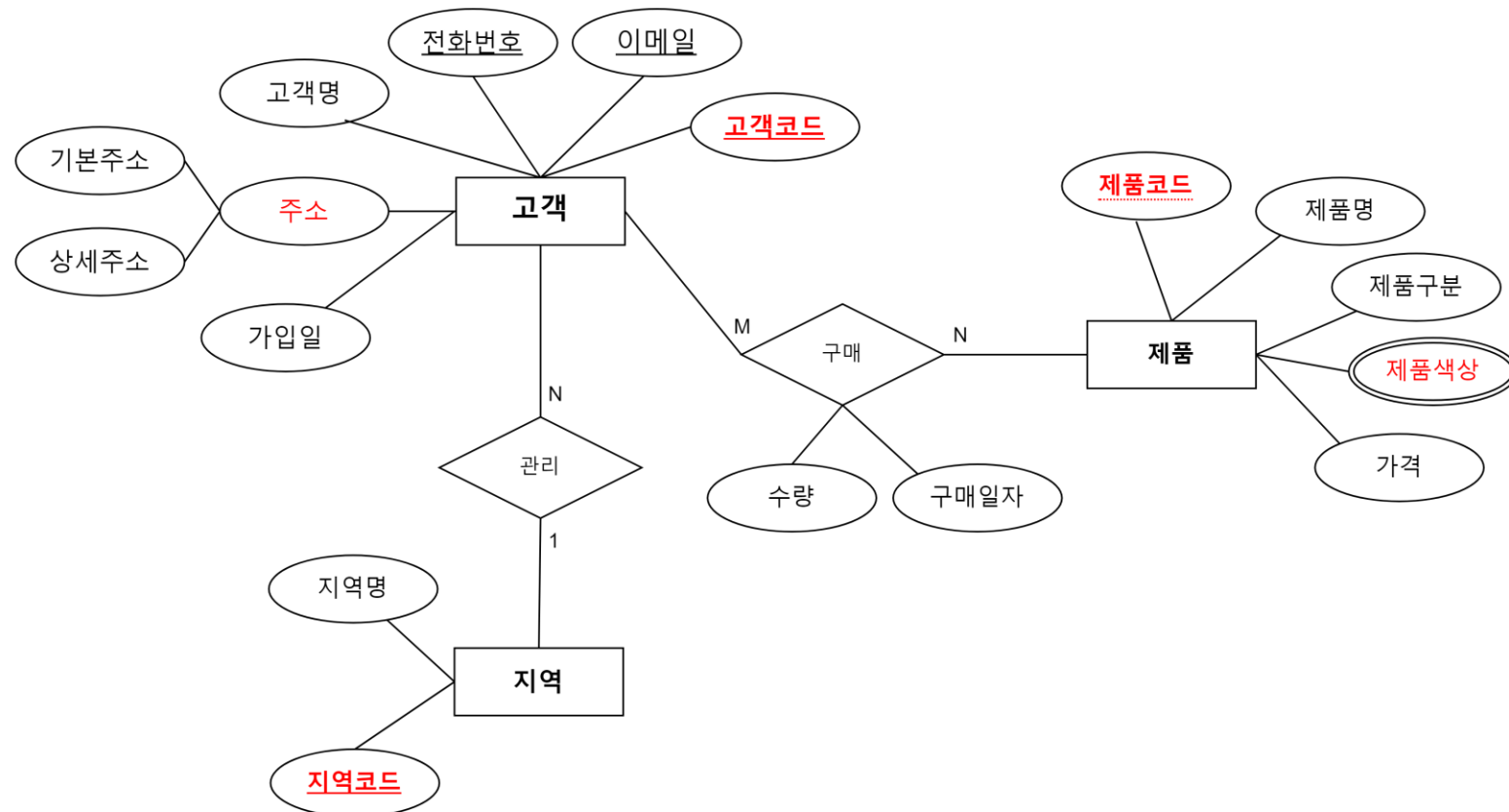
- 온라인 전자상거래 플랫폼 ERD 그리기
 - Initial Schema for COMPANY Database



Data Modeling #2 - Conceptual Data Modeling(ERD)

- 온라인 전자상거래 플랫폼 ERD 그리기

- Refined Schema for COMPANY Database



[7일 완성]생각하는 데이터베이스 모델링

Data Modeling #3 - Logical Data Modeling(RM)

온라인 전자상거래 플랫폼 관계모델 만들기

- Logical Data Modeling(논리적 모델링)

- 목적

- DBMS에 적합한 논리적 스키마 설계
 - 개념적 스키마를 논리적 데이터 모델을 이용해 논리적 구조로 표현

-> 논리적 모델링

일반적으로 관계 데이터 모델을 많이 이용

- 결과물

- 논리적 스키마 : 릴레이션 스키마

- 주요 작업

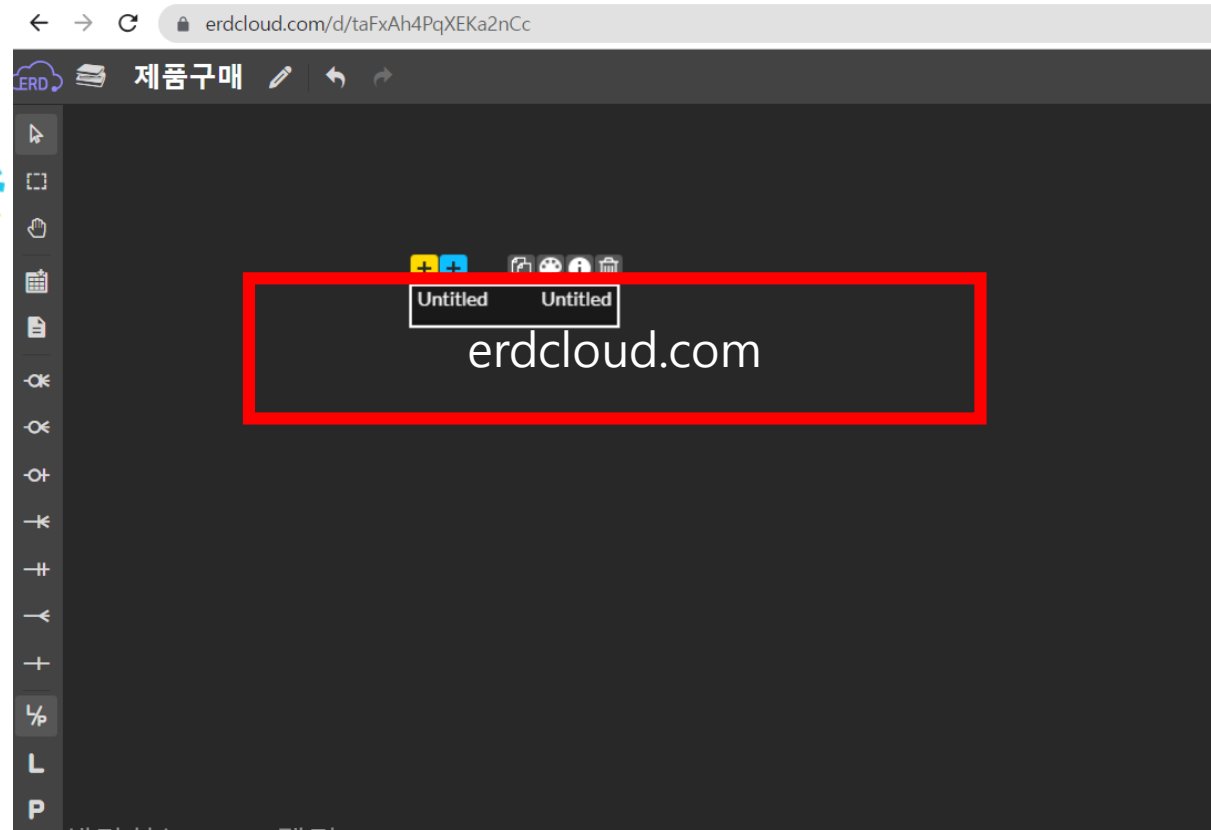
- 개념적 설계 단계의 결과물인 E-R 다이어그램을 릴레이션 스키마로 변환
 - 릴레이션 스키마로 변환 후 속성의 데이터 타입, 길이, 널 값 허용 여부, 기본 값, 제약조건 등을 세부적으로 결정하고 결과를 문서화시킴

Data Modeling #3 - Logical Data Modeling(RM)

- 온라인 전자상거래 플랫폼 관계모델 만들기
 - Logical Data Modeling TOOL

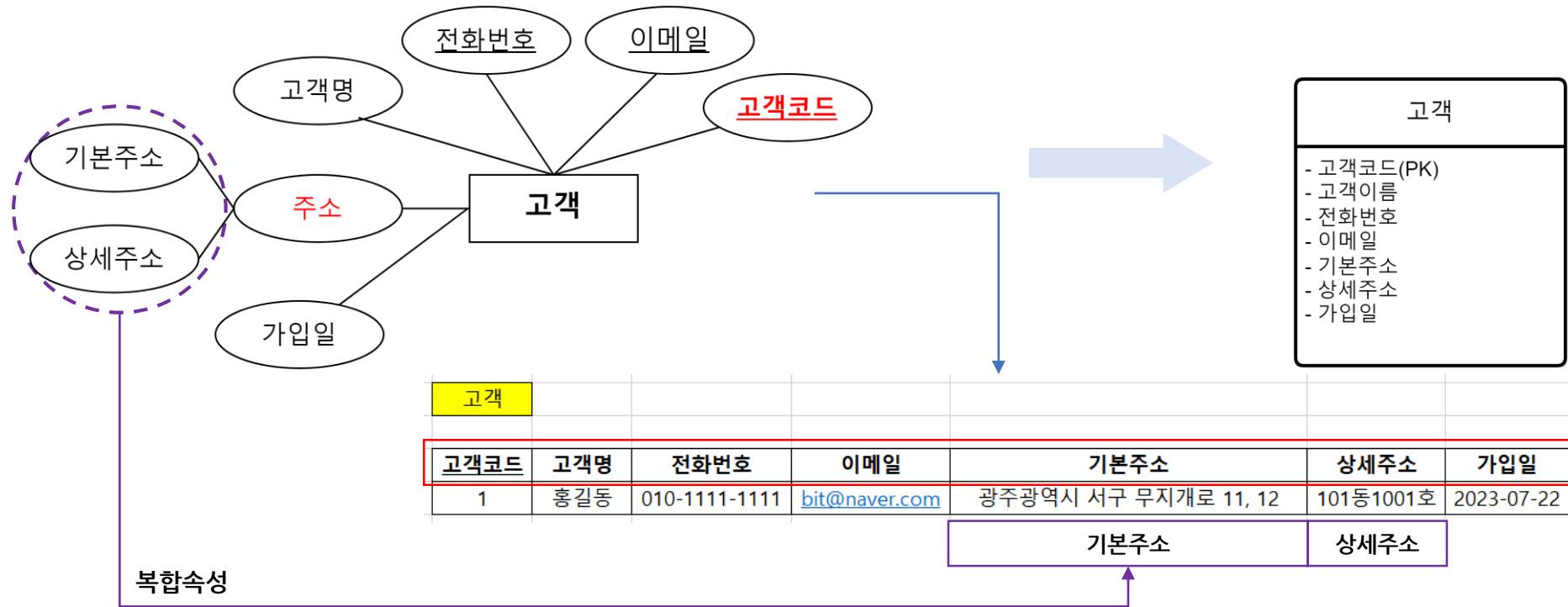


aquerytoo.com



Data Modeling #3 - Logical Data Modeling(RM)

- 온라인 전자상거래 플랫폼 관계모델 만들기
 - 고객 엔티티를 테이블 형식으로 만들기

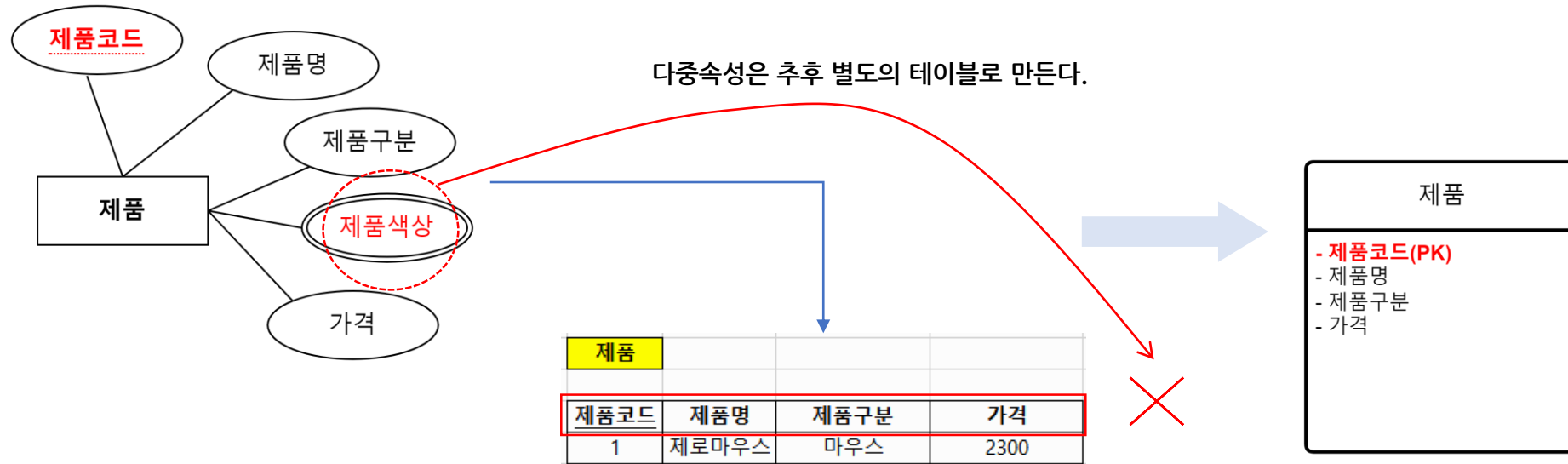


- 온라인 전자상거래 플랫폼 관계모델 만들기
 - 지역 엔티티를 테이블 형식으로 만들기



Data Modeling #3 - Logical Data Modeling(RM)

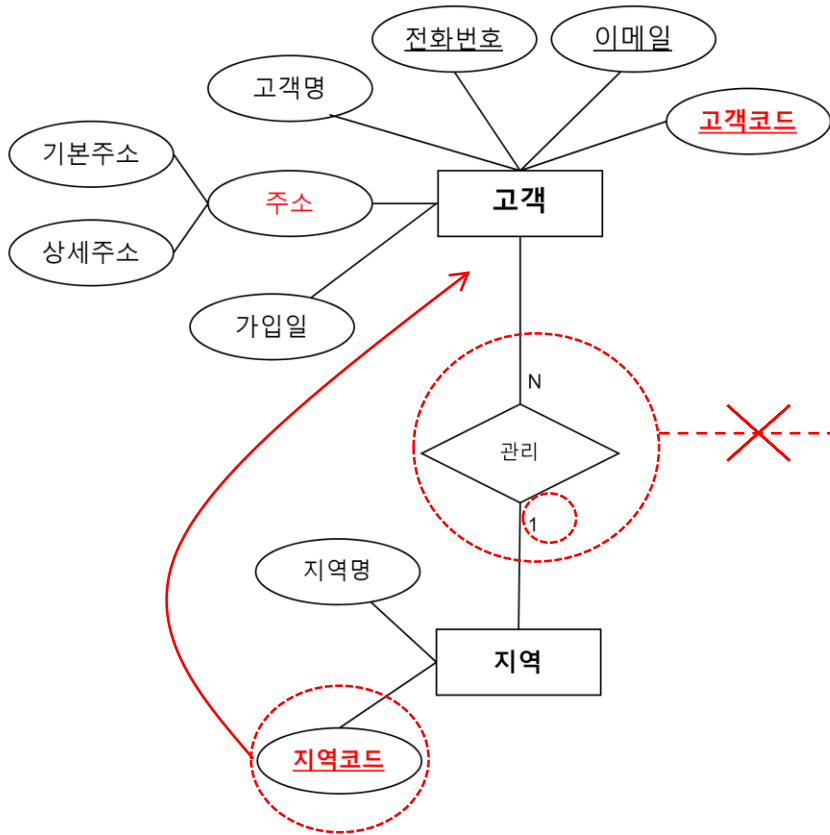
- 온라인 전자상거래 플랫폼 관계모델 만들기
 - 제품 엔티티를 테이블 형식으로 만들기



Data Modeling #3 - Logical Data Modeling(RM)

- 온라인 전자상거래 플랫폼 관계모델 만들기

- 고객 엔티티와 지역 엔티티 관계 설정하기



- RM(Relation Model) : 관계모델

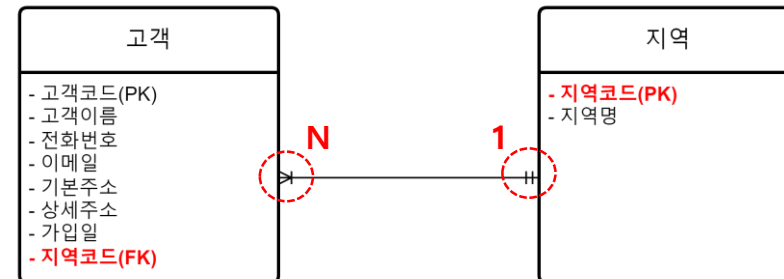
외래 키(FK : Foreign Key)

→ 두 테이블 간의 관계에서 다른 테이블의 기본 키 필드에 연결된 필드이다

외래 키(FK : Foreign Key)							
고객							
고객코드	고객명	전화번호	이메일	기본주소	상세주소	가입일	지역코드
1	홍길동	010-1111-1111	bit@naver.com	광주광역시 서구 무지개로 11, 12	101동1001호	2023-07-22	02

테이블로 만들어 지지 않고 외래키(Foreign Key)로 연결(참조관계)

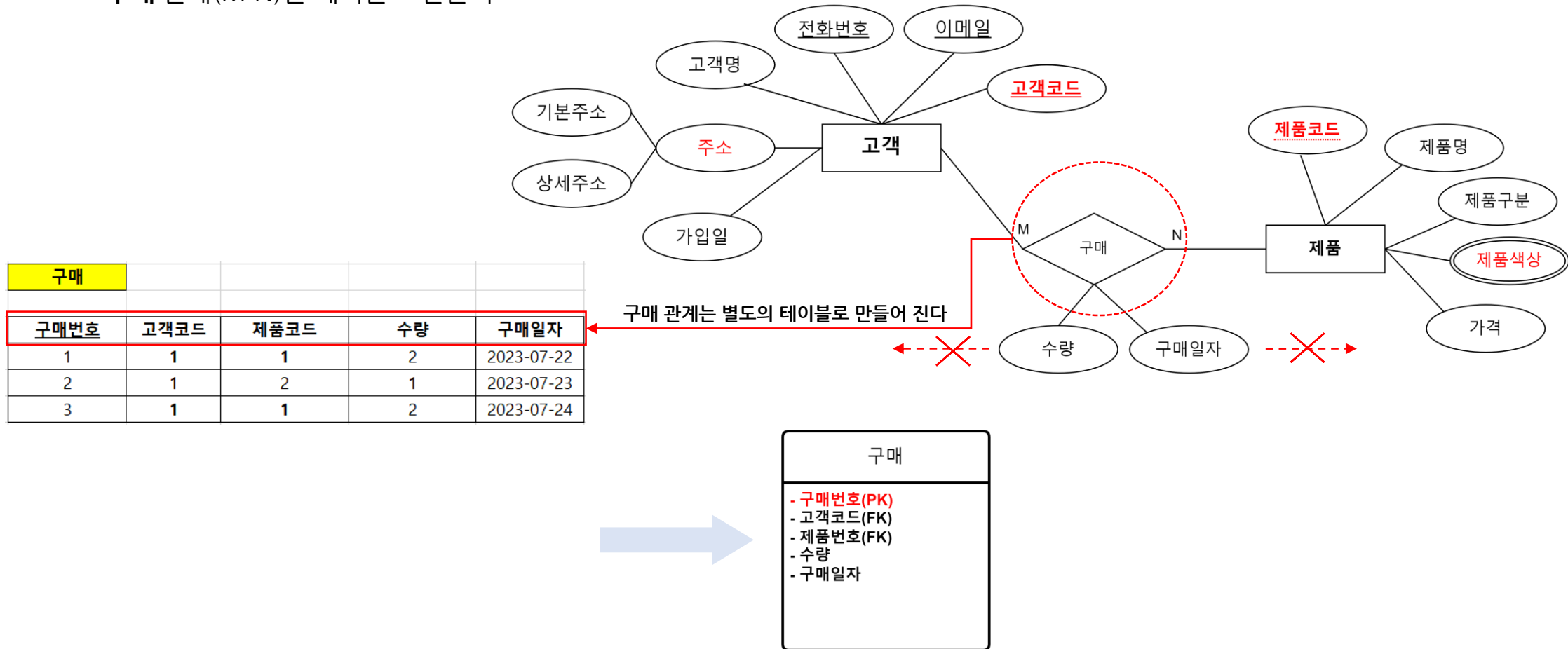
지역	
지역코드	지역명
02	서울특별시



생각하는 DB모델링

- 온라인 전자상거래 플랫폼 관계모델 만들기

- 구매 관계(M:N)를 테이블로 만들기



Data Modeling #3 - Logical Data Modeling(RM)

● 온라인 전자상거래 플랫폼 관계모델 만들기

- 고객 엔티티와 제품 엔티티 관계 설정하기(구매)

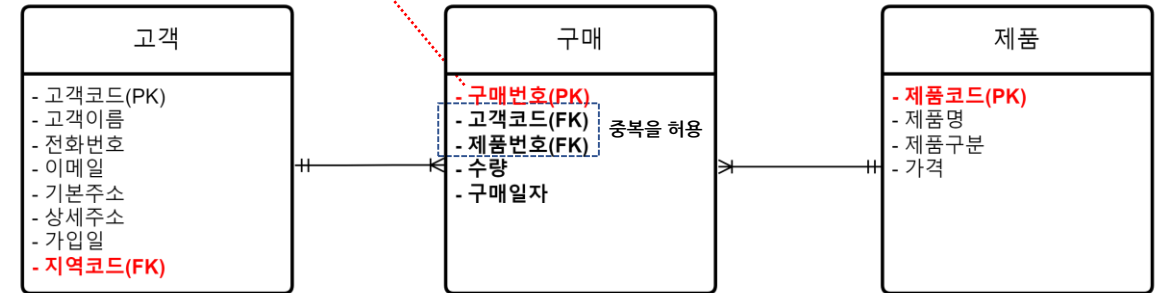
고객		1	M	구매		N	1	제품	
고객코드	고객이름			고객코드(FK)	제품코드(FK)			제품코드	제품명
1	나길동			1	1			1	마우스
2	조길동			1	3			2	키보드
3	홍길동			2	1			3	컴퓨터
				2	3				
				3	3				
				3	3				

중복을 허용 해야 됨

고객		1	M	구매			N	1	제품	
고객코드	고객이름			구매번호(PK)	고객번호(FK)	제품코드(FK)			제품코드	제품명
1	나길동			1	1	1			1	마우스
2	조길동			2	1	3			2	키보드
3	홍길동			3	2	1			3	컴퓨터
				4	2	3				
				5	3	3				
				6	3	3				

- M:N에서 Primary Key 지정(독립형 PK)

* 독립형 PK



Data Modeling #3 - Logical Data Modeling(RM)

- 온라인 전자상거래 플랫폼 관계모델 만들기
 - M:N에서 Primary Key 지정(상속형 PK)

학생		1	M	수강		N	1	과목	
학생코드	학생이름			학생코드(FK)	과목코드(FK)			과목코드	과목명
1	나길동			1	1			1	자바
2	조길동			1	3			2	C
3	홍길동			2	1			3	Python
				2	3				
				3	3				
				3	3				

수강번호(AK) : 후 보키
not null, unique

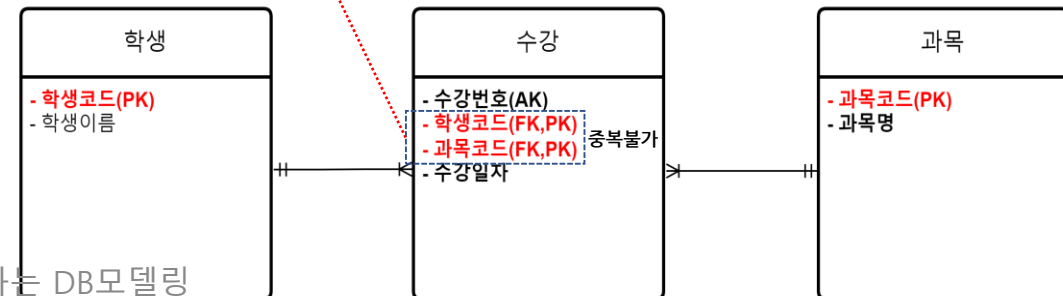
학생		1	M	수강		N	1	과목	
학생코드	학생이름			수강번호(AK)	학생코드(PK,FK)	과목코드(PK,FK)		과목코드	과목명
1	나길동			1	1	1		1	자바
2	조길동			2	1	3		2	C
3	홍길동			3	2	1		3	Python
				4	2	3			
				5	3	3			
				6	3	3			

중복을 막으려면 복합키 설정

확장

학생		1	M	수강		N	1	과목	
학생코드	학생이름			학생코드(PK,FK)	과목코드(PK,FK)			과목코드	과목명
1	나길동			1	1			1	자바
2	조길동			1	3			2	C
3	홍길동			2	1			3	Python
				2	3				
				3	3				
				3	3				

* 상속 PK인 경우



[7일 완성]생각하는 데이터베이스 모델링

Data Modeling #4 - Physical Data Modeling

온라인 전자상거래 플랫폼 물리적모델 만들기

- 온라인 전자상거래 플랫폼 물리적모델 만들기

- 테이블정의서 만들기

		작성일자			작성자				
테이블명							테이블 ID		
테이블설명									
컬럼명	컬럼 ID	타입	길이	NN	PK	FK	UK	CK	비고
Table 생성 스크립트									

Data Modeling #4 - Physical Data Modeling

- 온라인 전자상거래 플랫폼 물리적모델 만들기

- 테이블 만들기

```
-- Drop the tables if they exist
DROP TABLE IF EXISTS t_sales;
DROP TABLE IF EXISTS t_customer;
DROP TABLE IF EXISTS t_product;
DROP TABLE IF EXISTS t_region;
```

```
-- Create the t_region table
CREATE TABLE t_region (
  region_code varchar(3) not null,
  region_name varchar(10) not null,
  primary key(region_code)
);
```

```
-- Create the t_customer table
CREATE TABLE t_customer (
  id int not null auto_increment,
  customer_name varchar(10) not null,
  phone varchar(20) not null unique,
  email varchar(50) not null unique,
  address varchar(100) not null,
  regist_date datetime default now(),
  region_code varchar(3) not null,
  primary key(id)
);
```

```
-- Create the t_product table
CREATE TABLE t_product(
  id int not null auto_increment,
  product_code varchar(12) not null unique,
  product_name varchar(50) not null,
  price int,
  primary key(id)
);
```

```
-- Create the t_sales table
CREATE TABLE t_sales (
  id int not null auto_increment,
  customer_id int not null,
  product_code varchar(12) not null,
  qty int not null,
  sales_date datetime default now(),
  primary key(id)
);
```

- 온라인 전자상거래 플랫폼 물리적모델 만들기
 - 관계 설정하기

```
CREATE TABLE t_customer (  
  id int not null auto_increment,  
  customer_name varchar(10) not null,  
  phone varchar(20) not null unique,  
  email varchar(50) not null unique,  
  address varchar(100) not null,  
  regist_date datetime default now(),  
  region_code varchar(3) not null,  
  primary key(id),  
  foreign key (region_code) references  
  t_region(region_code)  
);
```

```
CREATE TABLE t_customer (  
  id int not null auto_increment,  
  customer_name varchar(10) not null,  
  phone varchar(20) not null unique,  
  email varchar(50) not null unique,  
  address varchar(100) not null,  
  regist_date datetime default now(),  
  region_code varchar(3) not null,  
  primary key(id),  
  CONSTRAINT fk_region_code FOREIGN KEY (region_code)  
  REFERENCES t_region(region_code)  
);
```

Data Modeling #4 - Physical Data Modeling

- 온라인 전자상거래 플랫폼 물리적모델 만들기

- 테이블을 만든 후 관계 설정하기

-- Add foreign key constraint to t_customer table

ALTER TABLE **t_customer**

ADD CONSTRAINT fk_region_code **FOREIGN KEY** (region_code) REFERENCES t_region(region_code);

ALTER TABLE **t_sales**

ADD CONSTRAINT fk_customer_id **FOREIGN KEY** (customer_id) REFERENCES t_customer (id),

ADD CONSTRAINT fk_product_code **FOREIGN KEY** (product_code) REFERENCES t_product (product_code);

- 제약조건 확인하기

select * from **information_schema.table_constraints**;

- 제약조건 삭제하기

ALTER TABLE **t_sales**

DROP FOREIGN KEY fk_customer_id, **DROP FOREIGN KEY** fk_product_code;

Data Modeling #4 - Physical Data Modeling

- 온라인 전자상거래 플랫폼 물리적모델 만들기

- 테이블에 데이터 입력

-- t_region 테이블에 데이터 추가

```
INSERT INTO t_region (region_code, region_name)
VALUES
```

```
('02', '서울특별시'),
('031', '경기도'),
('032', '인천광역시'),
('033', '강원특별자치도'),
('041', '충청남도'),
('042', '대전광역시'),
('043', '충청북도'),
('044', '세종특별자치시'),
('051', '부산광역시'),
('052', '울산광역시'),
('053', '대구광역시'),
('054', '경상북도'),
('055', '경상남도'),
('061', '전라남도'),
('062', '광주광역시'),
('063', '전라북도'),
('064', '제주특별자치도');
```

	region_code	region_name
▶	02	서울특별시
	031	경기도
	032	인천광역시
	033	강원특별자치도
	041	충청남도
	042	대전광역시
	043	충청북도
	044	세종특별자치시
	051	부산광역시
	052	울산광역시
	053	대구광역시
	054	경상북도
	055	경상남도
	061	전라남도
	062	광주광역시
	063	전라북도
	064	제주특별자치도

-- t_customer 테이블에 데이터 추가

```
INSERT INTO t_customer (customer_name, phone, email, address, region_code)
VALUES
```

```
('홍길동', '010-1234-5678', 'hong@example.com', '서울시 강남구', '02'),
('김철수', '010-9876-5432', 'kim@example.com', '경기도 수원시', '031'),
('이영희', '010-1111-2222', 'lee@example.com', '인천시 남구', '032'),
('박민지', '010-5555-7777', 'park@example.com', '강원도 춘천시', '033'),
('정기호', '010-9999-8888', 'jung@example.com', '대전시 중구', '042');
```

	id	customer_name	phone	email	address	regist_date	region_code
▶	1	홍길동	010-1234-5678	hong@example.com	서울시 강남구	2023-07-07 13:42:24	02
	2	김철수	010-9876-5432	kim@example.com	경기도 수원시	2023-07-07 13:42:24	031
	3	이영희	010-1111-2222	lee@example.com	인천시 남구	2023-07-07 13:42:24	032
	4	박민지	010-5555-7777	park@example.com	강원도 춘천시	2023-07-07 13:42:24	033
	5	정기호	010-9999-8888	jung@example.com	대전시 중구	2023-07-07 13:42:24	042

Data Modeling #4 - Physical Data Modeling

- 온라인 전자상거래 플랫폼 물리적모델 만들기

- 테이블에 데이터 입력

-- t_product 테이블에 데이터 추가

```
INSERT INTO t_product (product_code, product_name, price)
VALUES
('P001', '노트북', 1500000),
('P002', '스마트폰', 1000000),
('P003', '키보드', 50000),
('P004', '마우스', 30000),
('P005', '이어폰', 70000);
```

	id	product_code	product_name	price
▶	1	P001	노트북	1500000
	2	P002	스마트폰	1000000
	3	P003	키보드	50000
	4	P004	마우스	30000
	5	P005	이어폰	70000

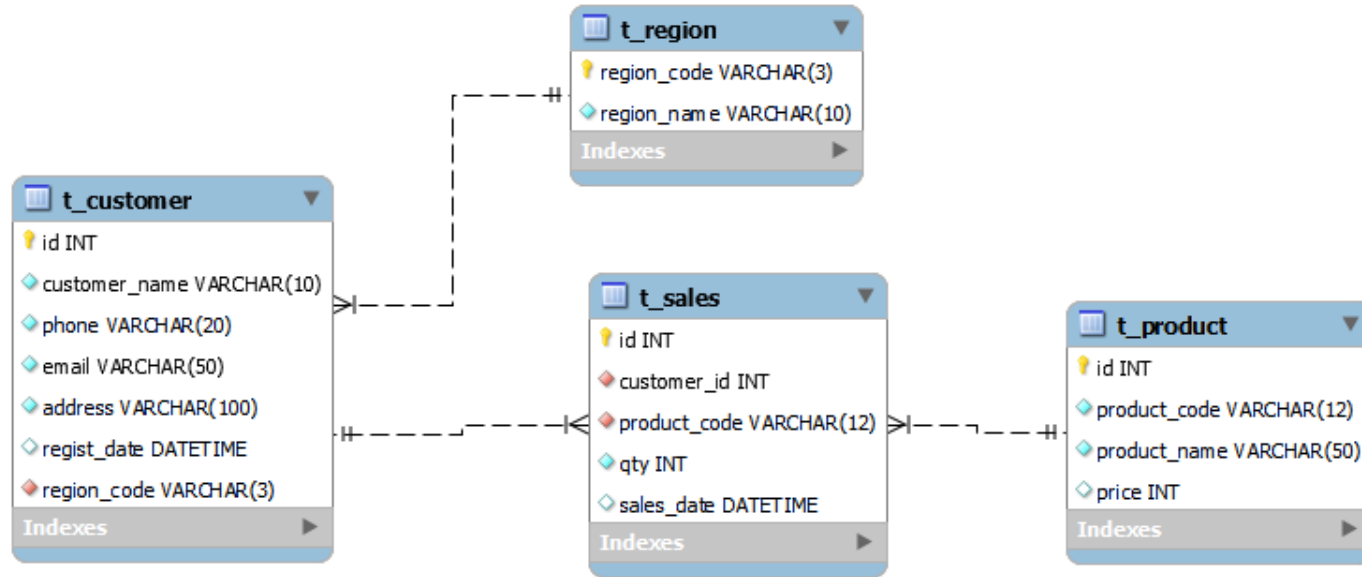
-- t_sales 테이블에 데이터 추가

```
INSERT INTO t_sales (customer_id, product_code, qty)
VALUES
(1, 'P001', 2),
(2, 'P002', 1),
(3, 'P003', 5),
(4, 'P004', 3),
(5, 'P005', 2),
(1, 'P002', 3),
(3, 'P001', 1),
(2, 'P004', 2),
(4, 'P003', 4),
(5, 'P005', 1);
```

	id	customer_id	product_code	qty	sales_date
▶	1	1	P001	2	2023-07-07 13:42:24
	2	2	P002	1	2023-07-07 13:42:24
	3	3	P003	5	2023-07-07 13:42:24
	4	4	P004	3	2023-07-07 13:42:24
	5	5	P005	2	2023-07-07 13:42:24
	6	1	P002	3	2023-07-07 13:42:24
	7	3	P001	1	2023-07-07 13:42:24
	8	2	P004	2	2023-07-07 13:42:24
	9	4	P003	4	2023-07-07 13:42:24
	10	5	P005	1	2023-07-07 13:42:24

Data Modeling #4 - Physical Data Modeling

- 온라인 전자상거래 플랫폼 물리적모델 만들기
 - 물리적 모델링 후 생성된 관계모델(RM)



Data Modeling #4 - Physical Data Modeling

- 온라인 전자상거래 플랫폼 물리적모델 만들기

- 화면 UI 설계와 DB관계

	region_code	region_name
▶	02	서울특별시
	031	경기도
	032	인천광역시
	033	강원특별자치도
	041	충청남도
	042	대전광역시
	043	충청북도
	044	세종특별자치시
	051	부산광역시
	052	울산광역시
	053	대구광역시
	054	경상북도
	055	경상남도
	061	전라남도
	062	광주광역시
	063	전라북도
	064	제주특별자치도

고객이름:

박매일

전화번호:

010-1111-1111

이 메 일:

abc@naver.com

주 소:

광주광역시 서구 무슨 로 11,11

지 역:

선택하세요 ▼

서울특별시
경기도
인천광역시

[등록] [취소]

자동증가

	id	customer_name	phone	email	address
▶	1	홍길동	010-1234-5678	hong@example.com	서울시 강남구
	2	김철수	010-9876-5432	kim@example.com	경기도 수원시
	3	이영희	010-1111-2222	lee@example.com	인천시 남구
	4	박민지	010-5555-7777	park@example.com	강원도 춘천시
	5	정기호	010-9999-8888	jung@example.com	대전시 중구

가입일자

regist_date	region_code
2023-07-07 13:42:24	02
2023-07-07 13:42:24	031
2023-07-07 13:42:24	032
2023-07-07 13:42:24	033
2023-07-07 13:42:24	042

생각하는 DB모델링

[7일 완성] 생각하는 데이터베이스 모델링








Data Modeling #5 - SQL(Structured Query Language)

SQL 실습(CRUD 연습)

Data Modeling #5 - SQL(Structured Query Language)

- SQL 실습(CRUD 연습)

SQL(Structured Query Language, 구조화 질의어)은 관계형 데이터베이스 관리 시스템(RDBMS)의 데이터를 관리하기 위해 설계된 특수 목적의 프로그래밍 언어

질의 내용	SQL
Q : 모든 고객의 정보를 출력하세요.	 <code>select * from t_customer;</code>
Q : 고객정보를 입력하세요.	 <code>insert into t_customer(customer_name, phone, email, address, region_code)</code> <code>values('손흥민','010-1234-7894','son@gmail.com','서울특별시 강남구 삼성동','02');</code>  <code>insert into t_customer(customer_name, phone, email, address, region_code)</code> <code>values('박태환','010-1234-5555','park@gmail.com','제주특별자치도 제주시 일도','064');</code>
Q : 손흥민 고객의 주소를 수정하세요.	 <code>update t_customer set address='서울특별시 강남구 일동' <u>where</u> id=5;</code>
Q : 홍길동 고객을 삭제하세요.	 <code>delete from t_customer <u>where</u> id=1;</code>
Q. 서울에 거주하는 고객을 출력하세요.	 <code>select * from t_customer <u>where</u> region_code='02';</code>
Q. 박태환 고객이 물건(노토)을 2개를 주문했을 경우 구매 테이블에 입력하세요.	 <code>insert into t_sales(customer_id, product_code, qty) values(6,'P00110',2);</code>

Data Modeling #5 - SQL(Structured Query Language)

- SQL 실습(CRUD 연습)

SQL(Structured Query Language, 구조화 질의어)은 관계형 데이터베이스 관리 시스템(RDBMS)의 데이터를 관리하기 위해 설계된 특수 목적의 프로그래밍 언어

Q1. 모든 고객의 이름과 이메일을 가져오는 질의

```
SELECT customer_name, email  
FROM t_customer;
```

Q2. 특정 지역(예: '서울특별시')에 사는 고객의 이름과 전화번호를 가져오는 질의:

```
SELECT customer_name, phone  
FROM t_customer  
WHERE region_code = '02';
```

Q3. 특정 제품의 판매량과 판매 일자를 가져오는 질의:

```
SELECT qty, sales_date  
FROM t_sales  
WHERE product_code = 'P00110';
```

- SQL 실습(CRUD 연습)

Q4. 제품별로 판매된 총 수량과 가격을 계산하는 질의:

```
SELECT product_name, SUM(qty) AS total_quantity, SUM(qty * price) AS total_price
FROM t_sales
JOIN t_product ON t_sales.product_code = t_product.product_code
GROUP BY product_name;
```

Q5. 특정 기간 동안의 판매량을 계산하는 질의

```
SELECT DATE(sales_date) AS sales_date, SUM(qty) AS total_quantity
FROM t_sales
WHERE sales_date BETWEEN '2023-01-01' AND '2023-07-30'
GROUP BY DATE(sales_date);
```

- SQL 실습(CRUD 연습)

Q6. 가장 최근에 등록된 고객의 정보를 가져오는 질의

```
SELECT *  
FROM t_customer  
ORDER BY regist_date DESC  
LIMIT 1;
```

Q7. 각 지역별로 고객 수를 계산하는 질의:

```
SELECT t_region.region_name, COUNT(t_customer.id) AS customer_count  
FROM t_region LEFT JOIN t_customer  
ON t_region.region_code = t_customer.region_code  
GROUP BY t_region.region_name;
```

FROM: FROM 절에서는 t_region 테이블과 t_customer 테이블을 사용합니다. t_region 테이블은 왼쪽 테이블로 설정되고, t_customer 테이블은 조인 조건에 따라 오른쪽 테이블로 설정됩니다.

LEFT JOIN: LEFT JOIN은 t_region 테이블을 기준으로 왼쪽 테이블의 모든 레코드를 선택하고, 조인 조건에 맞는 t_customer 테이블의 레코드를 연결합니다. 이 때, t_region.region_code와 t_customer.region_code가 일치하는 조인 조건을 사용합니다. LEFT JOIN을 통해 모든 지역에 대한 고객 수를 계산할 수 있습니다.

GROUP BY: GROUP BY 절에서는 t_region.region_name을 기준으로 그룹화합니다. 이렇게 함으로써 지역별로 고객 수를 계산할 수 있습니다.

SELECT: SELECT 절에서는 t_region.region_name과 COUNT(t_customer.id)를 선택합니다. COUNT(t_customer.id)는 각 지역에 속하는 고객 수를 계산하는 집계 함수입니다.

- SQL 실습(CRUD 연습)

Q8. 제품별로 판매된 총 가격과 평균 가격을 계산하는 질의

```
SELECT t_product.product_name, SUM(t_sales.qty * t_product.price) AS total_price, AVG(t_product.price) AS average_price
FROM t_sales
JOIN t_product ON t_sales.product_code = t_product.product_code
GROUP BY t_product.product_name;
```

Q9. 특정 고객이 구매한 제품의 목록을 가져오는 질의:

```
SELECT t_product.product_name
FROM t_sales
JOIN t_product ON t_sales.product_code = t_product.product_code
WHERE t_sales.customer_id = 3;
```

Q10. 고객이 속한 지역의 이름과 해당 지역의 판매량을 가져오는 질의:

```
SELECT t_region.region_name, SUM(t_sales.qty) AS total_quantity
FROM t_customer
JOIN t_region ON t_customer.region_code = t_region.region_code
JOIN t_sales ON t_customer.id = t_sales.customer_id
GROUP BY t_region.region_name;
```

- SQL 실습(CRUD 연습)

Q11. 고객이 구매한 제품의 평균 가격보다 높은 가격으로 판매된 제품의 목록을 가져오는 질의:

```
SELECT t_product.product_name, t_product.price
FROM t_product
WHERE t_product.price > (SELECT AVG(price) FROM t_product)
```

Q12. 고객 당 총 판매 금액:

```
SELECT t1.customer_name, SUM(t3.qty * t2.price) AS total_sales_amount
FROM t_customer t1
JOIN t_sales t3 ON t1.id = t3.customer_id
JOIN t_product t2 ON t2.product_code = t3.product_code
GROUP BY t1.customer_name;
```

Q13. 최고 판매 제품:

```
SELECT t2.product_name, SUM(t3.qty) AS total_quantity_sold
FROM t_product t2
JOIN t_sales t3 ON t2.product_code = t3.product_code
GROUP BY t2.product_name
ORDER BY total_quantity_sold DESC
LIMIT 5;
```

- SQL 실습(CRUD 연습)

Q14. 월간 판매 추이:

```
SELECT DATE_FORMAT(t3.sales_date, '%Y-%m') AS sales_month, SUM(t3.qty * t2.price) AS total_sales_amount
FROM t_sales t3
JOIN t_product t2 ON t2.product_code = t3.product_code
GROUP BY sales_month
ORDER BY sales_month;
```

Q15. 판매당 평균 수량:

```
SELECT AVG(t3.qty) AS average_quantity
FROM t_sales t3;
```

Q16. 복수 구매 고객:

```
SELECT t1.customer_name, COUNT(DISTINCT t3.product_code) AS num_products_purchased
FROM t_customer t1
JOIN t_sales t3 ON t1.id = t3.customer_id
GROUP BY t1.customer_name
HAVING COUNT(DISTINCT t3.product_code) > 1;
```

- SQL 실습(CRUD 연습)

Q17. 판매 테이블에서 고객의 ID, 이름, 주소와 함께 제품의 코드, 이름, 판매 날짜, 수량, 가격 및 계산된 총액이 포함되도록 질의:

```
SELECT t1.id, t1.customer_name, t1.address, t2.product_code, t2.product_name, t3.sales_date, t3.qty, t2.price, t3.qty * t2.price as total_amount
FROM t_customer t1
JOIN t_sales t3 ON t1.id = t3.customer_id
JOIN t_product t2 ON t2.product_code = t3.product_code
WHERE t1.id = 3;
```

Q18. 고객의 총 판매 금액과 함께 고객 정보를 검색하는 쿼리

```
SELECT c.customer_name, c.email,
(
SELECT SUM(s.qty * p.price)
FROM t_sales s
INNER JOIN t_product p ON s.product_code = p.product_code
WHERE s.customer_id = c.id
) AS total_sales_amount
FROM t_customer c;
```

Q19. 구매한 총 제품 수와 함께 고객 이름을 검색

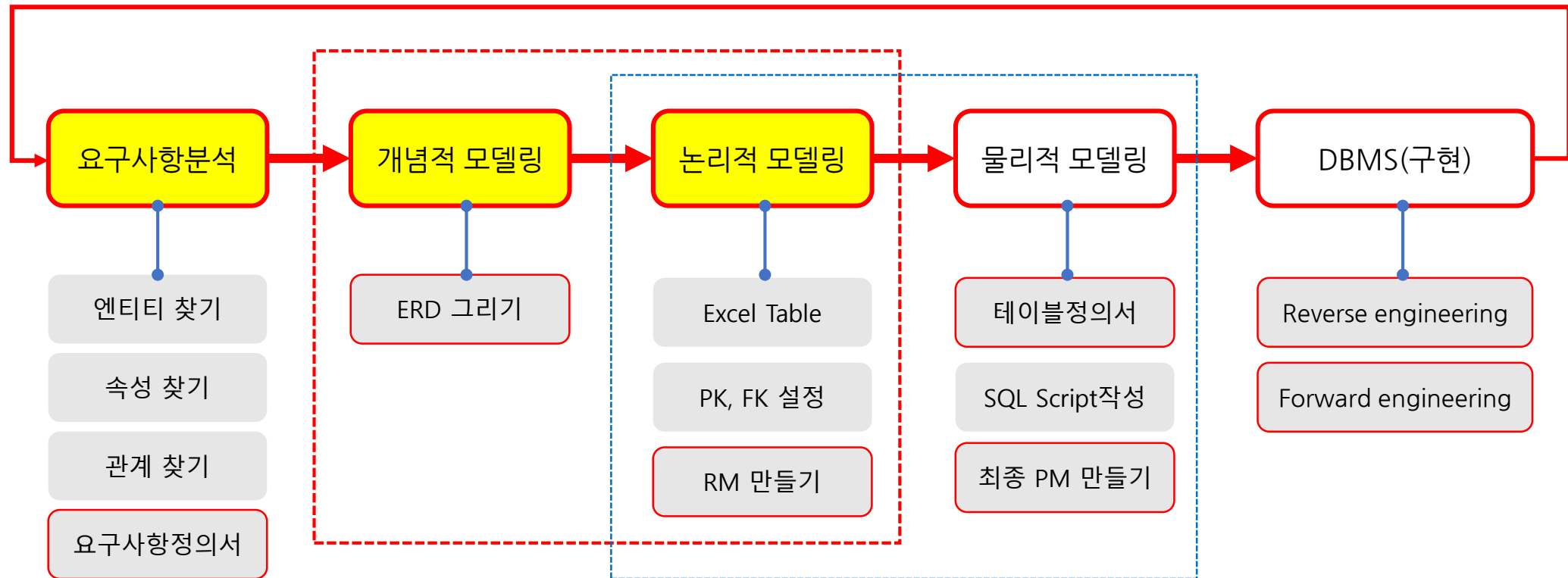
```
SELECT c.customer_name,
(
SELECT COUNT(*)
FROM t_sales s
WHERE s.customer_id = c.id
) AS total_products_purchased
FROM t_customer c;
```


[7일 완성]생각하는 데이터베이스 모델링

Data Modeling #6 - 데이터모델링 과제 수행

학사관리 시스템 구축 데이터 모델링

- 학사관리 시스템 구축 데이터 모델링



- 학사관리 시스템 구축 데이터 모델링

간단한 **학사관리 시스템**을 구축하려고 합니다.

고객과의 상담을 통해서 관리할 필요가 있는 다음 정보들을 파악 하였습니다.

학생은 학번, 이름, 키, 학과코드로 이루어져 있습니다.

학과는 학과코드, 학과명으로 이루어져 있습니다.

교수는 교수코드, 교수 명, 학과코드로 이루어져 있습니다.

개설과목은 과목코드, 과목명, 교수코드, 시작일, 종료일로 이루어져 있습니다.

수강은 학번, 과목코드로 이루어져 있습니다.

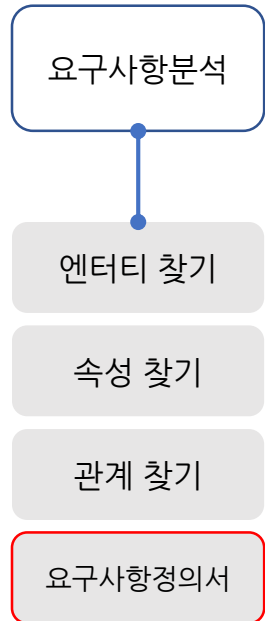
학과는 많은 학생을 가질 수 있고 학생은 한 학과의 소속된다.

학과는 많은 교수를 가질 수 있고 교수는 한 학과의 소속 된다.

교수는 많은 과목을 가르칠 수 있고 과목은 강의하는 교수 한 명이 지정된다.

과목은 수강하는 많은 학생을 가지고 학생은 많은 과목을 수강할 수 있다.

● 학사관리 시스템 구축 데이터 모델링



▪ 1. 요구사항 정의서

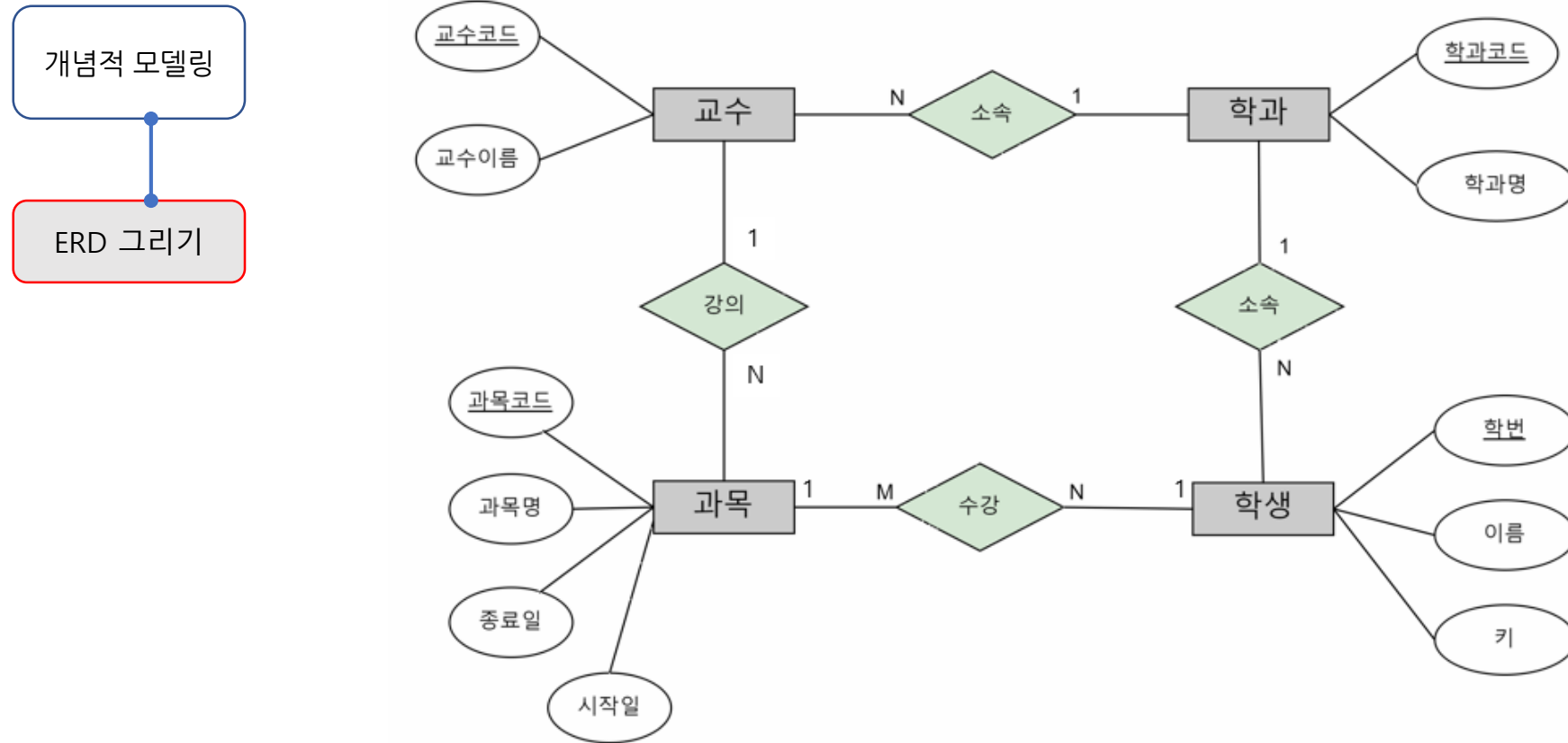
- ① 학생은 학번, 이름, 키, 학과코드로 이루어져 있습니다.
- ② 학과는 학과코드, 학과명으로 이루어져 있습니다.
- ③ 교수는 교수코드, 교수명, 학과코드로 이루어져 있습니다.
- ④ 개설과목은 과목코드, 과목명, 교수코드, 시작일, 종료일로 이루어져 있습니다.
- ⑤ 수강은 학번, 과목코드로 이루어져 있습니다.
- ⑥ 학과는 많은 학생을 가질 수 있고 학생은 한 학과의 소속된다.
- ⑦ 학과는 많은 교수를 가질 수 있고 교수는 한 학과의 소속 된다.
- ⑧ 교수는 많은 과목을 가르칠 수 있고 과목은 강의하는 교수 한 명이 지정된다.
- ⑨ 과목은 수강하는 많은 학생을 가지고 학생은 많은 과목을 수강할 수 있다.

▪ 2. 객체 정의서

객체 명	속성 명
학과	학과코드, 학과명
학생	학번, 이름, 키
교수	교수코드, 교수이름
과목	과목코드, 과목명, 시작일, 종료일

관계	참여객체	관계유형	속성
소속	학과와 학생 학과와 교수	1:N	
강의	교수와 과목	1:N	
수강	학생과 과목	M:N	

- 학사관리 시스템 구축 데이터 모델링



- 학사관리 시스템 구축 데이터 모델링

논리적 모델링

Excel Table

PK, FK 설정

RM 만들기

Relational Schema				
논리적인모델링				
학생				
학번(PK)	이름	키	학과코드(FK)	
학과				
학과코드(PK)	학과명			
교수				
교수코드(PK)	교수이름	학과코드(FK)		
과목				
과목코드(PK)	과목명	시작일	종료일	교수코드(FK)
수강				
번호(PK)	학번(FK)	과목코드(FK)		

● 학사관리 시스템 구축 데이터 모델링

물리적 모델링

테이블정의서

SQL Script작성

테이블명		교수		계정ID	root		
NO	컬럼ID	컬럼설명	TYPE	길이	NULL여부	PK	비고
1	professor_id	교수코드	INT		NN	Y	AI
2	professor_name	교수이름	VARCHAR	10	NN		
3	professor_code	학과코드	INT		NN		FK

테이블명		학생		계정ID	root		
NO	컬럼ID	컬럼설명	TYPE	길이	NULL여부	PK	비고
1	student_id	학번	INT		NN	Y	AI
2	student_name	학생이름	VARCHAR	10	NN		
3	student_height	학생키	DECIMAL		NN		
4	student_code	학과코드	INT		NN		FK

테이블명		학과		계정ID	root		
NO	컬럼ID	컬럼설명	TYPE	길이	NULL여부	PK	비고
1	department_code	학과코드	INT		NN	Y	AI
2	department_name	학과명	VARCHAR	20	NN		

테이블명		과목		계정ID	root		
NO	컬럼ID	컬럼설명	TYPE	길이	NULL여부	PK	비고
1	subject_code	과목코드	INT		NN	Y	AI
2	subject_name	과목명	VARCHAR	20	NN		
3	subject_start	시작일	DATE		NN		
4	subject_end	종료일	DATE		NN		
5	subject_professor	교수코드	INT		NN		FK

테이블명		학과		계정ID	root		
NO	컬럼ID	컬럼설명	TYPE	길이	NULL여부	PK	비고
1	class_id	수강번호	INT		NN	Y	AI
2	student_id	학번	INT		NN		FK
3	subject_code	과목코드	INT		NN		FK

● 학사관리 시스템 구축 데이터 모델링

```
-- Create Department table
CREATE TABLE Department (
  department_code INT PRIMARY KEY,
  department_name VARCHAR(50)
);

-- Create Student table
CREATE TABLE Student (
  student_id INT PRIMARY KEY AUTO_INCREMENT,
  name VARCHAR(50),
  height DECIMAL(5,2),
  department_code INT
);

-- Create Professor table
CREATE TABLE Professor (
  professor_code INT PRIMARY KEY AUTO_INCREMENT,
  professor_name VARCHAR(50),
  department_code INT
);

-- Create Course table
CREATE TABLE Course (
  course_code INT PRIMARY KEY AUTO_INCREMENT,
  course_name VARCHAR(50),
  professor_code INT,
  start_date DATE,
  end_date DATE
);
```

```
-- Create Student_Course table
CREATE TABLE Student_Course (
  id INT PRIMARY KEY AUTO_INCREMENT,
  student_id INT,
  course_code INT
);

-- Add foreign key constraints using ALTER TABLE
ALTER TABLE Student
ADD CONSTRAINT FK_Student_Department
FOREIGN KEY (department_code) REFERENCES Department(department_code);

ALTER TABLE Professor
ADD CONSTRAINT FK_Professor_Department
FOREIGN KEY (department_code) REFERENCES Department(department_code);

ALTER TABLE Course
ADD CONSTRAINT FK_Course_Professor
FOREIGN KEY (professor_code) REFERENCES Professor(professor_code);

ALTER TABLE Student_Course
ADD CONSTRAINT FK_Student_Course_Student
FOREIGN KEY (student_id) REFERENCES Student(student_id);

ALTER TABLE Student_Course
ADD CONSTRAINT FK_Student_Course_Course
FOREIGN KEY (course_code) REFERENCES Course(course_code);
```


● 학사관리 시스템 구축 데이터 모델링

-- 학생

-- Insert data into Student table

```
INSERT INTO Student (name, height, department_code) VALUES  
( '가길동', 170.5, 1),  
( '나길동', 165.2, 1),  
( '다길동', 180.0, 2),  
( '라길동', 175.8, 3),  
( '마길동', 160.7, 4),  
( '바길동', 168.3, 1),  
( '사길동', 172.1, 2),  
( '아길동', 175.0, 1);
```

-- 교수

-- Insert data into Professor table

```
INSERT INTO Professor (professor_name, department_code)  
VALUES  
( '가 교수', 1),  
( '나 교수', 1),  
( '다 교수', 2),  
( '빌 게이츠', 3),  
( '스티브 잡스', 4);
```

-- 개설과목

-- Insert data into Course table

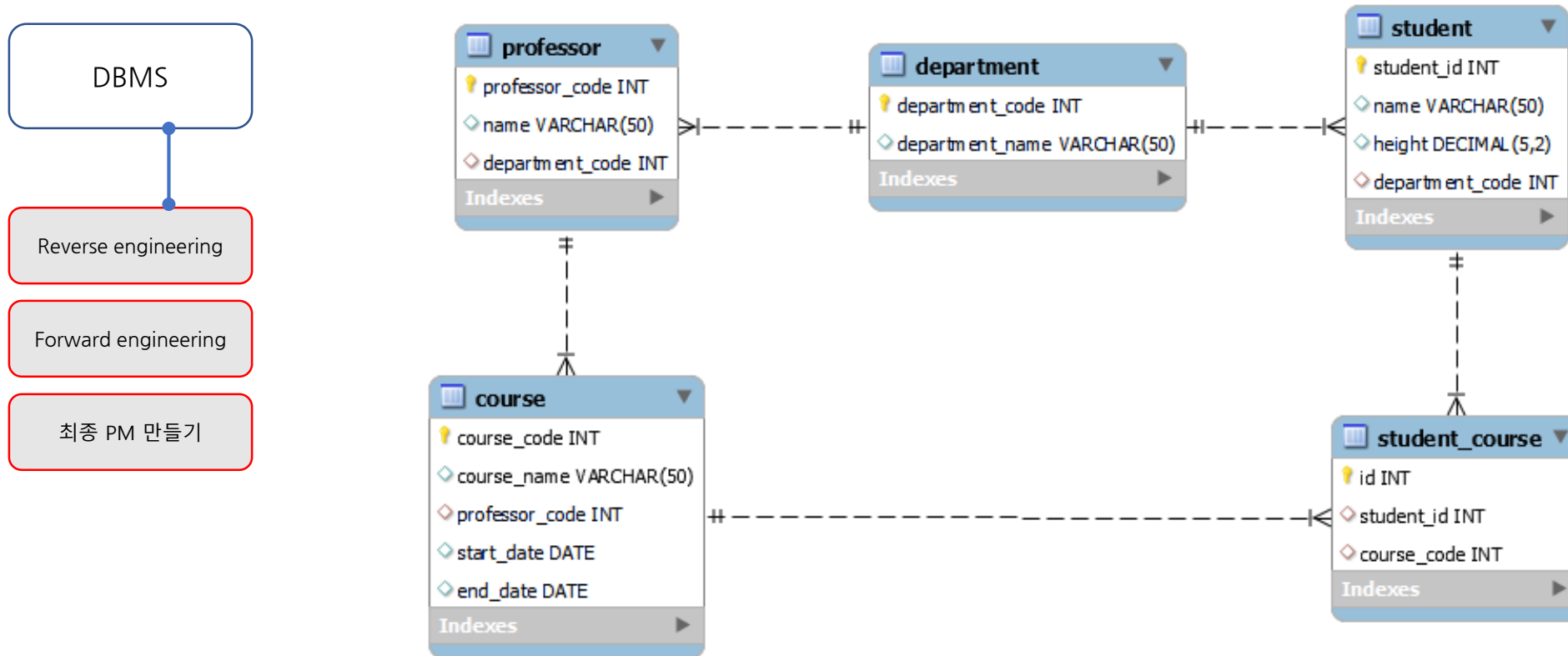
```
INSERT INTO Course (course_name, professor_code, start_date,  
end_date) VALUES  
( '교양 영어', 1, '2023-07-01', '2023-08-15'),  
( '데이터베이스 입문', 2, '2023-07-01', '2023-08-31'),  
( '회로이론', 3, '2023-07-15', '2023-09-15'),  
( '공학수학', 4, '2023-07-15', '2023-09-30'),  
( '객체지향 프로그래밍', 5, '2023-07-01', '2023-08-31');
```

-- 수강

-- Insert data into Student_Course table

```
INSERT INTO Student_Course (student_id, course_code) VALUES  
(1, 1),  
(2, 1),  
(3, 2),  
(4, 3),  
(5, 4),  
(6, 5),  
(7, 5);
```

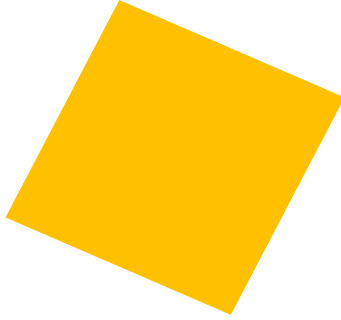
- 학사관리 시스템 구축 데이터 모델링



[7일 완성] 생각하는 데이터베이스 모델링

Data Modeling #7 - 데이터베이스 모델링 종합시험

데이터베이스 설계 및 SQL 활용 시험



생각하는 데이터베이스 모델링 종합시험

과 정 명 : 생각하는 데이터베이스 모델링

시험범위 : 데이터베이스 설계 및 SQL 활용

문 제 : 10문항

시험시간 : 2시간

시험일 : 과정종료일 직 후

제출방법 : bitcocom@empas.com

- 메모장, 워드 등 문서작성기를 이용하여 번호 별 정답 작성

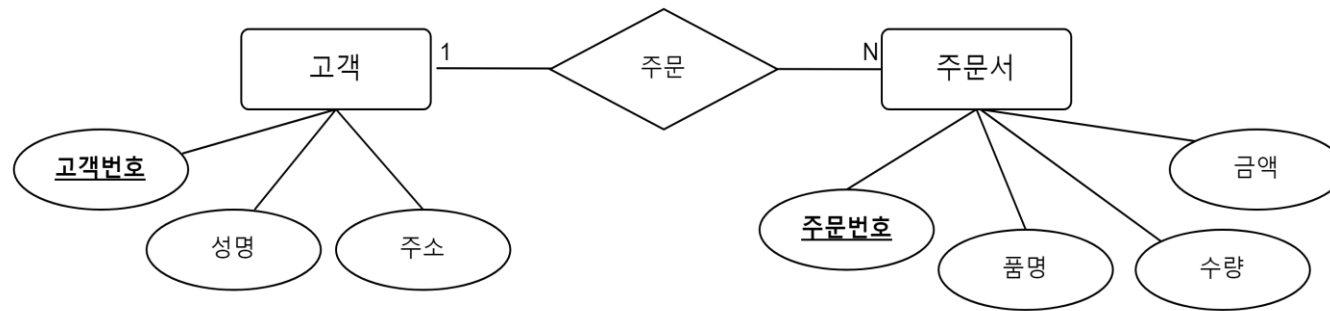
- 제출파일명 : DB시험(이름)

-주의사항-

시험형태 : 오픈 북(인터넷 미 사용), 실습

● 데이터베이스 설계 및 SQL 활용 시험

문제1. 다음은 고객과 주문서 간의 관계를 나타낸 E-R 다이어그램이다 질문에 답하시요.



1) 개체는 무엇인가

정답)

2) 고객의 속성을 기술하시요

정답)

3) 주문서의 속성을 기술하시요

정답)

4) 고객과 주문서의 주문 관계는 어떤 관계((대응 수, 다중도) 가 있는가

정답)

5) 고객과 주문서의 주문 관계를 고객측면과 주문서 측면에서 설명하시요.

정답) - 한사람의 고객은 ~

정답) - 한 개의 주문서는 ~

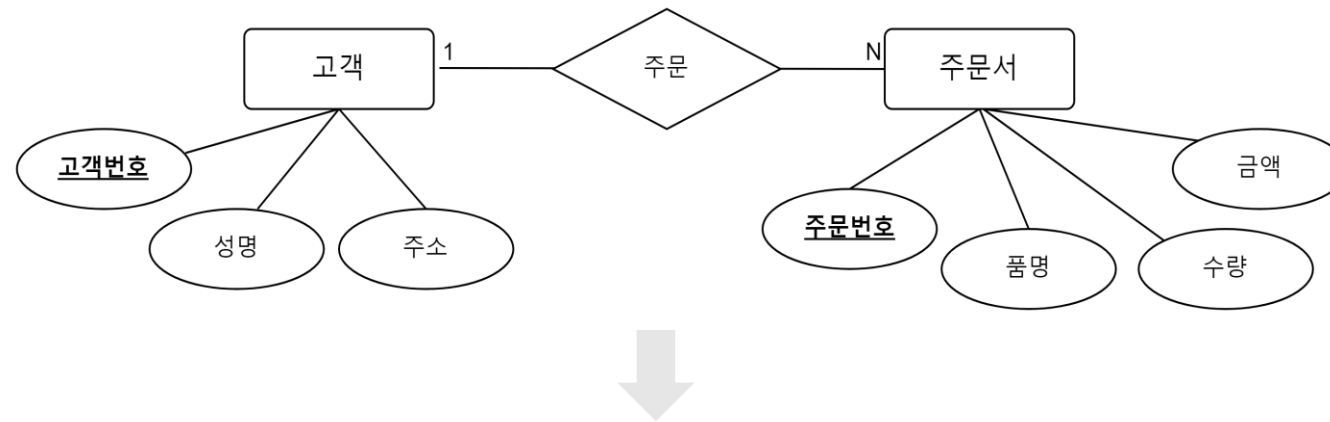
6) 기본 키(Primary Key) 속성을 기술하시요

정답)

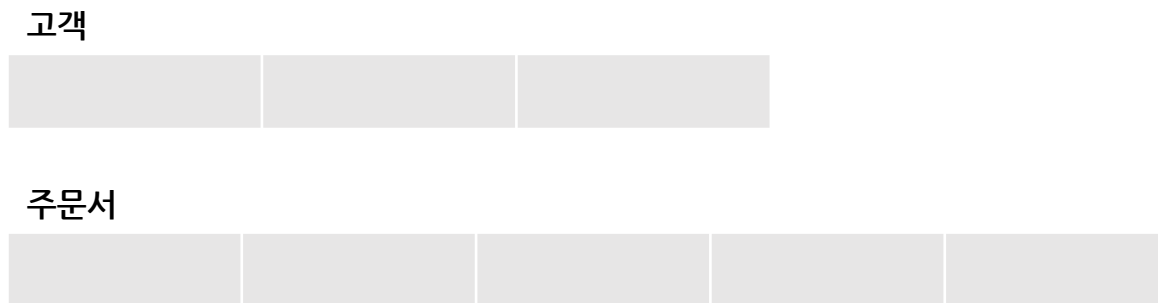
- 데이터베이스 설계 및 SQL 활용 시험

문제2. 다음은 고객과 주문서 간의 관계를 나타낸 E-R 다이어그램이다. Binary 1:N Relation Types을 논리적인 모델(Relational Model)로 설계하시요 (참조관계를 화살표로 표시하여 캡처 할 것)

ER Diagram



Relational Schema



- 데이터베이스 설계 및 SQL 활용 시험

문제3. 1번의 E-R다이어그램을 관계형 모델로 생성하고SQL DDL코드를 생성하시요.(실습)

- 고객 테이블 : t_cus - 고객번호(숫자), 성명(문자), 주소(문자)

-주문서 테이블: t_order -주문번호(숫자), 품명(문자), 수량(숫자), 금액(숫자),고객번호(숫자)

정답) 생성된 관계형 모델(그림을 캡처하기)

정답) 생성된 SQL코드를 쓰시오.(생성된 SQL 붙여넣기)

- 데이터베이스 설계 및 SQL 활용 시험

문제4. 3번에서 생성된 SQL코드를 이용하여 table을 생성하고 아래의 데이터를DML의 insert 명령문을 이용하여 저장하시요 (insert SQL을 기술하시요)

t_cus

고객번호	성명	주소
001	가길동	수원시
002	나길동	안산시
003	조길동	서울시
004	홍길동	안양시

t_order

주문번호	품명	수량	금액	고객번호
101	사과	2	300	001
102	우유	3	200	001
103	시금치	4	100	002
104	콜라	7	200	002
105	두부	5	300	003
106	햄버거	2	200	003
107	빵	3	100	003
108	감자	1	200	003
109	오이	5	200	004

정답) insert SQL코드를 쓰시오.

- 데이터베이스 설계 및 SQL 활용 시험

4번에서 저장된 데이터를 보고 아래 질의에 대한 쿼리를 작성하시요. (결과의 순서는 상관없다) (5번~10번)

문제5. 고객번호 003번이 주문한 주문 내용을 아래 처럼 출력하는 SQL문장을 기술하시요.(join쿼리)

성명	주소	품명	수량	금액
조길동	서울시	두부	5	300
조길동	서울시	햄버거	2	200
조길동	서울시	빵	3	100
조길동	서울시	감자	1	200

정답) insert SQL코드를 쓰시오.

- 데이터베이스 설계 및 SQL 활용 시험

문제6. 고객 번호 002번이 주문한 주문의 총 금액은 얼마인가?

TOTAL
300

정답) insert SQL코드를 쓰시오.

- 데이터베이스 설계 및 SQL 활용 시험

문제7. 주문테이블을 보고 고객번호별 주문 금액의 총합을 구하여 출력하시요.(group by절 사용)

고객번호	금액
001	500
002	300
003	800
004	200

정답) insert SQL코드를 쓰시오.

- 데이터베이스 설계 및 SQL 활용 시험

문제8. 고객테이블에서 이름을 기준으로 고객을 내림차순으로 정렬하여 출력하시요.

고객번호	성명	주소
004	홍길동	안양시
003	조길동	서울시
002	나길동	안산시
001	가길동	수원시

정답) insert SQL코드를 쓰시오.

- 데이터베이스 설계 및 SQL 활용 시험

문제9. 004번 고객의 이름을 홍길순으로 주소를 인천시로 수정하는 SQL문장을 작성하시요.

정답) insert SQL코드를 쓰시오.

문제10. order 테이블에서 품명이 3글자 이상인 품명을 출력하시요.

주문번호	품명
103	시금치
106	햄버거

정답) insert SQL코드를 쓰시오.

[7일 완성] 생각하는 데이터베이스 모델링

(생각하는 DB1탄)

END